Lecture Notes in Computer Science 4197

Martin Raubal   Harvey J. Miller
Andrew U. Frank   Michael F. Goodchild (Eds.)

Geographic Information Science

4th International Conference, GIScience 2006
Münster, Germany, September 20-23, 2006
Proceedings

Springer

Volume Editors

Martin Raubal
University of Münster
48149 Münster, Germany
E-mail: raubal@uni-muenster.de

Harvey J. Miller
University of Utah
Salt Lake City
UT 84112-9155, USA
E-mail: harvey.miller@geog.utah.edu

Andrew U. Frank
Vienna University of Technology
1040 Vienna, Austria
E-mail: frank@geoinfo.tuwien.ac.at

Michael F. Goodchild
University of California
Santa Barbara
CA 93106-4060, USA
E-mail: good@geog.ucsb.edu

Preface

The GIScience conference series (www.giscience.org) was created as a forum for all researchers who are interested in advancing research in the fundamental aspects of geographic information science. Starting with GIScience 2000 in Savannah, Georgia, USA, the conferences have been held biennially, bringing together a highly interdisciplinary group of scientists from academia, industry, and government to analyze progress and to explore new research directions. The conferences focus on emerging topics and basic research findings across all sectors of geographic information science. After three highly successful conferences in the United States, this year's GIScience conference was held in Europe for the first time.

The GIScience conferences have been a meeting point for researchers coming from various disciplines, including cognitive science, computer science, engineering, geography, information science, mathematics, philosophy, psychology, social science, and statistics. The advancement of geographic information science requires such interdisciplinary breadth, and this is also what makes the conferences so exciting. In order to account for the different needs of the involved scientific disciplines with regard to publishing their research results, we again organized two separate stages of paper submission: 93 full papers were each thoroughly reviewed by three Program Committee members and 26 were selected for presentation at the conference and inclusion in this volume. Then, 159 extended abstracts, describing work in progress, were screened by two Program Committee members each. Subsequently, 42 of them were selected for oral presentation, and 46 for poster presentation at the conference. All extended abstracts were published in a separate book in the Institute for Geoinformatics, University of Münster-Prints series.

The proceedings of GIScience 2006 provide ample evidence that the field of geographic information science is both maturing and growing. This year's number of full-paper submissions marked a record, topping the previous submission rate by more than 30%. Another important trend is the increasing competitiveness for entering the LNCS proceedings. The acceptance rate for full papers was below 28%, a clear indication of the quality of the papers in this book. The mixture of topics demonstrates that the field is still exploring classic topics, but also moving into novel directions. Among the traditional topics we find spatial representations and data structures, spatial and temporal reasoning, computational geometry, spatial analysis, and databases. A significant number of papers deal with navigation, interoperability, dynamic modeling, ontology, and semantics. Geosensors, location privacy, social issues and GI research networks rank among the important and exciting new directions.

In addition to the paper sessions, GIScience 2006 included four keynote addresses, five pre-conference workshops, an interactive poster session, and social

events. The water castle of Nordkirchen, one of the most beautiful sights in the area, served as the location for the gala dinner. The conference itself was held at Münster Castle—built from 1767 to 1787, formerly the Prince Bishop's residence, now part of the University—and located in one of the most livable cities in the world. In October 2004 Münster became the first German city to win the Gold LivCom Award (www.livcomawards.com) in the category of cities with 200,000 to 750,000 inhabitants. This year's conference turned out to be yet another highlight of the series combining excellent research with a picturesque background.

As Program Co-chairs, we thank the many people who helped to make GIScience 2006 a success: those who submitted their work and participated in the conference, the co-organizers, the General Chair, the Program Committee and additional reviewers, the Local Organizing Committee, the Web team, the staff of the Institute for Geoinformatics and the University of Münster, and last but not least our sponsors. Special thanks go to Carsten Keßler for his help with these proceedings.

July 2006
<div align="right">

Martin Raubal
Harvey Miller
Andrew Frank
Mike Goodchild
</div>

Organization Committee

Program Chair

Martin Raubal, University of Münster

Program Co-chairs

Harvey J. Miller, University of Utah
Andrew U. Frank, Vienna University of Technology
Michael F. Goodchild, University of California, Santa Barbara

General Chair

Werner Kuhn, University of Münster

Workshop Chair

Antonio Krüger, University of Münster

Program Committee

Dave Abel, CSIRO
Pragya Agarwal, University College London
Luc Anselin, University of Illinois, Urbana-Champaign
Claudia Bauzer Medeiros, University of Campinas
Michela Bertolotto, University College Dublin
Thomas Bittner, SUNY Buffalo
Barbara Buttenfield, University of Colorado at Boulder
Gilberto Câmara, INPE
Nicholas Chrisman, Laval University
Christophe Claramunt, Naval Academy Research Institute
Anthony Cohn, University of Leeds
Helen Couclelis, University of California, Santa Barbara
Isabel Cruz, University of Illinois at Chicago
Leila de Floriani, University of Genova
Jürgen Döllner, HPI - University of Potsdam
Matt Duckham, University of Melbourne
Geoffrey Edwards, Laval University

Max Egenhofer, University of Maine
Sara Fabrikant, University of Zurich
Peter Fisher, City University London
Christian Freksa, University of Bremen
Mark Gahegan, Penn State University
Ralf Güting, University of Hagen
Francis Harvey, University of Minnesota
John Herring, Oracle Corporation
Gerard Heuvelink, Wageningen University
Stephen Hirtle, University of Pittsburgh
Hartwig Hochmair, St. Cloud State University
Christian Jensen, Aalborg University
Chris Jones, Cardiff University
Marinos Kavouras, National Technical University of Athens
Daniel Keim, University of Konstanz
Menno-Jan Kraak, ITC
Benjamin Kuipers, University of Texas at Austin
Lars Kulik, University of Melbourne
Paul Longley, University College London
David Mark, SUNY Buffalo
Daniel Montello, University of California, Santa Barbara
Atsuyuki Okabe, University of Tokyo
Harlan Onsrud, University of Maine
Dimitris Papadias, University of Science and Technology, Hong Kong
Donna Peuquet, Penn State University
Dieter Pfoser, Research Academic Computer Technology Institute
Enrico Puppo, University of Genova
Jonathan Raper, City University London
Jochen Renz, The University of New South Wales Sydney
Claus Rinner, University of Toronto
Andrea Rodríguez, Universidad de Concepción
Christoph Schlieder, University of Bamberg
Timos Sellis, National Technical University of Athens
Monika Sester, University of Hannover
Shashi Shekar, University of Minnesota
Eric Sheppard, University of Minnesota
Takeshi Shirabe, Vienna University of Technology
Barry Smith, SUNY Buffalo
John Stell, University of Leeds
Kathleen Stewart Hornsby, University of Maine
Barbara Tversky, Columbia University
Marc van Krefeld, Utrecht University
Peter van Oosterom, Delft University of Technology
Achille Varzi, Columbia University
Agnes Voisard, Fraunhofer ISST and FU Berlin

Rob Weibel, University of Zurich
Stephan Winter, University of Melbourne
Mike Worboys, University of Maine
May Yuan, University of Oklahoma

Additional Reviewers

Mete Celik	Paola Magillo	Aristidis Sotiropoulos
Frank Dylla	Kyriakos Mouratidis	Emmanuel Stefanakis
Birgit Elias	Laura Papaleo	Giuliano Torrengo
Betsy George	Kostas Patroumpas	Jan Oliver Wallgrün
Seda Gürses	Kai-Florian Richter	Nico van de Weghe
Sangho Kim	Joern Schneidewind	Huiyong Xiao
Margarita Kokla	Tobias Schreck	Jin Soung Yoo

Sponsoring Institutions

ifgi
Institute for Geoinformatics
University of Münster
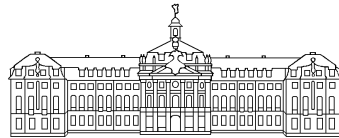
WESTFÄLISCHE
WILHELMS-UNIVERSITÄT
MÜNSTER

con terra

ESRI ++
ESRI Geoinformatik GmbH

ESRI
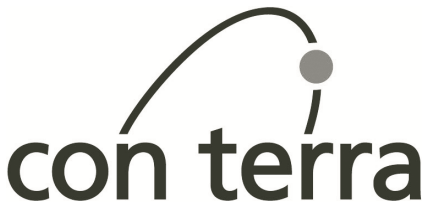
Table of Contents

A Social and Spatial Network Approach to the Investigation of Research Communities over the World Wide Web

Pragya Agarwal[1], Roderic Béra[1], and Christophe Claramunt[2]

[1] Department of Geomatic Engineering,
University College, London, UK
{pagarwal, rbera}@ge.ucl.ac.uk
[2] Naval Academy Research Institute, Lanvéoc-Poulmic,
BP 600, 29240 Brest Naval, France
claramunt@ecole-navale.fr

Abstract. The research presented in this paper introduces a graph-based and computational modelling approach that derives a social network and computes its emerging spatial and thematic properties from the semantics embedded in a series of Web pages. We apply several local and global graph-based operators, and complement them with thematic, spatial and similarity operators. This allows us to infer the degree of correlation between the different properties of the network. The approach is applied to the study of the GIS research communities as exemplified by the International COSIT and GIScience conferences.

Keywords: graph analysis, semantic space, social networks, GIScience, web communities.

1 Introduction

A social network is commonly defined as a set of people who share a common interest and have connections of some kind (Wasserman and Faust, 1994). Social networks provide useful insights into ways that the social communities are formed and interact. They have been widely studied over the past years, particularly in mathematical and statistical research (Strogatz, 2001; Newman, 2003), but have also been applied in many application domains such as epidemiology (Moore and Newman, 2000), environment (Dunne *et al.*, 2002) and scientific citation (Redner, 1998).

Social networks have been preferably studied using surveys, but these have certain limitations as connections are not always recorded in questionnaires. Surveys are also time consuming, can be subjective, costly and error prone. In order to address these limitations, new forms of studies have recently emerged to derive and compute social networks. In particular, the World Wide Web provides many opportunities for inferring and analysing social networks as web sites often embed information with respect to a specific domain of interest, and the actors that are part of it (Berners-Lee *et al.*, 2001; Greco *et al.*, 2002; Hou and Zhang, 2002; Bekkerman and McCallum,

2005). Derivation of a social network becomes a modelling and parsing problem where members of a group of interest and connections between them should be inferred from the information embedded on the web.

The research presented in this paper introduces a social and spatial network approach to infer and analyse the structure and properties of a semantic network derived from the relationships embedded and hidden in a subset of the World Wide Web. Our objective is to complement graph-based operators with spatial, thematic and network correlation measures that cannot be easily expressed by a graph layout, but can be virtually derived from the relationships embedded and hidden on the Web. This will help us in understanding the role of space and thematic classifications in a given social network, for instance, the degree to which network properties are correlated to other properties.

The domain used for our study is the semantic network formed by the scientific community closely related to Geographical Information Science. Network and graph-based analysis support exploration of patterns of collaborations in this field, and reveal to what degree trajectories of researchers impact the formation of a network of communities. The motivation for examining research networks is twofold. First, as members of this community, we are curious about the elaborations and extent of the field. Second, in a more general sense, the emergence of GIScience research is an instance of the broader phenomenon of new disciplines forming at the intersection of existing fields, and this research will lead the way forward for assessing how this community composes over time. To illustrate the approach we consider two major international conferences of the field: the Conference on Spatial Information Theory (COSIT) and Conference on Geographical Information Science (GIScience) that present the advantage of reflectingthe key players of the GIScience community. These two conferences appear relatively central in the field of Geographical Information Science, and are well documented on the Web. The main information inputs are sourced from the web sites of the last organised sessions of these conferences in 2004 and 2005, respectively, along with previous years to some degree. The components constituting the elements of the network are researchers that had papers accepted at these conferences, and connections given by their academic trajectories along universities. The objective of the study is not to study individuals and their networks, but rather to analyse the affiliation network of the universities and research centres that have supported the careers and progression of these researchers. The expected output is a network of universities active in this particular research field. We also assume that the trajectory of a given researcher from one university to another represents an implicit connection between these two universities. This information is derived and computed from the short biographies as they appear on the web pages of the researchers hyperlinked to the conference web sites. This supports derivation of a semantic network whose properties are analysed using graph measures, thematic and geographical-based correlations, and similarity measures.

In a preliminary study, we analysed the properties exhibited by these scientific network using graph-based measures and their distribution in space (Béra and Claramunt, 2005). This allowed us to infer the hubs and authorities of the social network, centrality patterns, and the distribution of these properties in space. This paper extends our previous work by enlarging the analysis in several directions. First, we generalise the measure of geographic dispersion to the thematic dispersion.

Secondly, we integrate a hierarchical component in the analysis in order to take into account several levels of abstraction in the thematic and spatial dimensions. Last, we apply a similarity measure to evaluate the dependencies between the two scientific networks. The remainder of the paper is organised as follows. Section 2 introduces the principles of our modelling and computational approach. The case study is developed in Section 3, and major results presented. Finally, Section 4 draws some conclusions and outlines future work.

2 Semantic Network Modelling

A semantic network is modelled as a hypergraph G (N,E) where N is a finite set of nodes, and E is a finite set of links between these nodes. A node is denoted as n_i, a link between two nodes $n_i, n_j \in N$ as a pair (n_i, n_j).

Network analysis combines global measures that outline the emergent structure of the represented graph, with local-based measures that characterise particularities in the graph. The most popular local measures are the degree (i.e. number of nodes connected to a node) and the clustering coefficient of a node. The clustering coefficient evaluates the extent to which the neighbours of a node i are also linked one to another. Global measures evaluate the centrality of a node in the graph. These include for example the average distance between nodes, the ratio of shortest paths a node lies on (Sabidussi, 1966), and accessibility of a node (Batty, 2004; Béra and Claramunt, 2003, White and Smyth, 2003). An essential measure of centrality is the *betweenness centrality* $C_b(i)$ that gives the fraction of shortest paths between node pairs that pass through a given node (Freeman, 1977; Brandes, 2001), and is defined as

$$C_b(i) = \sum_{k \neq j \neq i} \frac{S_{jk}(i)}{S_{jk}} \; , \tag{1}$$

where S_{jk} denotes the number of shortest paths from j to k and $S_{jk}(i)$ is the number of shortest paths from j to k that i lies on. The *betweenness centrality* can be generalised to multiple component graphs, as well as for directed graphs.

Although these graph-based measures are expected to exhibit local and global properties of a given network, they do not take into account the spatial and thematic dimensions. This leads us to introduce some additional geographically-related measures that correlate the structural and geographical properties of the semantic network. From a given semantic network $G(N, E)$, we consider the geographically-based subsets of N that group the nodes of N according to their membership to a given classification and partition of the underlying space (the same principles can be applied to a thematic classification and a measure of thematic dispersion). The objective is to analyse to which degree the structure of a network is correlated to a given classification. Let us consider a classification $C = \{c_1, c_2, \ldots, c_j, \ldots, c_p\}$ that forms a partition of N with $p > 1$. For a given node n_i of a class c_j of N, we define the geographic dispersion value $\mathrm{disp}(n_i)$ as follows:

$$\text{disp}(n_i) = \frac{|n_{i\rightarrow}|}{|N - c_j|} - \frac{|n_{i\leftarrow}|}{|c_j|} \qquad (2)$$

Where, $|n_{i\leftarrow}|$ gives the cardinality of the set of links that connect n_i to nodes that belong to the same class c_j of n_i, $|n_{i\rightarrow}|$ the cardinality of the set links that connect n_i to nodes that do not belong to the same class c_j of n_i. disp(n_i) values are given by the interval [−1,+1]. Values that tend to −1 (alternatively +1) denote a lower (alternatively higher) geographic dispersion of the links of n_i, values that tend to 0 denote a balanced geographic dispersion of the links of n_i. The aggregated coefficient of geographic dispersion of a semantic network is given by:

$$Disp(N) = \sum_{i=1}^{|N|} disp(n_i) = \sum_{i=1}^{|N|} \left(\frac{|n_{i\rightarrow}|}{|N - c_j|} - \frac{|n_{i\leftarrow}|}{|c_j|} \right) \qquad (3)$$

The analysis of the semantic network is completed by a local measure of heterogeneity that evaluates the ratio of the number of adjacent nodes of n_i belonging to another class to the number of adjacent nodes belonging to the same class as n_i:

$$Het(n_i) = \frac{|n_{i\rightarrow}|}{|n_{i\leftarrow}| + 1} \qquad (4)$$

This measure is generalised to the whole network in order to produce an aggregated measure of heterogeneity denoted by *Het (N)*:

$$Het(N) = \sum_{i=1}^{|N|} Het(n_i) = \sum_{i=1}^{|N|} \left(\frac{|n_{i\rightarrow}|}{|n_{i\leftarrow}| + 1} \right) \qquad (5)$$

A possible refinement could be to weight the measure of heterogeneity *Het(n_i)* so as to give more importance to well connected nodes, regardless of the homogeneity or heterogeneity of the nodes considered in terms of class membership. This weighted measure is given as follows:

$$WHet(n_i) = (|n_{i\rightarrow}| + |n_{i\leftarrow}|) \cdot Het(n_i) = \deg(n_i) \cdot \frac{|n_{i\rightarrow}|}{|n_{i\leftarrow}| + 1} \qquad (6)$$

Where, $\deg(n_i)$ is the degree of node n_i. The aggregated and weighted measure of heterogeneity is then:

$$WHet(N) = \sum_{i=1}^{|N|} WHet(n_i) = \sum_{i=1}^{|N|} \left(\deg(n_i) \cdot \frac{|n_{i\rightarrow}|}{|n_{i\leftarrow}| + 1} \right) \qquad (7)$$

Measures of geographical dispersion can be defined at different levels of granularity depending on the level of abstraction chosen, and different partitions of the same geographic space can be considered. Under similar principles, dispersion between thematic attributes can be derived. One can, for instance, assess the

permeability of a community into another. Although exclusive communities can be inferred, the most common situation is that of flexible and inter-penetrating communities. This implies that classifications cannot be assumed to be forming partitions, with entities of interest possibly belonging to several classes simultaneously.

Inferring the structural properties exhibited by the graph representation, and correlating them to geographic and thematic dimensions provide valuable inputs to the search for patterns and particularities in a semantic network. These computational measures also support cross-comparison between different semantic networks. However, semantic networks are interrelated in different ways, particularly on the web where they can appear implicitly or explicitly across different web sites. The subjective component of the nature of these relationships means that a   strategy is needed in order to evaluate the degree of semantic relationships and/or similarity between two given semantic networks. Given (and assuming) the fact that a semantic network can be qualified with some well defined labels, an intuitive measure of similarity between two semantic networks will be given by a cross-comparison of the number of pages where they are present in one form or another, or in other words by comparing their respective footprints left on the Web.

More formally, let $h(q)$ denote the set of hit pages returned by a search query q, q being a set of keywords $\{ k_1, k_2, ..., k_n \}$ that qualify a given semantic network N. A similarity measure between two semantic networks N_1 and N_2, respectively, materialised by the search queries q_1 and q_2, can be defined by the number of hits returned by the combined query $q = q_1 \cup q_2 = \{ k_{11}, k_{12}, ..., k_{1l} \} \cup \{ k_{21}, k_{22}, ..., k_{2m} \}$. The result of this query is a set of hit pages $h(q) = h(q_1 \cup q_2) = h(q_1) \cap h(q_2)$, since the pages fulfilling both query q_1 and query q_2 in the query $q_1 \cup q_2$ are at the intersection of pages fulfilling query q_1 and pages fulfilling query q_2. The number of hits retuned by the combined query $q_1 \cup q_2$ is then, $|h(q_1 \cup q_2)| = |h(q_1) \cap h(q_2)|$. The similarity measure between a search query q_1 and a search query q_2 is defined as:

$$Sim(q_1, q_2) = \frac{|h(q_1 \cup q_2)|}{|h(q_1)|} \tag{8}$$

3   Research Community Analysis

As mentioned in the previous section, this graph modelling approach described above, is illustrated by, and applied to, the web sites of the series of conferences GIScience 2004 and COSIT 2003 and 2005. For our study, only full papers accepted to these conferences were considered, which are 26 for COSIT 2003, 31 for COSIT 2005 and 25 for GIScience, so that all three samples can be considered of comparable size. A Google search was performed from the (complete) author and university names, looking for CVs and résumés on personal and/or departmental web pages, in order to

develop, as accurately as possible, trajectories from academic career-paths. Significant academic changes considered are any substantial changes in the university associations of the researcher. We chose not to use any prior or offline knowledge, to ensure that the whole graph is extracted from the World Wide Web (WWW). We are aware that some of the trajectory information may be missing. This will mainly be the case for authors not maintaining a personal web page or in departments providing little information for their staff. The online proceedings, accessible from Springer-Verlag's website, and the online reference database of the University of Trier (DBLP) were also helpful in deriving this trajectory information.

The Universities derived from the academic career-paths of the authors of the papers presented at these conferences provided the elements for the identification of the semantic network nodes, along with the relationships between the universities inferred from the academic trajectories. For example, the event of an academic who got her/his PhD at the University College London (UCL) and then moved to the University of Maine followed by the University of Edinburgh gives two directed links UCL-Maine and Maine-Edinburgh but also between UCL and Edinburgh in order to reflect cumulative relationships. This leads to the appearance of clusters and networks of universities within the graph.
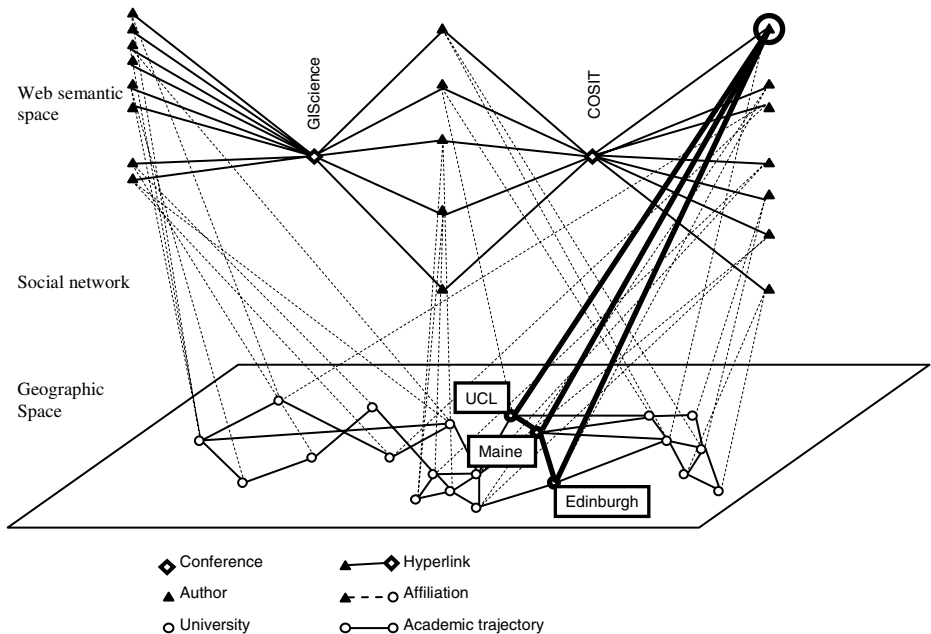
Fig. 1. Semantic networks derived for conference authors from the WWW indicative of social networks when superimposed over geographic locations. Highlighted is the example trajectory.

It is expected that the analysis of the graph and geographically-based emerging properties of the semantic network from the information implicitly available over the WWW information space should help in making apparent and for qualifying the

degree of integration of the research community. Figure 1 shows how the superimposition of the semantic network derived from the web and the geographic space derived from the world map leads to the derivation of the social network. In this case, the footprints of the different trajectories are also made apparent.

3.1 Structural Analysis

Thepatterns and trends that we consider in our analysis are : key university players in the research area of interest, connections between these universities, degrees of compactness versus spread of the research community, clusters, and peripheries. The networks of universities derived from the two conference web sites have several emerging characteristics as illustrated in figure 2 (note that the authors are represented in a semantic space made of three overlapping domains: COSIT 2003, GIScience 2004, and COSIT 2005).

Figure 2 shows in- and out-degrees through the number and direction of edges joining the network's vertices, and the *betweenness centrality* measure (undirected, with normalisation of values belonging to different components) of nodes is represented by their size. This figure, while showing the primary networks for all the three events, also shows the common space they share, provided by the actors and networks that have contributed to all three conferences. The central component of the graph connects most of the universities, with highly connected and attractive nodes reflecting 'key players', as represented by the relative sizes of the nodes in this case. Beside centrality, connectivity (in- and/or out-degrees) is a way of assessing the 'popularity' or attractivity vs. repulsivity of a node. For instance, this is noticeable for the University of California at Santa Barbara (UCSB), the State University of New York, Buffalo (SUNY Buffalo) and the University of Melbourne. The high (betweenness) centrality value, amounting to the larger size of the node, is a measure of the number of authors connected in one way or another during their trajectories to these 'key players'. Other highly attractive nodes are related to the academic changes of a very mobile researcher, who connects 6 universities together (i.e., SUNY at Buffalo, the University of Leipzig and Saarland University, Schaan, Sheffield, and Manchester). Also, some vertices are highly connected but appear rather repulsive (high out-degree, when compared to their in-degree), as is the case of the Technical University in Vienna (TUW). This is due to the fact that many junior researchers, getting their doctoral diploma at TUW move to other sites in the network, providing an important 'input' and attractiveness measure to other nodes in the research network. Only few other small isolates are present. As a measure of the integration (or 'unavoidability') of nodes, the *betweenness centrality* measure is indicative of how likely the members of the research community are to move through these nodes, which tend to be also well connected (thus forming hubs and authorities). Taking connectivity into account, the higher values are obtained essentially for universities involved in all three events (COSIT 2003 and 2005, GIScience 2004), and for the ones that have most authors associated to them along their academic trajectories.

The semantic networks show that the connectivity characteristics of COSIT differ from that of GIScience. Whereas in COSIT the graph is strongly dominated by a main
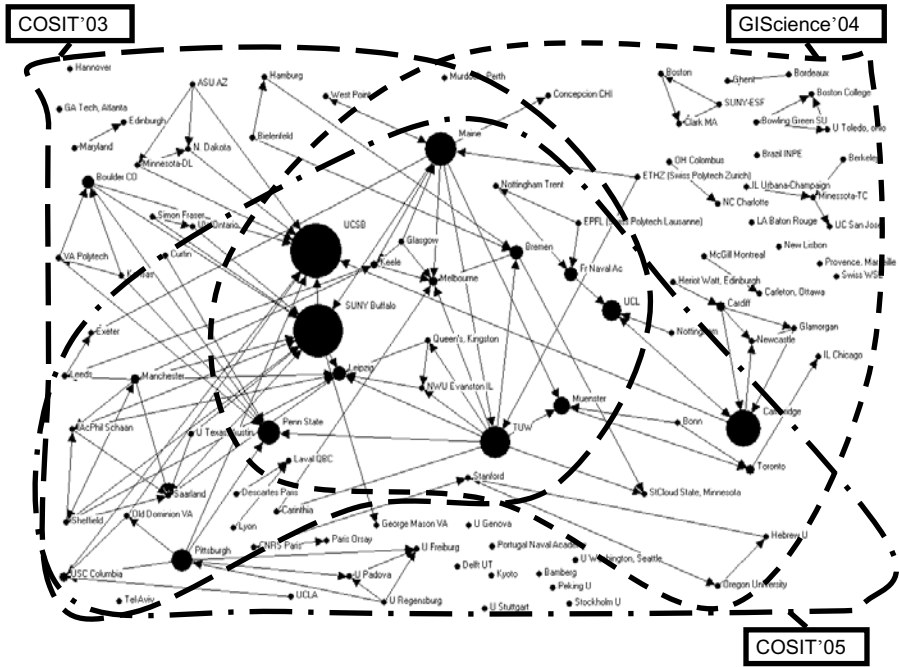
Fig. 2. Network of universities derived from COSIT 2003, COSIT 2005, and GIScience 2004. The nature of involvement is materialised by three overlapping sets; the central area (intersection of the above mentioned sets) is that of the universities present in all 3 events. The size of the nodes reflects the magnitudes of their betweenness centralities.

central component made of 33 nodes, GIScience, on the other hand, is less structured with respect to the academic trajectories, composed of numerous smaller and isolated components. Moreover, the community in GIScience conferences is less connected to the networks composing the COSIT conferences, and the nodes have relatively lower *betweeness centralities*, at least in parts of the network which are not common to both conference series. This suggests that the academic trajectories have played a smaller role in establishing the community for the GIScience conference as compared to the one for COSIT. However, it seems clear that there is some other process at work in the emergence and cohesion of the GIScience network, and that this process has a strong geographic component since almost exclusively North American universities are involved. Also it is worth noticing that in the central component of the graph, all of them except one (Concepcion University in Chile) are involved in COSIT 2003, whereas only a third of the total were involved in the GIScience conference.

Figure 3 shows the trajectories and the migratory patterns of the researchers, active in these communities. For clarity, cumulative links have been removed. This figure also presents the patterns of researchers gravitating towards certain major universities that can be derived as 'key players', as were evident from the networks displayed in figure 2. Sizes of the nodes are proportional to the degree. Here, it appears that best connected universities have a central role in shaping the academic flow along the
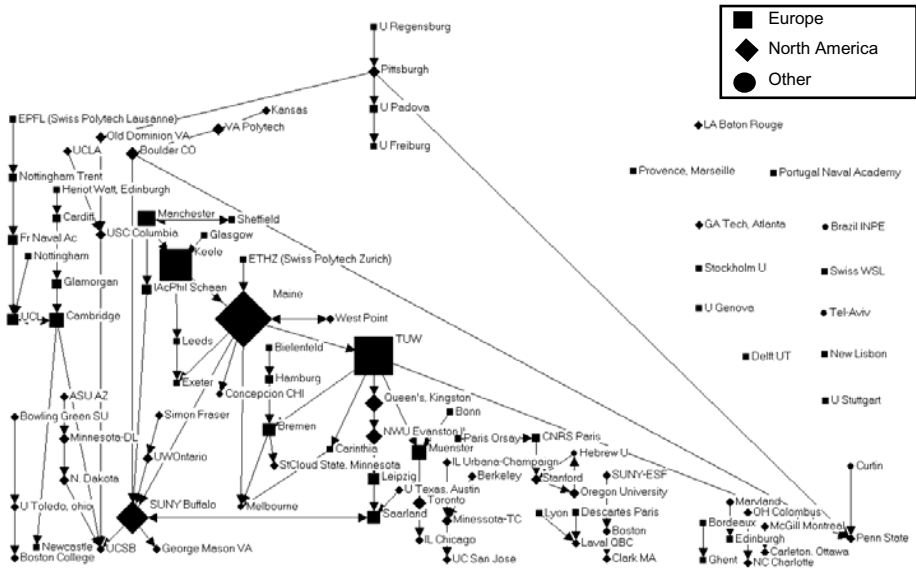
Fig. 3. Migratory Patterns of researchers. Trajectories flow top-down. Node sizes are proportional to the degrees of connectivity.
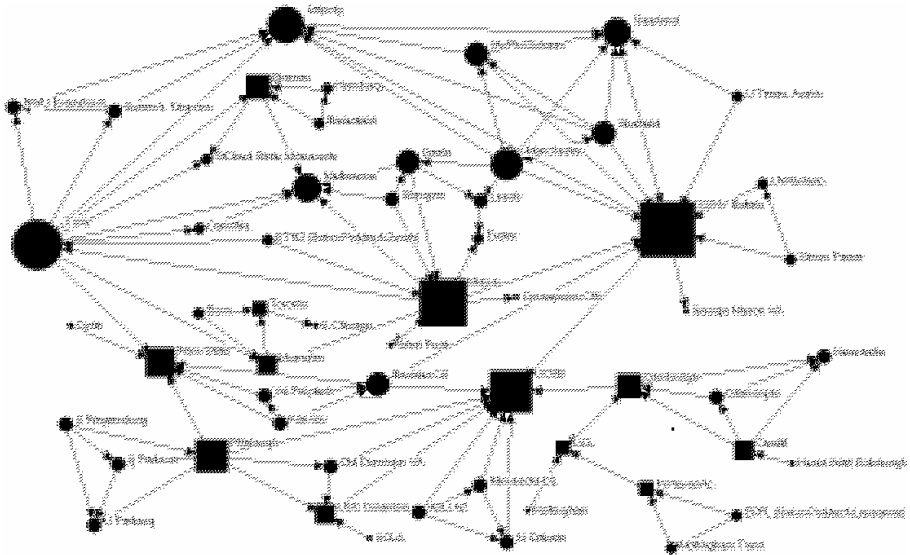
Fig. 4. Demonstrative network showing the role of 'key players' in maintaining the connectivity of the graph's main component. Cut points appear as squares. Node sizes are proportional to their degrees of connectivity.

network but are in no way end points for these trajectories. A geographic pattern is also apparent when considering the continental location of the Universities. For example, if the origins of the trajectories seem well distributed between Europe and North America, the end points have a clear trend towards being situated in North America.

The 'key players' in the network can also be considered from another perspective showing clearly their central role in maintaining cohesion in the graph and their roles as a bridge between smaller communities that otherwise would have been disconnected. Figure 4 shows these key players as cut points as well as nodes of high degree within the graph's main component. Overall, high betweenness centrality, high degree, and cut point situation appear to be correlated, as in the case of TUW, Maine, SUNY, or UCSB.

3.2   Dispersion Analysis

Figure 5 shows the semantic space partitioned according to a country-based classification. The dispersion factor (cf. equation 2) outlines the intra- vs. extra-national links. Highly negative dispersion reflects a university whose connections are predominantly within their own country, while large positive dispersion is the case of a university with higher connectedness to international counterparts. The highest geographical dispersion is obtained for the TUW, followed by University of Maine (i.e. many international trajectories transit or end there). It is worth noting the cases of the University of Melbourne (which has recently attracted several trajectories, mainly from Europe), the International Academy of Philosophy, Schaan (Liechtenschtein) and the University at Leipzig, all three being outliers with the characteristic of being on the path of international trajectories. For instance, the University of Melbourne has a high connectedness with universities outside Australia, while no connection within Australia.

Similarly, at a higher level of abstraction, an inter-continental partition system can be imposed on this component of the network space. Figure 6 shows the universities preferentially linked to those in other continents, when considering academic trajectories, using the dispersion index similar to that used for the previous analysis using countries. It appears that the Universities of Melbourne and Maine have a prominent centrality and positive dispersions with more inter-continental connections than within the same continent. University of Melbourne demonstrates a high dispersion because its attractivity and connectedness (as mentioned above) is by nature transcontinental (Australia is the only country represented in the set of conferences belonging to the whole area situated within the Indian and the Pacific oceans). Similarly, the University of Maine has attracted many researchers, mainly from Europe, at any moment in their academic trajectories. It is also worth noting that the TUW is, in this case, less prominent and with low centrality than the previous analysis as most of the connections and trajectories of researchers are linked to other universities in Europe. It is no major surprise that the least intercontinental nodes appear to be mainly European or American since both geographic areas contain a large number of Universities active in this particular discipline and, therefore, contain the majority of the graph's nodes. Also, most of the intra-continental (and also sometimes intra-country) nodes belong to the USA and to a lesser extent to the UK.

The least internationally connected node (though it is a key player in terms of centrality as well as in terms of connectedness and network cohesion is UCSB (it was shown in figure 4 as a cut point), with only 1 international node out of the total 9 nodes that it is connected to (figure 6).
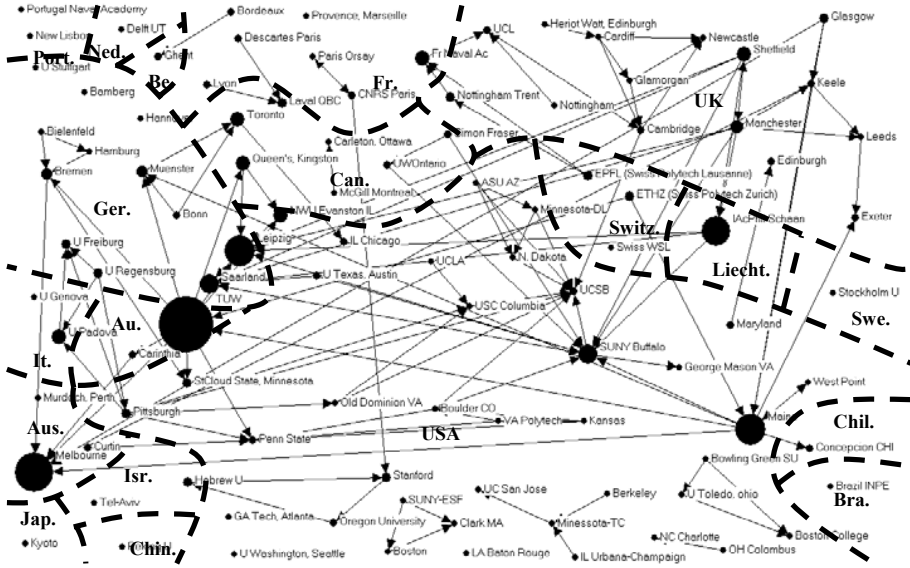
Fig. 5. Intra-national dispersion and researchers' trajectory network
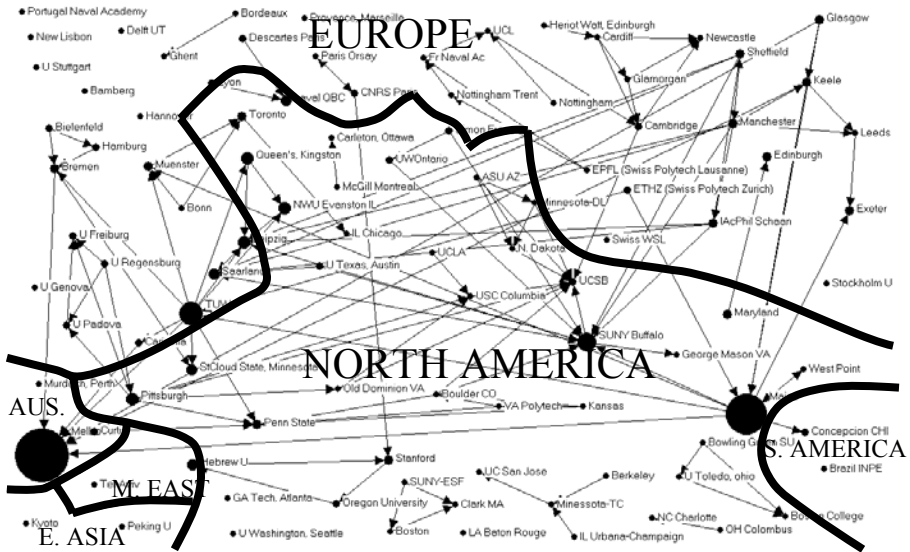
Fig. 6. Intra-continental dispersion and trajectory networks for researchers

3.3 Heterogeneity Analysis

Similar national and continental networks and patterns can be analysed using the heterogeneity index (cf. equation 4), which gives relative higher weighting to those nodes that are more connected to the nodes outside the host nation than to other nodes inside it. This also provides a comparative view on networks using different thematic parameters. In this case (see figure 7), when applying the heterogeneity index to an intra-national semantic space, it is noticed that University of Melbourne still stands out as the only prominent node within Australia (for similar reasons as explained in the previous section). On the other hand, several universities in Canada appear central and prominent. TUW maintains a central position too, as relation to its intra-national network, with researchers migrating to universities in other countries. Similarly, Universities of Saarland and Leipzig emerge as central in the research network as they support connections and movements of researchers to other nodes on the research network, more to countries lying outside Germany than within it. Within the UK, the universities fail to get high magnitude of heterogeneity, since most trajectories are intra-national, or at least intra-continental.
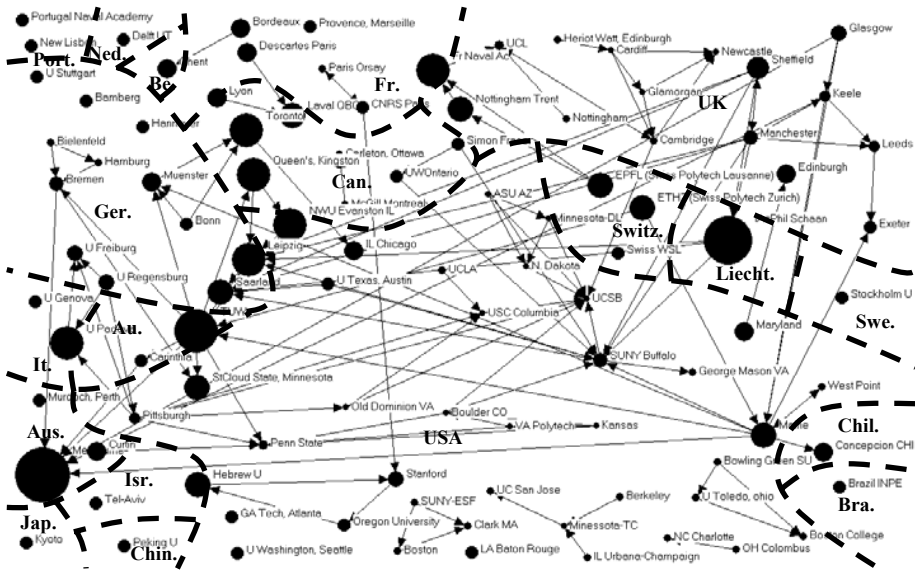
Fig. 7. Un-weighted intra-national heterogeneity

It is worth noting the changes in patterns of heterogeneity measures for the individual nodes when an intra-continental space is analysed (figure 8). It is observed that the University of Melbourne maintains its heterogeneity and attractivity in an intra-continental dispersion space, while there is a noticeable difference in TUW, which in this case appears to lose its prominence in terms of heterogeneity. This is because TUW has maximum connections within Europe, having hosted researchers at several points of time in their research trajectories and networks, as compared to those

outside Europe. St Cloud State, Minnesota appears quite heterogeneous, in this case, due to its connectivity primarily to Bremen and TUW, both outside North America, while not supporting any research collaborations or migrations with other Universities within the host continent.
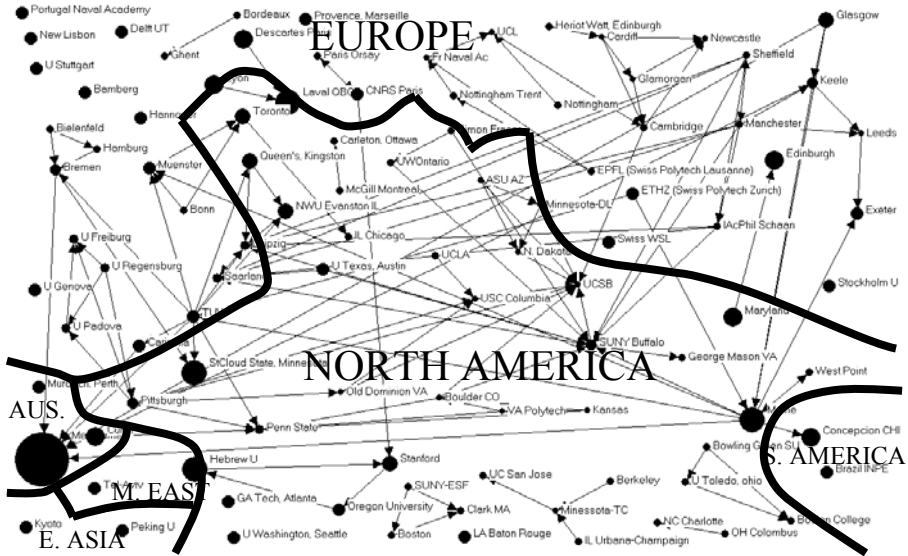
Fig. 8. Un-weighted intra-continental heterogeneity
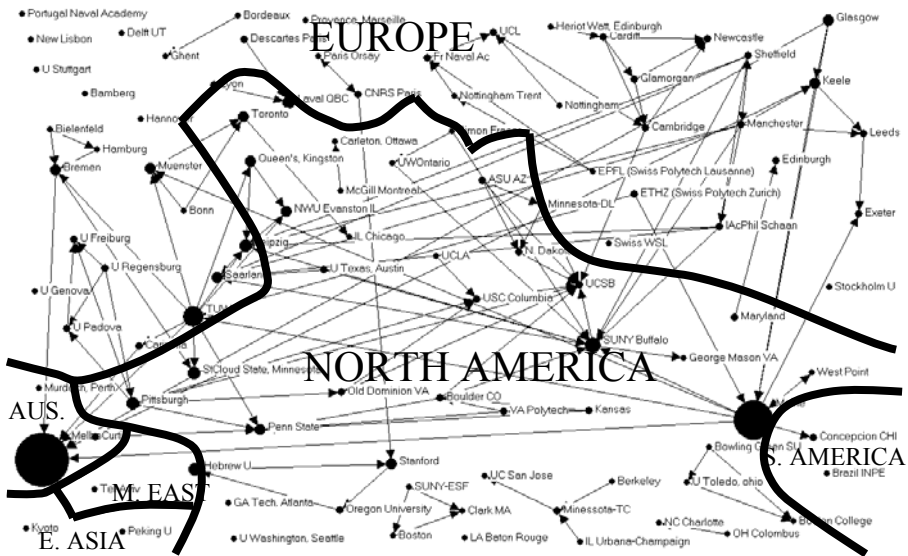
Fig. 9. Weighted continental heterogeneity

Figure 9 shows a comparative analysis to that shown in fig. 8 using the weighted measure for heterogeneity (cf equation 7) and the influence on node centralities using this measure, where the homogeneity or heterogeneity of the class is less important as compared to the relative connectivity of the node itself. The University of Maine appears significant because of the number of connections and networks that it hosts both internationally and within the USA. TUW appears prominent and has higher magnitude than that in figure 8 as, in this case, the relative geographic locations of the connected nodes are considered less important than the number of connections itself. St Cloud, Minnesota, on the other hand, seems to have 'lost' its centrality as the numbers of connections to and from this node are relatively few. No significant changes appear for the University of Melbourne.

The geographical dispersion index as well as the heterogeneity measures (in both weighed and unweighted version) support the analysis of the types of connections between the different universities. For instance, it appears that UCSB plays a key role in the USA from a national point of view, as it tends to be connected exclusively to other US universities. This is not the case of SUNY Buffalo or the University of Maine as both share, almost exclusively, connections with non-US universities. This highlights a difference in policy (conscious or not) of academic appointments and the university attractivity abroad. For example, the departments involved in the above mentioned conferences are mainly North-American and European, and North America appears more attractive to European academics than the reverse. Nevertheless, and for diverse reasons – of which one can imagine that geographical remoteness plays a role – the East Coast of the USA drags most of the Europeans, who might feel not too far away from home, as they are just 'on the other side of the pond'.

3.4 Similarity Analysis

We complement the graph-based measures by a similarity analysis applied to the two conference series COSIT and GIScience using keywords based on the application of search results from 'Google', a popular search engine over the WWW. Although previous analysis has shown that both conferences have their specific research themes and networks, these are far from being disconnected. The analysis in the previous sections also demonstrated the extent of overlaps between these two research communities and networks. One can then make an attempt at comparing the activity of these networks by measuring their respective footprints left on the web. A straightforward measure of popularity of a conference, and for a given year, is given by the number of Google "hits" derived from a search query defined using appropriate keywords such as the year, name and location of that conference. The following are results returned in February 2006:

$h(C2005) = h(\{COSIT+2005+Ellicottville\}) = 225$

$h(G2004) = h(\{GIScience+2004+University+Maryland\}) = 677$

$h(C2003) = h(\{COSIT+2003+Ittingen\}) = 346$

$h(G2002) = h(\{GIScience2002+Boulder\}) = 603$

$h(C2001) = h(\{COSIT+2001+Morro+Bay\}) = 427$

h(G2004 ∪ C2005) = 12

h(C2005 ∪ C2003) = 21

h(C2005 ∪ C2001) = 16

h(G2002 ∪ G2004) = 120

h(C2001 ∪ C2003) = 111

Similarity measures, using the similarity index introduced in equation 8, are then as follows:

Sim(G2004,C2005) = 0.017

Sim(C2005, C2003) = 0.09

Sim(C2005,C2001) = 0.07

Sim(G2002,G2004) = 0.19

Sim(C2001,C2003)= 0.26

These similarity figures show several patterns. First, GIScience, although more recent than COSIT, is much more apparent on the web than COSIT. Secondly, a more expected trend is that the two conferences appear in a relatively independent manner on the web, and the conferences in a series (e.g., Sim(G2002,G2004) = 0.19, Sim(C2001,C2003)= 0.26) are more related than conferences across different series (e.g., Sim(C2005, C2003) = 0.09, Sim(C2005,C2001) = 0.07). Third, more recent conferences have more visibility and higher similarity than the ones lying further apart. This could be because the format of the conference series COSIT, although relatively fluid, is becoming more standardised. These similarity figures are also representative of the quantitative strength of the represented communities, and will be useful in future work to analyse the values of the links and overlaps in the networks for individual communities as represented by individual conference series.

4 Conclusions and Future Work

In the present digitally connected age, the degrees of separation are becoming lower every day. The World Wide Web provides an important avenue for mapping the underlying semantic and social space that offers many opportunities for generating and studying social networks. The modelling framework proposed in this paper infers and derives a social network from the information on a domain knowledge embedded over the Web. The research communities as supported by the GIScience and COSIT conferences series were used as a demonstrative knowledge domain. Our proposal provides a preliminary contribution towards the inference and exploration of semantic and spatial relationships from the web-based information space. This approach also adds another dimension to the current research in social networks by combining thematic and geographic operators within such analysis and visualisation. Graph, thematic, geographic and similarity operators support a structural and spatial analysis of the network properties. Integration of the spatial and thematic dimensions within

the structural analysis permits the correlation of the structural properties with the underlying geographic space. The analysis provided several interesting patterns revealing the universities that play central or outlier roles within these research communities, and provided significant indications of the nature of the migratory patterns of researchers, as well as collaborative and academic research flows between universities. Most, if not all, of the results in the case study illustrate the theoretical fundamentals of our modelling approach, the intention being not to be exhaustive but representative of the possibility of the framework.

Further work will concern the extension of the approach to the whole series of COSIT and GIScience conferences, comparison with other conferences in the GIS domain, and integration of the temporal dimension in the study and analysis of researchers' trajectories. The migratory patterns for researchers are of great interest in establishing the network of communities within this discipline and this study will be extended as part of future work. Several dispersion and heterogeneity measures were introduced in this paper. Future work will also extend on the work presented here by exploring the comparative effectiveness and applicability of these different indices for social network analysis. This paper also presented an initial basis for calculating similarity measures between distinct social networks from the web. This will be further extended to validate the role of the 'key players' within this particular research community, and to find the distances and closeness between individual nodes in the research networks. This will be used to develop a more integrated and mathematical profile for the nature of collaborative networks and their respective strengths that form the basis of the research network and community in this particular discipline.

References

Batty, M., 2004, *Distance in space syntax*, Working Paper, n° 80, Centre for Advanced Spatial Analysis, UCL, London.

Bekkerman, R. and McCallum, A., 2005, Disambiguating web appearances of people in a social network, in *Proceedings of the 14th international conference on World Wide Web*, ACM Press, Chiba, Japan, pp. 463-470.

Béra, R. and Claramunt, C, 2003, Relative adjacencies in spatial pseudo-partitions, *Conference on Spatial Information Theory (COSIT '03)*, W. Kuhn, M. Worboys, and S. Timpf (eds.), Springer-Verlag, LNCS 2825, Ittingen, Switzerland, pp.218-234.

Béra, R. and Claramunt, C, 2005, Connectivity Inferences over the Web for the Analysis of Semantic Networks, C. Vangenot and K. J. Li (eds). In *Proceedings of the 5th International Workshop on Web and Wireless GIS (W2GIS'05)*, Springer-Verlag, LNCS 3833, Lausanne, Switzerland, pp. 222-234.

Berners-Lee, J., Hendler, J. and Lassila, O., 2001, *The Semantic Web*, Scientific American, vol. 184, n. 5, pp. 34-43.

Brandes, U., 2001, A faster algorithm for betweenness centrality, *Journal of Mathematical Sociology* 25(2):163-177.

Dunne, J. A., Williams, R. J. and Martinez, N. D., 2002, Food web structure and network theory; the role of connectance and size, *Proceedings of National Academy of Sciences*, Vol. 99, USA, 12917-12922.

Freeman, L. C., 1977, A set of measures of centrality based on betweenness, *Sociometry*, 40:35-41.

Greco, G., Greco, S. and Zumpano, E., 2002, A stochastic approach for modelling and computing web communities. In *Proceedings of the 3rd International Conference on Web Information Systems Engineering*, IEEE Press, Singapore, pp. 43-52.

Hou, J. and Zhang, Y., 2002, A matrix approach for hierarchical web page clustering based on hyperlinks. In *Proceedings of the 3rd International Workshops on Web Information Systems Engineering*, IEEE Press, Singapore, pp. 207-216.

More, C. and Newman, M. E. J., 2000, Epidemics and percolation in small-world networks, *Phys. Review E* 61, 5678-5682.

Newman, M. E. J., 2003, The structure and function of complex networks, *SIAM Review* 45, 167-256.

Redner, S., 1998, How popular is your paper ? An empirical study of the citation distribution. *European Phys. J. B.* 4, 131-134.

Sabidussi, G., 1966, The centrality index of a graph, *Psychometrika*, 31:581-603.

Strogatz, S. H., 2001, Exploring complex networks, *Nature* 410, 268-276.

Wasserman, S. and Faust, 1994, *Social Network Analysis*, Cambridge University Press, Cambridge.

White, S. and Smyth, P., 2003 Algorithms for estimating relative importance in networks, proc. 9th Conference of the ACM Special Interest Group on Knowledge Discovery in Data Mining (SIGKDD), Washington, D.C., 266-275.

Projective Relations in a 3D Environment

Roland Billen[1] and Eliseo Clementini[2]

[1] Geomatics Unit, University of Liege, 17 Allée du 6-Août,
B-4000 Liege, Belgium
`rbillen@ulg.ac.be`
[2] Dept. of Electrical and Information Engineering, University of L'Aquila,
I-67040 Poggio di Roio, L'Aquila, Italy
`eliseo@ing.univaq.it`

Abstract. This paper presents a model for positional relations among bodies of arbitrary shape in three dimensions. It is based on an existing model for projective relations among regions in two dimensions. The motivation is to provide a formal qualitative spatial relations model for emerging 3D applications. Two sets of relations are defined: ternary projective relations based on the concept of collinearity between a primary object and two reference objects and quaternary projective relations based on the concept of coplanarity between a primary object and three reference objects. Four sets of JEPD relations are defined for points and bodies in **R³**.

1 Introduction

The aim of this paper is to define a model for positional relations among bodies of arbitrary shape in three dimensions. The result is obtained by extending a model for projective relations among regions that was presented in [4, 6]. We approach the problem without considering any external frame of reference for defining the relations. Therefore, a kind of implicit frame of reference is determined by objects taking part of the relation. To explain this concept, let us consider two bodies in 3D space: without a frame of reference, the only relations that can be assessed are binary topological relations, such as being disjoint or intersecting [16]. If we consider three bodies in 3D, then we can express relations such as a body A is *before* bodies B and C, where bodies B and C implicitly define a direction, or such as a body A is *between* bodies B and C. In these cases, we talk about ternary projective relations, where the first object can be considered a *primary object* and the second and third one are the *reference objects*. In classical reasoning with orientation relations (see, for instance, [10]), we find a primary object, which is compared to a reference object in a given frame of reference. In our approach, we could say that the reference objects have the same role of a combination of the reference object and frame of reference in classical approach. Going a step farther, if we consider four bodies in 3D, we are able to define other projective relations, such as a body A is *above* or *below* the bodies B, C, and D. The need of considering quaternary projective relations arises because three bodies are necessary to define a concept of *coplanarity*, which in turn can define what is above or below. In this view, there is again a primary object A and three reference

objects B, C, and D. Quaternary relations are the main difference when we extend the model from 2D to 3D, since in 2D ternary relations are sufficient, being based on the concept of *collinearity* of three regions [5]. Previous work on projective relations among bodies is rather limited [2, 9].

With the development of 3D GIS, virtual reality, augmented reality, and robot navigation, qualitative relations are a key issue to perform relevant spatial analysis. Navigation in geographic environments, such as a city landscape [1], is a kind of application that can take advantage from our model to have a formal description of geometric relations among objects of the environment. Such a description is needed to build reasoning systems on these relations and facilitate a standard implementation in spatial database systems. Our work attempts to contribute to the enrichment of available formally-defined qualitative spatial relations.

The rest of the paper is organized as follows: after some mathematical background in Section 2, in Section 3 we describe the ternary projective relations among points in 3D. In Section 4, we deal with quaternary relations among points in 3D. In Section 5 and 6, we develop the model for ternary and quaternary relations among bodies, respectively. In Section 7, we outline further work and draw some conclusions.

2   Mathematical Background

Readers who are not familiar with projective geometry can find support in, e.g., [7]. Hereby, we limit the mathematical background to a brief outline of the general mathematical context and to a formal introduction to the concept of "body".

We consider ordinary objects of point-set topology, such as points and bodies, which are embedded in the Euclidean three-dimensional space $\mathbf{R}^3$. We avoid using any metric properties of objects, such as lengths, areas, and angles, and restrict ourselves to use the minimal number of geometric concepts in order to remain inside the domain of projective geometry. We take an axiomatic view of projective geometry, where fewer axioms than in Euclidean geometry are assumed. The classification of geometries based upon the action of a group of allowable transformations on a set was introduced by F. Klein [11]. Therefore, projective geometry is defined by projective transformations. The names "projective transformation", "homography", "collineation" and "projectivity" are all equivalent.
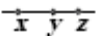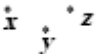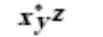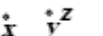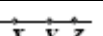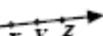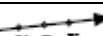
Bodies are bounded point-sets and will be indicated with capital letters *A*, *B*, *C*, etc. The interior of a body *A* is indicated with $A°$. The closure of a body A is indicated with $\overline{A}$. Bodies are *simple* if they are regularly closed (i.e., $A = \overline{A°}$) and without holes and disconnected components. Bodies are *complex* if they are regularly closed and have holes or disconnected components.

3   Ternary Projective Relations Between Points in a 3D Space

Ternary projective relations between points in 2D are extensively presented in [3, 4]. Note that other approaches have given similar results (see, e.g., [8, 13, 14]). As a brief

recall, we can say that the ternary projective relations model is based on an elementary concept of projective geometry: collinearity among points. At least three points are needed to define collinearity, and therefore it is intrinsically a ternary relation. Three points x,y,z (y and z being the reference points) are said to be *collinear* if they lie on the same line; we write $coll(x,y,z)$. From this basic projective relation, it is possible to build 9 other relations among three points. First, the *aside* relation is the negation of collinear relation. Knowing that the line joining the two reference points divides the space into two half-spaces; HP_{yz}^+ and HP_{yz}^-, it is possible to specialise the *aside* relation into *rightside* and *leftside* relations depending on which half-space the point x lies on. In the case the two reference points are coincident, we refine the relation *collinear* in the relations *inside* and *outside*. In the case the two reference points are distinct, we refine the relation *collinear* in the relations *between* and *nonbetween*. The relation *nonbetween* can be refined in the relations *before* and *after*. The set of ternary relations among points *rightside*, *leftside*, *between*, *before*, *after*, *inside*, *outside* is a jointly exhaustive and pairwise disjoint set of relations (JEPD) in $\mathbf{R^2}$.

Table 1. The definitions of ternary projective relations among points in $\mathbf{R^3}$

Name	short name	Definition	drawing
Collinear	$coll(x,y,z)$	$\exists\, line\ l : x \in l, y \in l, z \in l$	
Aside	$as(x,y,z)$	$\neg\, coll(x,y,z)$	
Inside	$in(x,y,z)$	$x=y \,\wedge\, y=z$	
Outside	$ou(x,y,z)$	$x \neq y \,\wedge\, y=z$	
Between	$bt(x,y,z)$	$y \neq z \,\wedge\, x \in [y,z]$	
Nonbetween	$nonbt(x,y,z)$	$coll(x,y,z) \wedge\, y \neq z \,\wedge\, x \notin [y,z]$	
Before	$bf(x,y,z)$	$coll(x,y,z) \wedge\, y \neq z \,\wedge\, x \in (-\infty_{yz}, y)$	
After	$af(x,y,z)$	$coll(x,y,z) \wedge\, y \neq z \,\wedge\, x \in (z, +\infty_{yz})$	

Ternary projective relations in $\mathbf{R^3}$ are almost totally equivalent to those in $\mathbf{R^2}$. All the definitions presented in Table 1 stand in $\mathbf{R^2}$ and $\mathbf{R^3}$ except the specialisation of the *aside* relation into *rightside* and *leftside* which is only possible in $\mathbf{R^2}$. Indeed, the line joining the two reference points does not partition $\mathbf{R^3}$ into two parts anymore (as it was the case in $\mathbf{R^2}$). Therefore, it is impossible to refine the *aside* relation. The set of ternary relations among points *aside*, *between*, *before*, *after*, *inside*, *outside* is a JEPD set of relations in $\mathbf{R^3}$ (figure 1).

Fig. 1. A decision tree for the ternary projective relations among points in **R³**

4   Quaternary Projective Relations Between Points in a 3D Space

Another set of projective relations can be defined in **R³** when considering one primary point and three reference points. Three non collinear points define one and only one plane in the space. Any plane in **R³** is a *hyperplane[1]* which means that it divides the whole space in two regions, called *halfspaces* (HS). Depending on the order of the three reference points (clockwise or counterclockwise), the plane can be oriented in **R³**, which allows to distinguish between a positive and a negative halfspace (HS⁺, HS⁻). Based on this partition, one can define projective relations between a point and three reference points: these relations are therefore *quaternary*.

A point w lying in a plane defined by the three reference points x, y and z is said to be coplanar with x, y and z, we write *copl (w,x,y,z)*. This *coplanar* relation can be refined depending on the relative position of the reference points. If the three reference points are *collinear*, then they define an infinity of planes that fill all the space. In such a degenerate case of coplanarity, we identify two quaternary projective relations called *inside* and *outside*. A point w is *inside* x, y and z if the three reference points are collinear and w belongs to the convex hull[2] of x, y and z (which corresponds to the segment joining the three points); otherwise, w is said to be *outside* x, y and z. These two relations are conceptually similar to *inside* and *outside* ternary relations. If the three reference points are *aside* then they generate two zones in the reference plane; a zone defined by the convex hull of the three points and its complementary

---

[1] A subset A of **R**ᵈ is an *affine subspace* if, for any distinct points x, y belonging to A, the straight line defined by x and y lies in A. Points, straight lines, planes, and **R**³ itself are the only affine subspace of **R**³. Their respective *dimensions* are 0, 1, 2 and 3. An affine subspace of dimension d-1 of **R**ᵈ is named *hyperplane*.

[2] The convex hull (*CH*) of two objects is formed by line segments joining each pair of objects' points; these line segments belong to objects' 1-transversals.

zone on the plane. A point w, coplanar with x, y and z, is said to be internal to x, y and z if it lies on the convex hull of x, y and z; otherwise it is said to be external to x, y and z. If the point w does not belong to a plane defined by the three reference points x, y and z, it is said to be non coplanar with x, y and z. This relation can be refined looking in which half-space the point w lies on. A point w which is non coplanar with x, y and z is said to be above x, y and z if it belongs to HS_{xyz}^{+}, or below x, y and z if it belongs to HS_{xyz}^{-}. Table 2 summarizes the quaternary projective relations between points in $\mathbf{R^3}$, which are illustrated in Figure 2.

The set of quaternary relations among points *above, below, internal, external, inside* and *outside* is a JEPD set of relations in $\mathbf{R^3}$ (Figure 3).

Table 2. The definitions of quaternary projective relations among points in a 3D space

Name	short name	Definition
Coplanar	*copl(w,x,y,z)*	$\exists\ plane\ p : w \in p, x \in p, y \in p, z \in p$
Non coplanar	*non_copl(w,x,y,z)*	$\neg\ copl(w,x,y,z)$
Above	*ab(w,x,y,z)*	$w \in HS_{xyz}^{+}$
Below	*be(w,x,y,z)*	$w \in HS_{xyz}^{-}$
Internal	*int(w,x,y,z)*	$aside(x, y, z)\ \wedge\ w \in CH(x, y, z)$
External	*ext(w,x,y,z)*	$copl(w,x,y,z)\ \wedge\ aside(x, y, z)\ \wedge$ $w \notin CH(x, y, z)$
Inside	*in(w,x,y,z)*	$coll(x,y,z)\ \wedge\ w \in CH(x, y, z)$
Outside	*ou(w,x,y,z)*	$coll(x,y,z)\ \wedge\ w \notin CH(x, y, z)$

Coplanar	*Non coplanar*	*Above*	*Below*
Internal	*External*	*Inside*	*Outside*

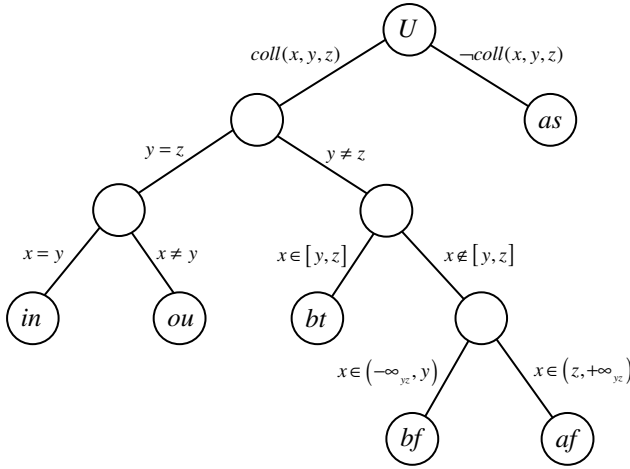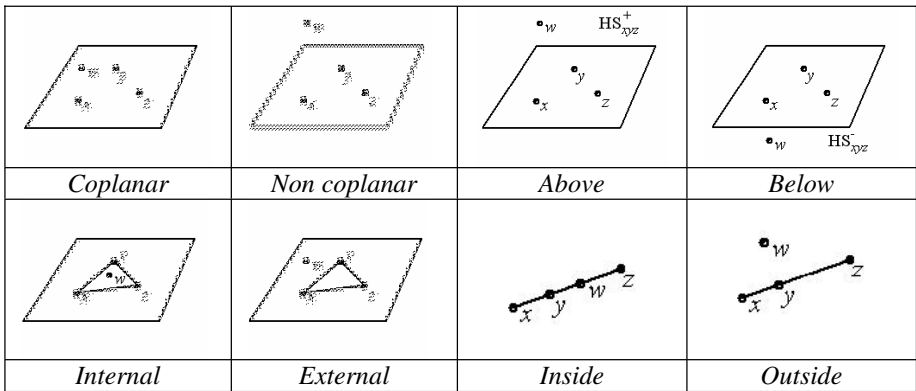**Fig. 2.** Quaternary projective relations among points in $\mathbf{R^3}$
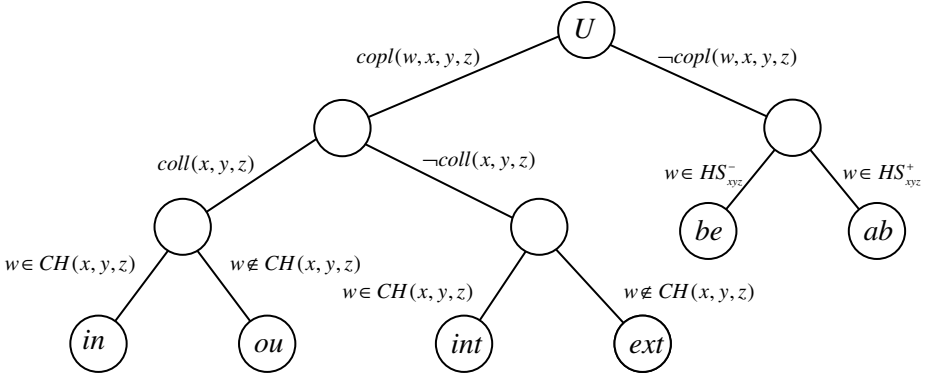
Fig. 3. A decision tree for the quaternary projective relations among points in $\mathbf{R}^3$

5   Ternary Projective Relations Between Bodies in a 3D Space

The importance of semantics of collinearity relies on the fact that modelling all ternary projective relations properties of spatial data can be done as a direct extension of such a property. The relation *collinear* among bodies can be introduced as a generalization of the same relation among points. Conceptually, collinearity relation between bodies in a 3D space does not differ from collinearity between regions in 2D. Among various definitions of collinearity [5], the most useful one is stated as follow:

Given three simple bodies A, B, $C \in \mathbf{R}^3$, $coll(A,B,C) \equiv_{def} \forall x \in A^\circ$ $[\exists y \in B^\circ$ $[\exists z \in C^\circ$ $[coll(x,y,z)]]]$;

Let us examine the geometric realization of collinearity among bodies. Similar to points, where we had a degenerate case of collinearity for coincident reference points, we have a degenerate case of collinearity among bodies if reference bodies have non-disjoint convex hulls: a body A is always collinear to bodies B and C if the intersection $CH(B) \cap CH(C)$ is non-empty. If the intersection $CH(B) \cap CH(C)$ is empty, we can identify a part of the space where a body A that is completely contained into it satisfies the relation *collinear*. Let us call this part of the space the *collinearity subspace* of B and C, $Coll(B,C)$. Such a subspace can be built by considering all the lines that are intersecting both B and C. Knowing that a line that intersects a family of convex sets is called a 1-transversal [15], the collinearity subspace can be equally defined as the set of all 1-transversals to (convex hulls of) bodies B and C.

In $\mathbf{R}^2$, the *collinearity zone* which is the 2D equivalent of the *collinearity subspace* was limited by the *internal* and *external tangents* of the two references objects (regions). In $\mathbf{R}^3$, the collinearity subspace is limited by two types of *tangent planes*; *internal tangent planes* that separates the two objects in different half-spaces (figure 4.a) and *external tangent planes* that leave the objects in the same half-space (figure 4.b). The intersection of all the half-spaces defined by internal tangent planes and containing the first reference body, union with the intersection of all the half-spaces

defined by internal tangent planes and containing the second reference body, union with the intersection of all the half-spaces defined by external tangent planes and containing the two reference objects define the *collinearity subspace* (figure 4.c).

| a. An internal tangent plane to two bodies | b. An external tangent plane to two bodies | c. Collinearity subspace of two bodies |
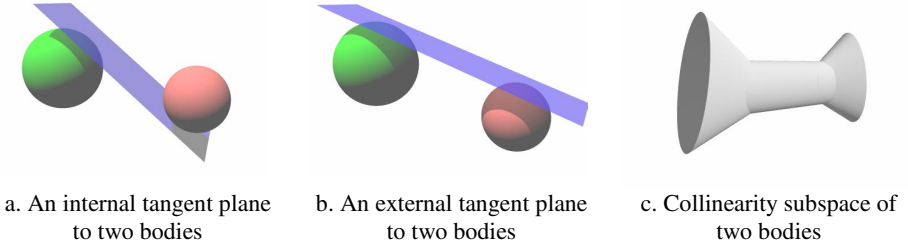
Fig. 4. Tangent planes to two bodies in **R³**

By definition, the convex hull of the union of two bodies is part of their collinearity subspace. The collinearity subspace of two bodies B and C (figure 5.a) can be subdivided in three parts; the convex hull of $B \cup C$ (figure 5.b) and two disjoint parts separated by the convex hull of $B \cup C$ which will be called *collinear subspace* $CS_{-\infty}$ (touching B) and *collinear subspace* $CS_{+\infty}$ (touching C) (figure 5.c).

| a. Collinearity subspace of bodies B and C | b. Convex hull of the union of B and C | c. Collinear subspaces $CS_{-\infty}$ and $CS_{+\infty}$ |
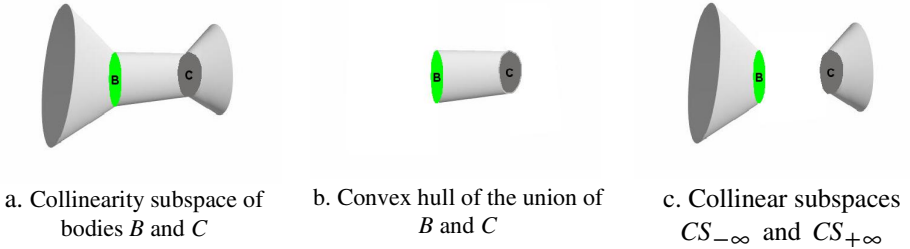
Fig. 5. Partition of the collinear subspace of two bodies in **R³**

In general, the part of the space where a body A that is completely contained into it satisfies a relation r is called the *acceptance subspace* of r. The collinearity subspace and all acceptance subspaces of relations are open sets: this corresponds to considering the interior of bodies in all the definitions of relations. This choice allows us to avoid limit cases: a point x in the boundary of body A that is falling in the boundary of an acceptance subspace does not influence the relation. If we had made the opposite choice, the point x would have contributed at the same time to the *collinear* and *aside* relations.

The definition of the relation *collinear* can be extended to complex bodies by considering the convex hulls of the reference objects in place of the reference objects. The definition of *collinear* together with other projective relations for bodies is given in Table 3. Besides each definition, also the corresponding acceptance subspace is given. The relation *collinear*, in the case the two reference bodies have not disjoint convex hulls, can be refined in two relations that are called *inside* and *outside*. The

relations *between* and *nonbetween* are refinements of the relation *collinear*, in the case the two reference bodies have disjoint convex hulls. The relation *nonbetween* can be refined in the two relations *before* and *after*, respectively.

Table 3. The definitions of projective relations among regions

relation	Definition	Acceptance subspace
$coll(A,B,C)$	$\forall x \in A^\circ \ [\exists y \in CH(B)^\circ$ $[\exists z \in CH(C)^\circ \ [coll(x,y,z)]]]$	$Coll(B,C) = (CS_{-\infty})^\circ \cup$ $(CS_{+\infty})^\circ \cup (CH(B \cup C))^\circ$
$as(A,B,C)$	$\forall x \in A^\circ \ [\forall y \in CH(B)^\circ$ $[\forall z \in CH(C)^\circ \ [as(x,y,z)]]]$	$Aside(B,C) = (\boldsymbol{R}^3 - Coll(B,C))^\circ$
$in(A,B,C)$	$CH(B) \cap CH(C) \neq \varnothing \ \wedge$ $A^\circ \subseteq (CH(B \cup C))^\circ$	$Inside(B,C) = (CH(B \cup C))^\circ$
$ou(A,B,C)$	$CH(B) \cap CH(C) \neq \varnothing \ \wedge$ $A^\circ \cap (CH(B \cup C)) = \varnothing$	$Outside(B,C) = \boldsymbol{R}^3 - CH(B \cup C)$
$bt(A,B,C)$	$CH(B) \cap CH(C) = \varnothing \ \wedge$ $A^\circ \subseteq (CH(B \cup C))^\circ$	$Between(B,C) = (CH(B \cup C))^\circ$
$nonbt(A,B,C)$	$Coll(A,B,C) \ \wedge$ $CH(B) \cap CH(C) = \varnothing \ \wedge$ $A^\circ \cap CH(B \cup C) = \varnothing$	$NonBetween(B,C) = (CS_{-\infty})^\circ \cup$ $(CS_{+\infty})^\circ$
$bf(A,B,C)$	$CH(B) \cap CH(C) = \varnothing \ \wedge$ $A^\circ \subset CS_{-\infty}$	$Before(B,C) = (CS_{-\infty})^\circ$
$af(A,B,C)$	$CH(B) \cap CH(C) = \varnothing \ \wedge$ $A^\circ \subset CS_{+\infty}$	$After(B,C) = (CS_{+\infty})^\circ$

Similarly to ternary projective relations between regions in $\boldsymbol{R}^2$ [4], we use the basic relations *aside*, *between*, *before*, *after*, *inside*, and *outside* to build a model for projective relations between three bodies of $\boldsymbol{R}^3$. The subspaces corresponding to the first four relations make a partition of the space $\boldsymbol{R}^3$ in the case the two reference bodies have disjoint convex hulls. If the two reference bodies have non disjoint convex hulls, the space is partitioned in two subspaces, corresponding to relations *inside* and *outside*.

Let us consider empty/non-empty intersections of a body A with the four subspaces:

$$(A \cap Before(B,C), A \cap Between(B,C), A \cap After(B,C), A \cap Aside(B,C))$$

A value 0 indicates an empty intersection, while a value 1 indicates a non-empty intersection. The 4-intersection can have $2^4 - 1$ different configurations. Each configuration corresponds to a projective relation among three bodies A, B, and C, where $CH(B) \cap CH(C) = \varnothing$. Four values equal to zero does not correspond to a relation. For $CH(B) \cap CH(C) \neq \varnothing$, we consider the following *2* intersections:

$$(A \cap Inside(B,C), A \cap Outside(B,C))$$

which can assume the values (0 1), (1 0), and (1 1). Overall, we obtain a model that is able to distinguish among a set of 18 ternary projective relations among three bodies of **R³**, which are JEPD.

We can use a linear notation for the relation, by listing 6 bits that represent the intersection of body *A* with *Before(B,C)*, *Between(B,C)*, *After(B,C)*, *Aside(B,C)*, *Inside(B,C)*, *Outside(B,C)*.

In this notation, the basic relations are expressed as follows: *bf(A,B,C)* = (1 0 0 0 | 0 0), *bt(A,B,C)* = (0 1 0 0 | 0 0), *af(A,B,C)* = (0 0 1 0 | 0 0), *as(A,B,C)* = (0 0 0 1 | 0 0), *in(A,B,C)* = (0 0 0 0 | 1 0), *ou(A,B,C)* = (0 0 0 0 | 0 1). Other relations correspond to cases with more than one non-empty value: for example, a relation that is a combination of two basic relations, such as a "*before and aside*", is indicated as: *bf:as(A,B,C)* = (1 0 0 1 | 0 0). Figure 6 contains three examples of ternary projective relations among bodies in **R³**.

a. *bf(A,B,C)*   b. *bt(A,B,C)*   c. *bf:as(A,B,C)*
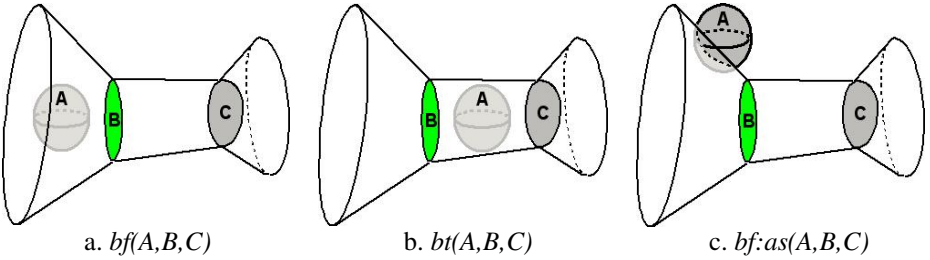
Fig. 6. Some examples of ternary projective relations among bodies in **R³**

6   Quaternary Projective Relations Between Bodies in a 3D Space

It is possible to extend the concept of quaternary relations between points in **R³** to bodies. The concept of *coplanarity* between four bodies can be introduced as a generalisation of the same relations among points. Following the same reasoning as when defining *collinearity* among bodies, we propose the following definition of coplanarity among bodies:

$$\forall w \in A^{\circ} \ [\exists x \in CH(B)^{\circ} \ [\exists y \in CH(C)^{\circ} \ [\exists z \in CH(D)^{\circ} \ [copl(w,x,y,z)]]]]$$

Let us examine the geometric realization of coplanarity among bodies. Similar to points where we had a degenerate case of coplanarity for collinear reference points, we have a degenerate case of coplanarity among bodies if reference bodies have at least a triplet of collinear points. If there exists at least one triplet of collinear points *x*, *y* and *z*, the relation coplanar with any point *w* of R³ is true; the coplanar subspace in this case is **R³** itself. Otherwise, we can identify a part of the space where a body *A* that is completely contained into it satisfies the relation *coplanar*. Let us call this part of the space the *coplanarity subspace* of *B, C* and *D*, *Copl(B,C,D)*. Such a subspace can be built by considering all the planes that are intersecting *B, C* and *D*. Knowing that a plane that intersects a family of convex sets is called a 2-transversal [15], the
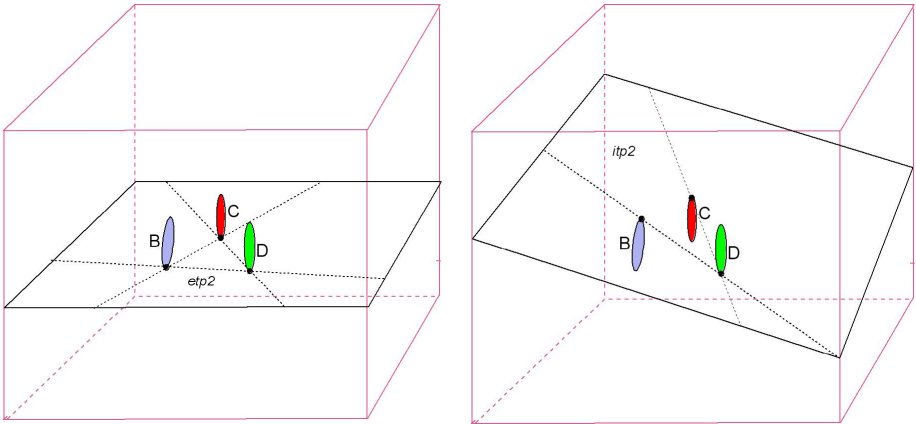
a. An external tangent plane (etp2) of *B*, *C* and *D*

a. An internal tangent plane (itp2) of *B*, *C* and *D*

Fig. 7. Internal and external tangent planes to three bodies in **R³**
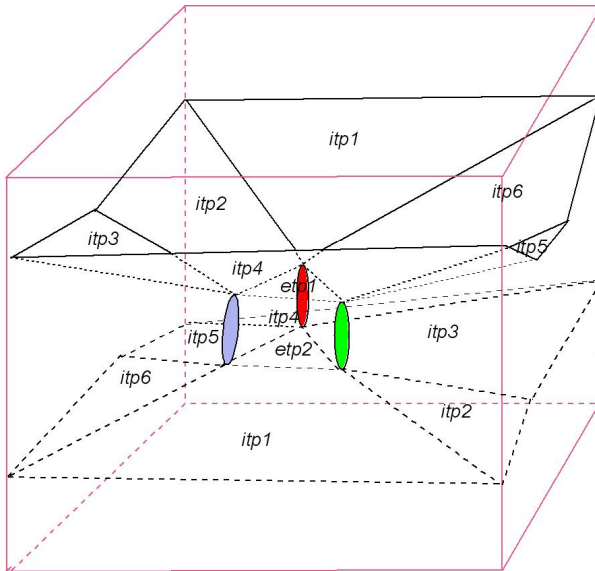
Fig. 8. Combination of internal and external tangent planes to three bodies in **R³**

coplanarity subspace can be equally defined as the set of all 2-transversals to (convex hulls of) bodies *B*, *C* and *D*.

Like the collinear subspace, the *coplanarity subspace* can be seen as the intersection of half-spaces defined by internal tangent planes combined with the intersection of half-spaces defined by external tangent planes and containing the three

reference objects. In this case, internal tangent planes separate two objects in one half-space and the third one in the other half-space and external tangent planes leave the three objects in the same half-space. By definition, there are 6 internal tangents planes and 2 external tangents planes to three bodies that do not contain triplets of collinear points [12]. Figure 7 shows an external tangent plane (etp2) and an internal tangent plane (itp2) of three bodies (*B*, *C* and *D*). The coplanar subspace partitions $\mathbf{R}^3$ in two distinct subspaces. Considering an order among the reference bodies (clockwise or counterclockwise), it is possible to consider a general orientation of the coplanarity subspace in $\mathbf{R}^3$. One ends up with three subspaces, the coplanar subspace, the non-coplanar subspace $NCS^+(B,C,D)$ and the non-coplanar subspace $NCS^-(B,C,D)$).

Figure 8 illustrates the combination of internal and external tangent planes. There are six internal tangent planes called *itp1*, …, *itp6* and two external tangent planes *etp1* and *etp2*. The six internal tangent planes form the top part of the coplanarity subspace (Figure 9.a) with *etp1* and the bottom part of the coplanarity subspace (Figure 9.a) with *etp2*. Figure 9.b represents the corresponding coplanarity subspace.

| a. Top and bottom of the coplanarity subsapce | b. Corresponding coplanarity subspace |
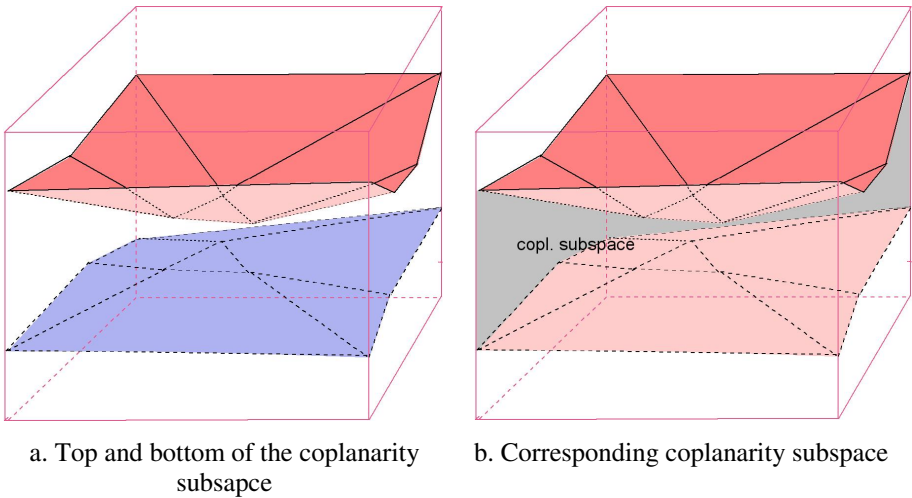
Fig. 9. Coplanarity subspace of three bodies in $\mathbf{R}^3$ (bodies are not represented)

Knowing that the convex hull of the union of three bodies is part of their coplanarity subspace, one can partition further the coplanarity subspace of three bodies *B*, *C* and D considering the convex hull of $B \cup C \cup D$ and its complement in the coplanarity subspace. These two subspaces are respectively called *Internal(B,C,D)* and *External(B,C,D)*.

The definition of the relation *coplanar* can be extended to complex bodies by considering the convex hulls of the reference objects in place of the reference objects. The definition of *coplanar* together with other quaternary projective relations for bodies is given in Table 4. Besides each definition, also the corresponding acceptance subspace is given. If there is no plane joining any points of *A*, *B*, *C* and *D*, body *A* is *non coplanar* with *B*, *C* and *D*. The relation *coplanar*, in the case the three reference

bodies are collinear, can be refined in two relations that are called *inside* and *outside*, depending if body *A* is included or not in the convex hull of $B \cup C \cup D$. Otherwise, the coplanar relation can be refined into *internal* and *external* relations depending if body *A* is included or not in the convex hull of $B \cup C \cup D$. The non coplanar relation can be also specialised into *above* and *below* relations depending in which non-coplanar subspaces body *A* is included.

Table 4. The definitions of projective relations among regions

relation	Definition	Acceptance subspace
copl(A,B,C,D)	$\forall w \in A°$ $[\exists x \in CH(B)° [\exists y \in CH(C)° [$ $\exists z \in CH(D)° [copl(w,x,y,z)]]]]$	$Copl(B,C,D) =$ $(CH(B \cup C \cup D))° \cup$ $External(B,C,D)$
non_copl(A,B,C,D)	$\forall w \in A°$ $[\forall x \in CH(B)° [\forall y \in CH(C)° [$ $\forall z \in CH(D)° [non\_copl(w,x,y,z)]]]]$	$Non\_copl(B,C,D) = \boldsymbol{R^3} -$ $Copl(B,C,D)$
in(A,B,C,D)	*coll(B,C,D)* $\wedge$ $A° \subseteq (CH(B \cup C \cup D))°$	$Inside(B,C,D) =$ $(CH(B \cup C \cup D))°$
ou(A,B,C,D)	*coll(B,C,D)* $\wedge$ $A° \cap (CH(B \cup C \cup D)) = \varnothing$	$Outside(B,C,D) = \boldsymbol{R^3} -$ $(CH(B \cup C \cup D))$
int(A,B,C,D)	*as(B,C,D)* $\wedge$ $A° \subseteq (CH(B \cup C \cup D))°$	$Internal(B,C,D) =$ $(CH(B \cup C \cup D))°$
ext(A,B,C,D)	*as(B,C,D)* $\wedge$ *copl(A,B,C,D)* $\wedge A° \cap (CH(B \cup C \cup D)) = \varnothing$	$External(B,C,D)$
ab(A,B,C,D)	*as(B,C,D)* $\wedge$ $A° \subset NCS^+$	$Above(B,C,D) = NCS^+$
be(A,B,C,D)	*as(B,C,D)* $\wedge$ $A° \subset NCS^-$	$Below(B,C,D) = NCS^-$

Like for the ternary projective relations, the all set of quaternary relations between four bodies can be obtained based on empty/non-empty intersections of the primary body *A* with the subspaces which satisfy the basic quaternary relations.

Let us consider empty/non-empty intersections of a body *A* with the four subspaces:

$(A \cap Internal(B,C,D), A \cap External(B,C,D), A \cap Above(B,C,D), A \cap Below(B,C,D))$

The 4-intersection can have 2^4-1 different configurations. Each configuration corresponds to a quaternary projective relation among four bodies *A*, *B*, *C* and *D*, where $CH(B) \cap CH(C) \cap CH(D) = \varnothing$. Four values equal to zero does not correspond to a relation. For $CH(B) \cap CH(C) \cap CH(D) \neq \varnothing$, we consider the following *2* intersections:

$(A \cap Inside(B,C,D), A \cap Outside(B,C,D))$

which can assume the values (0 1), (1 0), and (1 1). Overall, we obtain a model that is able to distinguish among a set of 18 quaternary projective relations among three bodies of $\mathbf{R^3}$, which are JEPD.

We can use a linear notation for the relation, by listing 6 bits that represent the intersection of body *A* with *Internal*(*B,C,D*), *External*(*B,C,D*), *Above*(*B,C,D*), *Below*(*B,C,D*), *Inside*(*B,C*), *Outside*(*B,C*).

In this notation, the basic relations are expressed as follows: *int(A,B,C,D)* = (1 0 0 0 | 0 0), ext*(A,B,C,D)* = (0 1 0 0 | 0 0), *ab(A,B,C,D)* = (0 0 1 0 | 0 0), *be(A,B,C,D)* = (0 0 0 1 | 0 0), *in(A,B,C,D)* = (0 0 0 0 | 1 0), *ou(A,B,C,D)* = (0 0 0 0 | 0 1). Other relations correspond to cases with more than one non-empty value: for example, a relation that is a combination of two basic relations, such as a "*internal and below*", is indicated as: *int:be(A,B,C,D)* = (1 0 0 1 | 0 0). Figure 10 contains four examples of quaternary projective relations among bodies in $\mathbf{R^3}$.
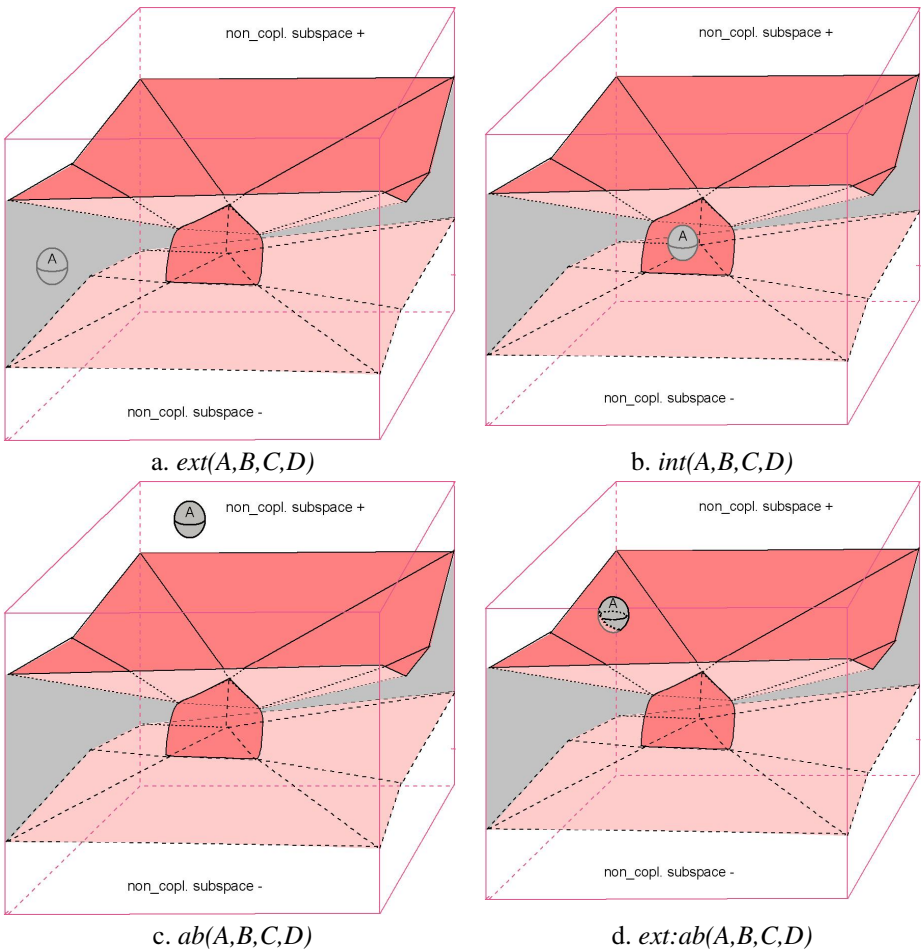
Fig. 10. Some Examples of quaternary projective relations among bodies in $\mathbf{R^3}$ (bodies are not represented)

7   Conclusions

In this paper, we have first presented ternary projective relations among points and among bodies in $\mathbf{R^3}$. The resulting model uses the concept of collinearity among bodies, which has been extended from previous work on points and regions in $\mathbf{R^2}$. The points' model provides a set of six JEPD ternary relations in $\mathbf{R^3}$ instead of seven in $\mathbf{R^2}$: *aside*, *between*, *before*, *after*, *inside* and *outside*. The bodies' model provides a set of eighteen JEPD ternary relations in $\mathbf{R^3}$ based on the same basic six relations.

As a second part, considering three reference objects, the concept of coplanarity has been introduced. In this case, the points' model provides a set of six JEPD quaternary relations in $\mathbf{R^3}$: *above, below, internal, external, inside* and *outside*, when the bodies' model provides a set of eighteen JEPD ternary relations in $\mathbf{R^3}$ based on the same basic six relations. Formal definitions and models to express all these relations have been proposed.

The study of the formal properties of relations and the development of a reasoning system is a future major issue. Development of algorithms to compute such 3D relations is needed and will complete the work already done for ternary relations among points and regions in $\mathbf{R^2}$. More generally, the study of properties of such relations will be important for optimizing the computation of more complex kinds of queries and for integrating ternary and quaternary projective relations as operators in a spatial database system. Finally, work needs to be done to "map" these concepts and models to specific 3D environments; e.g. to map the bottom of a coplanarity subspace on the topographic surface and use projective relations in the corresponding deformed space.

References

1. Bartie, P.J. and W.A. Mackaness, *Development of a Speech-Based Augmented Reality System to Support Exploration of Cityscape.* Transactions in GIS, 2006. **10**(1): p. 63-86.
2. Bennett, B., et al., *Region-based qualitative geometry*. 2000, University of Leeds, School of Computer Studies, LS2 9JT, UK. Technical Report 2000.07.
3. Billen, R. and E. Clementini. *Introducing a reasoning system based on ternary projective relations*. in *Developments in Spatial Data Handling, 11th International Symposium on Spatial Data Handling*. 2004. Leicester, UK: Springer-Verlag. p. 381-394.
4. Billen, R. and E. Clementini. *A model for ternary projective relations between regions*. in *EDBT2004 - 9th International Conference on Extending DataBase Technology*. 2004. Heraklion - Crete, Greece: Springer-Verlag. p. 310-328.
5. Billen, R. and E. Clementini, *Semantics of collinearity among regions*, in *OTM Workshops 2005 - 1st Int. Workshop on Semantic-based Geographical Information Systems (SeBGIS'05)*, R. Meersman, Editor. 2005, Springer-Verlag: Agia Napa, Cyprus. p. 1066-1076.
6. Clementini, E. and R. Billen, *Modeling and computing ternary projective relations between regions.* IEEE Transactions on Knowledge and Data Engineering, 2006. **18**(6): p. 799-814.
7. Coxeter, H.S.M., *Projective Geometry, 2nd ed.* 1987, New York: Springer-Verlag.

8. Freksa, C., *Using Orientation Information for Qualitative Spatial Reasoning*, in *Theories and Models of Spatio-Temporal Reasoning in Geographic Space*, A.U. Frank, I. Campari, and U. Formentini, Editors. 1992, Springer-Verlag: Berlin. p. 162-178.

9. Gapp, K.-P. *From Vision to Language: A Cognitive Approach to the Computation of Spatial Relations in 3D Space*. in *Proc. of the First European Conference on Cognitive Science in Industry*. 1994. Luxembourg. p. 339-357.

10. Hernández, D., *Qualitative Representation of Spatial Knowledge*. Lecture Notes in Artificial Intelligence. Vol. LNAI 804. 1994, Berlin: Springer-Verlag.

11. Klein, F., *Vergleichende Betrachtungen über neuere geometrische Forschungen.* Bulletin of the New York Mathematical Society, 1893. **2**: p. 215-249.

12. Lewis, T., B. von Hohenbalken, and V. Klee, *Common supports as fixed points.* Geometriae Dedicata, 1996. **60**(3): p. 277-281.

13. Ligozat, G.F. *Qualitative Triangulation for Spatial Reasoning*. in *European Conference on Spatial Information Theory, COSIT'93*. 1993. Elba Island, Italy: Springer Verlag. p. 54-68.

14. Scivos, A. and B. Nebel, *The Finest of its Class: The Natural Point-Based Ternary Calculus for Qualitative Spatial Reasoning*, in *Int. Conf. Spatial Cognition*. 2004, Springer. p. 283-303.

15. Wenger, R., *Progress in geometric transversal theory*, in *Advances in Discrete and Computational Geometry*, B. Chazelle, J.E. Goodman, and R. Pollack, Editors. 1998. Amer. Math. Soc.: Providence. p. 375-393.

16. Zlatanova, S., *3D GIS for Urban Development*. 2000, University of Graz - ITC.

A Multi-resolution Representation for Terrain Morphology

Emanuele Danovaro[1], Leila De Floriani[1,2], Laura Papaleo[1], and Maria Vitali[1]

[1] Department of Computer and Information Sciences, University of Genova, Italy
[2] Computer Science Department Center for Automation Research, Institute for Advanced Computer Studies, University of Maryland - College Park, Maryland
{danovaro, deflo, papaleo, vitali}@disi.unige.it

Abstract. Mesh-based terrain representations provide accurate descriptions of a terrain, but fail in capturing its morphological structure. The morphology of a terrain is defined by its critical points and by the critical lines joining them, which form a so-called surface network. Because of the large size of current terrain data sets, a multi-resolution representation of the terrain morphology is crucial. Here, we address the problem of representing the morphology of a terrain at different resolutions. The basis of the multi-resolution terrain model, that we call a *Multi-resolution Surface Network (MSN)*, is a generalization operator on a surface network, which produces a simplified representation incrementally. An MSN is combined with a multi-resolution mesh-based terrain model, which encompasses the terrain morphology at different resolutions. We show how variable-resolution representations can be extracted from an MSN, and we present also an implementation of an MSN in a compact encoding data structure.

1 Introduction

Modern acquisition devices, such as satellite or aerial photos, provide very large data sets for terrain models. These models are often excessively complex for applications such as visualization and real-time analysis. In this context, techniques for controlling the level of detail become crucial.

In general, a terrain model consists of a finite set of points in a domain in the xy plane with an elevation value f associated at each point. In case of data points regularly spaced in the domain, the terrain model is called a *Regular Square Grid* (RSG). Otherwise, the data points in the xy plane are connected in a *triangle mesh*, which provides a piecewise-linear interpolating function to the terrain data, called a *Triangular Irregular Network* (TIN) [13]. On the one hand, regular grids (RSGs) can be encoded in very compact data structures, since only the elevation values need to be stored. TINs, on the other hand, better adapt to the shape of the terrain, since their vertices are irregularly and adaptively sampled. In any case, a geometry-based description, such as an RSG or a TIN, provides an accurate representation of a terrain, but fails in capturing its morphological structure defined by critical points, like pits, peaks or passes, and integral lines,

like ridges or valleys. Beside being compact a morphological terrain description easily supports a knowledge-based approach to analyze, visualize and understand a terrain dataset, as required, for instance, in visual data mining applications.

In the last decades, there has been a lot of research focusing on extracting critical features (points, lines or regions) from images or terrain data described by an RSG or a TIN. More recent works in computational geometry concentrate on representing the morphology of terrains through a decomposition of the terrain surface into regions bounded by critical points (minima, maxima, saddle points) and integral lines [15]. These techniques are rooted in Morse theory and try to simulate the decomposition of a terrain induced by C^2-differentiable Morse functions in the discrete case.

A hierarchical representation of the terrain morphology is critical for interactive analysis and exploration of a terrain in order to maintain and analyze characteristic features at different levels of resolution. Current multi-resolution terrain models are just based on a geometric simplification process applied to a TIN describing a terrain at full resolution. In [8], a multi-resolution TIN which encompasses the morphology of the terrain has been presented. The proposed algorithm simplifies a constrained TIN through vertex removal, where the constraints are represented by the polygonal edges describing the integral lines connecting the critical points at different resolutions. However, the critical points have been maintained at the different levels of detail, so as to keep the morphology of the contour lines. In [4,5], a multi-resolution representation of a triangulated terrain has been proposed based on the notion of dependency introduced in [12].

In our work, we consider a combined multi-resolution terrain representation based on a multi-resolution constrained TIN, and on a hierarchical structural description of the terrain morphology. The multi-resolution constrained TIN is generated by simplifying the TIN following the generalization process which guides the simplification of the underlying morphology and, thus, the hierarchical morphological representation. Here, we focus on the multi-resolution morphological model. More precisely, we consider the surface network, which is a graph-based representation of the terrain morphology, and we discuss a generalization operator for simplifying such network. Based on such operator, we have developed a hierarchical representation of a surface network that we call a *Multiresolution Surface Network* (MSN). An MSN consists of a surface network representing the terrain morphology at a coarse resolution, and of a collection of refinement modifications, which reverse the generalization operators used in simplification, organized as a Directed Acyclic Graph (DAG). Variable-resolution surface networks can be extracted from an MSN through a simple DAG traversal.

The remainder of this paper is organized as follows. Section 2 reviews some background notions on morphological representations based on Morse theory. Section 3 reviews different approaches for computing discrete approximations of the terrain morphology by applying Morse theory. Section 4 formalizes a generalization operator for simplifying a surface network. Section 5 introduces the multi-resolution surface network (MSN). In Section 6, some concluding remarks are drawn.

2   Background Notions

In this Section, we introduce some basic notions on Morse theory and on the decomposition of the domain of a scalar field induced by a Morse function. For simplicity, we report the definitions only for the case of 2D scalar fields defined over a bounded region in the plane, recalling that they extend to arbitrary dimensional scalar field defined over a manifold in $\mathbb{R}^d$.

Morse theory is a powerful tool to capture the topological structure of a scalar field. It deals with the analysis of geometric shapes and the extraction of synthetic shape abstractions, preserving topological properties as well as main morphological characteristics.

Let f be a C^2-differentiable real-valued function defined over a domain $D \subseteq \mathbb{R}^2$. A point $p \in \mathbb{R}^2$ is called a *critical point* of f if and only if the Gradient of the function f vanishes at point p. In particular, the function f is said to be a *Morse function* if all its critical points are non-degenerate i.e., if and only if the Hessian matrix $Hess_p f$ of the second derivatives of f at p is non-singular (its determinant is $\neq 0$). The number of negative eigenvalues of $Hess_p f$ is called the *index* of a critical point p. In particular, in a neighborhood of each critical point p, the function f can be expressed in a canonical quadratic form in some neighborhood of a critical point. This implies that the critical points of a Morse function are isolated. Morse theory establishes a connection between the critical points and topological invariants of the shape, like the number of holes, concavities, etc. Finally, it provides a way of describing a manifold as a CW-decomposition [27,20].

In 2D, there are three types of non-degenerate critical points. A non-degenerate critical point p is a *minimum* (*pit*), a *saddle*, or a *maximum* (*peak*) if p has index 0, 1 or 2, respectively. For two-dimensional scalar fields, the formula $\#maxima - \#saddles + \#minima = 1$ is verified (see [16]). This formula is also known as the *Mountainer's equation*, for its interpretation for terrain models.

An *integral line* of a function f is a maximal path which is everywhere tangent to the gradient vector field. An integral line is emanating from a critical point or from the boundary of D, and it reaches another critical point or the boundary of D. An integral line which connects a maximum to a saddle or a minimum to a saddle is called a *separatrix line*. In Geographic Information Systems (GISs), separatrix lines that connect minima to saddles are usually called *ravine*, or *valley lines*, while those that connect saddles to maxima are called *ridge lines*.

The integral lines that converge to a maximum, a saddle and a minimum form a 2-dimensional, 1-dimensional and 0-dimensional region, respectively, and they are called *stable manifolds*. The integral lines that originate from a minimum, a saddle and a maximum form a 2-dimensional, 1-dimensional and 0-dimensional region, respectively, and they are called *unstable manifolds*. The stable (unstable) manifolds are pair-wise disjoint and decompose surface S into open cells which form a complex, since the boundary of every cell is the union of lower-dimensional cells. Such complexes are called *stable* and *unstable Morse complexes*, respectively. Figure 1(a) shows an example of a decomposition of the domain of a scalar field into an unstable Morse complex. In a 2D unstable

(stable) Morse complex, the 2-dimensional regions correspond to the maxima (minima), the 1-dimensional regions to the saddle points, and the 0-dimensional regions to the minima (maxima). A Morse function f is a *Morse-Smale function* when the stable and the unstable manifolds intersect only transversally. In two dimensions, this means that the stable and unstable 1-manifold cross when they intersect, and the crossing point is a saddle point.

A *Morse-Smale complex* is the complex defined by the intersection of the stable and unstable Morse complexes for a function f which is a *Morse-Smale function*. The 1-skeleton of a Morse-Smale complex consists of the critical points and the separatrix lines joining them, and it is called a *critical net* (see Figure 1 (b)). Figure 1(b) shows an example of a Morse-Smale complex for the same function as in Figure 1(a).

(a)   (b)
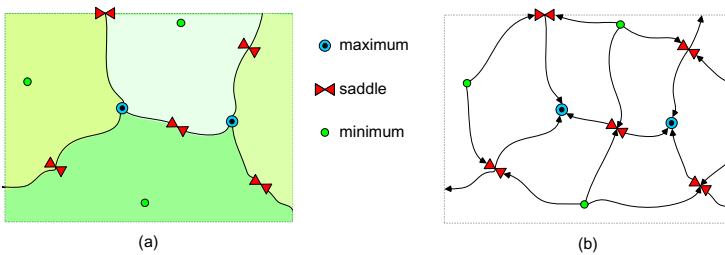
Fig. 1. (a) An example of an unstable Morse complexes (the 2-cells correspond to the minima). (b) The Morse-Smale complex. Its 1-skeleton is the critical net.

The *surface network* [22,26], widely used in Geographic Information Systems (GISs) for morphological terrain modeling, is a combinatorial representation of the critical net in the case of terrain models. The surface network is a planar graph in which the nodes correspond to the critical points, and the arcs to the integral lines connecting them. Thus, there exists an arc between a pair of nodes in the surface network if the two corresponding critical points are connected by an integral line in the critical net.

3   Computing Approximations of Morse and Morse-Smale Complexes

Several algorithms have been proposed in the literature for decomposing the domain of a scalar field f into an approximation of a Morse complex, or of a Morse-Smale complex. Such an approximation is obtained either by fitting a C^1- or C^2-differentiable surface on a terrain model, or by simulating a Morse-Smale complex, or a Morse complex in the discrete case by inferring properties from the C^2-differentiable case. Several recent algorithms working on TINs use this latter approach. Their assumption is that no two adjacent vertices in the TIN have the same elevation. This ensures that the critical points are isolated, as in the case of C^2-differentiable Morse functions.

All the algorithms proposed in the literature, with the exception of the one in [14], have been developed for 2D scalar fields. Almost all of them use what we called a *boundary-based* approach [7]. Basically, they extract an approximation of the critical net, by computing the critical points and then tracing the integral lines starting from saddle points and converging to minima and maxima.

Most of the existing algorithms working on TINs [3,2,15,21,29] detect first the critical points by comparing the elevation values at each vertex p with the elevations at the vertices adjacent to p on the TIN, and then compute the 1-cells of the complex by starting from the saddle points, and tracing two paths of steepest descent and two paths of steepest ascent on the underlying triangle mesh which stop at minima and maxima, respectively. The main difference among them lies in the way they select the vertices to be added in the four paths. The algorithms in [29,2] compute paths along the edges of the triangle mesh by selecting either the vertex of highest (or lowest) elevation at each step [2,29], while in [15] the steepest ascending or descending edge is selected. The algorithms in [3,21], instead, estimate the gradient along edges and triangles, and successively compute the ascending and descending paths by also cutting triangles in order to follow the actual paths of steepest ascent, or descent. Some algorithms [1,26,25] compute the arcs of the critical net from a 2D regular model (RSG), through a technique conceptually very similar to the one used for TINs. All three algorithms fit a surface with a certain degree of continuity to the input data set in order to extract the critical points. All these approaches try to enforce the Smale condition in the discrete case, by avoiding to connect two saddle points.

The algorithm in [1] uses a globally C^1-differentiable Bernstein-Bèzier bi-cubic interpolant, locally defined on each square cell. The integral lines are computed through a Runge-Kutta integration technique [23]. The algorithm proposed in [25] uses a bilinear C^0-differentiable interpolating function on each 2-cell of the grid. Minima and maxima can occur only at grid vertices, but additional saddles may be introduced by the interpolation inside the cell. Integral lines can follow grid edges, or go through 2-cells. When an integral line crosses a grid cell, it can be approximated with small (linear) steps, or computed exactly, by solving a linear system of differential equations. The algorithm in [26] fits a bi-quadratic polynomial by considering the 8-adjacent neighbors for each vertex p of the regular grid (four along grid edges and four diagonally). The algorithm produces a globally discontinuous approximation, formed by local surface patches, and computes the the first and second derivatives analytically (as in [25]). Such information is used to trace the integral lines starting from the saddles.

In [8,9], we have proposed an entirely different approach, that we call *region-based*. It consists of computing the stable and unstable Morse complexes considering as input a terrain model represented by a TIN. We only assume that the piecewise linear function defining the TIN is a discrete Morse function. Recall that the unstable (stable) manifold of a point p for a Morse function f is the set of points q such all that the ascending (descending) integral lines from q reach p. Our algorithms simulate this definition in the discrete case. First all minima

and maxima are identified, and then the ascending and descending complexes are computed independently by applying a region-growing approach. The first algorithm [9] computes the stable complex starting from a minimum m and initializing the 2-dimensional region R_m to all the triangles incident at m. At a generic step of the algorithm, the R_m is extended by adding a new triangle t sharing an edge e with R_m, provided that the vertex of t not bounding e has an elevation value highest than that of the extreme vertices of e. The computation of the stable complex ends when it is not possible to add other triangles to any of the computed regions.

In [8], the same region-growing approach is applied, but the gradient for each triangle t in the TIN is computed, and the angles between the normal vector at each edge of t and the gradient are evaluated. The edge e of t corresponding to the largest angle is marked as an *exit* edge of t, the one corresponding to the smallest angle is marked as *entrance* edge of t. Under these conditions, a new triangle t is added to a region R_m associated to a minimum m if it shares an edge e with a triangle t' already in R_m such that e is an entrance edge for t and an exit edge for t' (see Figure 2).
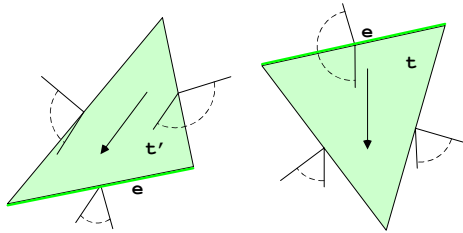
Fig. 2. Triangle t is adjacent to t' along the edge e. e is best exist for t' and best entrance for t. Under this condition t is added to the region of t'.

In both approaches [8,9], the unstable complex is computed in a completely symmetric way starting from the maxima. Finally, the intersection of the stable and unstable Morse complexes approximates the Morse complex, which is a Morse-Smale complex if the boundary of the computed regions in the two complexes intersect at a point. So, saddle points are extracted as the intersection of the two complexes. Although proposed for TINs, the approach can be extended to 3D scalar fields whose domain is discretized as a tetrahedral mesh.

An approximation of the Morse-Smale complex in the discrete case can also be computed by applying the discrete watershed transform. The watershed transform in the C^2-differentiable case provides a decomposition of a the domain of a C^2-differentiable function into regions of influence of the minima, called *catchment basins*. The boundary of the catchment basins form the *watershed lines*. If f is a Morse function, it can be seen that the catchment basins of the minima of f are the 2-dimensional regions in the ascending Morse complex of f and the watershed lines are 1-dimensional regions in such complex. Through a change in the sign of function f, the descending manifolds of the maxima can be extracted, and thus we can obtain the descending Morse complex for the original function.

Thus, watershed algorithms can be used to compute the Morse complexes and possibly the Morse-Smale complex (as an overlay of the two Morse complexes) if function f satisfies the Smale condition. Many sequential algorithms have been developed to compute watershed transforms, see e.g. [24] for a survey. Two major approaches proposed in the literature, are those based on the discretization of the topographic distance [19], and those based on simulating the immersion of a catchment basin in water [30] (and thus, the intuitive definition of watershed transform). Both types of watershed algorithms have been developed for images can be applied to regular models of scalar fields.

In [11] we have developed and compared implementations of the two watershed approaches applied also to TINs.

4   Generalization of Morse-Smale Complexes

Given a Morse-Smale decomposition of a scalar field, it is extremely interesting to maintain a multi-resolution representation of this decomposition in order to analyze the structure of the field starting from regions/lines/points of more importance and adding details incrementally.

Two major issues arise when computing a representation of a scalar field as a Morse, or a Morse-Smale complex. The first issue is the over-segmentation due to the presence of noise in the data sets. To this aim, *generalization algorithms* have been developed by several authors to locally simplify the structure of a Morse-Smale complex [31,15,4,29,28,17]. The second issue is related to the large size and complexity of available scientific data sets. Thus, a multi-resolution representation is crucial for an interactive exploration of such data sets. There exist just a few proposals in the literature for multi-resolution representations for 2D scalar fields [4,5,8,18].

The generalization of a Morse-Smale complex for a two-dimensional scalar field consists of collapsing a maximum-saddle pair into a maximum, or a minimum-saddle pair into a minimum, so as to maintain the consistency of the underlying complex. Usually, this operation is viewed as the *cancellation* of a pair of critical points, namely, a maximum and a saddle or a minimum and a saddle. A cancellation simulates the smoothing of the scalar field by modifying the gradient flows around two critical points.

We have formalized a cancellation in terms of the combinatorial representation of the critical net, defined by the surface network, as described below (see [10] for more details).

Let $S_N = (C, A)$ denote the surface network for a 2D scalar field. Let p and s be two critical points (i.e., two nodes in S_N such that arc $(p, s) \in A$, and p is a minimum (maximum) and s is a saddle. We call the *Influence set I^+* of (p, s) the collection of arcs $a_1, \ldots, a_k$ in A which are incident either in p or in s.

$$I^+ = \{e \equiv (t, v) \in A, |\{p, s\} \cap \{t, v\} \neq \emptyset\}$$

We call *relevant saddles* those saddles, different from s, which are connected to p through an arc belonging to I^+. Let R_s denote the set of relevant saddles with respect to the pair (p, s).

$$R_s = \{s_i \in C, s_i \neq s | \exists e \in I^+ \land e \equiv (s_i, p)\}$$

Moreover, because of the definition of surface network, if p is a minimum (maximum) there must exist exactly one other minimum (maximum) q, different from p connected to s through an arc in I^+. Let I^- denote the set of arcs connecting q with all the relevant saddles. Thus, $I^- = \{(q, s_i) | s_i \in R_s\}$ and, obviously, $I^- \cap A = \emptyset$. The generalization transformation (*cancellation*) on a surface network $S_N = (C, A)$ is thus defined as follows:

$$C = C \setminus \{p, s\}, A = (A \setminus I^+) \cup I^-$$

In other words, points p and s are removed from C and the arcs in I^- are replaced with I^+ in S_N. Figure 3 (a) shows an example of a surface network $S_N = (C, A)$. The arcs in I^+ are highlighted in dark blue. Figure 3 (b) illustrates the surface network $S'_N = (C', A')$ obtained from S by canceling the points s and p, where s is a saddle and p is a minimum. The arcs in I^- are highlighted in purple.

(a)   (b)
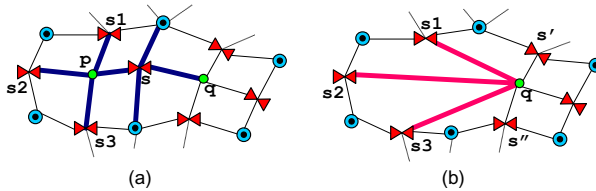
Fig. 3. (a) A surface network $S_N = (C, A)$. The arcs in I^+ are highlighted in dark blue. p is a minimum and s is a saddle. (b) The surface network $S'_N = (C', A')$ obtained from S_N by cancellation of saddle s and minimum p. The arcs in I^- are highlighted in purple. q is a minimum and $s1, s2, s3, s', s''$ are saddles.

Generalizations (or contractions) of surface networks have to be such that the resulting surface network should always be topologically consistent. The main difference among the methods proposed in the literature is in the way pairs of critical points to be canceled are selected. In [31], a minimum (maximum) p is chosen for cancellation together with its lowest (highest) adjacent saddle s. The order in which the minima and maxima are chosen for the cancellation is not specified. In [15], and in [3], a saddle s is selected together with its adjacent maximum at lower elevation, or its adjacent minimum at higher elevation. The order in which the pairs of points are canceled is determined based on the notion of *persistence* (see [3] for more details on persistence). In the generalization algorithm proposed in [28], a pair of adjacent critical points p and s is chosen in such a way that the difference in elevation between p and s is minimal among all (unsigned) differences in elevation between a saddle and an adjacent minimum, or a saddle and an adjacent maximum.

The problem of generalizing 3D Morse-Smale complexes has been recently investigated [17]. This method extends the technique discussed in [4,15] to functions defined on 3-manifolds. The extension is not trivial, since in 3D there are

three possible types of legal cancellations: minimum and 1-saddle, 1-saddle and 2-saddle, and 2-saddle and maximum. The cancellations are performed in order of persistence. Conditions on the elevations of the critical points involved in the generalization operation similar as in the 2D case have to be imposed. While the two cancellations involving a minimum or a maximum are similar to the ones performed in the 2D case, the saddle-saddle cancellation does not have an analog in lower dimensions. In order to ensure the separation of the minima and maxima originally separated by the two saddles, additional cells have to be introduced in the Morse-Smale complex, and thus this cancellation may not produce a reduction of the number of its cells. However, the authors claim that all these cells are removed by subsequent saddle-extremum cancellations [17].

5   Multi-resolution Surface Networks

Let us consider a sequence of legal generalizations applied to the surface network $S_N = (C, A)$ at the maximum resolution. This sequence produces a surface network at the coarsest resolution, the we call the *base network*. We invert the cancellation sequence, by considering the base network plus a sequence of refinements. A *refinement* is the inverse operation with respect to a cancellation. We observe that some refinements do not have to be necessarily applied in the same order as in the sequence. We define a *dependency relations* among refinements. Intuitively, two refinements are considered to be independent if they do not affect the same portion of surface network. If u and w are two independent refinements, then u can be applied before w, or w before u. Thus, a multi-resolution representation for a surface network encodes the surface network at the coarsest resolution, plus the a collection of refinements, reversing the cancellation sequence, and a dependency relation among them. The hierarchical Morse-Smale complexes introduced in [4,5] can be seen as an instance of such representation.

A cancellation applied to a surface network $S_N = (C, A)$ can be expressed as a pair (I^+, I^-), as explained in the previous section. We denote with $S'_N = (C', A')$ the surface network obtained from S_N by replacing I^+ with I^-. Then, the inverse refinement transformation, applied to S'_N consists of replacing the arcs in I^- with those in I^+, thus yielding network S_N as result. We call the pair $u = (I^-, I^+)$ a *refinement update*. A refinement update $u = (I^-, I^+)$ can be applied to a surface network $S_N = (C, A)$, if and only if $I^- \subset A$ and also $u' = (I^+, I^-)$ satisfies the requirements to be a feasible cancellation transformation (as defined in Section 4). Thus, we can define a *dependency relation* between pairs of refinement updates $u_1 = (I_1^-, I_1^+)$ and $u_2 = (I_2^-, I_2^+)$ as follows: u_2 *directly depends* on u_1 if and only if u_2 removes some of the arcs inserted by u_1, thus if and only if $I_1^+ \cap I_2^- \neq \emptyset$. We denote the *direct dependency relation* as $\prec$.

Figure 4 shows a surface network at the coarsest resolution (base network) and the refinement transformations with their dependency relations. The updates $1 \equiv u_1 = (I_1^-, I_1^+)$ and $2 \equiv u_2 = (I_2^-, I_2^+)$ involving pairs (s_1, p_1) and (s_2, p_2), respectively, are independent. Note that $I_1^- \cap I_2^+ = \emptyset$ and $I_2^- \cap I_1^+ = \emptyset$. The
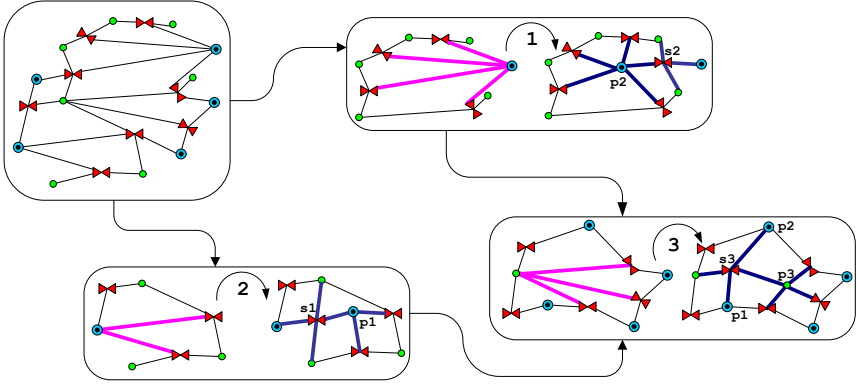
Fig. 4. The DAG of the refinement updates. On the top-left of the picture the base network is depicted. The refinement updates 1 and 2 are independent, while the update 3 depends on both 1 and 2.

refinement update $3 \equiv u_3 = (I_3^-, I_3^+)$ depends on both the updates 1 and 2, since $I_3^- \cap I_1^+ = p_1$ and $I_3^- \cap I_2^+ = p_2$.

The transitive closure of relation $\prec$ can be shown to be a partial order. Thus, we call the pair $M = (U, \prec)$ a *Multi-resolution Surface Network (MSN)*. In Figure 4 an example of an MSN is depicted. Note that the direct dependency relation is represented as a Directed Acyclic Graph (DAG). The refinements belonging to any subset U' on the set of refinements U, that is closed with respect to the partial order (i.e., such that, for every refinement $u' \in U'$ also the refinements preceding u' belong to U') can be applied to the base network S_B in any total order that extends the partial order, thus producing a surface network at an intermediate resolution.

It can be easily seen that any extracted surface network is a planar graph. The base network S_B is a planar graph, since each refinement removes a set of arcs I^- which are all internal to a cycle defined by the relevant saddles and the new set of arcs I^+ are internal to the same cycle.

The basic operation that we need to perform on a multi-resolution surface network consists of extracting a surface network from an MSN satisfying some application-dependent requirements based on the level of detail (LOD), such as the density of the critical points, the difference in elevation of pairs of removable points, etc. The LOD criterion can be uniform, or variable in space. Such operation is called *selective refinement*.

A selective refinement algorithm traverses the partially ordered set U of updates and constructs a closed subset U' of updates that, when applied to the base network S_B, gives the network which is the answer to the selective refinement query. Figure 5 shows examples of networks extracted from a triangulated terrain. Figure 5 (a) illustrates the coarsest surface network, Figure 5 (b) and Figure 5 (c) represent intermediate refined surface networks. Finally, Figure 5 (d) illustrates the surface network at full resolution.

A direct encoding of an MSN by storing, for each update $u = (I^-, I^+)$, the set of arcs in I^- and I^+ can be inefficient in term of space. Thus, we have developed a compact representation that we describe below. The direct dependency relation is encoded as a DAG in which the nodes correspond to refinements and the arcs describe the direct dependency relation. Updates are described not as collections of arcs, but procedurally. The encoded information must be sufficient to perform both cancellations and refinement on any currently extracted surface network. A refinement is required when extracting a network at some intermediate resolution by top-down traversal of the DAG, while we need to perform cancellations to coarsen locally any extracted network.
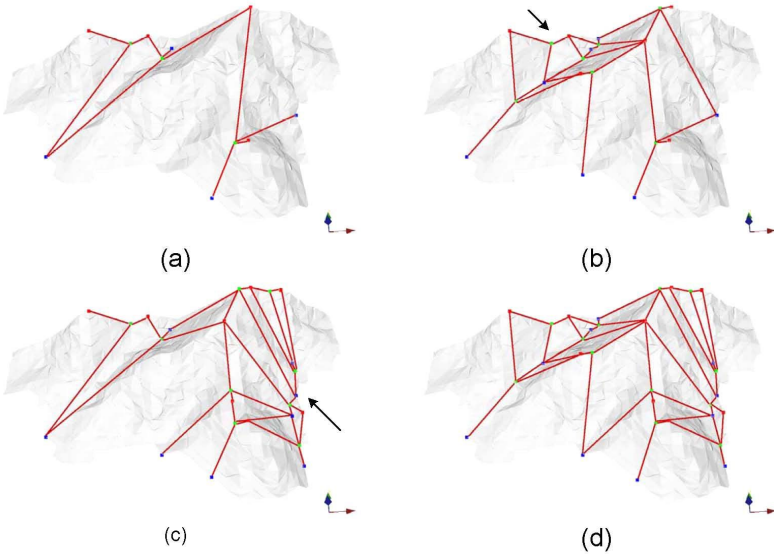
Fig. 5. (a) The initial triangulated terrain and the coarsest surface network. (b)-(c) Two intermediate surface networks. (d) The surface network at full resolution.

The cancellation transformation is entirely specified by the pair of critical points (s, p) removed, where s is a saddle point and p a minimum (maximum). To perform the inverse refinement transformation $u = (I^-, I^+)$ we need to specify the two critical points p and s inserted by the refinement (coordinates and field value), and an implicit description of I^-. This is obtained by specifying:

- the critical point q which is the extreme vertex of every arc in I^-.
- ordered pair of relevant saddle points (s', s'') satisfying the following condition:
  - there exist two arcs (s', q) and (s'', q) which are incident in q.
  - I^- is the set of arcs incident in q which are between arcs (s', q) and (s'', q) by considering the arcs incident in q in counterclockwise order around q (see Figure 3-(b)).

Note that I^+ is then completely defined since the end-points of the arcs in I^- which are not q define the relevant saddles (see Section 4). Thus, an update is composed by the indices of q, s' and s'', and coordinates and field value of p and s. We assume to store coordinate, field value, and index in 4 bytes: p and s require 12 bytes each, since we encode two coordinates and the field value, the three indices of q, s' and s'' requires 12 bytes. In this way, the total cost sums up to 36 bytes.

6 Concluding Remarks

In this work, we have considered the problem of representing the morphology of a terrain model at different resolutions. The terrain morphology can be described by the use of a Morse-Smale decomposition of its domain, and the 1-skeleton of this decomposition, namely the critical net, can be abstracted using a *surface network*. We have presented and compared different approaches proposed in the literature for the computation of a Morse or Morse-Smale decomposition starting from a terrain model. We have then formalized the *generalization operator* on a surface network and we have defined a hierarchical representation in the form of a multi-resolution surface network (MSN for short) to be combined with a multi-resolution geometry-based terrain model, which encompasses the morphology simplification at different resolutions. Finally, we have shown an efficient implementation of an MSN and how variable resolution representations can be extracted from it.

Further development of this work will involve developing efficient techniques for computing Morse complexes for 3D and 4D scalar fields, and investigate multi-resolution morphological representations of 3D scalar fields based on Morse decompositions. To this aim, we are currently extending the algorithm in [8] to the 3D case. It would be also interesting from an application point of view to study generalization of the structural terrain representations, such as channel and ridge networks [6].

Acknowledgments

References

1. C. L. Bajaj, V. Pascucci, and D. R. Shikore. Visualization of scalar topology for structural enhancement. In *Proceedings IEEE Visualization'98*, pages 51–58. IEEE Computer Society, 1998.
2. C. L. Bajaj and D. R. Shikore. Topology preserving data simplification with error bounds. *Computers and Graphics*, 22(1):3–12, 1998.

3. P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci. A multi-resolution data structure for two-dimensional Morse functions. In G. Turk, J. van Wijk, and R. Moorhead, editors, *Proceedings IEEE Visualization 2003*, pages 139–146. IEEE Computer Society, October 2003.

4. P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci. A topological hierarchy for functions on triangulated surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 10(4):385–396, July/August 2004.

5. P.-T. Bremer, V. Pascucci, and B. Hamann. Maximizing adaptivity in hierarchical topological models. In *International Conference on Shape Modeling and Applications*, pages 300–309. IEEE Computer Society, 2005.

6. Werner C. Formal analysis of ridge and channel patterns in maturely eroded terrain. *Annals of the Association of American Geographers*, 78(2):253–270, 1988.

7. L. Comic, L. De Floriani, and L. Papaleo. Morse-Smale decomposition for modeling terrain knowledge. In A.G. Cohn and D.M. Mark, editors, *Spatial Information Theory: International Conference*, volume 3693 of *Lecture Notes in Computer Science*, pages 426–444, 2005.

8. E. Danovaro, L. De Floriani, P. Magillo, M. M. Mesmoudi, and E. Puppo. Morphology-driven simplification and multi-resolution modeling of terrains. In E. Hoel and P. Rigaux, editors, *Proceedings ACM-GIS 2003 - The 11th International Symposium on Advances in Geographic Information Systems*, pages 63–70. ACM Press, November 2003.

9. E. Danovaro, L. De Floriani, and M. M. Mesmoudi. Topological analysis and characterization of discrete scalar fields. In T. Asano, R. Klette, and C. Ronse, editors, *Theoretical Foundations of Computer Vision, Geometry, Morphology, and Computational Imaging*, volume 2616 of *Lecture Notes on Computer Science*, pages 386–402. Springer Verlag, 2003.

10. E. Danovaro, L. De Floriani, L. Papaleo, and M. Vitali. Multi-resolution morse-smale decomposition of triangulated terrains. Technical Report DISI-TR-2005-21, Department of Computer and Information Sciences, University of Genova, May 2005.

11. E. Danovaro, L. De Floriani, L. Papaleo, and M. Vitali. Watershed algorithms for TINs: Implementations and Comparisons. Technical Report DISI-TR-06-03, Department of Computer and Information Sciences, University of Genova, February 2006.

12. L. De Floriani, E. Puppo, and P. Magillo. A formal approach to multi-resolution modeling. In W. Strasser, R. Klein, and R. Rau, editors, *Geometric Modeling: Theory and Practice*, pages 302–323. Springer-Verlag, 1997.

13. L. De Floriani, E. Puppo, and P. Magillo. Applications of computational geometry to Geographic Information Systems. In J. R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, chapter 7, pages 333–388. Elsevier Science, 1999.

14. H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Morse-Smale complexes for piecewise linear 3-manifolds. In *Proceedings 19th ACM Symposium on Computational Geometry*, pages 361–370, 2003.

15. H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse complexes for piecewise linear 2-manifolds. In *Proceedings 17th ACM Symposium on Computational Geometry*, pages 70–79. ACM Press, 2001.

16. H. B. Griffiths. *Surfaces*. Cambridge University Press, 1976.

17. Attila Gyulassy, Vijay Natarajan, Valerio Pascucci, Peer-Timo Bremer, and Bernd Hamann. Topology-based simplification for feature extraction from 3d scalar fields. In *Proceedings of IEEE Conference on Visualization*, 2005.

18. Ware J.M and C.B. Jones. A multiresolution topographic surface database. *International Journal of Geographical Information Systems*, 6(6):479–496, 1992.
19. F. Meyer. Topographic distance and watershed lines. *Signal Processing*, 38:113–125, 1994.
20. J. Milnor. *Morse Theory*. Princeton University Press, 1963.
21. V. Pascucci. Topology diagrams of scalar fields in scientific visualization. In S. Rana, editor, *Topological Data Structures for Surfaces*, pages 121–129. John Wiley and Sons Ltd, 2004.
22. J. L. Pfaltz. Surface networks. *Geographical Analysis*, 8:77–93, 1976.
23. W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical recipes in c - second edition*. Cambridge University Press, 1992.
24. J. Roerdink and A. Meijster. The watershed transform: definitions, algorithms, and parallelization strategies. *Fundamenta Informaticae*, 41:187–228, 2000.
25. B. Schneider. Extraction of hierarchical surface networks from bilinear surface patches. *Geographical Analysis*, 37:244–263, 2005.
26. B. Schneider and J. Wood. Construction of metric surface networks from raster-based DEMs. In S. Rana, editor, *Topological Data Structures for Surfaces*, pages 53–70. John Wiley and Sons Ltd, 2004.
27. S. Smale. Morse inequalities for a dynamical system. *Bulletin of American Mathematical Society*, 66:43–49, 1960.
28. S. Takahashi. Algorithms for extracting surface topology from digital elevation models. In S. Rana, editor, *Topological Data Structures for Surfaces*, pages 31–51. John Wiley and Sons Ltd, 2004.
29. S. Takahashi, T. Ikeda, T. L. Kunii, and M. Ueda. Algorithms for extracting correct critical points and constructing topological graphs from discrete geographic elevation data. *Computer Graphics Forum*, 14(3):181–192, 1995.
30. L. Vincent and P. Soille. Watershed in digital spaces: an efficient algorithm based on immersion simulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, 1991.
31. G. W. Wolf. Topographic surfaces and surface networks. In S. Rana, editor, *Topological Data Structures for Surfaces*, pages 15–29. John Wiley and Sons Ltd, 2004.

A Spatiotemporal Model of Strategies and Counter Strategies for Location Privacy Protection

Matt Duckham[1], Lars Kulik[2], and Athol Birtley[2]

[1] Department of Geomatics
University of Melbourne, Victoria, 3010, Australia
mduckham@unimelb.edu.au
[2] Department of Computer Science and Software Engineering
University of Melbourne, Victoria, 3010, Australia
lars@csse.unimelb.edu.au
a.birtley@ugrad.unimelb.edu.au

Abstract. Safeguarding location privacy is becoming a critical issue in location-based services and location-aware computing generally. Two drawbacks of many previous models of location privacy are: 1) the models only consider a person's location privacy protection, but not the invasion of location privacy by external agents; and 2) the models are static and do not consider the spatiotemporal aspects of movement. We argue that, to be complete, any model of location privacy needs to enable the analysis and identification of techniques both to protect and to invade an individual's location privacy over time. One way to protect an individual's location privacy is to minimize the information revealed about a person's location, termed *obfuscation*. This paper presents an explicitly spatiotemporal model of location privacy that models a third party's limited knowledge of a mobile individual's location. We identify two core strategies that a third party can use to refine its knowledge, so potentially invading that mobile individual's location privacy. A global refinement strategy uses the entire history of knowledge about an agent's location in a single step. A local refinement strategy iteratively constructs refined knowledge over time. We present a formal model of global and local refinement operators, and show how this formal model can be translated into a computational model in a simulation environment.

1 Introduction

Location privacy can be defined as a special type of *information privacy* that concerns the claim of individuals to determine for themselves when, how, and to what extent location information about them is communicated to others [6], cf. [21]. The emergence of low-cost location-aware computing, which combines powerful mobile computing platforms, wireless communication capabilities, and ubiquitous integrated positioning systems, has led to location privacy being acknowledged as a key challenge in information science (e.g., [17]). A failure to safeguard location privacy has been linked to a range of undesirable effects, including unsolicited marketing and *location-based* "*spam*"; decreased *personal safety*, such as might result from stalking or assault; and *intrusive inferences*, where other personal data about an individual is inferred from that individual's location over time [11, 19, 14].

This paper presents a new model of location privacy based on a formal analysis of mobile individuals and the knowledge a third party may have about the location of those individuals over time. The approach extends previous work in two significant ways.

- The approach models both the strategies an individual may employ to protect his or her location privacy and the *counter strategies* that a third party might employ to subvert these strategies. Most previous work has focused solely on strategies for location-privacy protection.
- The approach adopts a *spatiotemporal* model of location privacy. This model can be used to account for an individual's movements over time and the refinements a third party may make to its knowledge using such spatiotemporal information.

Following a brief literature review in Section 2, we present the key concepts that underlie this approach to location privacy in Section 3. The formal model, founded on set-based and algebraic techniques, is presented in Section 4. Section 5 discusses the implementation of refinement operators and Section 6 summarizes our formalization of refinement operators.

2   Background

Conventional techniques for protecting (location) privacy can be categorized into three classes: *regulation* [9], *privacy policies* [20, 16], and *anonymity* [10, 8]. Each of these approaches plays an important role in providing a complete solution to location privacy, but also has its limitations. A survey of techniques for location privacy protection is given in [6].

Regulation (such as data protection legislation) and privacy policies are trust-based mechanisms for proscribing unacceptable uses of private location information. Ultimately, any trust-based mechanism is vulnerable to inadvertent or malicious disclosure of private information. Anonymity-based techniques dissociate information about an individual, such as location, from that individual's actual identity [18]. Anonymity also has limitations, especially in spatial application domains. A person's identity can often be inferred from his or her location, making anonymity vulnerable to data mining [7, 1]. Further, using anonymity presents a barrier to authentication and personalization, required for many location-based applications [15, 12].

In [4, 5], the authors presented obfuscation as a fourth type of location privacy protection technique. *Obfuscation* is the process of deliberately degrading the quality of information about a person's location, with the aim of protecting that person's location privacy. Using obfuscation, individuals may choose to reveal varying degrees of information about their location (for example, suburb, block, street, or precise coordinate-level information). In common with anonymity-based techniques, obfuscation aims to hide information. However, obfuscation is an explicitly spatial privacy protection technique that aims to hide information about location rather than identity. Some recent work has begun to extend privacy policy and anonymity approaches with spatial capabilities (e.g., "spatial cloaking" [10] and the "need-to-know principle" [20]). Obfuscation is complementary to existing privacy protection techniques, but can extend these techniques by

enabling greater use of authentication and personalization and by increasing the security of location information from data mining and unauthorized access.

Unlike previous work which has focused primarily on ways to effectively hide location information to protect location privacy, in this paper we are also interested in understanding the ways in which a third party can refine information about an agent's location. The third party could be a hostile agent attempting to invade an individual's privacy or a legitimate security agency attempting to locate an agent more precisely. Our core thesis is that *a complete treatment of location privacy requires an understanding of both the strategies for hiding location information and the counter strategies for revealing hidden location information.*

3   Approach

The approach taken in this paper builds on recent work in [4, 5], introduced in the previous section. We assume an agent is moving around a geographic environment, carrying or interacting with some location-aware technology (such as a location-aware navigation system or a simple cell phone). Information about the agent's location is being tracked by third party (3P). Examples of 3Ps might include: co-workers or employers, telecommunications companies, location-based service providers, legitimate security organizations, or jealous spouses. However, either because of technological limitations or because the agent wishes to protect its location privacy, we assume the 3P's knowledge (3PK) of the agent's location is imperfect[1].

The quality of the 3PK may be lowered by a lack of detail (imprecision). For example, location in a conventional cell phone is typically resolved to the level of a single cell, usually an area of between 200m and 2km in diameter. The system can generate highly accurate knowledge about the presence or absence of an agent in a particular cell, but the precise location of agent inside the cell is unknown. The quality of the 3PK may also be lowered by inaccuracy. For example, an agent that wishes to protect its location privacy may deliberately introduce inaccuracies into the location information it reveals to a 3P (one type of obfuscation discussed in [5]). The overall scenario is summarized graphically in Figure 1, where combinations of obfuscation and/or technical limitations give rise to imprecision and/or inaccuracy in the 3PK of an agent's location.

In such a situation, two important questions are:

1. How can a 3P improve its knowledge of the location of an individual (thereby potentially invading an agent's location privacy)?
2. What strategies and spatial behaviors can an agent adopt to counter a 3P's attempts to subvert the agent's location privacy?

In this paper we focus primarily on answering question 1. However, the answer to question 2 is also a important focus and intended future outcome of this research. Further, we argue that the answer to question 2 depends, in part, on a thorough understanding of the answer to question 1.

---

[1] Conversely, if the 3P possesses perfect knowledge of the individual's location, the individual's privacy becomes a matter for regulatory- and policy-based systems rather than obfuscation- or anonymity-based systems.
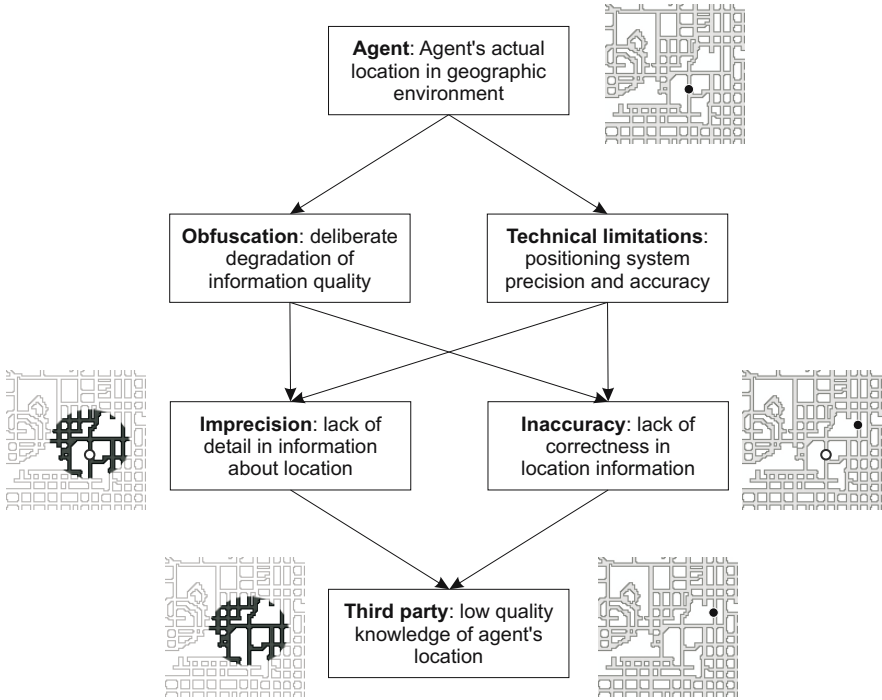
Fig. 1. Summary of location privacy scenario

3.1   Refinement Through Assumptions

There are a variety of ways in which a 3P might attempt to gain more information about an agent's location. A 3P may have access to additional information, for example generated by secret surveillance devices or through processing of an agent's device's mobile IP address (e.g., [3]). However, in this paper we assume the only sources of information to which a 3P has access are: 1) information about the geographic environment; and 2) imperfect information about an agent's location over time. Thus, the only way for the 3P to refine its knowledge of an agent's location is by making assumptions about the spatial and temporal characteristics of that agent's movements. Some common assumptions might include:

1. *Maximum/minimum/constant speed*: A 3P might assume certain values for the agent's maximum, minimum, or average speed of movement, or assume that the agent travels at constant speed.
2. *Direction*: A 3P might assume that the agent is heading in a particular direction (e.g., north, or as near as possible to it at each intersection).
3. *Goal-directed movement*: A 3P might assume that an agent is engaged in goal-directed movement, for example, taking the shortest path from some (unknown) source to some (unknown) destination.
4. *Connected locations*: A 3P might assume that an agent can only access locations that are spatially connected (for example, accessing locations which are connected

by road) or spatiotemporally connected (for example, accessing two locations connected by a railway line at a time when a train service between those locations is operating).

5. *Temporal granularity*: A 3P might assume that its knowledge about the agent's location is at a suitably fine temporal granularity such that the 3P has complete, if imperfect, knowledge of the agent's location (i.e., there are no gaps in the 3PK and the agent never manages to "sneak away" to locations outside the spatial extent of the 3PK).

While this list of assumptions is by no means complete, it gives a flavor of the diversity of potential spatial and temporal assumptions a 3P might make. Each of these assumptions may be used to refine information about an agent's location. For example, a 3P might possess imprecise information about the location of an agent at two times, t_1 and t_2. However, if the 3P assumes that an agent has a maximum speed of movement, it may be possible to discount some of the locations for time t_2 as being impossible to reach in the available time from any of the locations reported for t_1. In general, the more numerous and stronger assumptions a 3P makes, the more it can refine its information about an agent's location. At the same time, more numerous and stronger assumptions make it more likely that one or more assumptions will be invalid (e.g., an agent actually travels faster than the maximum assumed speed). Invalid assumptions will introduce inaccuracy into the 3PK.

In the following section we formalize our location privacy system. This system is able to model a variety of assumptions a 3P might make in order to refine its knowledge about an agent's location, including those presented above. The key observation is that the assumptions can be formalized into two classes, *local* and *global* assumptions, each of which have differing formal properties and computational characteristics.

4   Formal Model

Section 3 introduced a number of potential assumptions that a third party might make in order to refine its knowledge about an agent's location. These assumptions are formalized in this section as *refinement operators*. The refinement operators take a 3PK as input, and refines this knowledge by excluding those possible agent locations that are inconsistent with the assumptions. The formal analysis categorizes the assumptions introduced in Section 3 into two classes: *global* and *local refinement operators*. After introducing the model foundations, we introduce the general structure and properties of refinement operators (Section 4.2). The class of global refinement operators (Section 4.3) and the class of local refinement operators (Section 4.4) are then presented along with specific examples of how each of the assumptions in Section 3 can be formalized using a refinement operator.

4.1   Model Foundations

The foundations of the formal model are as follows:

Definition 1. *A* geographic environment *is modeled as a graph* $G = (V, E)$ *where* V *is a set of vertices and* $E \subseteq V \times V$ *is a set of edges connecting vertices. Each edge of a*

graph is associated with a non-negative weight using a weighting function $w : E \to \mathbb{R}^+$, representing the distances along edges within the graph.

Definition 2. *An* agent locator *is a function $l : T \to E$ where T is a finite set of discrete totally ordered time instants $T = \{t_1, ..., t_n\}$ and E is the set of edges in the graph G.*

Definition 3. *A* knowledge function *models a third party's knowledge of the location of an agent as the function $k_l : T \to 2^E$, where 2^E is the powerset of E.*

There are several points to note about the choice of these structures as model foundations. First, in keeping with previous work (e.g., [4,5]), the geographic environment is modeled as a graph. Graphs are a natural choice for this task in location-based services as they may be used to model constraints to movement (such as posed by a road network) and non-metric spaces (such as travel-time networks); may be embedded within two- or higher-dimensional Euclidean space; and are computationally efficient discrete structures [4, 13]. In later sections, the following simple definitions drawn from standard graph theory are required.

- $ShortestPaths(G)$ denotes the set of all shortest paths (sequences of adjacent vertices $(v_1, ..., v_n)$) in the graph G. Computing the set of all shortest paths for a graph is known as the *all-pairs shortest path* (APSP) problem. A number of common $O(n^3)$ algorithms exist for computing APSPs, including the Bellman-Ford algorithm, Floyd's algorithm, or simply iterating Dijkstra's algorithm [2].
- $IsReachable(V, G)$ denotes the set of vertices in the graph G that can be reached by some path in G from at least one vertex in V within a single "clock tick." In defining $IsReachable$, we therefore assume that the time domain T has a minimum granularity (akin to the concept of "chronons" in spatial databases). Similarly, $IsReachable(E, G)$ denotes the set of edges in the graph G that can be reached by some path in G from at least one edge in E within a single clock tick. Reachable vertices or edges can be found using standard shortest path algorithms (possibly requiring additional assumptions about the speed of travel).
- $Connected(V, G)$ denotes the set of vertices in the graph G that are connected by some path in G to at least one vertex in V. Similarly, $Connected(E, G)$ denotes the set of edges in G that are connected by some path in G to at least one edge in E. Again, standard algorithms exist for finding the set of connected locations in a graph, including computing the transitive closure of the graph.
- $Incident(e, p)$ indicates that an edge e is incident with path p (i.e., e has at least one vertex in common with p). Similarly, the symbol $Incident(E, p)$ indicates that the edge set E contains at least one edge which is incident with path p.

Second, the agent locator represents the location of an agent at a particular time instant as an *edge* of the graph (rather than the more obvious choice of a node). Edges are preferred as locations in our formal model because this approach:

- provides an inherently qualitative representation of location that does not assume the existence of precise coordinates (useful, for example, where location-sensing technology like GPS cannot provide completely precise and accurate location information, even if perhaps augmented with map matching techniques);

- can be extended to represent continuous movement along an edge, rather than assuming instantaneous movement between adjacent nodes (acknowledged as a drawback in previous work, [4]); and
- leads to an efficient and compact formal structure.

At the same time, by subdividing edges with additional vertices (termed *graph homomorphisms*), arbitrarily fine spatial detail can still be represented by the edges in a graph.

Finally, a knowledge function represents a third party's knowledge of the location of an agent as a set of edges for a particular time t. The set of all times T is discrete and finite. These times provide the smallest granularity of time that may be represented in the system (similar to the concept of a *chronon* or *tick* in spatiotemporal databases). The set of knowledge functions has a number of characteristic features, including:

- A knowledge function k_l is *accurate* if $l(t) \in k_l(t)$ for all $t \in T$; otherwise the knowledge function is *inaccurate*.
- A knowledge function k_l is *precise* if $|k_l(t)| = 1$ for all $t \in T$; otherwise the knowledge function is *imprecise*.
- A knowledge function k_l contains *ignorance* if there exists a t such that $k_l(t) = E$, where E is the set of all edges. Ignorance is a special case of imprecision.
- A knowledge function k_l is *inconsistent* if there exists a t such that $k_l(t) = \varnothing$; otherwise a knowledge function is *consistent*.
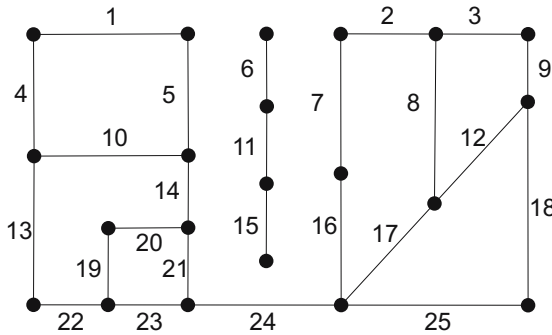
Fig. 2. Example geographic environment graph

For example, Figure 2 depicts a simple graph (geographic environment) $G = (V, E)$ with 25 edges (locations). Note that a geographic environment need not be connected. The movement of an agent for time domain $T = \{1, 2, 3\}$ from edge 2 at time 1 along edge 8 at time 2 to edge 17 at time 3 is represented by the function $l = \{(1, 2), (2, 8), (3, 17)\}$. A 3PK of the agent's movement might be represented by the function $k_l = \{(1, \{2, 3\}), (2, E), (3, \{17, 8, 16\})\}$. At any particular time the third party does not know exactly on which edge the agent is located: the agent has a degree of location privacy. In this case, the knowledge function k_l is imprecise, accurate, consistent, and contains ignorance.

4.2   Refinement Operators

In this section we provide some preliminary definitions leading up to the general definition of a refinement operator. First, we define the knowledge function space as the set of all knowledge functions.

Definition 4. *The* knowledge function space K *is the set of all knowledge functions* $(2^E)^T$, *i.e.,* $K = \{k | k : T \rightarrow 2^E\}$.

Next we define a relation $\leq$ on the set of knowledge functions K:

Definition 5. *The relation $\leq$ is a relation on the set of knowledge functions K such that* $k_1 \leq k_2$ *if and only if* $k_1(t) \subseteq k_2(t)$ *for all* $t \in T$.

It is easy to see that the relation $\leq$ forms a partial order, since $\leq$ is reflexive ($k \leq k$), antisymmetric ($k_1 \leq k_2 \wedge k_2 \leq k_1 \rightarrow k_1 = k_2$), and transitive ($k_1 \leq k_2 \wedge k_2 \leq k_3 \rightarrow k_1 \leq k_3$). Finally, we define a refinement operator as follows:

Definition 6. *A* refinement operator $\mathcal{R}$ *is a function* $\mathcal{R} : K \rightarrow K$ *such that* $\mathcal{R}(k) \leq k$ *and* $\mathcal{R}(k) \leq \mathcal{R}(\mathcal{R}(k))$. *A refinement operator is said to be* monotonic *if* $k_1 \leq k_2 \rightarrow \mathcal{R}(k_1) \leq \mathcal{R}(k_2)$.

The property that $\mathcal{R}(k) \leq k$ is a defining property of refinement: applying an assumption should always increase or maintain the precision of a 3PK, but never decrease its precision. The property $\mathcal{R}(k) \leq \mathcal{R}(\mathcal{R}(k))$ provides idempotency: refinement operators must not increase the precision of a 3PK through repeated application. Monotonicity is a useful property that is exhibited by all the refinement operators discussed in this paper. A monotonic operator is always idempotent. However, in general it is possible to conceive of non-monotonic refinement operators. Based on this general definition of refinement, in the following sections we provide definitions for two distinct classes of refinement operators: global and local refinement.

4.3   Global Refinement

A *global refinement operator* $\Gamma : K \rightarrow K$ is a refinement operator that uses the entirety of a third party's knowledge about an agent's location over time in order to compute a refinement of the agent's location over time. The key feature of a global refinement, when compared with local refinement, is that it utilizes the 3PK in a *single step*. There are potentially any number of different global refinement operators that might be defined based on this general specification. In the context of location privacy, we discuss two natural global refinements: connected refinement and goal-directed refinement.

Connected Refinement. A basic assumption introduced in Section 3 is that an agent can only access locations that are connected in the geographic environment. Under this assumption an agent can never be located at any edge that is disconnected from any other edge at which it has been located. This concept can be formalized as a global refinement operator $\Gamma_C : K \rightarrow K$ as follows:

$$\Gamma_C(k_l)(t) = \left(\left(\bigcap_{t' \in T} Connected(k_l(t'), G) \right) \cap k_l(t) \right)$$

The operator Γ_C is well formed in that it satisfies the basic properties of refinement operators, i.e., $\Gamma_C(k) \leq k$ and $\Gamma_C(k) \leq \Gamma_C(\Gamma_C(k))$. Further, for any sets of edges E_1, E_2 of a graph $G = (V, E)$ then $E_1 \leq E_2 \rightarrow Connected(E_1, G) \subseteq Connected(E_2, G)$. It follows that Γ_C is a *monotonic* refinement operator.

To illustrate, consider again Figure 2. Given the knowledge function $k_l = \{(1, \{1, 5, 6\}), (2, \{10, 5, 14\}), (3, \{20, 21, 15\})\}$, and assuming the agent can only travel between connected edges in the graph, then the agent cannot be located at edges 6 or 15. Formally: $\Gamma_C(k_l) = \{(1, \{1, 5\}), (2, \{10, 5, 14\}), (3, \{20, 21\})\}$.

Goal-Directed Refinement. Another assumption proposed in Section 3 is that an agent is moving in a goal-directed manner, e.g., it is taking the shortest path between two (unknown) locations. Under this assumption, the agent's true location can only be amongst those locations that are incident with at least one shortest path that travels through all previous known location sets. This assumption can be formalized as a global refinement operator $\Gamma_D : K \rightarrow K$ as below.

$$\Gamma_D(k_l)(t) = k_l(t) \cap \{e \in E | \exists p \in ShortestPaths(G).$$
$$\big(\forall t' \in T.Incident(e, p) \wedge Incident(k_l(t'), p)\big)\}$$

By definition Γ_D satisfies the defining properties of a refinement operator. The operator Γ_D is also monotonic as a consequence of the basic properties of the incidence relation.

To illustrate, consider $k_l = \{(1, \{3, 9\}), (2, \{12, 17\}), (3, \{7, 16\})\}$. Assuming Euclidean distances as edge weights for the graph in Figure 2 then $\Gamma_D(k_l) = \{(1, \{3, 9\}), (2, \{12, 17\}), (3, \{16\})\}$. There is no shortest path from edge 9 or 3, passing through edge 12 or 17 which also passes through edge 7.

4.4 Local Refinement

Operators belonging to the local refinement class are built up from the recursive application of atomic refinement operations. Unlike global refinement operators, local refinement operators construct a refined knowledge function piecewise, considering consecutive time instants. The most basic component of a local refinement operator is the *alpha combination* of two sets of edges at consecutive time instants. Based on the knowledge of an agent's location at t_{i-1} and t_i, the alpha combination provides the refinement of the agent's location at time t_i. Specific local refinement operators work by providing specific definitions for the alpha-combinations. The next level of the local refinement operator is the *beta combination* of a knowledge function between two time instants. Beta combinations recursively combine alpha combinations. Formally:

Definition 7. *An* alpha-combination *is a function* $\alpha : 2^E \times 2^E \rightarrow 2^E$.

In order to be well formed, the alpha combination must have the properties that $\alpha(E_1, E_2) \subseteq E_2$ (refinement property) and $E_1' \subseteq E_1 \rightarrow \alpha(E_1', E_2) \subseteq \alpha(E_1, E_2)$ (monotonic with respect to first argument). Alpha-combinations of (local) pairs of edge sets are recursively combined using *beta-combinations*, defined as follows:

Definition 8. *A* beta-combination *is a function* $\beta : K \times T \times T \to K$, *where*

$$\beta : (k, t_s, t_j) \mapsto \begin{cases} \text{if } j = s & \left\{ \left(t_s, k(t_s) \right) \right\} \\ \text{if } j = s+1 & \beta(k, t_s, t_s) \cup \left\{ \left(t_{s+1}, \alpha(k(t_s), k(t_{s+1})) \right) \right\} \\ \text{otherwise} & \beta(k, t_s, t_{j-1}) \cup \\ & \left\{ \left(t_j, \alpha(\beta(k, t_s, t_{j-1})(t_{j-1}), k(t_j)) \right) \right\} \end{cases}$$

Finally, the top level local refinement operator returns a beta-combination:

Definition 9. *For any* $T = \{t_0, ..., t_n\}$ *a* local refinement operator Λ *is a function*

$$\Lambda : K \to K, \ k_l \mapsto \beta(k_l, t_0, t_n)$$

Temporal Granularity Refinement. An assumption introduced in Section 3 is that the 3PK is at a temporal granularity fine enough to preclude the possibility that the agent has ever traveled outside the spatial extent of the 3PK. This assumption may be modeled as a local refinement operator Λ_G where for each pair of consecutive times t_i, t_{i+1} the agent can only be located at those edges that are connected in the subgraph formed by the union of $k_l(t_i)$ and $k_l(t_{i+1})$. To define this operator on a graph $G = (V, E)$ simply requires a redefinition of the alpha combination used for Λ_G as follows:

$$\alpha_G(E_1, E_2) = Connected(E_1, (V, E_1 \cup E_2))$$

Because $\alpha_G(E_1, E_2) \subseteq E_2$, Λ_G satisfies the defining properties of a refinement operator, and is monotonic in a similar way to the global connected refinement operator.

For the knowledge function $k_l = \{(1, \{1, 5\}), (2, \{13, 14\}), (3, \{16, 21\})\}$ (see Figure 2) the following the series of steps leads to the temporal granularity refinement $\Lambda_G(k_l) = \{(1, \{1, 5\}), (2, \{14\}), (3, \{21\})\}$:

$$\begin{aligned} \Lambda_G(k_l) &= \beta(k_l, 1, 3) \\ &= \beta(k_l, 1, 2) \cup \left\{ \left(3, \alpha_G(\beta(k_l, 1, 2)(2), k_l(3)) \right) \right\} \\ &= \{(1, \{1, 5\}), (2, \{14\})\} \cup \left\{ \left(3, \alpha_G(\{14\}, \{16, 21\}) \right) \right\} \\ &= \{(1, \{1, 5\}), (2, \{14\}), (3, \{21\})\} \end{aligned}$$

Maximum Speed Refinement. Given $T \subset \mathbb{R}$ and a weighting function w associated with the graph G which represents the distance along each edge, the travel time along each edge for an agent moving at a specified maximum speed can easily be calculated in order to decide whether two edges are reachable from one another. The assumption that an agent can only travel at a specified maximum speed (Section 3) forms the basis of the maximum speed refinement operator Λ_S. Again, Λ_S depends on the definition of the alpha combination for the operator, as follows:

$$\alpha_S(E_1, E_2) = IsReachable(E_1, G) \cap E_2$$

For similar reasons to the global goal-directed refinement operator, Λ_S satisfies the defining properties of a refinement operator and is monotonic. Definitions for further local refinement operators, including those derived from minimum speed and direction assumptions, are omitted here, but may be constructed in a similar way.

4.5   Combining Operators

One of the goals of providing definitions for refinement operators is to enable the combination of different refinement operators in a structured and predictable way. Consider the set of all refinements (global or local) of a knowledge function k. A pair of refinements $\mathcal{R}_1$ and $\mathcal{R}_2$ of k may be combined by taking the intersection of the two independent refinements. This refinement combination $\circ$ can be defined as follows:

Definition 10. *For two refinement operators $\mathcal{R}_1$ and $\mathcal{R}_2$ and a knowledge function k, let $\mathcal{R}_1(k)$ and $\mathcal{R}_2(k)$ be the corresponding refined knowledge functions. The combination operator $\circ$ of two knowledge functions $\mathcal{R}_1(k)$ and $\mathcal{R}_2(k)$ is then defined as*

$$(\mathcal{R}_1(k) \circ \mathcal{R}_2(k))(t) = (\mathcal{R}_1(k))(t) \cap (\mathcal{R}_2(k))(t).$$

As a consequence of the properties of set intersection, the operator $\circ$ is commutative and associative for all refinement functions of a knowledge function k. Further, consider the empty refinement $\mathcal{E}(k)$, where $\mathcal{E}$ is a local refinement operator with alpha combination $\alpha_\mathcal{E}(E_1, E_2) = E_2$. This refinement forms an *identity* for the combination $\circ$, i.e., $\mathcal{R}(k) \circ \mathcal{E}(k) = \mathcal{R}(k)$. Consequently, $\circ$ has the algebraic structure of a *monoid* (closure, commutativity, associativity, and identity) on the set of all refinements of a knowledge function k.

In addition, local refinement operators may be combined by taking the intersection of alpha combinations from each of the two refinement operators, resulting in the combination $\bullet$ as follows:

Definition 11. *For two local refinement operators Λ_1 and Λ_2 with alpha combinations α_1 and α_2, respectively, and a knowledge function k, $\Lambda_1(k) \bullet \Lambda_2(k)$ is the local refinement operator with alpha combination $\alpha_\bullet \equiv \alpha_1 \cap \alpha_2$.*

Again, we can show that $\bullet$ forms a monoid, with $\mathcal{E}(k)$ as the identity element.

Lemma 1. *For two local refinement operators Λ_1, Λ_2 and a knowledge function k, then:*

$$\Lambda_1(k) \bullet \Lambda_2(k) \le \Lambda_1(k) \circ \Lambda_2(k)$$

Proof. By induction. Let $T = \{t_0, \dots, t_n\}$, $i \in \{0, \dots, n\}$. Let α_1, β_1 and α_2, β_2 be the alpha and beta combinations of Λ_1 and Λ_2, respectively. Let $\beta_{\alpha_1 \cap \alpha_2}$ denote the beta combination of $\alpha_1 \cap \alpha_2$. The lemma is trivially true for $i = 0$, because all beta combinations depend in this case on k only. If $i = 1$, then

$$\beta_{\alpha_1 \cap \alpha_2}(k, t_0, t_1)(t_1) = \beta_1(k, t_0, t_1)(t_1) \cap \beta_2(k, t_0, t_1)(t_1).$$

If the lemma is true for $i - 1 > 1$, i.e.

$$\beta_{\alpha_1 \cap \alpha_2}(k, t_0, t_{i-1})(t_{i-1}) \subseteq \beta_j(k, t_0, t_{i-1})(t_{i-1}), \qquad j = 1, 2,$$

then we obtain

$$\alpha_1\big(\beta\alpha_1 \cap \alpha_2(k, t_0, t_{i-1})(t_{i-1}), k(t_i)\big)$$
$$\cap \, \alpha_2\big(\beta\alpha_1 \cap \alpha_2(k, t_0, t_{i-1})(t_{i-1}), k(t_i)\big)$$
$$\subseteq \alpha_j\big(\beta_j(k, t_0, t_{i-1})(t_{i-1}), k(t_i)\big) \qquad j = 1, 2,$$

because alpha combinations are monotonic with respect to their first argument. This implies

$$\beta_{\alpha_1 \cap \alpha_2}(k, t_0, t_i)(t_i) \subseteq \beta_j(k, t_0, t_i)(t_i), \qquad j = 1, 2,$$

and therefore

$$\beta_{\alpha_1 \cap \alpha_2}(k, t_0, t_i)(t_i) \subseteq \beta_1(k, t_0, t_i)(t_i) \cap \beta_2(k, t_0, t_i)(t_i),$$

which proves the lemma.

The importance of the algebraic properties of combinations of refinement operators is that for a given knowledge function we can say that the order in which refinement operators are applied will not affect the results of the refinement itself.

5   Computational Model

The formal model presented above has intentionally been kept simple. As such, several aspects of the formal model are not computationally viable. Here we discuss some of the steps taken to implement privacy simulation software based on the formal model. The aim of the simulation environment is to show how the formal model can be translated into a practical computational model, and to allow empirical exploration of different privacy protection and invasion strategies in future work. First we will present some general observations concerning the development of the simulation environment. Then we discuss the implementation of global operators, followed by the implementation of local operators. Finally, we examine some methods for extending the formal model to accommodate more complicated assumptions about the behavior of an agent.

5.1   The Simulation Environment

Our simulation environment is a simple model of a mobile phone network. Streets are represented by the edges of a graph, along which an agent moves. The location of an agent is not precisely known but is given by the set of all streets that partially lie within a cell of the phone network. The simulation environment enables the application of a variety of refinements to a 3PK in order to generate a more precise and accurate representation of the agent's current location.

The simulation is based on discrete events. At each time instant (tick) the simulation determines the set of edges that overlap with the cell in which the agent is currently located. This edge set forms the 3PK, called the knowledge base (KB) entry, for that time instant. The KB is implemented as a list of sets of edges. The list structure performs the role of the time values in the formal model—ordering the edge sets. Additional information, such as time deltas, could easily be added to the KB if required.

In our simulation, the KB is updated at each tick with new knowledge about the agent's location. The refinement operators are applied after each update. To increase computational efficiency, the KB is only updated when the set of potential agent locations has changed for many refinement operators. This corresponds to the point in time when the agent is detected as having transitioned between two cells. By minimizing KB entries

in this manner, we minimized both the number of calls to our refinement functions and the size of the KB when those refinement functions were invoked. Some refinement operators, for example the "maximum speed" operator cannot be optimized this way as they may require the KB to be updated even if the set of potential edges remains unchanged. The implementation of the "maximum speed" operator is discussed at the end of this section.

5.2 Global Operators

Two global operators were implemented for this simulation: the "goal directed" operator and the "connected path" operator. These operators are defined in Section 4.3. One optimization in our implementation is to precompute all pairs of shortest path in the graph. Used this precomputed result, Algorithm 1 returns all edges in the graph that are connected to at least one of the elements in an input set of edges.

Algorithm 1. Connected algorithm, $Connected(S)$

Data: S, a set of edges
Result: R, a set of edges connected to at least one edge in S
Local variables: a list of shortest paths L; shortest path s; nodes n, n_1, n_2; edges e, e'
$R \leftarrow \varnothing$;
for *each edge* $e \in S$, *where* $e = (n_1, n_2)$ **do**
 $\quad L \leftarrow GetAllShortestPathsFrom(n_1) \cup GetAllShortestPathsFrom(n_2)$;
 $\quad$ **for** *each* $s \in L$ **do**
 $\quad\quad n \leftarrow GetNodeAtEndOfPath(s)$;
 $\quad\quad$ **for** *each edge* e' *starting from* n **do**
 $\quad\quad\quad R \leftarrow \{e'\} \cup R$;

On the basis of Algorithm 1, we can define the algorithm for computing the connected path refinement operator, *ConnectedPath* (Algorithm 2). The first loop in Algorithm 2 builds the set S of all edges in the graph that are connected to at least one edge in the input KB. The second loop intersects S with each edge set in the input KB to generate the refined KB.

Algorithm 2. Connected path algorithm, $ConnectedPath(K_p)$

Data: K_p, the knowledge base (ordered list of sets of edges) to process
Result: K_o, the refined knowledge base
Local variables: sets of edges S, S'
$S \leftarrow \varnothing$;
$K_o \leftarrow K_p$;
for *each edge set* $S' \in K_o$ **do**
 $\quad S \leftarrow S \cup Connected(S')$;
for *each edge set* $S' \in K_o$ **do**
 $\quad S' \leftarrow S' \cap S$;

The directed path refinement operator is computed by Algorithm 3. The first loop finds all the shortest paths that share at least one edge with every set of edges in our input KB. The edges of each path that satisfies this criterion are added to an edge set S. In the second loop, the algorithm simply intersects each unrefined edge set with S to produce a refined edge set.

Algorithm 3. Direct path algorithm, $DirectedPath(K_p)$

Data: K_p, the knowledge base (ordered list of sets of edges) to process
Result: K_o, the refined knowledge base
Local variables: sets of edges S, T, U, V; shortest path s; Boolean b
$V \leftarrow \varnothing$;
$K_o \leftarrow K_p$;
for *each shortest path s in the graph* **do**
    $b \leftarrow true$;
    $T \leftarrow GetPathEdges(s)$;
    for *each edge set $U \in K_p$* **do**
        if $(|T \cap U| = 0)$ **then** $b \leftarrow false$;
    if b **then** $V \leftarrow V \cup T$;
for *each edge list $S \in K_o$* **do**
    $S \leftarrow S \cap V$;

5.3   Local Operators

A key feature of local operators is their recursive nature. Implementations of the local operators can take advantage of this recursion to improve operator performance, by only processing updates to the knowledge function each time the operator is called. Algorithm 4 implements the generic local operator. The algorithm requires one argument, a KB K_l that is persistently stored externally to the algorithm. Whenever new edge sets are added to the unrefined KB, we also add the same edge sets to our stored KB K_l. Thus, for the first call of the algorithm, K_l will simply be a copy of the unrefined KB. The global index l is initially set to 1, so the entire KB will be processed. The final line of the Algorithm 4 stores the cardinality of the KB K_l in the global index l. So, for subsequent calls of the algorithm, only new, unrefined edge sets are processed, increasing the efficiency of the local operator implementation.

In Section 4.5 we defined the combination of local operators using the • operator. A combination of operators can be implemented using a single refined KB, shared between

---

Algorithm 4. Local processing algorithm, $LocalRef(K_l)$

Data: K_l, externally stored knowledge base to process (possibly partially refined)
Result: K_l, the fully refined version of the input knowledge base
Static variables: global index l initialized as $l \leftarrow 1$
Local variables: local index i
for $(i \leftarrow l; i < |K_l|; i = i + 1)$ **do**
    $K_l[i] \leftarrow DoAlphaOperation(K_l[i - 1], K_l[i])$;
$l \leftarrow |K_l|$;

---

all local operators. The result of each operator's refinement is intersected with the shared KB at every time step. In this manner, processing a local operator still only requires an operation on a single pair of edge sets. However, if the active local operators change, the shared KB needs to be rebuilt from the start.

5.4 Extending the Base Model

The computational model outlined above can be extended to handle more complicated assumptions. As an example, we discuss the implementation of the maximum speed local refinement operator. Formally, this operator is defined by the set of edges that are reachable from any edge of the KB entry in a single tick, assuming some maximum speed. As mentioned earlier, new entries in the KB are only added when the agent transitions between cells. However, edges may become reachable over time even if the unrefined KB does not change. The maximum speed operator therefore needs to refine the KB (i.e., perform an alpha operation) at each tick.

Two specific problems arose in implementing this extension. First, it is unlikely that the time between each tick will be uniform. To achieve framerate-independent refinement, the simulation must take into account the actual time delta between ticks. Second, as an agent can only be located to the precision of an edge, its precise location on the edge is unknown. A naive implementation then allows an agent can reach all adjacent edges in zero time, with the unfortunate side-effect that the refined set of potential agent edges will expand recursively at each update frame, no matter how small the time delta or how slow the agent is (see Figure 3).

◣ Edge on which agent might be located, using naive maximum speed assumption
◠ Edge on which agent might be located, using unrefined knowledge base
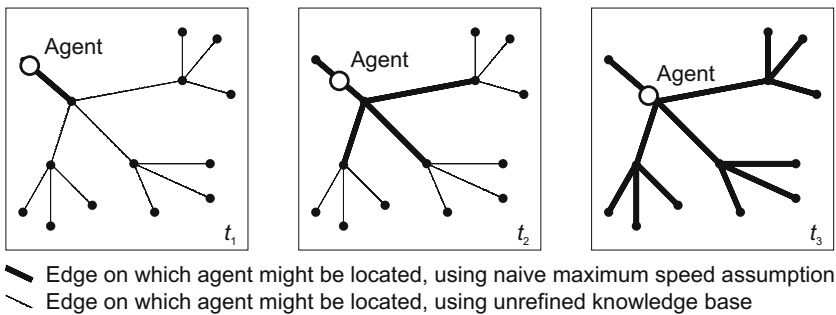
Fig. 3. A naive implementation of the maximum speed operator expands the potential agent location set at each tick, regardless of any assumed maximum speed

The first problem was solved by determining the set of reachable edges based on physical speeds (meters per second) rather than tick speed (meters per tick). This implied a modification of the alpha operator implementation to accept a time delta in addition to the two edge sets. The second problem was solved by simply discarding our refinements until the KB actually changed. This meant that we would always update with respect to the "original" refinement, albeit with a constantly increasing time delta. A simple example is provided in Figure 4.

Tick	Tot. time	Time Δ	Edge Set	Processing
1	0	11	E_1	—
2	11	10	E_2	$a(E_1, E_2, (11 - 0) + 10)$
3	21	9	E_2	$a(E_1, E_2, (11 - 0) + 10 + 9)$
4	30	9	E_3	$a(R(3), E_3, (30 - 11) + 9)$
5	39	11	E_4	$a(R(4), E_4, (39 - 30) + 11)$
6	40	10	E_4	$a(R(4), E_4, (39 - 30) + 11 + 10)$
7	50	12	E_4	$a(R(4), E_4, (39 - 30) + 11 + 10 + 12)$
8	62	10	E_5	$a(R(7), E_5, (62 - 39) + 10)$

Fig. 4. Example update calls for the maximum speed operator. Function a, a modified implementation of an α-combination for a maximum speed local refinement operators, accepts two edge sets and a time delta; R returns the refined knowledge base for a particular tick.

6 Discussion

This paper has set out a formal model of location privacy based on obfuscation. This model extends previous work in two ways:

1. The approach models both the strategies an individual can employ to protect his or her location privacy and the counter strategies that a 3P might employ to subvert these strategies. Counter strategies require that the 3P make one or more assumptions about the movement of the agent. These assumptions and their combinations are formally modeled as refinement operators. Combinations of refinement operators have predictable algebraic properties, ensuring that the order in which operators are combined does not affect the results of that combination.
2. The approach adopts an explicitly spatiotemporal model of location privacy. A 3P can use knowledge of an individual's movement over time to build refinements of that individual's location. Two distinct classes of refinement operator, global and local, are identified. These refinement operator classes differ in the way they handle spatiotemporal information. Global refinement operators work on the entire history of movement in a single step. Local refinement operators work recursively on knowledge of changing location, one clock tick at at time.

The paper has also indicated how some of the over-simplifications inherent in the formal model can be overcome when the formal model is translated into a computational model, implemented as a simulation environment.

Future work will focus on the empirical evaluation of location privacy protection and invasion strategies by experimenting using simulations of various 3P assumptions upon mobile agents. These experiments will investigate the trade-off between accuracy and precision of 3PK. Increased use of assumptions by a 3P will increase the precision of 3PK about the location of an agent. However, using incorrect assumptions may introduce inaccuracies into the 3PK. Thus, achieving a balance between the most precise but still accurate refined knowledge of an individual's location is the key to effective invasion of location privacy (and so understanding this balance is the key to effective counter-strategies to invasion of location privacy).

Further work will also investigate other strategies an individual might use to counter the refinement strategies of a 3P. For example, certain routes through a geographic environment might be more private than others (e.g., ones that are better connected might provide better protection from temporal granularity refinement operators). Understanding how refinement strategies operate is a prerequisite to understanding how an individual might protect against these refinements. One final open question that has not been addressed in this paper is the interaction between mobile individuals. Individuals do not move in isolation of other individuals, rather our movements are to some extent coordinated by spatiotemporal constraints, such as rush hour traffic or attendance at a soccer match or large concert. Where a 3P has (imperfect) knowledge of the location of many individuals, the 3P may be able to refine its knowledge based on inferences between individuals within groups. Modeling the aggregate movement of changing groups of individuals is an important future challenge.

Acknowledgments

References

1. A.R. Beresford and F. Stajano. Location privacy in pervasive computing. *IEEE Pervasive Computing*, 2(1):46–55, 2003.
2. T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. MIT Press, 2001.
3. R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation Onion router. In *Proc. 13th USENIX Security Symposium*, 2004.
4. M. Duckham and L. Kulik. A formal model of obfuscation and negotiation for location privacy. In H.W. Gellersen, R. Want, and A. Schmidt, editors, *Pervasive 2005*, volume 3468 of *Lecture Notes in Computer Science*, pages 152–170. Springer, Berlin, 2005.
5. M. Duckham and L. Kulik. Simulation of obfuscation and negotiation for location privacy. In D.M. Mark and A.G. Cohn, editors, *COSIT 2005*, volume 3693 of *Lecture Notes in Computer Science*, pages 31–48. Springer, Berlin, 2005.
6. M. Duckham and L. Kulik. Location privacy and location-aware computing. In J. Drummond, R. Billen, D. Forrest, and E. João, editors, *Dynamic and Mobile GIS: Investigating Change in Space and Time*, chapter 3. CRC Press, Boca Raton, FL, 2006.
7. S. Duri, M. Gruteser, X. Liu, P. Moskowitz, R. Perez, M. Singh, and J-M. Tang. Framework for security and privacy in automotive telematics. In *Proc. 2nd International Workshop on Mobile Commerce*, pages 25–32. ACM Press, 2002.
8. F. Espinoza, P. Persson, A. Sandin, H. Nyström, E. Cacciatore, and M. Bylund. GeoNotes: Social and navigational aspects of location-based information systems. In G.D. Abowd, B. Brumitt, and S. Shafer, editors, *Ubicomp 2001: Ubiquitous Computing*, volume 2201 of *Lecture Notes in Computer Science*, pages 2–17. Springer, 2001.
9. W. W. Görlach, A. Terpstra and A. Heinemann. Survey on location privacy in pervasive computing. In *Proc. First Workshop on Security and Privacy at the Conference on Pervasive Computing (SPPC)*, 2004.

10. M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. MobiSys '03*, pages 31–42, 2003.

11. M. Gruteser and D. Grunwald. A methodological assessment of location privacy risks in wireless hotspot networks. In D. Hutter, G. Müller, and W. Stephan, editors, *Security in Pervasive Computing*, volume 2802 of *Lecture Notes in Computer Science*, pages 10–24. Springer, 2004.

12. J. I. Hong and J. A. Landay. An architecture for privacy-sensitive ubiquitous computing. In *Proc. 2nd International Conference on Mobile Systems, Applications, and Services*, pages 177–189. ACM Press, 2004.

13. C. S. Jensen. Database aspects of location-based services. In J. Schiller and A. Voisard, editors, *Location-based services*, chapter 5, pages 27–39. Morgan Kaufmann, 2004.

14. E. Kaasinen. User needs for location-aware mobile services. *Personal and Ubiquitous Computing*, 7(1):70–79, 2003.

15. M. Langheinrich. Privacy by design—principles of privacy-aware ubiquitous systems. In G. D. Abowd, B. Brumitt, and S. Shafer, editors, *Ubicomp 2001: Ubiquitous Computing*, volume 2201 of *Lecture Notes in Computer Science*, pages 273–291. Springer, 2001.

16. M. Langheinrich. A privacy awareness system for ubiquitous computing environments. In G. Borriello and L. E. Holmquist, editors, *UbiComp 2002: Ubiquitous Computing*, volume 2498 of *Lecture Notes in Computer Science*, pages 237–245. Springer, 2002.

17. R.R. Muntz, T. Barclay, J. Dozier, C. Faloutsos, A.M. Maceachren, J.L. Martin, C.M. Pancake, and M Satyanarayanan. *IT Roadmap to a Geospatial Future*. The National Academies Press, Washington, DC, 2003.

18. A. Pfitzmann and M. Köhntopp. Anonymity, unobservability, and pseudonymity—a proposal for terminology. In H. Federrath, editor, *Designing Privacy Enhancing Technologies*, volume 2009 of *Lecture Notes in Computer Science*, pages 1–9. Springer, 2001.

19. B. N. Schilit, J.I. Hong, and M. Gruteser. Wireless location privacy protection. *IEEE Computer*, 36(12):135–137, 2003.

20. E. Snekkenes. Concepts for personal location privacy policies. In *Proc. 3rd ACM conference on Electronic Commerce*, pages 48–57. ACM Press, 2001.

21. A. F. Westin. *Privacy and freedom*. Atheneum, New York, 1967.

Incorporating Landmarks with Quality Measures in Routing Procedures

Birgit Elias and Monika Sester

ikg – Institute of Cartography and Geoinformatics, University of Hannover,
30167 Hannover, Germany

Abstract. In this paper we present an approach to providing landmark-based routes using a shortest-path algorithm. We start from the assumption, that at one junction there can be several landmarks to choose among, in order to find an optimal description of a route. The landmark selection used for describing the route is optimized taking the quality measures for the landmarks into account. Therefore, it is necessary to define quality measures. In the paper different types of quality measures are introduced and their integration in the route graph, as well as in a routing algorithm is presented. The usability of the approach is demonstrated using test data.

1 Introduction

The improvement of navigation systems and automatically generated routing description is an area of considerable research interest.

Most automated navigation systems rely on finding the shortest path in an underlying database network. But numerous cognitive studies have shown that human navigators use more than this single concept to select a path — criteria like least time, fewest turns and others play a vital role in human route planning.

Another finding of spatial cognition research is the importance of landmarks in the wayfinding process. Route descriptions are a sequence using two basic elements: instructions (actions at critical points along the route) and describing elements (characteristics and position of landmarks). Therefore, incorporating landmarks into automatically generated route instructions is essential to improving the usability of navigation systems.

It is well-known that humans try to minimize the complexity of route descriptions in order to reduce their cognitive load. For example, the capacity of short-term memory restricts it to a limited number of information pieces given in a route description. So, reducing the number of instructions is one possibility to decrease the description complexity. Furthermore, the linguistic complexity (e.g. number of words used) of each instruction or the cognitive complexity of the task described in the instruction has to be optimized.

In this paper we introduce an approach to optimizing the route selection considering the cognitive complexity of the route. For that purpose we incorporate point-like landmarks in the routing process. A graph using the geometric network and the landmark information is built up. At each junction in the road

network, a variable number of landmarks can occur (0. . . n). The landmarks are introduced with quality measures to generate abstract route-specific route directions that take the dependency between each landmark and chosen path segment as well as interdependence between different landmarks into account.

The paper is structured as follows: First, a review of the relevant literature dealing with landmarks in directions and aspects of route optimization is given in Sect. 2. In Sect. 3 the idea of incorporating landmarks with quality measures in a shortest-path calculation is introduced. In Sect. 4 the data model and some results using a test data set are presented. The paper closes with discussion of results and outlook to future work in Sect. 5.

2   Related Work

2.1   Landmarks in Directions

Directions are routing instructions, mostly given in verbal form. They are generated to help someone finding his way who is unfamiliar with the environment. This process consists of three different cognitive stages [1]: first, the mental representation of this area is activated. Secondly, the optimal path considering different criteria like shortest path, fewest turns etc. is selected. Thirdly, the abstract conceptualization is transformed to verbal (or graphical) output. For conveying the wayfinding information via speech it is necessary to break the route down in single, sequential segments. These consist of describing elements (position and characteristics of landmarks) and moving actions (especially at critical points along the route) [2,3].

There are two different approaches to provide local landmarks for route directions. In the first approach, a formal model of landmark saliency grounding on the characterization of Sorrows and Hirtle [4] is specified: the measures 'visual', 'semantic' and 'structural' attraction for building objects are established and assessed via hypothesis testing [5]. As an additional factor the advance visibility of objects while approaching a decision point is taken into account [6]. In a last step, the structural component is enhanced considering the *spatial chunking* of elements into 'higher order route description elements' (HORDE) [7,8].

The second approach deals with the extraction of building landmarks from existing spatial databases. Therefore, the task is split into two subtasks: in a first step for each potential decision point salient objects are identified as *potential landmarks*. These are determined using data mining methods to detect the salient objects in the local neighborhood of a junction [9,10]. Subsequently, the selection of potential landmarks has to be reduced according to the characteristics of a specific chosen route to the *route-specific landmarks*. For that step aspects of landmark visibility and perception, landmark quality, as well as cognitive and linguistic optimization of the communication process play an important role [11].

2.2   Aspects of Route Optimization

In automated navigation systems shortest-path calculations applying the Dijkstra algorithm [12] (or the A* algorithm) are used. Therefore, the road database

is stored as a graph network with nodes and edges and so the shortest-path problem is translated to a graph-theoretical problem. The distances between the nodes are used as 'travel costs' and are applied as weights to the edges of the network. It is possible to substitute the distance with travel time or other weights and create further route solutions (e.g. fastest route, *simplest paths* [13], *clearest route* [14]). The use of additional node weights is possible without modifications of the routing algorithm. Handling costs of turns requires the storage of edge-edge relations in the graph. For that purpose, the concept of a line graph is introduced [15]. This graph represents all pairs of consecutive edges and allows individual weighting.

Studies in the field of spatial cognition indicate that different criteria are used for path selection in human wayfinding: in addition to shortest distance and least time, also fewest turns, most scenic or straightest route criteria (minimizing angular deviation) and others are taken into account [16,17,18]. For this reason different route optimization approaches are introduced recently. They differ in optimizing a given criteria for the route selection or for its description.

Simplest Paths. Very often not the shortest but the simplest route is needed to give wayfinding instructions to someone who is unfamiliar with the environment. Cognitive studies indicate that people choose the straightest possible routes as opposed to more meandering routes [18]. That behavior leads to complexity reduction of the environment and ease the requirements of human short-term memory capacity: following the findings of Miller [19] people find it easy to remember (short-term) up to seven items (give or take two). So, reducing the number of turn elements in a route minimizes the needed memory capacity and therefore the complexity of the route.

Duckham and Kulik [13] propose an algorithm that can be used to select routes that minimize the complexity of instructions. This idea to 'ease the description' was first introduced from Mark [20] and assumes that the number and type of turns burden the route with a specific weight. While Mark [20] uses a weighting function to join metric distances with the instruction complexity, which are represented by a number counting the instruction elements needed, Duckham and Kulik [13] completely rely on the measure of instruction complexity. Nonetheless, the simplest paths are on average only 16% longer than the shortest paths.

Clearest Route. The Landmark Spider approach is similar to the simplest paths as far as it uses only specific weights and no geometric distance information in the shortest-path calculation [14]. But these weights represent the relevance of landmarks at each node with respect to the traveler's movement. Therefore, the combination of the salience of a landmark, its distance to the node, and the orientation of the landmark with respect to the traveler's heading build up the cost function. So, this approach uses a subset of all available landmarks, which are most prominent and easy to find, and determines the *clearest route* in terms of spatial reference.

Until now there has been no testing with real data to assess the performance of this approach. It is expected to be identical to the shortest path solution in the best-case scenario, but a low landmark density will lead to problems [14].

Context-Specific Route Directions. This approach provides a formal model that focus on the optimization of the route directions taking the surrounding context into account [21,22]. For a given route the minimal number of distinct parts in the abstract route directions on the highest granularity levels possible are calculated. Thereby one-one relations between decision point/action pairs and route instruction represent a low granularity. A high granularity stands for a many-one relation expressed by one route instruction covering more than one decision point of the route. Using this criterion supports the idea that the number of description elements in a route is connected to the complexity of the entire route directions.

These different granularity levels for the abstract route elements are produced by applying spatial chunking to them. This method groups several actions at decision points into one segment called 'higher order route directions elements' according to the three types 'numerical', 'landmark', and 'structural' chunking [8]. This approach aims at producing the cognitive simplest routes, because the conceptualization process of the route is eased. It complements the simplest paths approach providing an optimal description for their routes.

3   'Landmark Route'

Here, we want to introduce an approach that optimizes the route selection according to landmarks. The approach simplifies the conception of landmarks in only dealing with point-like landmarks like buildings. From the network and the landmark information a graph is constructed which supports a different routing option besides the shortest-path solution: the chosen route takes landmark quality and distance information into account and also considers landmark chunking rules.

3.1   Modelling Quality Measures for Landmarks in a Graph

To optimize the landmark selection, quality measures for the given landmark set are needed. Therefore, it is necessary to distinguish between different quality measures for landmarks representing the quality of the landmark object itself, the interdependencies between incoming route segment and landmark, and the quality of the landmark with respect to a turn. In the end, the final landmark quality is a summarization of all single quality aspects.

If each landmark is considered as a point-like feature, then it is possible to assign them to a junction in a path network and a 'landmark graph' can be built up. The landmarks represent the nodes of the graph. Their location corresponds to the geographic position of the junctions of the road network, which they are assigned to. Because more than one landmark at one junction is allowed, the connection between each neighbored landmark pair, which has a corresponding

connection in the road network, is represented by an edge. So, each node and edge from the road network is multiple modeled in the landmark graph depending on the number of potential landmarks given for each node (see Fig. 1).
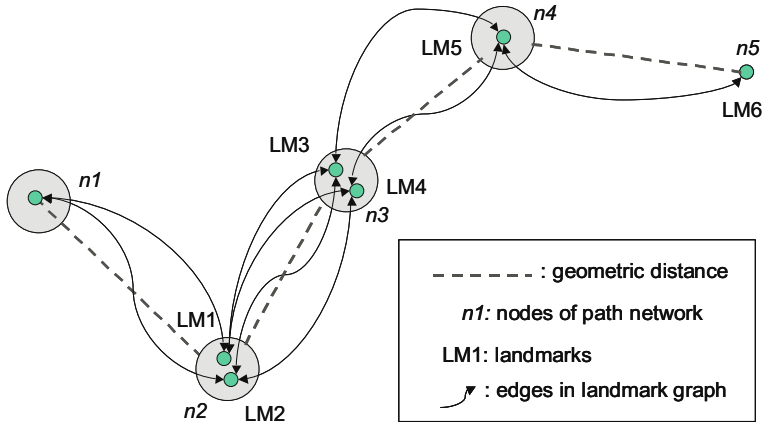
Fig. 1. Different landmark quality measures

Quality measures for landmarks can be assigned to different elements of the graph according to their impact on node-, edge-, edge-edge- or chunked edge-weights (see Fig. 2):

▷ If a quality criterion is only affecting the landmark object itself, the resulting 'landmark object quality weight' is assigned to the corresponding node in the graph (see Fig. 2, top left: weights for *n1*, *n2* and *n3*).
▷ If the landmark quality criterion is depending on the incoming route segment, it is assigned as 'landmark quality segment weight' to the directed edge. This stands for the dependency between the landmark quality and the direction the landmark is approached. (See Fig. 2, top right: the weight assigned to edge *e12* (representing the connection from node *n1* to *n2*) models the landmark quality of *n2* with respect to the edge *e12*).
▷ If the quality measure depends on the turn configuration, an edge-edge relation has to be modeled in the graph (for example using a line graph [15]). Onto these kind of edges the 'landmark turn quality weight' can be applied (see Fig. 2, down left: the weight applied to 'double edge' *de123* models the cost of the turn from edge *e12* onto *e23* visiting landmark *n2*). This represents the fact, that the configuration of incoming and outgoing edge is relevant for the weight – this is relevant to model turning restrictions or preferences. However, for using this kind of specific edges in most navigation systems the routing is not performed on the node-edge-graph, but on the dual construct: the edges are modeled as nodes, turns then represent edges; all edge (and node) weights have to be merged into one combined weight and assigned to this new dual edge.

▷ If the quality measure depends on a concatenation of chunked elements, the 'landmark chunk quality weight' is applied to the chunk element (see Fig. 2, down right: Spatial chunking operations result in a HORDE modeled as edge *chunk13* and representing the connection from *n1* to *n3* via *n2*. Therefore, a completely new edge in the graph is created. Its geometric representation is the sequence of the chunked segments, its graph representation is the direct connection between *n1* and *n3*. The weight of the edge depends on the sum of weights of all chunked edges belonging to the chunk multiplied with a factor taken the improvement achieved by the chunking into account.
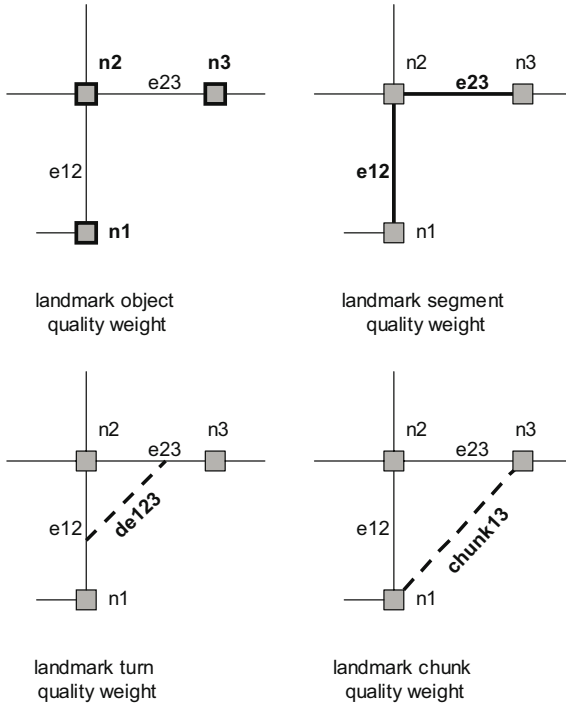
Fig. 2. Different Landmark Quality Measures

3.2   Establishing Weights for Landmark Quality

In Elias [10,11] a procedure to select potential landmarks is described. In that approach buildings were used as potential landmarks. The selection process is based on a visibility analysis and a Minimum Description Length Principle by selecting objects as potential landmarks that can be described in a compact way. The result is a list of equal suited potential landmarks for a junction. After the selection, the question arises, which aspects have to be modeled as 'quality measures' for these objects to determine the best fitting object as a route-specific landmark in a particular navigation context. In this paper, the modeling of quality aspects is tailored to the characteristics of potential landmarks.

To determine the characteristics of 'good landmarks' for car navigation systems Burnett et al. [23] conducted a direction-given study. The reasons for using a specific landmark are analyzed and result in a list of factors:

1. **Permanence:** The likelihood of the landmark being present. We assume that here all buildings have the same value. (Probably this aspect gains more attention when it is necessary to determine the weight between different landmark object types, e.g. like buildings and parks.)
2. **Visibility:** The importance of this aspect for the quality of a landmark is approved generally. But besides the existing 'visibility' of objects, also the question of 'visual perception' arises. Cognitive tests reveal the human characteristic to focus his spatial attention according to his expectation [24]. That means, if we expect to make a turn right at the next junction, our visual focus rest on the right side of the street. Landmarks situated on the 'wrong' side will not be noticed.
3. **Usefulness of Location:** Describes whether the landmark is located close to navigational decision points or not. In the extraction process of landmarks this factor is modeled so far that only objects, which are located 'near' a potential decision point (junction), are investigated. But if this junction is not a decision point in the chosen route (means the path is following straight on), the impact of this factor is getting minor.
4. **Uniqueness:** Represents the likelihood of the landmark not being mistaken for other objects. One factor is the highly individual appearance, which, in our case, is already checked in the procedure to select the potential landmarks. The second aspect covers preventing confusion between the landmark and similar looking objects. The landmark must be identified easily and with no mistakes in the environment. Therefore, it is necessary to check whether there are objects in the 'neighborhood' of the route, which are misleading. This local area of attention consists not only of the immediate surrounding of the decision point. Also the incoming route segment has to be inspected, because a misleading object ahead of the correct landmark can lead to a wrong navigation decision [11].
5. **Brevity:** Stands for the conciseness of description associated with a landmark. It is related to the number of terms/words used to refer to the object.

To model these quality aspects in a routing graph it is necessary to link each quality aspect to a type of weight (see Fig. 2). In Table 1 the classification of quality weights is given. Here, we assume that the potential landmarks are determined following the approach of Elias [11], so some of the quality aspects are considered already in that pre-processing step. They are separately marked in the table. Applying the quality weights onto the landmark selection approach of [5] is also possible, but some of the aspects introduced here are already taken into account in the approach itself (e.g. visibility). Therefore, a modification of the weight modeling in respect to the used landmark selection procedure would be needed. Chunk weights are not addressed in this table, because they represent a composition of different quality factors, which are specified at the end of this chapter.

Table 1. Classification of landmark quality weights

Quality Factors	Potential Landmarks	Object Weight	Segment Weight	Turn Weight
1. Permanence	●			
2. Visibility	(●)		●	
2. Perception				●
3. Usefulness of Location	●			
3. Decision Point				●
4. Individual Appearance	●			
4. Uniqueness			●	
5. Brevity		●		

Constituting the different types of weights needs the merging of several aspects in one weight measure. Because no other elements of the graph are affected, considering the brevity of the landmark description builds up the **object weight**. To establish the **segment weight** it is necessary to merge the aspect of visibility and uniqueness into one weight. Both are dependent on the route segment traveled to reach the landmark: the object has to be visible and no confusable objects similar to the landmark itself must be on the route segment leading toward the landmark. For the routing it is necessary that also the distance of a segment is taken into account. Therefore, this segment weight has to be combined with the distance weight and then constitutes a new weight measure for the edge.

To determine the **turn weight** the influence of perception and location has to be combined. Both need an edge-edge relation to be modeled, as the measures depend on the turn instruction and the relative position of the landmark at the junction. Because the turn costs vary dependent on the approaching direction and turn instruction, the combination of incoming and outgoing edge is needed to store this quality weight.

The kind of spatial chunking applied to the data is the so called 'landmark chunking' [8]. It chunks all straight following route segments until a turn instruction is required and located by a landmark ('turn right at the church'). A chunk is stored as a new segment in the graph and its weight is handled as a segment weight accordingly. To define the **chunk weight** it needs a careful consideration of all the influencing factors and their validity for the chunk element. The following options are possible:

▷ the geometric distance is the sum of all single segments chunked together
▷ the only valid object weight is the measure of the last landmark in the chunk chain (represented by the last node)
▷ only the visibility of the end landmark is necessary; the value can be taken from the last segment in the chunk
▷ the uniqueness measure has to be determined again, this time taking the complete path of the chunk into account
▷ a reduction factor has to be applied on the segment measure representing the simplification achieved by chunking elements (for example depending on the number of elements chunked)

4   Incorporating Quality Weights into Test Graph

To check the usability of using the weights within a shortest-path search algorithm, a few tests with test data are conducted using the Dijkstra Algorithm.

4.1   Graph Modeling Using Time Penalty as Weights

As visualized in Fig. 1 at each junction new nodes have to be introduced that represent the landmarks, and edges between all nodes have to be established. In our scenario, we will describe a setup where different node weights will be modeled, as well as weights for edges and chunking. No tests with the modeling of turn weights are conducted, as they require the dual modeling, which has not been done so far. Therefore, the tests in this paper are restricted to object, segment and chunk weights.

For developing a common framework to assess the graph, seconds are chosen as reference dimension units for geometric distances and quality weights. For that purpose the geometric distances (given in meters) are transformed into seconds assuming an average pedestrian speed of 1,5 m/sec. As far as that, the shortest-path in the network is equal to the shortest-time solution of the Dijkstra analysis. The test network used here is shown in Fig. 3.
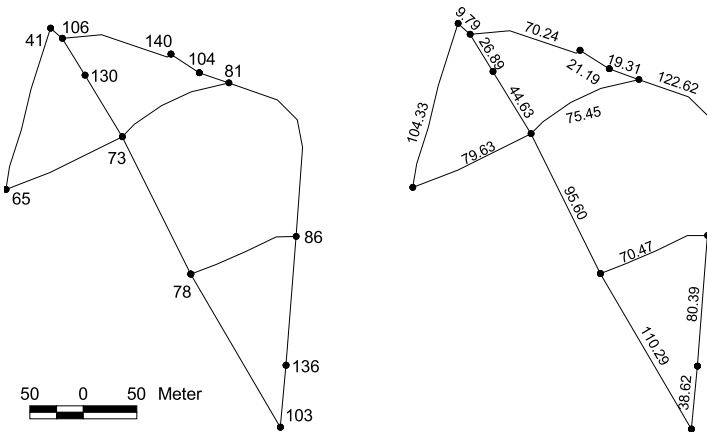
Fig. 3. Test data: nodes of the network (left) and distances (given in seconds) between the nodes (right)

Integrating landmarks in the process can be regarded as an improvement of the navigation task. If the task is regarded as a process with a specific duration, the navigation process of a route described by 'good landmarks' has to be faster than without any landmark information. Therefore, all segments in the landmark graph without any landmark information at their end node have to be punished with a general time delay. This penalty shall represent the additional time that

is needed by a pedestrian to orientate himself at the end of each route segment (e.g. causing stops at each decision point). The same holds for the quality of the landmarks: good landmarks get no penalty, whereas bad ones or junctions with no landmark at all are punished with an extra time delay. Thus, additional landmark information leads to a simplification of the navigation task and the time delay decreases. It neutralizes if the landmark is 'good' in all quality aspects, because the navigation process can be completed without any stops or time delay for re-orientation purposes.

This approach is represented in the data model for the landmark graph as follows:

Distances: The metric distances are taken from the underlying path network. They are converted into seconds modelling an average walking speed of 1,5 m/sec of a pedestrian. The global time delay is applied as a penalty to all distances without any landmark information in their end node and here is proposed with 90 sec.

Object Weights: The nodes of the graph represent the potential local landmarks located near the decision points of the route. Because more than one landmark at a decision point is allowed, the nodes and their connections are multiple modeled. The landmark object weight is applied to the nodes of the landmark graph and causes a maximum penalty of 30 sec (that means, if the landmark is not very good, at most 30 sec are added as penalty on the landmark node).

Segment Weights: The edges of the landmark graph are directed edges representing the distances between the landmarks as well as the landmark segment weights. These weights reflect the impact of the landmark depending on its quality according to the segment influencing aspects. If the landmark is clearly visible from the route at an early stage, its value is 0 sec. The maximum penalty for a bad landmark weight is here given with 30 sec for each edge. The distance measures and the segments weights are added up.

Chunking: We assume that a route description that entails identical landmarks in a sequence can be memorized better than a route where the landmarks change from junction to junction. Chunking of instructions can be integrated in two different ways: either by introducing new edges in the graph that serve as 'shortcuts', or by inclusion in the routing algorithm (see Sect. 4.3).

The first option is to introduce new edges in the graph by connecting route segments that can be described by subsequent identical landmark objects. This inclusion can be automated in a kind of region growing approach, known from digital image processing (see e.g. [25]): Starting from a seed node, routes are followed and gathered, that have identical landmark types. After at least two nodes with the same type have been found, a new edge is included that now constitutes a kind of shortcut in the route graph. For this edge, an appropriate weight has to be determined as described above. After all nodes have been visited via all possible edges, the algorithm terminates, as all new shortcut edges are found.

In this paper, the idea of incorporating landmarks and their quality into the routing process is introduced. The quality weights are modeled and applied using a time delay for the traveler. There are no studies or research findings so far examining the actual impact of these aspects with respect to the navigation process. So as a start, we have chosen reasonable penalty values for each weighting type and assume that all types have the same maximum effect.

4.2   Examples for Integrating Object and Segment Weights

In this section an example showing the use of object and segment weights in the landmark graph is introduced (see Figure 4 and Table 2 accordingly). The optimal route between junction 65 and 86 is requested. Here, we investigate the route costs of two different options: the route can lead either from 65 via junction 73 and 78 toward junction 86 (called D1) or from 65 via junction 73 and 81 to 86 (called D2). The shortest-path calculation of both routes using the original Dijkstra algorithm reveals, that the distance of route D1 is shorter than of D2 (see Table 2).
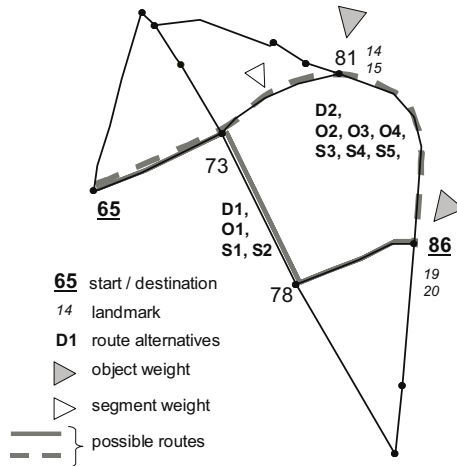
Fig. 4. Example using weights: route alternatives between junctions 65 and 86

Now, we introduce multiple landmarks at one node (at junction 81 landmarks 14 and 15, at junction 86 landmark 19 and 20) and apply landmark object weights to all nodes of the landmark graph. We assume that only 81_15 and 86_20 are 'good' landmarks in respect to object quality and therefore have no time penalty at all. All other nodes represent landmarks of 'bad' quality in respect to object quality and are punished with a time delay of 30 sec accordingly. Different route alternatives are calculated (called O1–O4) and presented in Table 2. The results show that only a combination of both weighted landmarks 81_15 and 86_20 accelerates the navigation task and leads to the shortest (or here better 'fastest')-path. All other combinations would still prefer the O1 (that is equal to the route D1).

Table 2. Result of Dijkstra Algorithm: Route between junctions 65 and 86 using object and segment weights

	nodes	shortest-path		using object weights				using object and segment w.				
start:	65	D1	D2	O1	O2	O3	O4	S1	S2	S3	S4	S5
73	dist	79,63	79,63	79,63	79,63	79,63	79,63	79,63	79,63	79,63	79,63	79,63
	node			30	30	30	30	30	30	30	30	30
	edge							30	30	30	30	30
81_14	dist		75,45		75,45					75,45		
	node				30					30		
	edge									30		
81_15	dist		(75,45)			75,45	75,45				75,45	75,45
	node					0	0				0	0
	edge										0	0
78	dist	95,60		95,60				95,60	95,60			
	node			30				30	30			
	edge							30	30			
86_19	dist	70,47	122,62	70,47	122,62	122,62		70,47		122,62	122,62	
	node			30	30	30		30		30	30	
	edge							30		30	30	
86_20	dist	(70,47)	(122,62)				122,62		70,47			122,62
	node						0		0			0
	edge								30			30
total [sec]:		**245,70**	277,70	335,7	367,7	337,7	**307,7**	425,7	395,70	457,70	397,70	**367,70**

In a next step segment weights are introduced additionally. They are applied comparable to the node weights: only the 'good' landmarks in respect to the segment aspects are weighted with 0 sec, all other landmarks get a penalty of 30 sec for their 'bad' quality in respect to their segment characteristics. In our case, only the edge from 73 leading to 81_15 receives a good segment weight. Different route alternatives (using different landmark combinations called S1–S5) are processed. The results show, that using the landmark node combination 81_15 and 86_20 leads to the fastest route. This route has to be preferred because it is the best landmark route available.

4.3   Integrating Chunking in Dijkstra Algorithm

Chunking can be included directly into the routing algorithm, in our case into the Dijkstra Algorithm. The idea is that route elements between adjacent nodes of a different type get a penalty as opposed to route elements that link two nodes of the same type. In the original Dijkstra Algorithm those nodes are expanded, that have currently been identified as being accessible on the shortest path from the start node. When the accessible successor nodes are added, not only the edge weights between them are used, but also a penalty value, if the adjacent nodes are of a different type. For the penalty, an appropriate value has to be chosen. In our case we used 60 sec, assuming that this additional time is needed to memorize different types.

In the following example (Fig. 5), the shortest route between junctions 73 and 140 is searched. At each of these junctions there are a varying number of

landmarks of different type. E.g. at node 130, there are landmarks 30, 31, and 32 with landmark types 2, 4 and 1, respectively. The routing between junctions 73 to 140 is possible along all landmark nodes in the graph. Also, there will be four target points in node 140, as that node consists of four landmarks.
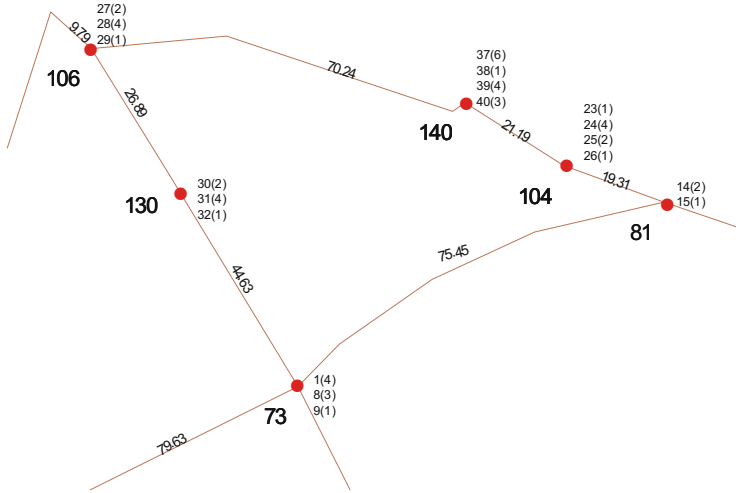
Fig. 5. Example for chunking in Dijkstra Algorithm: junctions with different landmark-nodes, including their type, in brackets.

Applying the modified Dijkstra Algorithm to this route yields the results given in Table 3.

Table 3. Result of Dijkstra Algorithm: route between junctions 73 and 140

route nr	start (type)	dist.	via node (type)	dist.	via (type)	dist.	end (type)	penalty (from-to)	total distance
1	9(1)	75,45	15(1)	19,31	26(1)	21,19	37(6)	60 (26-37)	175,95
2	9(1)	75,45	15(1)	19,31	26(1)	21,19	38(1)	0	115,95
3	9(1)	44,63	31(4)	26,89	28(4)	70,24	39(4)	0	141,76
4	9(1)	75,45	15(1)	19,31	26(1)	21,19	40(3)	60 (26-40)	175,95

As shown in the table, the routing from junction 73 to junction 140 yields four different possibilities. Three of them use the junctions 81 and 104 with different selections of their landmarks. One is taking the north route via junctions 130 and 106. Route 2 is the shortest with 115,95 sec. It starts with landmark 9 and uses landmarks 15, 26 and 38, that all have the same type (1). Routes 1 and 4 have different end nodes (37(6) and 40(3)) - thus they have to 'pay' for this type-change with 60 sec (see column penalty). Route 3 uses only landmarks of type 4 and thus no penalty is applied.

The best route among the four possibilities is route number 2, which uses only landmarks of type 1: LM1. The derived route description can be as follows: "Start from junction 73, turn left at LM1, go straight at the next LM1, until you reach the following LM1, which is your target."

In the example, chunking was used to join any subsequent nodes of same type. In reality this might not be useful. As stated in [8], chunking is typically used to aggregate descriptions of the same type along a straight line, until a turning instruction appears. Taking such aspects into account is also possible, however, then also the geometric properties of the graph have to be taken into account: only those adjacent nodes of same type that go straight will get no punishing value.

The proposed chunking operations are not verified if they are cognitively plausible. They are introduced to show the feasibility of incorporating the idea of chunking into the routing process by means of creating new edges and time penalty values. The actual chunking strategies has to be analyzed and fit into the proposed model.

5   Conclusion and Outlook

In the paper we presented an automatic approach that is able to generate an optimal route description using landmarks. From a set of potential landmarks assigned to single junctions, the optimal ones are selected that can be used to describe the best route. When transferring the problem to a graph structure, a crucial factor is the modeling of the weights needed for the determination of the optimal route. Here, we try to model all distances and weights by means of seconds, that represent the duration of the different aspects (like moving or re-orientation). We made some assumptions that also led to satisfying results. Here further work is needed to prove the correctness of the penalty concept for all eventualities and specify the time dimension for each weighting factor. Especially the penalty model for the chunking weight has to be adopt to the actual chunking strategy in wayfinding descriptions. Also further investigations and experiments are needed to assess the time delay of a pedestrian. Therefore, user tests have to be conducted to determine the impact of the different aspects in reality.

However, the presented method gives the technical framework for the calculation of shortest paths. It takes different factors into account, as it allows to modeling all the aspects in terms of weights in the graph. Moreover, an approach was shown including the introduction of possible chunks in the routing calculation process of the Dijkstra Algorithm.

References

1. Daniel, M.P., Denis, M.: Spatial descriptions as navigational aids: A cognitive analysis of route directions. Kognitionswissenschaft **7** (1998) 45–52
2. Denis, M., Pazzaglia, F., Cornoldi, C., Bertolo, L.: Spatial discourse and navigation: An analysis of route directions in the city of venice. Applied Cognitive Psychology **13** (1999) 145–174

3. Tversky, B., Lee, P.: Pictorial and verbal tools for conveying routes. In Freksa, C., Mark, D., eds.: Spatial Information Theory: Cognitive and Computational Foundations of Geographic Information Science, Springer Verlag (1999) 51–64

4. Sorrows, M., Hirtle, S.: The nature of landmarks for real and electronic spaces. In Freksa, C., Mark, D., eds.: Spatial Information Theory: Cognitive and Computational Foundations of Geographic Information Science, Springer (1999) 37–50

5. Nothegger, C., Winter, S., Raubal, M.: Computation of the salience of features. Spatial Cognition and Computation **4** (2004) 113–136

6. Winter, S.: Route adaptive selection of salient features. In Kuhn, W., Worboys, M., Timpf, S., eds.: Spatial Information Theory: Foundations of Geographic Information Science; International Conference, COSIT 2003. Volume 2825 of Lecture Notes in Computer Science., Springer (2003) 320–334

7. Klippel, A., Winter, S.: Structural salience of landmarks in route directions. In Cohn, A.G., Mark, D.M., eds.: COSIT 2005. Volume 3693 of Lecture Notes in Computer Science., Springer (2005) 347–362

8. Klippel, A., Tappe, H., Habel, C.: Pictorial representations of routes: Chunking route segments during comprehension. In Freksa, C., Brauer, W., Habel, C., Wender, K.F., eds.: Spatial Cognition III — Routes and Navigation, Human Memory and Learning, Spatial Representation and Spatial Learning. Lecture Notes in Artificial Intelligence, Springer; Berlin (2003) 11–33

9. Elias, B.: Extracting landmarks with data mining methods. In Kuhn, W., Worboys, M., Timpf, S., eds.: Spatial Information Theory: Foundations of Geographic Information Science; International Conference, COSIT 2003. Volume 2825 of Lecture Notes in Computer Science., Springer (2003) 398–412

10. Elias, B., Brenner, C.: Automatic generation and application of landmarks in navigation data sets. In Fisher, P., ed.: Developments in Spatial Data Handling, Springer (2004) 469–480

11. Elias, B.: Extraktion von Landmarken für die Navigation. PhD thesis, Universität Hannover (2006) (in press).

12. Dijkstra, E.W.: A note on two problems in connexion with graphs. In: Numerische Mathematik. Volume 1. Springer (1959) 269–271

13. Duckham, M., Kulik, L.: "'simplest"' paths: Automated route selection for navigation. In Kuhn, W., Worboys, M., Timpf, S., eds.: Spatial Information Theory: Foundations of Geographic Information Science; International Conference, COSIT 2003. Volume 2825 of Lecture Notes in Computer Science., Springer (2003) 169–185

14. Caduff, D., Timpf, S.: The landmark spider: Representing landmark knowledge for wayfinding tasks. In T. Barkowsky, C. Freksa, M.H., Lowe, R., eds.: Reasoning with Mental and External Diagrams: Computational Modeling and Spatial Assistance, AAAI Press, Stanford, CA, USA (2005) 30–35

15. Winter, S.: Modeling costs of turns in route planning. GeoInformatica **6** (2002) 345–361

16. Golledge, R.G.: Path selection and route preference in human navigation: A progress report. In: Spatial Information Theory: A Theoretical Basis for GIS, International Conference COSIT '95, Semmering, Austria, September 21-23, 1995, Proceedings. Volume 988 of Lecture Notes in Computer Science., Springer (1995) 207–222

17. Golledge, R.D.: Human wayfinding and cognitive maps. In: Wayfinding Behavior. John Hopkins Press (1999) 5–45

18. Dalton, R.C.: The secret is to follow your nose: Route path selection and angularity. Environment and Behavior **35** (2003) 107–131

19. Miller, G.: The magical number seven, plus or minus two: Some limits on our capacity for processing information. The Psychological Review **63** (1956) 81–97
20. Mark, D.M.: Automated route selection for navigation. IEEE Aerospace and Electronic Systems Magazine **1** (1986) 2–5
21. Richter, K.F., Klippel, A.: A model for context-specific route directions. In Freksa, C., Knauff, M., Krieg-Brückner, B., eds.: Spatial Cognition IV. Reasoning, Action and Interaction: International Conference Spatial Cognition 2004. Volume 3343 of Lecture Notes in Computer Science., Springer (2004) 58–78
22. Richter, K.F., Klippel, A., Freksa, C.: Shortest, fastest, - but what next? a different approach to route directions. In Raubal, M., Sliwinski, A., Kuhn, W., eds.: Geoinformation und Mobilität - von der Forschung zur praktischen Anwendung. Beiträge zu den Münsteraner GI-Tagen 2004. IfGIprints, Institut für Geoinformatik; Münster (2004) 205–217
23. Burnett, G., Smith, D., May, A.: Supporting the navigation task: Characteristics of 'good' landmarks. In Hanson, M.A., ed.: Contempory Ergonomics 2001, Taylor and Francis (2001) 441–446
24. Maaß, W.: How spatial information connects visual perception and natural language generation in dynamic environments. In Frank, A., Kuhn, W., eds.: Spatial Information Theory, Springer Verlag (1995) 223–240
25. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. 2. edn. Prentice Hall (2002)

What Is the Region Occupied by a Set of Points?

Antony Galton[1] and Matt Duckham[2]

[1] School of Engineering, Computer Science, and Mathematics
University of Exeter, Exeter EX4 4QF, UK
A.P.Galton@exeter.ac.uk
[2] Department of Geomatics
University of Melbourne, Victoria 3010, Australia
mduckham@unimelb.edu.au

Abstract. There are many situations in GIScience where it would be useful to be able to assign a region to characterize the space occupied by a set of points. Such a region should represent the location or configuration of the points as an aggregate, abstracting away from the individual points themselves. In this paper, we call such a region a 'footprint' for the points. We investigate and compare a number of methods for producing such footprints, with respect to nine general criteria. The discussion identifies a number of potential choices and avenues for further research. Finally, we contrast the related research already conducted in this area, highlighting differences between these existing constructs and our 'footprints'.

1 Introduction

There are many situations in GIScience where it would be useful to be able to assign a region to a set of points, to represent the location or configuration of the points as an aggregate, abstracting away from the individual points themselves.

In map generalisation, for example, what appears at one level of detail as a set of discrete points may be better represented, at a coarser level of detail, as a region. The region indicates where the points are located and can give a general idea of their configuration, but does not indicate how many points there are or their individual locations. In Figure 1, for example, the configuration of points represented at one scale as in the left-hand illustration might appear at a smaller scale as in the right-hand illustration.

One might refer to the 'outline' of a group of trees, or a flock of birds, or generally of any aggregation of discrete point-like objects, but it is important to appreciate that the outline is not by any means uniquely determined. To illustrate this, note that each of the illustrations in Figure 2 represents a possible outline for the same set of points; depending on one's purpose in requiring an outline, some may be 'better' solutions than others, but none is absolutely 'correct'.

Different choices of outline can have an effect on how we understand phrases like 'among the trees' or 'in the wood'. The trees can be considered to form a
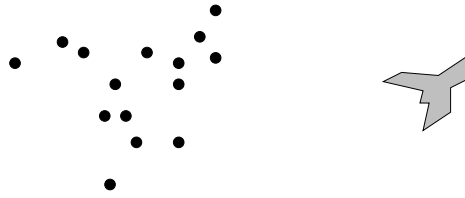
Fig. 1. Generalization and scale for points and regions
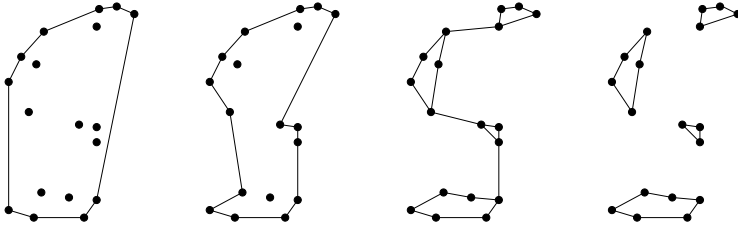
Fig. 2. Possible outlines for the region occupied by a set of points

collection of point-like objects distributed over an area of space. Which other spatial points should we count as 'among the trees'? The obvious answer is 'the points enclosed within the outline of the trees', but as we have seen, this does not provide us with a determinate answer.

Sometimes a configuration of objects is naturally divided into a number of separate clusters. The clustering may be 'obvious', as in Figure 3, where the set of points on the left is shown as falling into three clusters by the fact that the region associated with it (on the right) has three connected components.

Fig. 3. Clustering in regions occupied by a set of points

In other cases, there may be several equally good answers, as for example in Figure 4, where the same set of points is presented as being grouped into either two or four clusters.

One criterion for 'good' clustering may of course be visual salience, but visual salience may in turn serve as a pointer towards some underlying causal mechanisms which lead to the points being clustered in the way that they are. Evaluation of particular solutions will always be subject to the vagaries of human judgment; this is closely tied to the notion of gestalt perception, that faculty of the human perceptual system which enables us to perceive complex configurations as wholes
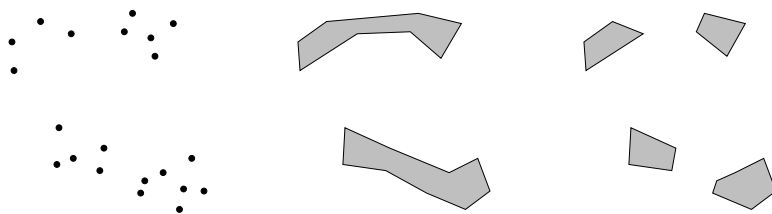
Fig. 4. Ambiguity in clustering

with global properties which cannot be simply derived by aggregating local properties. Our inquiry must therefore go beyond computational geometry to engage with more human-oriented disciplines such as cognitive science.

The criteria for evaluating any particular solution to the problem of associating a region with a set of points, or indeed any general *method* of solution, must in the last instance be determined by the particular application context in which it is being evaluated. We can, however, point to a number of general concerns which will be relevant to a wide range of application contexts and which we can therefore take to represent a basic set of criteria on top of which more refined criteria can be built as the application demands. There follows a list of such concerns; for convenience we shall use S to refer to the given set of points, and $R(S)$ to refer to the region proposed as representing their collective location.

1. Should every member of S fall within $R(S)$, as in Figure 5(a,c–g), or are outliers permitted, as in Figure 5(b)?
2. Should any points of S be allowed to fall on the boundary of $R(S)$, as in Figure 5(a–b,d–g), or must they all lie in its interior, as in Figure 5(c)?
3. Should $R(S)$ be topologically regular, as in Figure 5(a–c, e–g), or can it contain exposed point or line elements, as in Figure 5(d)?
4. Should $R(S)$ be connected, as in Figure 5(a–d,f,g), or can it have more than one component, as in Figure 5(e)?
5. Should $R(S)$ be polygonal, as in Figure 5(a–e,g), or can its boundary be curved, as in Figure 5(f)?
6. Should $R(S)$ be simple, i.e., its boundary is a Jordan curve, as in Figure 5(a–c,f), or can it have point connections as in Figure 5(g)?

Note that questions (1) and (2) are concerned with the relationship between the region $R(S)$ and the points in S, whereas (3)–(6) are all concerned with what kinds of regions we are prepared to admit to play the role of $R(S)$.

A further question, which is difficult to frame precisely, let alone answer, is concerned with the amount of 'empty space', unoccupied by points, that is allowed inside a region. Of course, *all* the area of the region apart from the points themselves (which mathematically may be of dimension zero, but in applications, as in our diagrams, might just as well be 'small regions') is empty space; what is meant is rather illustrated in Figure 6: on the left, Figure 5(a) is repeated, with the the largest circle contained in $R(S)$ but disjoint from S highlighted; on
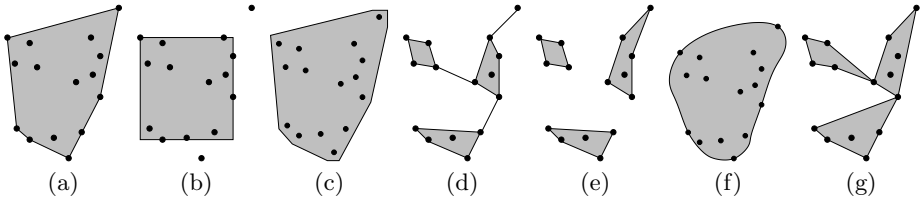
Fig. 5. Examples to illustrate the evaluation criteria

the right we do the same for Figure 5(d). In the latter, $R(S)$ clearly contains less 'empty space', and therefore for some purposes may be regarded as more desirable.
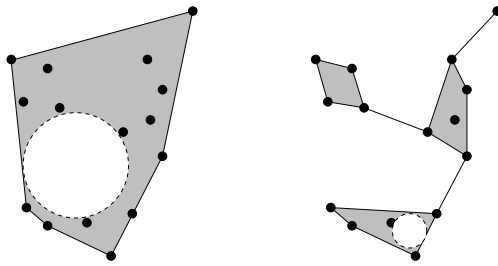
Fig. 6. Empty space as a criterion for region forming

Our question is therefore:

7. How big is the largest circular (or other specified) subregion of $R(S)$ that contains no elements of S?

Further evaluation criteria relate to the *methods* of solution themselves, rather than the results produced by them:

8. How easily can the method used be generalised to three (or more) dimensions?
9. What is the computational complexity of the algorithm?

Finally, it is important to note that we have deliberately chosen objective geometric and computational criteria for comparing $R(S)$ and methods of generating $R(S)$. Our list might also be extended with innumerable further perceptual criteria (e.g., the ability of human visual perception to discern different elements in $R(S)$) and wider cognitive criteria (e.g., the aesthetics of $R(S)$). However, such extensions would make comparison much harder and more laborious.

2   Convex Hulls

Ask a mathematician to define a region associated with a set of points, and there is a fair chance that they will nominate the convex hull. The convex hull

of a set of points S in the plane is the smallest convex polygon that encloses S. The convex hull has the advantage that it is uniquely defined, in a very simple way, for an arbitrary set of points. Moreover, algorithms for computing convex hulls have been thoroughly researched and a good deal is known about their computational properties (e.g., [1,2]).

How do convex hulls fare with respect to our list of evaluation criteria? Note that we are here assuming that the set of points S is finite; if infinite sets are allowed, then the answers to some of the questions will be different.[1]

1. Outliers are not allowed.
2. There are always points of S on the boundary of $R(S)$.
3. The convex hull is topologically regular (unless the points are collinear).
4. The convex hull is connected (a disconnected region is by nature non-convex).
5. The convex hull is polygonal.
6. The boundary of the convex hull is a Jordan curve (unless the points are collinear).
7. The amount of empty space included within the convex hull can be large.
8. The construction readily generalises to n dimensions.
9. There exist convex-hull algorithms of complexity $O(n \log n)$.

Characteristics 4 and 7 in particular mean that the convex hull is unsuitable for many purposes. A good outline—one which represents the shape of a point configuration in a cognitively salient way—must not allow too much empty space. Clustering can also be cognitively salient, and consequently in many cases $R(S)$ must be allowed to be disconnected to provide a satisfactory outline (although clustering might also be treated as a separate problem from outline generation).

These points are illustrated in Figure 7(a) and (b). In (a), we see that the convex hull entirely fails to capture an essential feature of the set of points under consideration, namely that they occupy a roughly C-shaped region. In (b), likewise, the convex hull fails to reveal that the points under consideration fall into two distinct clusters separated by a point-free gap that is comparable in size to the clusters themselves.

For this reason, we have been led to look for methods for generating *non-convex* 'hulls' of various kinds—as shown for the same sets of points in Figure 7(c) and (d). We shall refer to regions of this kind, which are intended to represent the location of a set of points, as *footprints* for that set. A natural way to approach this is by way of extending or generalising existing convex-hull algorithms. We shall consider some examples of this approach first before turning our attention to some alternative methods. Throughout, it should be borne in mind that, unlike the convex hull, footprints are not unique; it never makes sense to ask whether a proposed footprint is 'correct'—certainly not in isolation, but even with respect to any specified application context. For this reason, evaluation of the results is not, and never can be, an exact science, relying as it must to some extent on subtle and unquantifiable elements of human judgment.

---

[1] For example, if S is an open set, then none of its points lies on the boundary of the convex hull; and if S is a convex curvilinear figure then its convex hull (i.e., S itself) is non-polygonal.
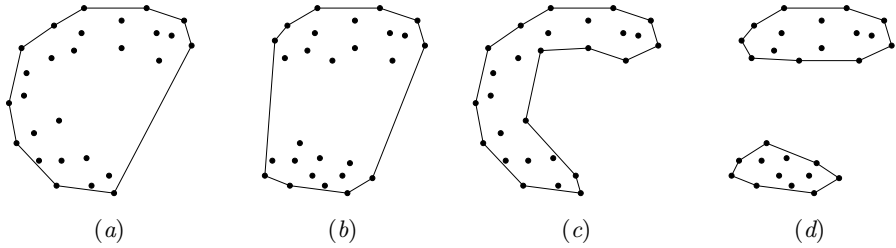
Fig. 7. Two convex hulls and two non-convex 'footprints'

3 Gift-Wrapping and the Swinging Arm

Our first procedure generalises the well-known 'gift-wrap' algorithm for generating convex hulls. In the planar (two-dimensional) version of this algorithm, the hull is generated by a sequence of swings of a half-line L, initially anchored at an extremal point of S. At each step, L is anchored to the last point added to the footprint, and rotated clockwise until it hits another point in S. This is added as the next vertex of the footprint, and the procedure is repeated until we return to the starting point.

The Swinging Arm procedure for generating footprints (see Appendix A) is essentially identical to this except that instead of a half-line we use a line-segment of some constant length r specified as an input to the algorithm. This is the 'swinging arm'. If r is not less than the longest side of the convex hull perimeter, then the procedure will generate the convex hull; but if it is shorter than that, the polygon it generates is non-convex. For sufficiently short arms, the algorithm will initially generate a footprint component which does not include all of S; in this case, we repeat the algorithm starting from a new extremal point selected from the points of S not already included in one of the components. This is repeated until every point of S is included in one of the components. The footprint thus generated will then be disconnected. When the length of the arm is less than the minimal separation of any two points in S, then the swinging arm will never encounter any points at any stage in the process: in this case the footprint is identical to S.

Figure 8 shows the sequence of steps by which the algorithm generates a footprint for a set of nine points, given an arm-length r. At step j, a circle of radius r is inscribed about the latest point $(H_{1,j})$ recruited to the footprint, and the next edge to be added to the footprint $(H_{1,j}H_{1,j+1})$ is indicated by a dashed line, previous edges being shown solid.

The footprint obtained will vary with the length of the swinging arm, as shown in Figure 9 for the same set of points as before. The values of r shown are those at which the footprint changes; for example, the footprint shown for $r = 3$ will result for any arm-length in the range $3 \leq r < \sqrt{10}$. For smaller values of r (e.g., $r = \sqrt{5}$), the polygons constituting the footprint may be degenerate, and for $r < \sqrt{5}$, the footprint is identical to S.
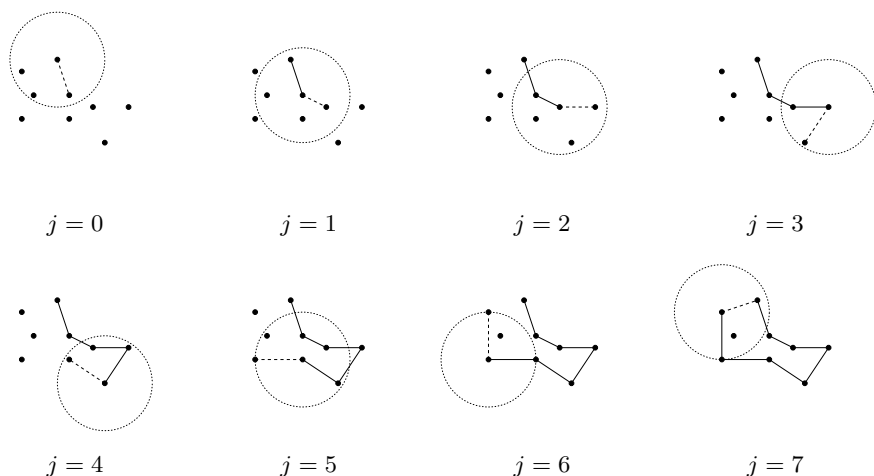
Fig. 8. Construction of a footprint using the 'swinging arm' algorithm

In the swinging arm algorithm presented above, the arm is stipulated to swing clockwise. Does it make any difference if instead we make it swing anticlockwise? In the example considered, there is one case where it does make a difference. The first two illustrations in Figure 10 show the results for arm-length $\sqrt{26}$, swinging clockwise and anticlockwise. Inspection of these figures will reveal the characteristic configuration of points which gives rise to the difference.

Since the direction in which the arm is swung is essentially arbitrary, it is unsatisfying that the choice can lead to different results. For this reason, one might prefer to use the union of the clockwise and anticlockwise swinging-arm footprints for a given set of points, as shown in the third illustration in Figure 10. This idea leads naturally on to the further idea that, in order to obtain this, we could discard the swinging arm altogether, and instead simply join together *all* pairs of points whose separation is less than or equal to the arm length, as shown in the rightmost illustration.[2] To obtain the footprint, we then simply include all points lying within any closed polygon formed out of these joins. This method, which we shall call the *Close Pairs* method, could also be seen as a direct generalisation of a method for generating the convex hull, since if we join together all pairs of points regardless of their separation then we obtain the convex hull.

How do the Swinging Arm (SA) and Close Pairs (CP) methods stand with respect to our list of general evaluation criteria? Except where otherwise stated, the comments below apply equally to both methods.

1. Every member of S falls within $R(S)$.
2. There must be points of S on the boundary of $R(S)$.
3. $R(S)$ is in many cases topologically non-regular.

---

[2] This the same as the '$r_{\min}$-circle graph' of [3].

$r = \sqrt{5}$ $r = 3$ $r = \sqrt{10}$ $r = \sqrt{13}$

$r = 4$ $r = 5$ $r = \sqrt{52}$ $r = \sqrt{53}$
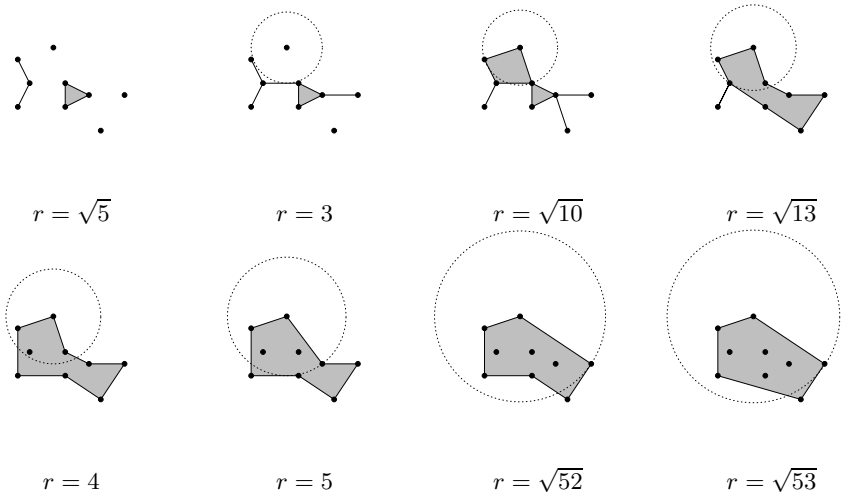
Fig. 9. Effect of varying arm-length on the footprint

Clockwise   Anticlockwise   Union   Close pairs
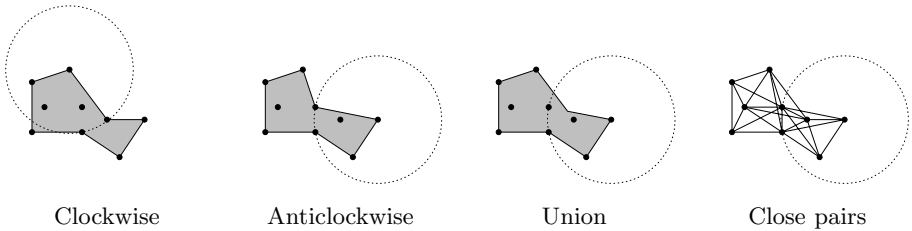
Fig. 10. Clockwise *vs* anticlockwise swinging arm ($r = \sqrt{26}$)

4. $R(S)$ is in many cases disconnected.

5. $R(S)$ is always polygonal, so long as this term is understood to include degenerate polygons such as polylines or isolated points.

6. In many cases, a large area of empty space included within the convex hull will be excluded from the footprint when r is made short enough (for example the large concavity of a 'C'-shaped distribution of points). This is not always the case, however, for example if the points of S are, say, the vertices of a many-sided regular polygon; if r is at least as long as the side-length of the polygon, the SA and CP algorithms will both give the convex hull, including the large empty interior.

7. $R(S)$ may have point connections (i.e., non-Jordan boundary).

8. Generalising SA to three dimensions is possible, but not straightforward—we need to use a 'swinging flap', and whereas in two dimensions the boundary of a region is linear, and hence can be constructed sequentially, in three-dimensions there is the added complication that we have always to decide which edge to swing the flap about next.

Generalising CP seems more straightforward: first we must join together all pairs of points whose separation is less than r; next include the interior of any planar polygon formed from the edges thus obtained; and finally include the interior of any polyhedron bounded by those polygons. A complication here is that one might also wish to include the interiors of non-planar polygons, but then it is not necessarily obvious how the interior of such a polygon is to be defined.

9. For both SA and CP the computational complexity is larger than one might imagine at first sight.

   With SA, in the course of generating a footprint-component, the swinging arm might encounter a point which already lies in the interior of that component; such points should not be included in the perimeter of the footprint. In the algorithm this is accomplished at step 3(e)(ii) in which such interior points are marked as unavailable; this adds significantly to the complexity. However, even with this complication, the worst-case complexity is $O(n^3)$, and in practice it is usually much better than this ($O(n^2)$ or less). Note that the complexity depends on r as well as the size of the input set—in general, the complexity decreases as r increases, up to the point at which the footprint becomes convex.

   With CP, having constructed the edges one must then identify the closed polygons. Note that the diagrams in Figure 12 were produced simply by selecting the edges of length less than r, for a range of suitable values of r. This is sufficient for the human eye to perceive the relevant footprint, but a CP algorithm needs to identify that footprint as an explicit polygon. Although we have not implemented this, initial investigations suggest that such an algorithm would be computationally quite expensive. Overall, a time complexity of $O(n^2)$ seems the best that could possibly be hoped for, with the likelihood that a more thorough investigation of a CP algorithm would reveal a worse complexity.

Regarding item 6, a simple refinement of the CP algorithm allows one to eliminate large areas of empty space. Instead of including the interiors of *all* polygons formed by the edges of length less than r, we can select only those polygons with less than some specified number of sides. For example. we could restrict it to just the triangles. A triangle whose sides are of length at most r has area at most $0.433r^2$ (i.e., $\frac{\sqrt{3}}{4}r^2$)—but the largest *circle* that can be inscribed in such a triangle has area at most $0.262r^2$ (i.e., $\frac{1}{12}\pi r^2$).

4   A Delaunay-Based Method

Work by Duckham et al. [4] has used Delaunay triangulations as the basis for defining a footprint for a set of points. The method, here designated DT, begins by building the Delaunay triangulation of the points. Then the algorithm 'shaves off' boundary edges in decreasing order of length, subject to constraints that ensure that the final shape contains all the input points and is both regular and

has a Jordan curve boundary (topologically isomorphic to the unit cell). Figure 11 provides an illustration of the process.

The DT algorithm terminates at a uniquely defined final shape. In Figure 6, this terminal shape is shown in step 8, where no further edges can be removed without violating the regularity and Jordan constraints. However, in general, running the algorithm to its terminal shape can produce very sinuous shapes with low visual salience. Therefore, Duckham et al. suggest a number of ways of parameterizing the algorithm based on a minimum edge-length which leads to shapes that have higher visual salience. The final algorithm presented by Duckham et al. is highly efficient in terms of time complexity.
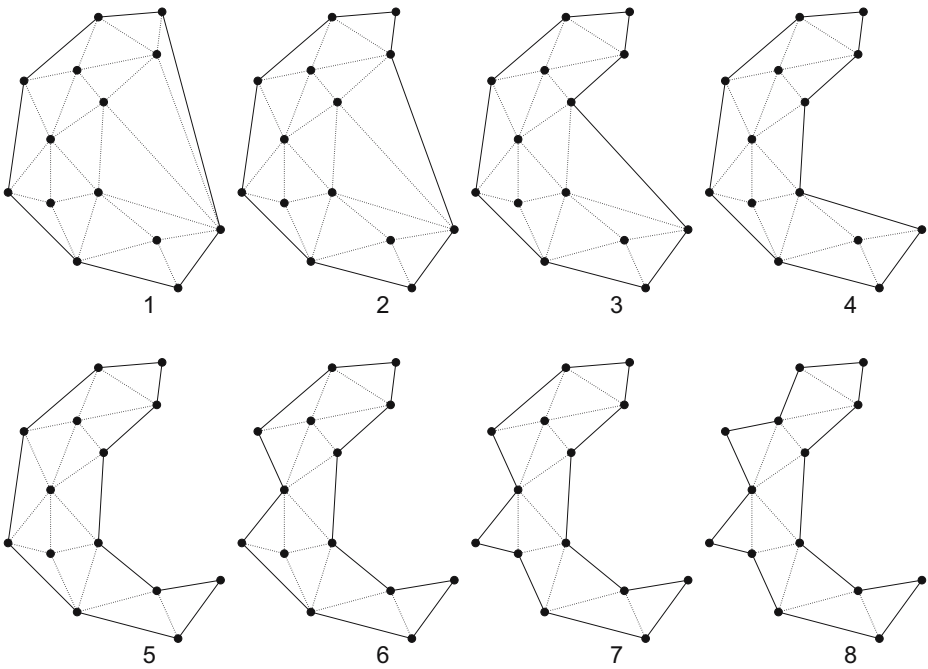
Fig. 11. Some footprints produced by edge-removal from a Delaunay triangulation

A comparison of the DT method (Figure 11) with the CP method (Figure 12) reveals that while for some values of r, the CP-footprint coincides with the corresponding DT footprint, the CP method also produces footprints that are inaccessible via the DT method. Figure 12 shows a selection of CP-footprints generated for the same set of points used in Figure 11. Note in particular that footprint 3 cannot be a DT footprint since two of the vertices arise from the crossing of two edges. Note further that as r becomes small (as in footprints 7 and 8), non-regularity sets in. Figure 12.6 has a point-connection—i.e., two triangles joined only at a shared vertex; topologically, this is regular (the footprint

is the closure of its interior), but this feature is also disallowed by the DT method. Footprints 7 and 8 are not only non-regular, they are also disconnected, another feature not possible for a DT-footprint.
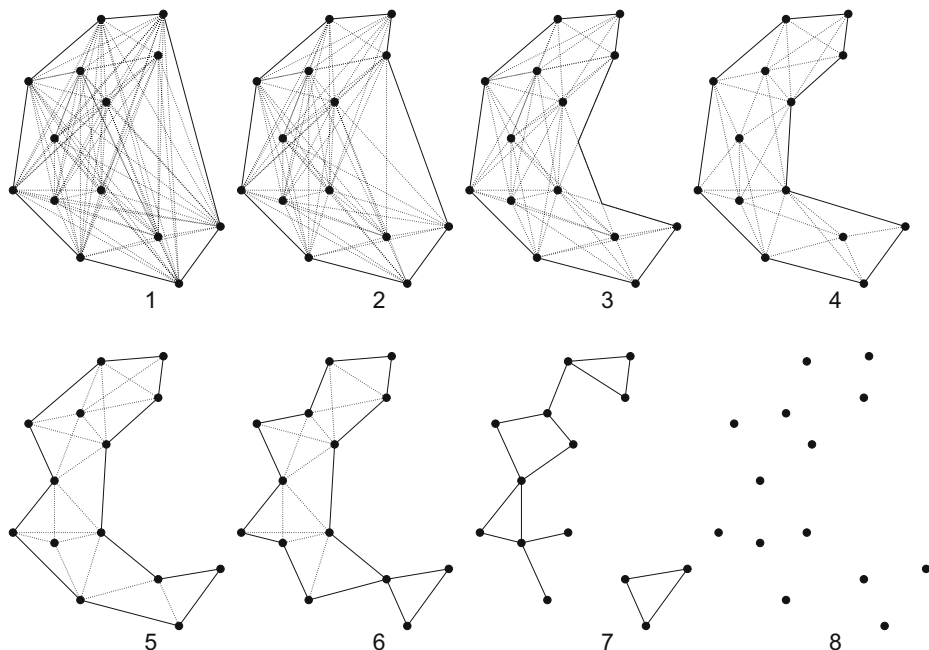
Fig. 12. Some footprints produced by the 'close pairs' method (cf. Figure 11)

Our list of evaluation criteria can be applied to the DT method of [4], with results as follows:

1. Every member of S falls within $R(S)$.
2. There are always points of S on the boundary of $R(S)$.
3. $R(S)$ will always be topologically regular.
4. $R(S)$ will always be connected.
5. $R(S)$ will always be polygonal.
6. The boundary of $R(S)$ will always be a Jordan curve.
7. It is possible for $R(S)$ to contain a large amount of empty space, like a convex hull—although since $R(S)$ must be a subset of the convex hull, it cannot contain *more* empty space. The requirements for regularity and simplicity mean, for example, that it is not possible to generate polygons with holes using the DT algorithm. Thus, a 'ring' of points generates a circular polygon hull rather than an annulus.
8. Extensions to three dimensions are problematic (see below).
9. Duckham et al. report that their algorithm achieves a computational complexity of $O(n \log n)$, although this efficiency depends on the regularity and simplicity constraints, as well as only holding in two dimensions.

Duckham et al. argue that the regularity constraints are useful both from the perspective of visual salience, and from the perpective of algorithm efficiency. It is conceivable that a variety of different algorithms might be developed based on Delaunay triangulations and without regularity constraints. These would allow, for example, the generation of disconnected regions and regions with holes (hence, the criteria 3, 4, 6 and 7 listed for the DT method above do not in general hold for all Delaunay triangulation-based methods). However, we can observe a number of features of any Delaunay-based algorithm, regardless of the specific algorithm used.

1. Starting with a Delaunay triangulation immediately constrains the solution space for the boundary of the resulting region, much more so than the SA and CP methods. Some edges that might be added using SA and CP methods are not present in the Delaunay triangulation, and so can never be present in the final shape; the converse is not true.
2. Developing an algorithm that must compute the Delaunay triangulation immediately places a lower bound of $\Omega(n \log n)$ on the optimal time complexity for the entire algorithm (in itself not a negative feature, since this clearly still allows for the specification of efficient, scalable Delaunay-based algorithms).
3. Although Delaunay triangulations can be extended beyond two dimensions, there are implications for extending Delaunay-based algorithms to higher-dimensional spaces. For example, the possibility of unshellable triangulations in more than two dimensions presents problems for extending Duckham et al.'s method to three or more dimensions.

5   Extended Footprints

Why should we insist on finding a polygon (or similar) region whose vertices are points of the original set? To represent the region occupied by the points at a certain granularity, it is natural to suppose that each point has an 'area of influence', and to represent the region by the aggregation of all the areas of influence of the points in the set. The points themselves would then all lie in the interior of the region, with none on the perimeter, as in Figure 5(c): this leads to solutions which differ from all the ones considered up to now with respect to our second criterion. Let us call such a region an *extended footprint* for the set of points.

An obvious way of generating an extended footprint is as follows. For each point in the set, construct a closed circular disc of radius $\frac{1}{2}r$ centred on that point, and let the region be the union of these discs. Let us call this the Covering Discs (CD) method. An example is shown in Figure 13. To produce a smoother outline, we could draw in the shared tangents of any pair of discs which overlap, and include also the area between the tangents. This is the Covering Discs with Tangents (CDT) method; an example is shown in the right-hand illustration of Figure 13.

The CD/CDT method is closely related to the dilation and erosion operations common in computer graphics and mathematical morphology (e.g., [5]). Simple and obvious though this method is, the footprints produced do not seem very
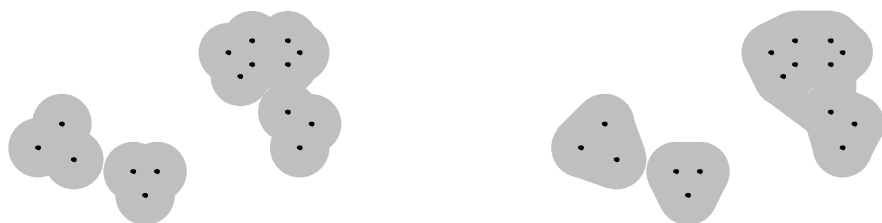
Fig. 13. An extended footprint without and with added tangents

natural. An important question concerns how far the footprint should extend into the 'outer space'; the CD method assumes that this distance should be the same all round the periphery of the footprint, and this may indeed be the most natural default position. However, one situation where we may be able to make a more informed choice about how far the footprint should extend beyond the points is when we have additional information about points which are to be *excluded* from the region. This allows one to make use of the methods of [6] and [7] which are briefly discussed in the next section.

6   Related Work

In terms of motivation, the closest work we are aware of is that of Edelsbrunner *et al.* [8], in which the concept of 'α-shape' is developed as a generalisation from a certain definition of convex hull. The α-shape of a point-set S is a straight-line graph derived from the α-hull, which is defined to be the intersection of all closed discs of radius $1/\alpha$ that contain all the points of S. (If $\alpha < 0$, the closed disc of radius $1/\alpha$ is interpreted as the complement of the open disc of radius $-1/\alpha$.) For $\alpha = 0$, we have the convex hull; for sufficiently large negative values of α, we have the set S itself. Unlike with our footprints, some of the points of S may lie outside the α-shape (cf. criterion 1).

Traka and Tziritas [9] give an algorithm for constructing a non-convex hull $R(S)$ by starting with the convex hull of S and successively adding extra points from S to the perimeter; but for their purposes it is required that *all* points of S lie on the perimeter of $R(S)$ (which is therefore a Hamiltonian circuit of the complete graph on those points), whereas for our more general purposes this condition is unnecessarily stringent.

In the context of GIS, a method based on Voronoi diagrams has been suggested by [6]. This is applicable where the points S are representative localities within some region R, and in addition we are given a set S' of points known to lie outside R. In this Dynamic Spatial Approximation Method (DSAM), the Voronoi diagram of $S \cup S'$ is constructed, and an approximation to region R is derived as the union of the Voronoi cells containing members of S. This approximation to R is a kind of footprint, typically non-convex, of the set S. It is, of course, an extended footprint. An example is shown in Figure 14(a), where the initial set of points S (shown by the black circles) is supplemented by additional

points (shown by the white circles) which are supposed to be definitely outside the target region R. The dotted lines give the Voronoi tessellation for the full set of points, and the solid line shows the boundary suggested by the DSAM method for region R. The resulting approximation to R is shown shaded. It consists, in fact, of all those points for which the nearest classified point is in S.

A drawback is that this method can only be used if a suitable set S' is given, which is often not the case; however, one might consider modifying the method to cover cases where all we are given is the set S, by introducing some arbitrary set of points lying outside the convex hull and using these as S'. The method could be iterated: once a footprint has been formed by the DSAM method, we create a new S' from points selected to lie outside that footprint, and then generate a new footprint. Successive iterations will lead to footprints that cling more tightly to the original set S.[3]

A related method, involving Delaunay triangulations, is suggested in [7]. This is illustrated in Figure 14(b). Here the Delaunay triangulation of the same set S is shown; each edge of the triangulation is either dotted (if it joins two 'white' points), dashed (if it joins two 'black' points), or solid (if it joins a 'white' and a 'black' point). Region R (shown shaded) is obtained by joining up the midpoints of the solid edges in the triangulation. It will be seen that while it resembles the Voronoi-based approximation, it has the advantage of giving a smoother outline.

In three-dimensions, the Power Crust method of [10,11], which is based on Medial Axis Transforms, provides good 'footprints'[4] very much in the spirit of our own requirements. However, the application context of this work is surface reconstruction from data points captured from real objects. As a consequence, the initial set of points are required to lie on the surface of the output region, which means that the problem under consideration is somewhat different—albeit related—to ours.

Our notion of a footprint is in some ways superficially reminiscent of, though in essence quite different from, the *relative convex hull* of [12]. The relative convex hull of a region $R \subseteq R'$ is the figure with the minimum perimeter whose perimeter lies entirely in the closure of $R' \setminus R$.

The fact that our footprints are parametrised by r, the swinging arm-length (or disc diameter), recalls multiresolution approaches to shape such as [13] and indeed reflects the same underlying motivation of revealing different kinds of structure existing at different granularities. However, these approaches are essentially concerned with the shapes of regions, whereas we are concerned with constructing a region to represent a discrete set of points.

Having obtained such a region, we still need to be able to describe the resulting shape—what is it, for instance, about Figure 7(c) which makes it a C-shape: where does the *linearity* of the 'C' come from? Features such as this might be revealed by some form of *skeletonisation* procedure. Skeletonisation techniques

---

[3] This method has been investigated by Nicholas Talbot in an unpublished final-year undergraduate project supervised by Antony Galton at Exeter.

[4] The inverted commas here merely reflect the slight oddity of referring to a three-dimensional region as a footprint.
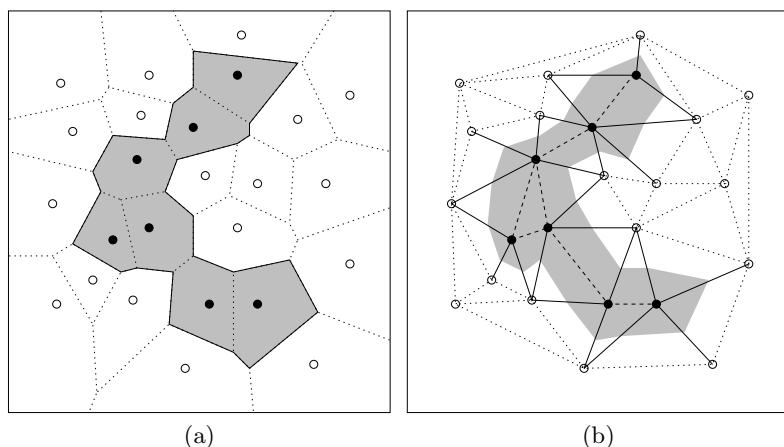
Fig. 14. Approximations to the region defined by a set of points, using (a) a Voronoi tessellation (after Alani *et al.* [6]) and (b) a Delaunay triangulation (after Arampatzis *et al.* [7]).

are being extensively investigated by a number of researchers, e.g., [14], but not in connection with our footprint problem.

7 Conclusions and Further Work

In this paper we have only scratched the surface of what is potentially a rich field for investigation. We have described a number of methods for generating footprints for sets of points, and have hinted at some possible applications for such techniques. However, for the further evaluation of these different methods, it will be necessary to adopt a sharper focus in this respect: ultimately, it will only be in specific contexts of application that a final evaluation of any given method can be given, and this must serve as a pointer to further work.

Other topics for investigation include the mathematical properties of the footprints derived by the various methods, a closer examination of the computational properties of the algorithms (and in particular a more detailed analysis of their complexity), and, of course, a generalisation of the methods to three dimensions.

In relation to the first of these topics, it would be useful to investigate the relationship between our various kinds of footprint and the general geometrical notion of a *hull*. Klette and Rosenfeld [15] give the conditions for a hull operator H as

H1. $S \subseteq H(S)$
H2. $S_1 \subseteq S_2 \rightarrow H(S_1) \subseteq H(S_2)$
H3. $H(H(S)) \subseteq H(S)$.

If H2 is replaced by

H4. $S_1 \subseteq S_2 \rightarrow area(H(S_1)) \leq area(H(S_2))$

then we have what is called a 'near-hull'. If neither H2 nor H4 is stipulated then we have a 'pseudo-hull'. Our footprints are clearly related to these various kinds of hull but for the most part do not exactly correspond to any of them. In particular, note that property H3 will not be applicable to those footprint-generating methods which require a finite set of points as input, since in that case the footprint operator R cannot be applied to a second time, so there is no such region as '$R(R(S))$'.

Another issue that needs to be investigated is the relationship between footprint formation and clustering. Algorithms which can generate footprints with multiple components do, by that very fact, function as clustering algorithms; but they are not necessarily very good clustering algorithms, and it is arguable that since clustering is a very different topic from footprint formation, it should not be attempted to accomplish both using a single algorithm. A better solution might be to apply some dedicated clustering algorithm first, and then derive footprints from each cluster individually. One obvious possibility would be the nearest neighbor (single link) cluster algorithm, which generates clusters based on a sub-graph of the Delaunay triangulation [16]. Although not clustering algorithms, the Gabriel graph [17] and the relative neighborhood graph [18] are two other well-studied sub-graphs of the Delaunay triangulation that would be potentially useful in characterizing the region occupied by a set of points.

Acknowledgments

References

1. de Berg, M., Schwarzkopf, O., van Kreveld, M., Overmars, M.: Computational Geometry: Algorithms and applications. 2nd edn. Springer, Berlin (2000)
2. O'Rourke, J.: Computational Geometry in C. 2nd edn. Cambridge University Press, Cambridge, UK (1998)
3. van Kreveld, M.: Finding the wood by the trees. CG Tribune (1998) http://www.inria.fr/prisme/personnel/bronnimann/cgt/cgt10.ps.
4. Duckham, M., Kulik, L., Worboys, M., Galton, A.: Efficient characteristic hulls. (2006, in preparation)
5. Serra, J.: Image Analysis and Mathematical Morphology. Academic Press, New York (1982)
6. Alani, H., Jones, C.B., Tudhope, D.: Voronoi-based region approximation for geographical information retrieval with gazetteers. International Journal of Geographical Information Science **15** (2001) 287–306

7. Arampatzis, A., van Kreveld, M., Reinbacher, I., Jones, C.B., Vaid, S., Clough, P., Joho, H., Sanderson, M., Benkert, M., Wolff, A.: Web-based delineation of imprecise regions. In: Workshop on Geographic Information Retrieval (SIGIR 2004). (2004) (http://www.geo.unizh.ch/~rsp/gir/abstracts/arampatzis.pdf, accessed 8/7/05).
8. Edelsbrunner, H., Kirkpatrick, D.G., Seidel, R.: On the shape of a set of points in the plane. IEEE Transactions on Information Theory **IT-29** (1983) 551–559
9. Traka, M., Tziritas, G.: Panoramic view construction. Signal Processing: Image Communication **18** (2003) 465–481
10. Amenta, N., Choi, S., Kolluri, R.: The power crust. In: Sixth ACM Symposium on Solid Modeling and Applications. (2001) 249–260
11. Amenta, N., Choi, S., Kolluri, R.: The power crust, unions of balls, and the medial axis transform. Computational Geometry: Theory and Applications **19** (2001) 127–153
12. Sklansky, J., Kibler, D.: A theory of nonuniformly digitized binary pictures. IEEE Transactions on Systems, Man, and Cybernetics **6** (1976) 637–647
13. Cinque, L., Lombardi, L.: Shape description and recognition by a multiresolution approach. Image and Vision Computing **13** (1995) 599–607
14. Borgefors, G., Nyström, I., Sanniti di Baja, G.: Computing skeletons in three dimensions. Pattern Recognition **32** (1999) 1225–1236
15. Klette, R., Rosenfeld, A.: Digital Geometry: Geometric Methods for Digital Picture Analysis. Morgan Kaufmann (2004)
16. Xu, R., Wunsch, D.: Survey of clustering algorithms. IEEE Transactions on Neural Networks **16** (2005) 3
17. Gabriel, K.R., Sokal, R.R.: A new statistical approach to geographic variation analysis. Systematic Zoology **18** (1969) 259–278
18. Toussaint, G.T.: The relative neighborhood graph of a finite planar set. Pattern Recognition **12** (1980) 261–268

A   The Swinging Arm Algorithm

Input: A finite set S of points in the plane, and a positive real number r.
Output: A footprint of S.

1. Mark all points of S as 'available' and 'unvisited'.
2. Let $i = 0$.
3. Repeat
   (a) Increase i by 1.
   (b) Let $H_{i,0}$ be the available point in S with maximal y-coordinate (if there is more than one such, choose the one with minimal x-coordinate).
   (c) Let L be a line-segment of length r anchored at $H_{i,0}$ parallel to the positive y-axis.
   (d) Let $j = 0$.
   (e) Repeat
       i. Rotate L clockwise about $H_{i,j}$ until either it meets another available point in S or it returns to its starting position. In the former case, let $H_{i,j+1}$ be the point found (if more than one, choose the one closest to $H_{i,j}$); in the latter, let $H_{i,j+1} = H_{i,j}$.

    ii. If $H_{i,j+1}$ is marked as visited, then identify the greatest $k \leq j$ such $H_{i,j+1} = H_{i,k}$, and mark as unavailable all points of S in the interior of the polygon with vertices $H_{i,k}, H_{i,k+1}, \ldots, H_{i,j}$.

    iii. Mark $H_{i,j+1}$ as visited.

    iv. Let L be a line-segment of length r anchored at $H_{i,j+1}$ and passing through $H_{i,j}$.

    v. Increase j by 1.

    until $H_{i,j} = H_{i,0}$.

(f) Add the polygon with vertices $H_{i,0}, H_{i,1}, \ldots, H_{i,j-1}$ as component C_i of the footprint, and mark all those vertices as unavailable.

  until all points in S are unavailable.

4. Return components $C_1, \ldots, C_i$.

Voronoi Hierarchies

Christopher Gold and Paul Angel

School of Computing, University of Glamorgan, Pontypridd CF37 1DL Wales UK
`cmgold@glam.ac.uk, pangel@glam.ac.uk`

Abstract. Voronoi diagrams are widely used to represent geographical distributions of information, but they are not readily stacked in a hierarchical fashion. We propose a simple mechanism whereby each index Voronoi cell contains the generators of several Voronoi cells in the next lower level. This allows various processes of indexing, paging, visualization and generalization to be performed on various types of data. While one-level Voronoi indexes have been used before, and hierarchies of the dual Delaunay triangulation have been used for fast point location, we believe that this is the first time that the advantages of their integration have been demonstrated.

1 Introduction

People (often) like hierarchies – they give structure to an organization. So do computers – fast indexes are based on tree structures, and so are hard discs. Similarly in geography – we often organize administrative districts hierarchically. However, one of the particularly useful spatial structuring tools – the Voronoi diagram (VD) – is not usually capable of being put in a hierarchical context.

The main property of the VD that is of interest here is that it partitions space into proximal regions such that any location is associated with its nearest Voronoi generator. This allows the automated tessellation of space, and provides an adjacency structure that is valuable for generating topology [11]. The VD also provides a "geometric walk" facility [12], [17] that permits the navigation of the VD – or the dual Delaunay triangulation (DT). It also provides a simple indexing system, where the Voronoi cell may be a container for a collection of data points – e.g. [20].

The VD, and the dual DT, are basic tools in spatial analysis. The VD is used for topology generation, for volumetric calculations (e.g. as Thiessen polygons for rainfall, or for runoff calculation) and the DT is the basis of the Triangulated Irregular network (TIN) terrain model. (While any triangulation is possible, only the DT, because of its basis in the VD, is guaranteed to be locally modifiable.) They are also well studied in Computational Geometry, and form the basis of many spatial algorithms. For key surveys, see [2] and [22].

What is often missing from the VD is a useable hierarchical model. This would permit proper hierarchical indexing, generalization and paging of spatial data. The problem is simple to state: a hierarchical structure of Voronoi cells (VCs) within Voronoi cells means that the boundaries of the secondary cells are cut off by the boundaries of the (larger) primary cells. An example of this approach is given in [27] for

image retrieval and in [24] for graph drawing algorithms. This means that VCs are not fully hierarchical to each other, unlike a quad tree where each region is a subdivision of a higher region. The "recursive Voronoi diagram" of [3] consists of repeatedly adding the Voronoi centres as Delaunay nodes, rapidly densifying the original diagram.

Spatial partitions may be based on the coordinates alone, as in the grid or quad tree; on manual boundaries, as in the choropleth map; or automatically from previously-defined generators, as in the VD. For regularly spaced generators the VD may be hierarchical, as in the case of generators on a grid or generators associated with irregular boundaries [15] but in the general case this is not so. An interesting case is the work of Christaller [4]. Here, for the administration-optimizing model in particular, a regular pattern of generators is produced so that lower-level administrative district centres fall within higher-order districts. The tributary area of a higher-order central place includes the whole of the tributary areas of the neighbouring lower-order central places, even though the boundaries of the lower-order central places do not necessarily conform to the higher order boundaries if they were taken alone – see Fig. 1, and [18], p. 363.
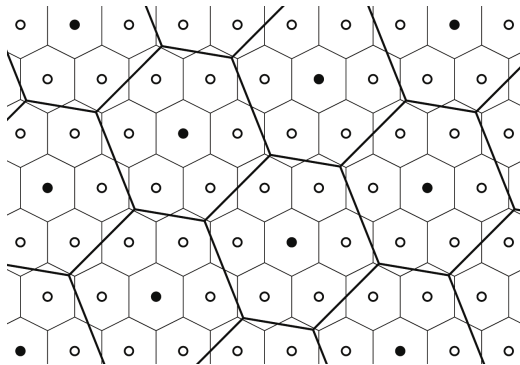
Fig. 1. Christaller hierarchy, after [18]

This forms the basis for our Voronoi hierarchy: a VC of the (higher) level-i contains all Voronoi generators of the (lower) level-i-1 that are closer to that level-i generator than to any other. (We will refer to the base level of actual data as level-0.) Thus a level-1 index cell will contain several (or many) level-0 points, and a level-2 index cell will contain several level-1 generators. Thus a form of hierarchical VD may be envisaged, where the index VC contains a set of *generators* of the underlying VD, without necessarily containing all of their *generated* VCs. This system is fully hierarchical, with as many levels (indexes of indexes) as desired, with each higher level forming a generalization of the level below. Okabe and Sadahiro [23] proposed a related structure where the index generators were local maxima of the attribute (e.g. elevation) of the base data points, and similarly higher up the index structure.

We will discuss previous work that addresses parts of our approach, various potential applications of the concept, and details of the algorithms. Section 2 describes spatial indexing using this approach, Section 3 describes how this may be used as a disc paging mechanism for spatially distributed data, and Section 4 summarizes the algorithms and data structures. Section 5 describes several applications where this approach is useful, and Section 6 provides our conclusions.
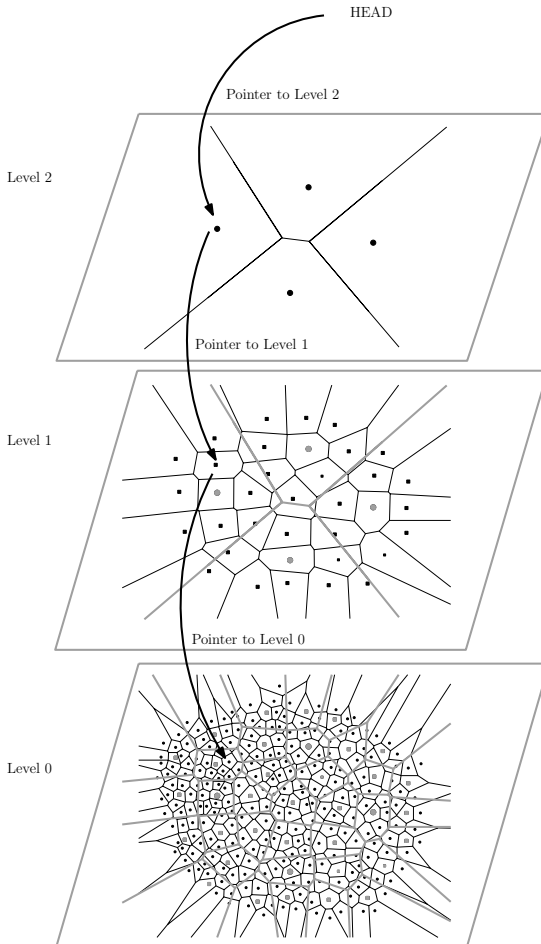
Fig. 2. Three-level Voronoi hierarchy

2 Spatial Indexing

Straightforward algorithms may be devised to implement this structure, given the standard VD, stored as a quad-edge structure where edges of both the Voronoi and Delaunay subdivisions are stored explicitly [17]: each index cell generator has a

pointer to a generator in the next lower level. Fig. 2 illustrates the structure. Note that a reference to a level-0 generator consists of a chain of pointers from the head pointer of the whole structure. Thus a reference to the parent of a generator is always preserved by the indexing system. Fig. 3 shows three levels superimposed. The index cells may be distributed in a variety of ways, depending on the application requirements. The intention is to have a dynamic system – permitting deletion as well as insertion of points, and hence allowing updating. The only implementation difficulties occur if an index cell is allowed to be empty of child generators – here decisions must be made as to how to handle the down pointer. Nevertheless, in the ordinary case a variety of strategies and applications appear to be interesting.
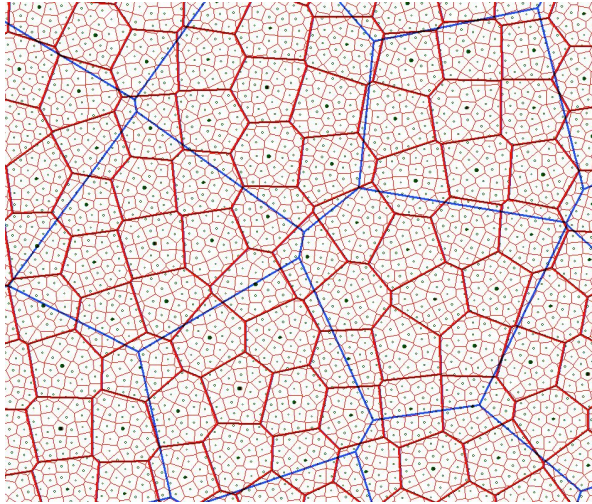
Fig. 3. Three levels of Voronoi cells

The most-studied aspect of the hierarchical model is for spatial indexing, but this has focused exclusively on the DT, not the VD. The most obvious application is to generate a fast and flexible point-location strategy – using a simple "walk" algorithm, start at some initial index cell generator, and search until the closest one to the desired location is found. Then follow the down pointer to the next level, and continue the walk as before. After following all the levels, the closest base-level data point is found. Details of the method (only using data points as index cell generators, and working only with the DT) are given in [9] but other similar papers, such as [21] and [7] may be found. Amenta et al. [1] used a "biased randomized insertion order" to improve the disc behaviour of very large data sets, but this is not intended for user-specified insertion and deletion. The simple walk algorithm of [12] and [17] consists of starting at some arbitrary location in the DT structure and testing the desired destination location p against each edge of the initial triangle using a simple determinant

test (called CCW in [17]). If *p* is outside any edge then the alternate triangle for that edge is selected and the process repeated until *p* is inside all three triangle edges. It is important to note that this is a "geometric" test, using the coordinates of *p* to find the containing triangle or closest data point.

Devillers' method [9] consist of starting at the top of the index structure, walking to the closest generator, following a pointer down to some generator in the next lower level, and continuing until the closest point in the base level is found. In his method index generators are obtained as a subset of those of the next level down using a specific form of random selection in order to achieve an analysis of efficiency, and are entirely focused on a rapid search in the DT. Nevertheless, the VD is implicit in his use of the DT. Our approach focuses on the VD and guarantees that the generator pointed to in level-i is inside the VC of the generator in level-(i+1), thus achieving the Christaller model mentioned previously.

The generators in our approach may be obtained in any fashion desired for the application: a subset of the generators below; a regular square or hexagonal pattern; by a random process – perhaps with some rejection radius to prevent too great a proximity; or any other method. Nevertheless, all indexing methods are significantly faster than a simple walk without any indexing. The number of levels desired is dependent on the size of the data set and the application.

Index structure and data set properties are particularly intimately linked when data deletion is permitted. In Devillers' method if a generator is deleted when it is the object of the index cell's down pointer then that index cell generator is deleted – and so on as far up the hierarchy as necessary. In our more general approach the down pointer may be associated with any generator within the index generator's VC, so deletion of the lower point means that its parent's down pointer must be examined, and modified, before deletion occurs. This is straightforward as all point-location queries start at the top and preserve all down pointers to the base point. Again depending on the desired application, the index generator may have its down pointer adjusted to point to any other child generator – either a neighbour to the deleted generator or the closest point to the index generator. The only complications occur when the number of an index generator's children approaches zero – here the generator's pointer may be set to null or else the generator may be deleted, allowing adjacent cells to occupy the space.

3   Paging

An interesting extension of this concept is when the dataset is too large for machine memory, and the data structure must be broken up – invalidating pointers across page (or index cell) boundaries. Here the quad-edge structure is convenient – edges that cross page boundaries, so that one node is on each page, are duplicated on each page, along with *both* end nodes. Upon walking through the structure as usual, when a page-crossing edge is detected (because the node at the other end of the edge is outside the page's index cell) the walk algorithm reverts to the index cell level (the next pointer back in the pointer chain) in order to find the index cell containing the "to" node that fell off the previous page. Following the down pointer into this cell/page,

the new structure is traversed to find the node that was previously off-page. All neighbouring edges to this node are then tested to find the duplicated edge (based on the coordinates of the "from" node on the previous page). Regular processing may then resume. (The FastWalk algorithm described in the next section performs all these actions, except the search round the "to" node, as its normal mode of operation.)

Page recovery from disc of all spatially adjacent nodes and edges within the page index cell may be invoked as needed. Thus duplication of the boundary edges and associated nodes, together with the geometric walk of the page-level index VD, permits direct navigation between pages, and thus allows storage of spatially-local data on each disc page, greatly speeding up the processing of large data sets. This has been a significant concern in the handling of large graph-based maps. The paging structure is identical to the index structure except for the duplication of page-crossing edges at the page level. This paging structure is also useful when computing tasks are to be shared between processors, and may be used for the parallel processing of large maps.

4   Algorithm Summaries

Any polygon structure may be represented by its dual graph. Voronoi cells are "proximal" cells – they contain all points in space closer to the generating point than to any other. The dual of the VD is the DT. Voronoi cells are based on a set of "generators" – which on the lowest level (level-0) are the original data points. The VD/DT must be stored using an appropriate data structure – e.g. the quad-edge structure of [17]. It is assumed that the structure is "dynamic" – points may be inserted or deleted at any time, and construction is incremental.

This allows a simple "walk" algorithm to find the closest generator to any (x, y) location, starting from any other location see [12] and [17]. This walk algorithm is fast, but theoretically inefficient for large data sets. A hierarchical or grid index is needed.

Index structures are built in the same fashion, using a smaller number of generators. Each index generator must have a pointer to a generator in the next level down that falls within the index VC. There may be one or more levels of indexes. The pointer to a base-level generator is a chain of pointers starting at the "Head" (Fig. 2), and the parent of any generator (at any level) is therefore always available without extra searching simply by reverting to the next-higher pointer in the chain.

4.1   FastWalk

To find the closest point to a desired location *p*: start at the top index level, find the closest generator G to *p* and follow its "down" pointer to a lower level generator within the VC of G. Repeat this on each level until the base level is reached. The "address" of this base generator is the sequence of "down" pointers, starting from the top. This gives any level of index cell containing the base generator. This reduces the walk over the whole map to a local walk at each level. It may be used to find both the closest generator and the enclosing triangle. Devillers [9] gives an efficiency analysis of a similar algorithm.

4.2   InsertPoint

To insert a new base point: i) perform FastWalk to get the enclosing triangle; ii) insert the new point into the base structure, using the incremental algorithm of [17]; iii) optionally update the index generator's down pointer to point to the new base point. Since all the levels are dynamic a key issue is the maintenance of the index down pointer as the map is modified.

4.3   DeletePoint

To delete a base point: i) FastWalk to the base point's location; ii) if the down pointer is to the deleted point then redirect it to an adjacent base point in the same index cell; iii) delete the base point, e.g. using the algorithm of [8]. If there are not sufficient generators in the lower level, then either delete the index generator (letting the neighbouring Voronoi cells take over) or else set the pointer to null (requiring searching the index level for a valid pointer).

4.4   Subdivision Traversal

The placement of index generators in the hierarchy has not been defined in this process. They may fall on a regular grid, which gives a tessellation of square Voronoi cells. They may be generated randomly (with some minimum separation), or they may be selected on the basis of the data distribution. The intention is to have approximately the same number of base points in each cell.

   This indexing structure may also be used for subdividing a large base structure. Each index cell on one specified level references one "page" of the data. The upper index structure may be walked through, the down pointer followed, and the walk continued on the set of data held on the lower levels. A modification of the "Scan" algorithm [14], which requires no extra storage and no flagging of elements, may be used to traverse the whole page of data.

4.5   Partitioning

The quad-edge structure has pointers both to the nodes (generators) at each end and to the adjacent edges. At a page (boundary) fault, some quad-edges will cross from one page to an adjacent one, having a node on each page. In this case the connecting edge will be copied onto both pages, along with both end nodes. These boundary-crossing pointers can not be resolved by direct pointer-following as the only linking between adjacent pages is by geometric walking at a higher level. Once a boundary quad-edge has been found, the other copy of a boundary-crossing edge may be located by using FastWalk to move to the adjacent index cell, and locate the end node belonging on the new page. A search is then made of all edges around this node in order to find the node belonging on the previous page. The matching copy of the original boundary-crossing edge is thus located. The original operation may then continue. To avoid repeated switching between pages during actions close to the boundary, it is desirable

that the currently active page, along with its Voronoi-neighbour pages, are held in memory at the same time.

4.6   Parallel Processing

Partitioning large datasets based on the hierarchical grouping of the original Voronoi patches lends itself to parallel / concurrent implementation. The FastWalk algorithm fits well into a messaging infra-structure, allowing search to pass between separate patches that may be managed by separate concurrent processes. Such approaches are popular in object-oriented software design, given the inherent modularity of the software, and are scalable in that messaging systems manage concurrent access to data more effectively than traditional state-based models, especially in large multi-processing environments.

4.7 Tessellation Maintenance

Maintenance of the tessellation at any index level requires the deletion or insertion of an index generator in the same way as for a base point described above (together with the possible similar modification of higher index levels), and the redistribution of child points/generators in the immediate neighbourhood. This last is automatic for index layers merely used for generalization or FastWalk, using the Voronoi condition, but involves physical reallocation of children when the layer is used for disc partitioning. In all cases the down pointers of the set of immediate Voronoi neighbours need to be validated (but not always changed). Attributes or properties associated with base points need to be assigned during the insertion process. Kinetic point movement is managed by the usual Delaunay edge flips ([6], [16]) and FastWalk is used to verify the index cell currently containing the moving point.

5   Applications

In addition to the basic indexing and paging processes, the Voronoi hierarchy is directly relevant to a variety of spatial analyses, in addition to the original Christaller work mentioned previously.

5.1 Terrain Modelling

Paging systems probably involve thousands of points in each index cell, and only one index level. If the "spread" between each level is smaller (perhaps 5-10) then a steeper hierarchical structure may be built, and each index node may be considered as a generalization of the lower points. For example, if the data represents terrain elevation (Fig. 4) then the index generator may represent the average of its child nodes (Fig. 5) and again for a second level (Fig. 6). Further – if the Voronoi cells are considered, rather than the triangulation, then each node represents a Voronoi prism of known volume – perhaps with water runoff volumes as well (Fig. 7). The average

may then be based on these values, preserving the local volumes of the rock. Since each data cell is assigned to only one index cell, the overall volume is also preserved. Alternatively, since the local slope is the most important additional information after the elevation of a point [5] the child nodes may be used to estimate slopes at index nodes. (Figs. 8 and 9 show the differences in the various surfaces.)

The effectiveness of this form of generalization depends on the strategy for the selection of the index generators, as well as on the distribution of the original data. If the objective is to perform flow modelling on the resulting surfaces then a uniform random pattern of generators works well, with the addition of a fairly large rejection radius in order to produce a set of approximately equal-sided index cells – see [13]. Since the finite-difference modelling used in their approach stores a rock height and a water height for each cell, the generalization process can preserve the overall water (and rock) volumes while reducing the level of detail and thus speeding the iterative simulation.

If the data is poorly distributed, e.g. from bathymetric data along ships' tracks, then the previous approach will produce many empty index cells. In this case an interpolation step, using the Voronoi-based Sibson interpolation [26] and the random generators may be used prior to the generalization. Alternatively, generators may be based on a subset of the data points, and this repeated at each level as in Devillers' approach for rapid point location.

5.2   Terrain Visualization and Level of Detail

One of the particular problems in terrain visualization is defining an appropriate Level of Detail (LOD) so that the scene may be perceived with sufficient detail, but without overloading the graphics pipeline with unnecessary triangles. Any one of the hierarchical levels previously described may serve to give the appropriate LOD for a particular map scale or view. View-dependent LOD requires a finer level of detail close to the viewer and a coarser level further away. Algorithms for LOD are well described in the computer graphics literature. (See the LOD Book [19].) For 3D perspective viewing the problem is more complex – the LOD must decrease away from the viewer and in "fly-through" applications the viewpoint changes rapidly. While several specialised algorithms have been developed, the hierarchical approach described in this paper may be sufficient for some applications. The hierarchical index provides a selection of patches of differing LOD: these may be selected on the basis of the distance (range) of the generator from the viewer, which gives a decreasing density of points with increasing distance from the viewer. Adaptive approaches can also be applied, creating deeper hierarchies based on the homogeneity properties of locally connected patches. This allows flatter regions of terrain to be represented by larger scale Voronoi patches, while leaving regions that change significantly, in height for example, to be represented by the higher resolution patches.

The overall triangulation is broken at the boundaries between patches of differing levels, but these patches may be sewn together using the "divide and conquer" approach of [17].
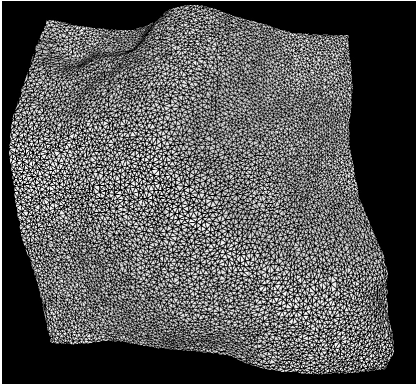
Fig. 4. Terrain model – level 0
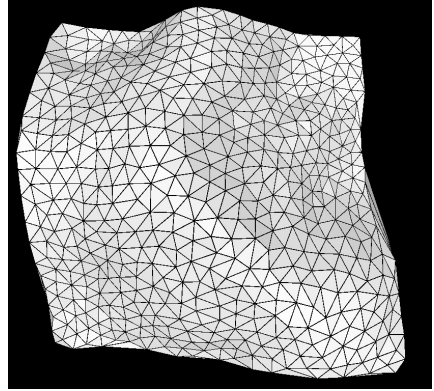
Fig. 5. Level 1 generators
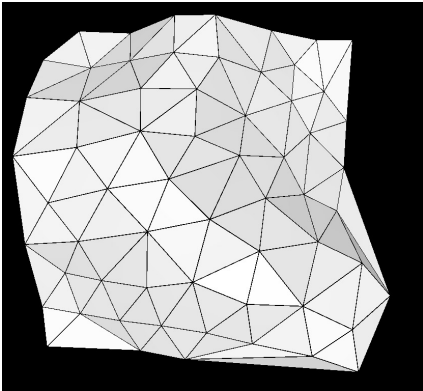
Fig. 6. Level 2 generators
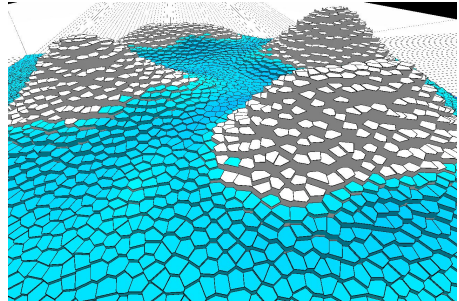
Fig. 7. Voronoi cells of some generators
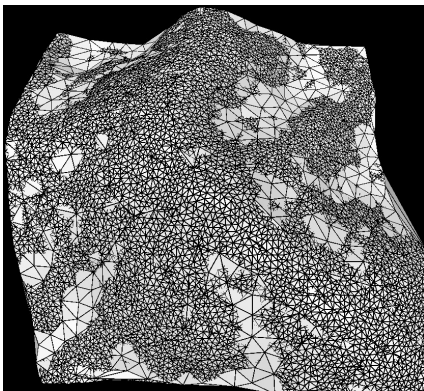
Fig. 8. Overlapping level 0 and level 1
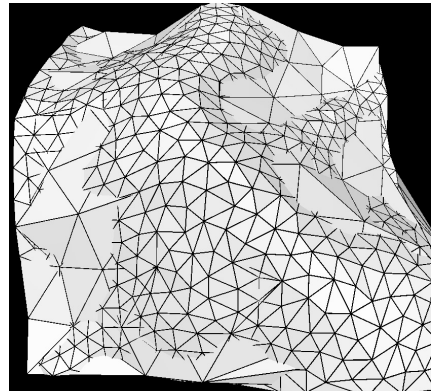
Fig. 9. Overlapping level 1 and level 2

5.3 Boundary and Image Generalization

In other problems, the boundary of the index cell may be used as a simplification of the boundary of the set of child nodes. The utility of this depends strongly on how the index nodes are positioned. Fig. 10 shows the result for random data and index generators.
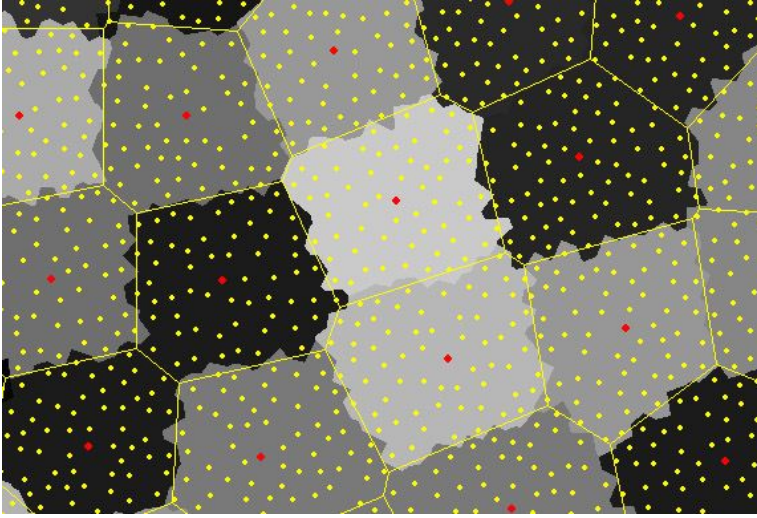
Fig. 10. Boundary approximation

It is also possible to adjust the index cells to the data: Fig. 11 shows a set of Lidar data for an urban setting. The intention is to model the buildings by approximating their boundaries with VC boundaries. This is achieved by adjusting the index generators' locations so as to maximise the differences between the average values calculated for the index generators (Fig. 12). The resulting "Voronoi City Model" [28] has vertical cell boundaries approximating the building walls. Similar methods may be used for generalizing images – see [10] and [25].
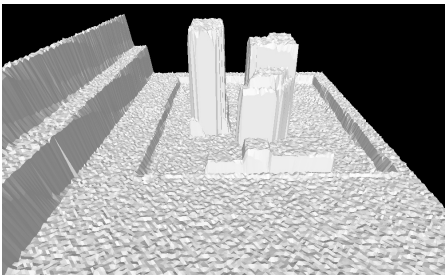
Fig. 11. Raw Lidar data
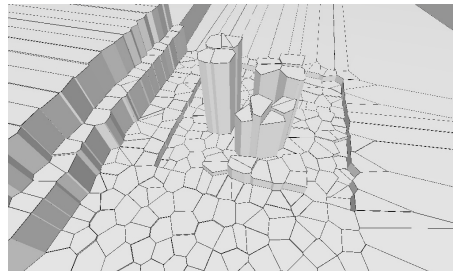
Fig. 12. Voronoi cell approximation

6   Conclusions

We believe that the simple form of Voronoi hierarchy described here is sufficiently useful to be of general interest as a straightforward extension of the simple Euclidean VD for points. There are certainly other applications waiting to be described. The methods described here are applicable to dynamic maps – where data points may be inserted and deleted at will, and they may be directly transferred to the VD on the sphere for global applications. The simple indexing structure is directly valid in higher dimensions, but the disc partitioning method used here depends on an edge-based structure (with data points at each end) to navigate across partitions. The appropriate data structure is less obvious in 3D and higher. The approach may also be applied to connected map structures, for example the Constrained Delaunay Triangulation, and the Line Segment Voronoi Diagram [6] as well as to kinetic structures ([16] and [6]).

Acknowledgements

References

1. Amenta, N., Choi, S. and Rote, G., (2003). Incremental constructions con BRIO. 19th ACM Symposium on Computational Geometry, San Diego
2. Aurenhammer, F., (1991), Voronoï diagrams - a survey of a fundamental geometric data structure. ACM Computing Surveys, v. 23, pp. 345-405
3. Boots, B. and Shiode, N., (2003). Recursive Voronoi diagrams. Environment and Planning B, v. 30, no. 1, pp. 113-124
4. Christaller, W., (1966) (translation), Central Places in Southern Germany. Prentice-Hall, NJ.
5. Dakowicz, M. and Gold, C. M. (2003), Extracting Meaningful Slopes from Terrain Contours , International Journal of Computational Geometry Applications, v. 13, pp. 339-357
6. Dakowicz, M. and Gold, C.M. (In Press). Structuring Kinetic Maps. In Proceedings of the Twelfth International Symposium on Spatial Data Handling, Vienna, July 2006
7. De Floriani, L., (1989). A pyramidal data structure for triangle-based surface description. Computer Graphics and Applications, v. 9, no. 2, pp. 67-78
8. Devillers, O., (1999). On deletion in Delaunay triangulation. Proc. 15th ACM Symp. Comp. Geom.., pp. 181-188
9. Devillers, O., (2002). The Delaunay Hierarchy. Int. J. Foundations of Computer Science, v. 13 no. 2, pp. 163-180
10. Dobashi, Y. Haga, T., Johan, H. and Nishita, T., (2002). A Method for Creating Mosaic Images Using Voronoi Diagrams," Proc. EUROGRAPHICS 2002, pp. 341-348
11. Gold, C. M. (2000), Chapter 4: An Algorithmic Approach to Marine GIS, In Marine and Coastal Geographical Information Systems, (Ed.: Wright, D. and Bartlett, D.), Taylor and Francis, London, England, pp.37-52

12. Gold, C. M., Charters, T. D. and Ramsden, J. (1977), Automated contour mapping using triangular element data structures and an interpolant over each triangular domain, In Proceedings: Sigraph '77. Computer Graphics, v. 11, (Ed.: George, J), San Francisco, USA, pp.170-175

13. Gold C. M. and Dakowicz M. (2005). The Crust and Skeleton – Applications in GIS. In: Proceedings, 2nd International Symposium on Voronoi Diagrams in Science and Engineering, Seoul, Korea, pp. 33-42

14. Gold, C. M. and Maydell, U. M. (1978), Triangulation and spatial ordering in computer cartography, In Proceedings, Canadian Cartographic Association third annual meeting, Vancouver, BC, Canada, pp.69-8l

15. Gold, C. M., Nantel, J. and Yang, W. (1996), Outside-in: an alternative approach to forest map digitizing, International Journal of Geographical Information Systems, v. 10, pp.291-310

16. Guibas, L., Mitchell, J.S.B. and Roos, T., (1991), Voronoï diagrams of moving points in the plane. In Proceedings, 17th. International Workshop on Graph Theoretic Concepts in Computer Science: Lecture Notes in Computer Science, v. 570, (Berlin: Springer-Verlag), pp. 113-125

17. Guibas, L. and Stolfi, J., (1985), Primitives for the manipulation of general subdivisions and the computation of Voronoï diagrams. Transactions on Graphics, v. 4, pp. 74-123

18. Haggett, P. (1979) Geography – A Modern Synthesis, 3rd ed. Harper and Row, NY, 627 p.

19. Luebke, D., Reddy, M., Cohen, J., Varshney, A., Watson, B. and Huebner, R. (2003). Level of Detail for 3D Graphics. Morgan Kaufmann, San Francisco, 390 p.

20. Lukatela, H., (2000). A Seamless Global Terrain Model in the Hipparchus System. International Conference on Discrete Global Grids, Santa Barbara. http://www.geodyssey.com/

21. Mucke, E.P., Saias, I. and Zhu, B., (1996). Fast randomized point location without preprocessing in two and three dimensional Delaunay triangulations. Proc. 12th ACM Symp. Comp. Geom., pp. 274-283

22. Okabe, A., Boots, B., Sugihara, K. and Chiu, S.N., (2000). Spatial Tessellations - Concepts and Applications of Voronoi Diagrams (2nd edition). John Wiley and Sons, Chichester, 671p.

23. Okabe, A. and Sadahiro, Y., (1996). An illusion of spatial hierarchy: spatial hierarchy in a random configuration. Environment and Planning A, v. 28, pp. 1533-1552

24. Pulo, K.J., (2001). Recursive space decomposition in force-directed graph drawing algorithms. Proceedings, Australian Symposium on Information Visualization, Sydney, December, v. 9, pp. 95-102

25. Schussman, S., Bertram, M., Hamann, B. and Joy, K.I., (2000). Hierarchical data representations based on planar Voronoi diagrams. In van Liere, R., Hermann, I., and Ribarsky, W., eds., Proceedings of VisSym '00 -- The Joint Eurographics and IEEE TVCG Conference on Visualization, Amsterdam, The Netherlands, May 2000, 63-72

26. Sibson, R., (1981), A brief description of natural neighbour interpolation. In), Interpreting Multivariate Data, edited by Barnett, V. (New York: John Wiley and Sons), pp. 21-36

27. Swets, D.L. and Weng, J., (1999). Hierarchical discriminant analysis for image retrieval. IEEE Trans. PAMI, v. 21, no. 5, pp. 386-401

28. Tse, R., Gold, C.M. and Kidner, D. (2006) A New Approach to Urban Modelling Based on LIDAR. Proceedings, WSCG 2006, Plzen, Czech Republic, http://wscg.zcu.cz/WSCG2006/contents.htm

Characterising Meanders Qualitatively

Björn Gottfried

Artificial Intelligence Group, TZI
University of Bremen, Germany
`bg@tzi.de`

Abstract. In this paper, a qualitative shape representation is described for the purpose of characterising linear geographical and artificial objects. In particular, we focus on the curve progression telling us how objects spread across the landscape. For instance, sinuosities of rivers provide important information about imperilled locations in the case of flood waters. However, precise geometrical descriptions are overdetermined and frequently difficult or sometimes impossible to obtain. By contrast, we introduce a concept which allows curves to be classified on the basis of a qualitative representation that defines properties of linear objects, which derive from how segments of objects are located relative to other segments, arriving at conclusions such as how twisty a curve is. Especially, the new method can be applied if only coarse information is available and even then if objects are given incompletely.

1 Introduction

In geographical information systems topological relations between geographical objects are useful [3]. It may, for example, be crucial to a particular query that there is a forest and that there is a river which is not connected to the forest; but it does not matter at all what the boundary of the forest looks like, or how far the river is from the forest provided that they are not in contact; such geometrical relationships are not important when we are interested in those cases where only the given topological relationships hold. Precise correspondences would retrieve fewer results than there are actually in the database. But sometimes topological relationships do not sufficiently characterise the query. For example, it might be crucial to take the curve progression of the river into account. How can different meanders of rivers, such as those in Fig. 1, be described? Similarly, there are many kinds of geographical and artificial objects for which curvature information is important, including among others contour lines in topographic maps, coastlines, borders of countries and other regions, transportation networks, such as roads and railways, irrigation networks, and sewer systems.

Modern geographic information systems demand concepts that provide means which are closely related to how people deal with spatial information, since it is desirable that user interfaces become more natural. Capturing commonsense knowledge about objects, qualitative representations lend themselves to provide
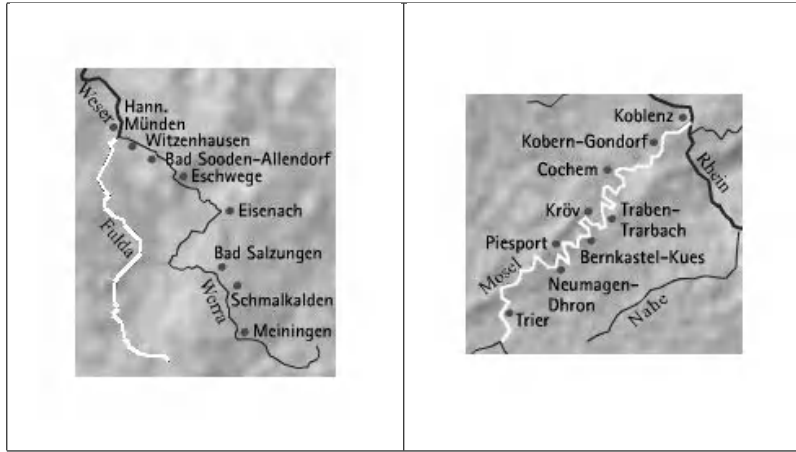
Fig. 1. Rivers differing in their curvatures

appropriate representational concepts [2]. Especially, qualitative boundary based approaches which are related to linear objects record features of boundaries while walking along them. Those which have been devised most recently include [6,11,14,10]. [11] base their approach on a discretisation of tangent bearing and curvature, considering the rate of change of curvature, and they give a set of tokens from which higher-level tokens can be derived. By contrast, all other approaches rely on polygonal approximations of the underlying boundaries. There are several reasons for this. Most importantly, polygons are at the core of geographical information systems; the discrete space used in computer representations inherently deals with polygons; depending on the application at hand objects can be approximated at a number of different granularity levels when using polygons; frequently, polygons are directly given (when recording the path of a navigating system the position of which is measured at intervals, for example); polygons provide concise representations of even highly complex objects, forming an appropriate basis for the qualitative characterisation of shapes from both the local and global points of view. Considering linear objects such as rivers, we are actually concerned with open polygons, i.e. simple polylines. Below, we simply speak of polygons, denoting both simple polylines and simple polygons.

This paper addresses the problem of how to represent smooth curves by polygons in such a way that the curve progression of linear objects is made explicit. Improving user interfaces in such a manner that distinctions made by the system are easily comprehensible by humans, a qualitative representation is recommended. As such section 2 introduces the qualitative representation on which we shall define qualitative features for the purpose of characterising meanders of linear objects in section 3. A number of examples in section 4 illustrate
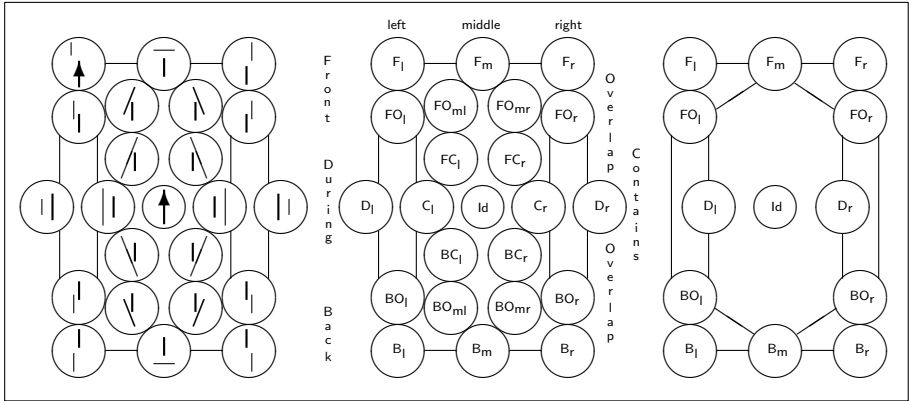
Fig. 2. The conceptual neighbourhood graphs of $\mathcal{BA}_{23}$ (middle) and $\mathcal{BA}_{13}$ (right)

how those qualitative features apply to linear objects. An extended concept of neighbourhood relations is introduced in section 5, which forms the basis for the introduced concepts and which generalises to other qualitative representations. We conclude in section 6, suggest topics for future research, and summarise in section 7.

2 A Qualitative Feature Scheme

In this section we will summarise previous work on a qualitative feature scheme on which the current approach is based. This feature scheme consists of a number of relations which describe arrangements between line segments in two dimensions. Such relations have been referred to as *bipartite arrangements*, $\mathcal{BA}$ for short [7]. Fig. 2 shows these relations.

If one line of a polygon is made the basis, the position of every other line can be described relative to it, using the relations of $\mathcal{BA}_{23}$ which are shown on the left hand side of Fig. 2. In this way, the qualitative context of a polygonal line, x, is considered, and for a polygon with n lines we obtain a list of relations to which we refer to as the *course* of reference segment x, in short $C(x)$:

$$C(x) \equiv (x_{y_1}, .., x_{y_n}), \; x_{y_i} \in \mathcal{BA}_{23}, i = 1, .., n \tag{1}$$

with x_{y_1} meaning that line segment y_1 is described with respect to reference segment x.[1] In particular, it holds that $x_x = \mathsf{Id}$. A subset $\mathcal{BA}_{13} \subset \mathcal{BA}_{23}$, shown on the right hand side of Fig. 2, provides a set of atomic relations in the sense that all other relations can be obtained by combining $\mathcal{BA}_{13}$ relations (e.g. $C_l = BO_l D_l FO_l$).

---

[1] Instead of the common infix notation we use indices in order to be able to list many $\mathcal{BA}_{23}$ relations in a compact way.

The entire range of relations (accordingly to $\mathcal{BA}_{13}$) where $\mathsf{C(x)}$ runs along is called its scope, and the number of different $\mathcal{BA}_{13}$ relations involved is called the extent, $\eta(\mathsf{C(x)})$ for short. The shortest extent is 0 in which case there is no other line segment than the reference segment x itself; the largest possible extent is 12 (which is $|\mathcal{BA}_{13} \setminus \{\mathsf{Id}\}|$) in which case the course runs completely around x. Scopes of such courses are also referred to as universal scopes since all relations of $\mathcal{BA}_{23}$ are realisable within these scopes. Eventually, there exist different scopes which have the same extent, for instance, one course may go from $\mathsf{F_l}$ to $\mathsf{F_r}$ and another one from $\mathsf{B_l}$ to $\mathsf{B_r}$; both courses, however, have an extent of 3.

While [9] distinguishes local and global features, here we simplify matters by representing both local and global features by bipartite arrangements. That is, we conceive each pair of line segments in a simple polygon as to be free of intersections, meaning that especially adjacent line segments have no point in common. For this purpose, we stipulate that each point which connects two line segments belongs to the first line segment in an oriented polygon. This enables us to get by with $\mathcal{BA}_{23}$ relations, as depicted in Fig. 2.

Having a polygon with n line segments there exist n courses, each one comprising n bipartite relations, i.e. such a feature scheme comprises a total of n^2 relations. Writing down all courses, one below the other, for a polygon with six lines the following matrix is obtained (compare the polygon on the left hand side of Fig. 3, and note that singular relations, such as between x and y, are dealt with in accordance to [8]):

$$
\begin{array}{llllll:llllll}
\mathsf{Id} & \mathsf{u_v} & \mathsf{u_w} & \mathsf{u_x} & \mathsf{u_y} & \mathsf{u_z} & \mathsf{Id} & \mathsf{D_r} & \mathsf{D_r} & \mathsf{BO_r} & \mathsf{B_r} & \mathsf{B_m} \\
\mathsf{v_u} & \mathsf{Id} & \mathsf{v_w} & \mathsf{v_x} & \mathsf{v_y} & \mathsf{v_z} & \mathsf{D_r} & \mathsf{Id} & \mathsf{F_l} & \mathsf{F_m} & \mathsf{F_r} & \mathsf{C_r} \\
\mathsf{w_u} & \mathsf{w_v} & \mathsf{Id} & \mathsf{w_x} & \mathsf{w_y} & \mathsf{w_z} & \mathsf{B_m} & \mathsf{B_l} & \mathsf{Id} & \mathsf{F_r} & \mathsf{C_r} & \mathsf{B_r} \\
\mathsf{x_u} & \mathsf{x_v} & \mathsf{x_w} & \mathsf{Id} & \mathsf{x_y} & \mathsf{x_z} & \mathsf{B_r} & \mathsf{B_r} & \mathsf{B_r} & \mathsf{Id} & \mathsf{D_r} & \mathsf{D_r} \\
\mathsf{y_u} & \mathsf{y_v} & \mathsf{y_w} & \mathsf{y_x} & \mathsf{Id} & \mathsf{y_z} & \mathsf{F_r} & \mathsf{FO_r} & \mathsf{D_r} & \mathsf{D_r} & \mathsf{Id} & \mathsf{F_r} \\
\mathsf{z_u} & \mathsf{z_v} & \mathsf{z_w} & \mathsf{z_x} & \mathsf{z_y} & \mathsf{Id} & \mathsf{D_r} & \mathsf{D_r} & \mathsf{BO_r} & \mathsf{B_r} & \mathsf{B_r} & \mathsf{Id}
\end{array}
$$

3   Towards the Characterisation of Meanders

A bend in a river is referred to as a meander, emerging from a stream flowing through a wide valley or flat plain thereby tending to form a meandering stream course as it alternatively erodes and deposites sediments along its course. The result is a snaking pattern as the stream meanders back and forth across its floodplain.

In the following, we shall generalise this concept, using the term meander broadly for linear objects which are shaped by several twists and turns, and we will characterise winding courses, i.e. how a polygon develops from the point of view of one of its line segments, to which we will refer to as the reference segment. The simplest way of characterising a course consists in referring to its $\mathcal{BA}_{23}$ relations. But what does a list of many $\mathcal{BA}_{23}$ relations tell us? (Have a look at the matrix above.) We shall rather identify more abstract features

| $\rho(\mathsf{C}(\mathsf{u})) = \mathsf{r}$ | $\rho(\mathsf{C}(\mathsf{u})) = \mathsf{rl}$ |
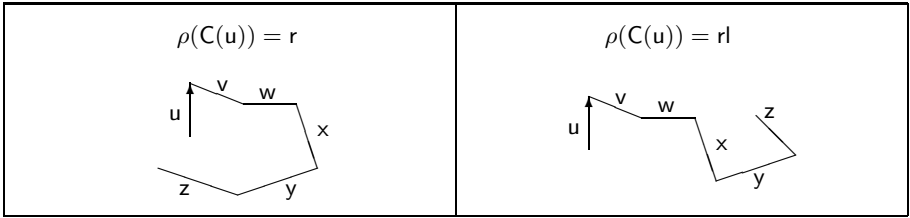
Fig. 3. Circulation directions around a reference line

which derive from this matrix, and which concisely describe a course, such as whether it runs around the reference segment clockwise or anticlockwise, whether it comprises reversals, and towards which directions the course runs along. In this way we will characterise differently shaped meanders.

Formally, at the finest level sequences of $\mathcal{BA}_{23}$ relations are considered (equation 1), and at a coarser level sequences of generalised directions are considered (section 3.1). In the latter case atomic $\mathcal{BA}_{23}$ relations are put together in order to form yet a coarser description than $\mathcal{BA}_{23}$ relations. Following a list of $\mathcal{BA}_{23}$ relations or generalised directions a course either orbits clockwise or anticlockwise around the reference segment (section 3.2). Taking the first derivative of a course amounts to consider changes in direction to which we refer to as reversals (section 3.3). Positions at which such changes occur are analogous to local minima and maxima of functions and they are referred to as inflection segments (section 3.4).

3.1   Generalised Directions

Suppose a polygon is described with respect to one of its line segments, x. As far as we are only interested in the overall direction towards which the course of x runs along, different $\mathcal{BA}_{23}$ relations can be put together which make up the same direction at a coarser granularity level. Such coarser directions form subsets of $\mathcal{BA}_{23}$. Leaving out the identity relation, there are $2^{22} = 4194304$ such subsets. Some of them are particularly useful as coarse directions; for example, a course may run somewhere left of the reference segment:

$$\mathsf{C}(\mathsf{x}) = \mathsf{l} \equiv \forall_{\mathsf{y} \neq \mathsf{x}} : \mathsf{x}_\mathsf{y} \in \{\mathsf{F}_\mathsf{l}, \mathsf{FO}_\mathsf{l}, \mathsf{D}_\mathsf{l}, \mathsf{C}_\mathsf{l}, \mathsf{BO}_\mathsf{l}, \mathsf{B}_\mathsf{l}\} \qquad (2)$$

These six relations combine to $2^6 = 64$ different sets which form the basis of several different courses, all of them running left of the reference segment. $\mathsf{C}(\mathsf{x}) = \mathsf{r}$ (right of), $\mathsf{C}(\mathsf{x}) = \mathsf{F}$ (in front of), $\mathsf{C}(\mathsf{x}) = \mathsf{B}$ (back of), etc., can be defined in the same way. By this means, complex concepts are based on fewer relations than when taking individual $\mathcal{BA}_{23}$ relations. Altogether, we introduce generalised directions as to be defined over sets of atomic $\mathcal{BA}_{13}$ relations:
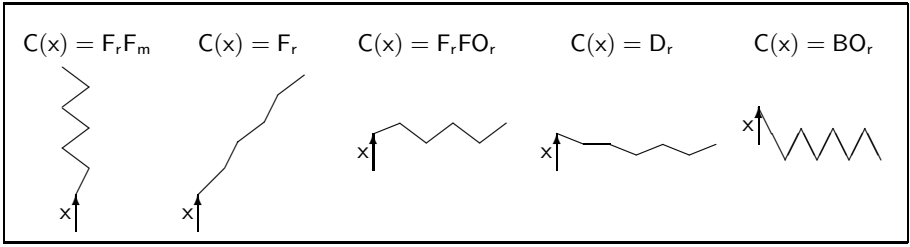
Fig. 4. Five polygons with three of them leading in a single direction (F_r, D_r, and BO_r) with respect to line segment x, while two of them are almost limited to one direction

Definition 1 (Generalised direction)
x *is line segment of a polygon and* $C(x)$ *is its course. A generalised direction of* $C(x)$, $\omega(C(x))$, *is a set* $M \in \mathcal{P}(\mathcal{BA}_{13})$.

For instance, the right hand side of Fig. 3 shows a polygon which can be described as to run (somehow) right of x, thus $\omega(C(x)) = r$; but this is not entirely true for the polygon on the left hand side.

How do generalised directions of different reference segments combine? Taking into account simultaneously the directions of all courses, properties such as the convexity of closed polygons can be derived. For an arbitrary closed polygon, P, which is oriented anticlockwise it holds

Proposition 1
$convex(P) \Leftrightarrow \forall_{x \in P} : \omega(C(x)) = l$

Proof: reduction on the triangle orientation: for three adjacent points of a convex polygon P which is oriented anticlockwise, their triangle orientation is anticlockwise. Accordingly, the first two points define a reference segment, x, while the third point lies left of this line, as does each following point. As a consequence, each line segment, y, defined by two such following points, lies left of x. As x can be defined by two arbitrary points of P, for each course it holds that all its line segments lie left of the reference segment, and it holds that $\forall_{x \in P} : \omega(C(x)) = l$.

Conversely, if $\forall_{x \in P} : \omega(C(x)) = l$, then for each pair of lines it holds that the primary segment lies left of the reference segment. The same holds for the end-points of such pairs of line segments, which define triangle orientations which are all oriented anticlockwise, indicating that the polygon is convex.   □

While both $\mathcal{BA}_{23}$ relations and generalised directions describe how parts of a polygon are located relative to other parts of that same polygon, considering meanders we are interested in how these relative locations change when following the courses of the reference segments.
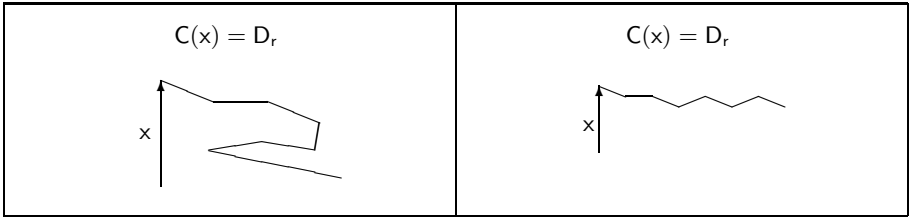
Fig. 5. Two polygons describing the same course with respect to line segment x

3.2   Circulation Direction

Irrespective of generalised directions introduced in the previous section, it can be distinguished whether a course circulates *left* around or *right* around the reference segment:

Definition 2 (Circulation direction)
x *is line segment of a polygon. The circulation between two neighbouring line segments,* y *and* y′*, is left of* x*, i.e.* $\rho(x_y, x_{y'}) = l$*, if the path from the first relation* x_y *to the second relation* $x_{y'}$ *runs anticlockwise around* x*; otherwise, it is right of* x*, in short* r*. If the direction does not change it holds that* $\rho(x_y, x_{y'}) = \varepsilon$*, as it does if* y *and* y′ *are not adjacent line segments.*

$\rho(x_y, x_z) = \varepsilon$ denotes the case that it cannot be determined from the point of view of x whether the circulation direction from y to z is clockwise or anticlockwise with respect to x. This indeterminacy can either be compensated by another reference segment, or y and z are not adjacent in which case their circulation direction cannot be derived from y and z alone.

Fig. 3 shows two examples. On the left hand side the course of x circulates entirely right of x. On the right hand side it also runs right of x, but it then turns back and as a consequence runs left of x. Note how this is different from the generalised direction $\omega(C(x)) = r$.

Further examples clarify the meaning of the circulation direction: $\rho(F_l FO_l) = l$, $\rho(FO_l F_l) = r$, $\rho(F_l F_m) = r$, $\rho(F_l FO_{mr} BO_r) = rr$, $\rho(F_l FO_{mr} FO_r FO_r D_r) = rlr$, and $\rho(FO_l F_l ld D_r BO_r) = rr$. Equal neighbouring directions can be omitted in order to obtain only the changes, i.e. changes between left and right, or anticlockwise and clockwise, respectively. It then holds that the number of changes between left and right of $\rho(C(x))$ is less than or equal to the length of the course.

Taking each of the line segments of the polygon on the left hand side of Fig. 3 as a reference segment, the circulation direction is always either r (clockwise) or ε (indeterminate). But the circulation directions of different reference segments of the same polygon are not always equal, as demonstrated by the polygon on the right hand side of Fig. 3. Here, $\rho(u_w, u_x) = \rho(D_r, BO_r) = r$, whereas $\rho(z_w, z_x) = \rho(F_l, FO_l) = l$. The circulation direction depends on the position of the reference segment with respect to the other line segments of the same

$C(x) = F_rFO_rBO_rB_mBO_lFO_lF_lF_mFO_rBO_rB_mBO_l$ $C(x) = F_rFO_rD_rBO_rB_r$
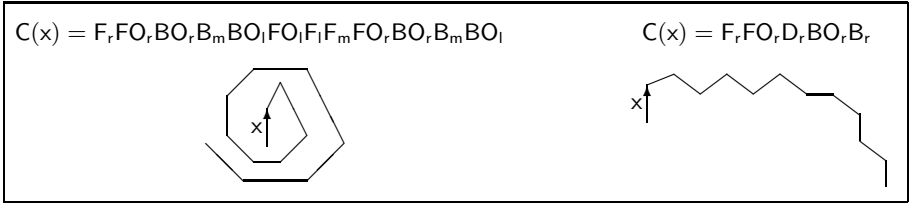
Fig. 6. Polygons without reversals from the point of view of x

polygon[2]. Changing the circulation direction is what we will focus on in the next section.

3.3   Reversals

As soon as a course changes its circulation direction from left to right or right to left, the course includes a reversal, as indicated by its first derivative. For example, $C(x) = F_rFO_rD_rFO_rF_r$ comprises a reversal since the circulation direction is changed after D_r with the second FO_r relation.

Definition 3 (Reversal)
x is a line segment of a polygon and $C(x)$ is its course. If $C(x)$ comprises two sections which circulate in different directions around x, $C(x)$ contains a reversal, and it holds $\varrho(C(x))$.

Examples of courses without reversals from the viewpoint of line segment x are depicted in Figs. 4 and 6. Examples of courses with exactly one reversal with respect to x are depicted in Fig. 7. As the left hand side of Fig. 5 shows, all reversals of a polygon cannot always be deduced from single courses. It is rather necessary to test all courses of a polygon in order to determine whether there are reversals in the polygon.

   An algorithm which determines reversals of a course $C(x)$ derives from

Proposition 2

$$\varrho(C(x)) \Leftrightarrow \exists_{u \neq x, v \neq x, w \neq x}(u < v < w \wedge x_u \neq x_v \wedge x_u = x_w \wedge \eta(C(x, u, w)) < 12) \quad (3)$$

Proof: u, v, and w are line segments which are unequal to x, and $u < v < w$ ensures an ordering among them, i.e. there is a section, s, of the course $C(x)$ which consists of at least three line segments. As it holds that $x_u \neq x_v$ there is at least one change in direction in s while $x_u = x_w$ ensures that the first direction in s is again satisfied at the end of s. Let $C(x, u, w)$ refer to the section s of $C(x)$ which starts with line segment u and ends with line segment w. Then, the course of s must have been turned backwards in the meantime since it holds that the extent of s, i.e. $\eta(C(x, u, w))$, is less than the extent of the universal scope.   □

---

[2] It depends also on its orientation relative to the other line segments. However, the current work solely relies on relative positions and analyses their relationships.

$$C(x) = F_rFO_rD_rFO_rF_r$$

$$C(x) = D_rBO_rB_rBO_rD_r$$
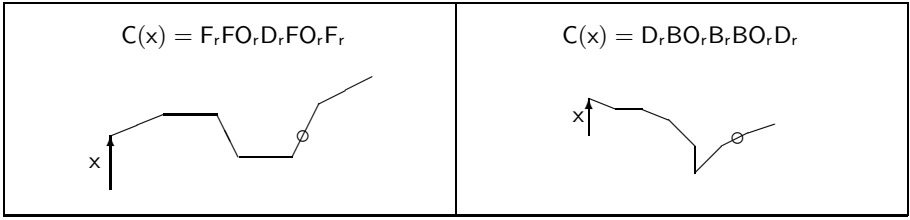
Fig. 7. Polygons with reversals; note that relations are put together when they are equal and adjacent

3.4   Inflection Segments

Determining line segments at which reversals occur enables reversals to get distinguished according to $\mathcal{BA}_{23}$, that is, by describing inflection segments regarding their $\mathcal{BA}_{23}$ relation. Such line segments are defined as follows:

Definition 4 (Inflection segment)
x *is a line segment of a polygon,* $C(x)$ *is its course, and it holds that* $\varrho(C(x))$. ′y *denotes the predecessor of* y *and* y′ *its successor. Then it either holds that* $\rho(x′_y, x_y, x_{y'}) = lr$ *or* $\rho(x′_y, x_y, x_{y'}) = rl$. *The first line segment after that reversal, namely* y′, *is called the inflection segment.*

While $\rho(x′_y, x_y) = l$ and $\rho(x_y, x_{y'}) = r$, y′ denotes the inflection segment which runs around x the other way round than that line segment which is immediately before y′.

Consider, for example, Fig. 7. Inflection segments are marked by circles. Note that these segments are inflection segments from the point of view of reference segment x. There are probably other inflection segments when considering other reference segments than x. To identify each inflection segment of a polygon it is necessary to analyse all courses, for the same reason as it is necessary to analyse all courses in order to find all reversals.

4   A Case Study: Differentiating Meanders

To illustrate the method, we shall compare the similarity of a number of German streams. What we want to know is whether their meanders can be distinguished in determining their reversals, and how the streams relate when ranking them regarding the number of reversals they comprise. For instance, while the Mosel has many twists and turns, the Fulda is less curved but comprises also some windings. Does the concept of reversals as introduced above account for those distinctions?

In order to obtain polygons the rivers have been approximated by the polygonal approximation algorithm of [12]. Using 1:5,000,000 small scale maps[3], the

---

[3] Note that the images shown are scaled up and down slightly to fit in the layout.
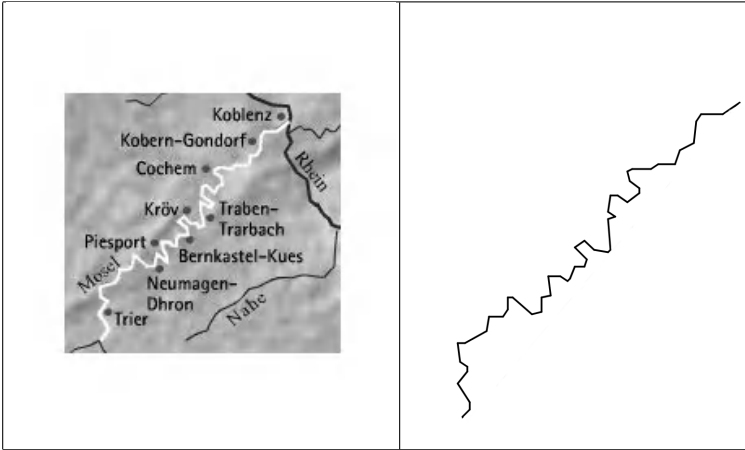
Fig. 8. The river Mosel approximated and simplified by a polygon

Mosel has been approximated by 51 line segments while the Fulda has been approximated by 22 line segments, allowing for the same error in both cases, namely not exceeding a cell in a raster representation. The Fulda comprises 8 reversals (on average, 0.36 per line segment) and the Mosel comprises 94 reversals (on average, 1.84 reversals per line segment). This shows that the method identifies the Mosel as to be five times more curved than the Fulda. But is this still a qualitative difference? It is at least based on qualitative distinctions, namely at the level of $\mathcal{BA}$ relations where we are rather concerned with differences in kind than of measurement. However, counting the number of reversals we obviously turn to some quantitative measurement. On the other hand, the number of reversals involved is in itself without meaning (nor is the normalisation). Instead it is the ordering of those quantities which matters, allowing the streams to get ordered, this ordering being of relative (and hence qualitative) nature.

For the Fulda in Fig. 9 we have printed the inflection segments in pale-grey, in order to visualise how inflection segments appear at those parts of a polygon where indeed large turns are made. Note that such locations can be identified on the basis of qualitative relations alone without requiring precise computations. But instead of precise inflection points of zero curvature we obtain extended portions of the polygon at which the curvature changes.

In Figs. 10 and 11 there are fourteen of the largest streams in Germany and their polygonal approximations. They have been ordered regarding their meanders, i.e. for each stream the number of reversals has been determined and normalised with the number of line segments involved. The polygonal approximations have been printed enlarged in order to allow them to be compared more easily. The first stream with the fewest reversals is the river Havel, that stream with the most reversals is the river Mosel. The shown ordering can in fact be comprehended very well: beginning with the Havel the streams getting more and
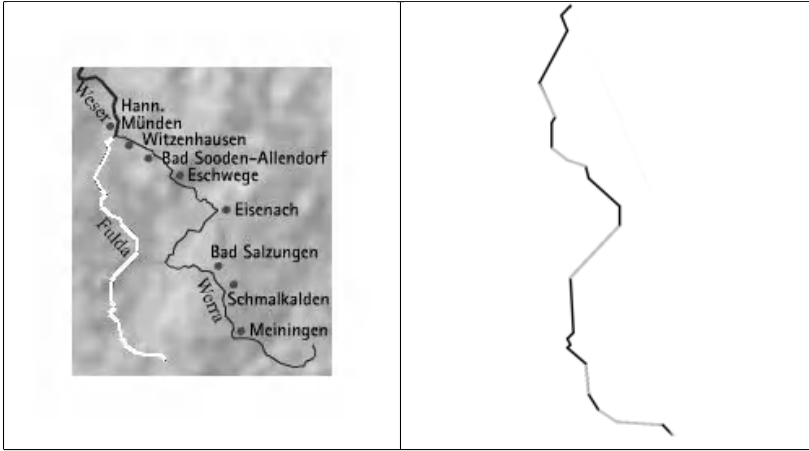
Fig. 9. The river Fulda approximated and simplified by a polygon; inflection segments are printed in pale-grey

more complex. It shows that both little meanders like in the case of the Mosel and larger meanders which extend larger portions of the stream like in the case of the Elbe and the Fulda are equally identified by this method.

5   Extending the Concept of Neighbourhood Relations

According to [5] two relations are conceptual neighbours if one of the objects involved can be continuously enlarged, shortened, or moved, so that one relation transforms into the other one without passing any third relation. We shall extend this concept of neighbouring relations by including the circulation direction. That is, while transforming one relation into another one we distinguish whether the line segment which undergoes some deformation circulates around the reference segment clockwise or anticlockwise during this deformation step. This changes nothing about the notion of conceptual neighbourhoods but allows a finer distinction to be made, either during a transformation step or while describing the course of the reference segment.

Definition 5 (Directed neighbourhood relation)
x *is a line segment of a polygon and* $C(x)$ *is its course. Then, two relations,* x_y *and* $x_{y'}$, *in* $C(x)$ *are left-directed neighbours, if they can be directly transformed into one another by continuously deforming (i.e. shortening, lengthening, or moving)* y *by circulating left around* x *without passing any third relation. Right-directed neighbours are defined accordingly.*

This definition adapts to all those qualitative representations which define conceptual neighbourhood graphs, such as [1], [4], and [13]. For the time domain, this amounts to distinguish a transformation towards the past and towards the

Havel	1; 0.10	Inn	2; 0.14

Elbe	3; 0.231	Oder-Neisse	7; 0.233

Leine	8; 0.36	Isa	9; 0.37

Fulda	10; 0.45	Hochrhein	13; 0.48
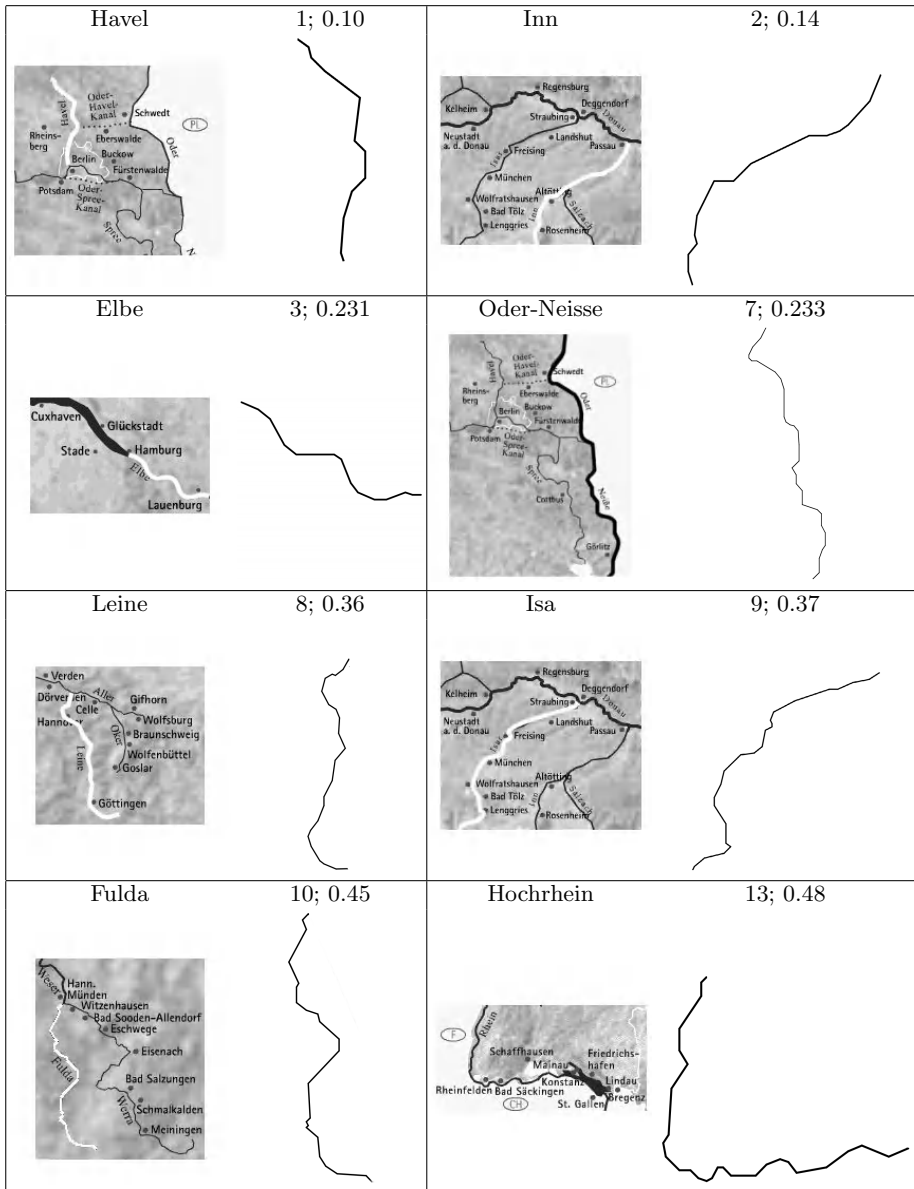
Fig. 10. German streams and their reversals: shown is the absolute number of reversals (first number) and the number of reversals on average per segment (second number)

future; in the domain of regions this allows for the differentiation whether two moving regions merge or separate. Directed neighbourhood relations are uniquely defined for these domains. Informally, this can be read off the neighbourhood

Donau	17; 0.56	Saale	24; 0.68

Weser	20; 0.76	Spree	34; 0.80

Neckar	70; 1.11	Mosel	119; 2.33
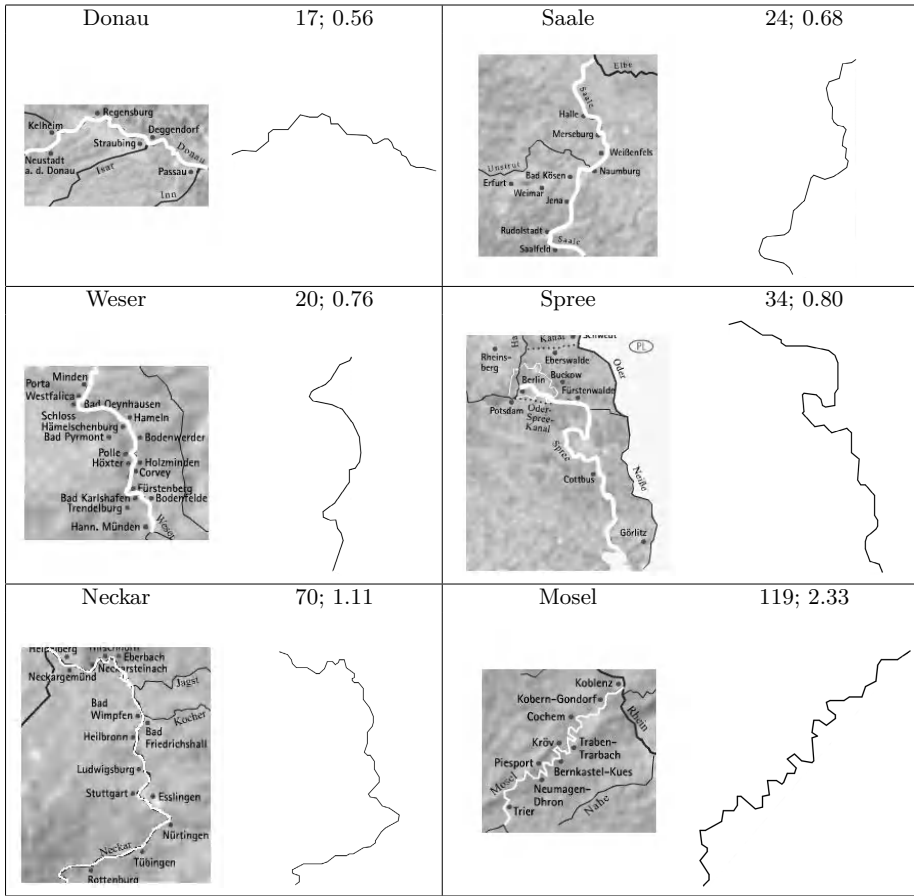
Fig. 11. German streams and their reversals: shown is the absolute number of reversals (first number) and the number of reversals on average per segment (second number)

graphs. In fact, this differentiation is partly implied in the ordering of relations in the time domain, since Allen distinguishes relations such as before and after.

In our case directed neighbourhood relations are necessary in order to distinguish two circulation directions, and as a consequence, in order to be able to recognise reversals as well as inflection segments. But directed neighbourhood relations are yet more fundamental. Qualitative representations frequently define sets of binary [4] or sometimes sets of ternary relations [15] among a number of two or three objects, respectively. But here we are concerned with single objects which are to be characterised qualitatively. The idea behind the qualitative characterisation of single objects relies on the partitioning of objects into parts in order to describe relations among those parts. However, this requires to take into account the ordering of qualitative relations between parts accordingly to how those parts define some object. For instance, concepts such as the circulation direction can

then be defined, and this makes the distinction between figure and ground possible which requires the orientation of an object's boundary to be determined. In this sense, the concept of directed neighbourhood relations is essential as an additional means for determining specific features among objects or parts.

6   Discussion

Having defined generalised directions and reversals, polygons can be classified accordingly. A polygon, for instance, which comprises reversals for neither of its courses would be classified as to be straight. Conversely, the more reversals a polygon has the more curved it is. Describing reversals in combination with generalised directions allow yet finer classifications to be made. Accordingly, sections without reversals can be described by generalised directions. Similarly, inflection segments can be referred to either by $\mathcal{BA}_{23}$ relations or generalised directions.

An advantage of this approach is its ability to cope with indeterminacy: Incompleteness arises if there are gaps in the polygon; nonetheless, all relations between the discernible segments can be used in the described way; this allows to describe the incomplete shape at least partly. Imprecision is dealt with implicitly by the coarseness of $\mathcal{BA}_{23}$ relations. Overall, the strength of these techniques is that the approach is particularly well suited for dealing with coarse shape information. Once precise shape details matter it may be appropriate to apply quantitative methods instead. However, even if information about a stream has been acquired incompletely or imprecisely, it is possible to determine reversals. In order to analyse the robustness of the method, we currently investigate to what extent the chosen precision of the polygonal approximation algorithm affects the number of reversals which are detected.

This generalises to the question of how qualitative features do depend on the granularity level at which polygons approximate linear objects. It is worth investigating to what extent the number of reversals, for example, depends on the chosen granularity level. Whenever there is a part of a course which crosses a singularity this will be the case regardless of whether the course is approximated with many or only with a few line segments. For instance, a course might run first of all left of some reference segment and afterwards in the front of that segment. In this case the course must contain the $\mathsf{FO_l}$ relation somewhere, and this course must contain another $\mathsf{FO_l}$ relation later if it contains a reversal. Dealing with a reversal which is even visible at a coarse approximation level, this same reversal will also be perceivable at each finer granularity level. But things are unfortunately not always that simple: a quite small reversal might be probably smoothed away at coarser granularity levels. However, objects should be approximated at a level of abstraction so that distinctions can be made which are crucial for the application at hand. Otherwise, if it is not possible to find a granularity level on the basis of the given application, objects which are to be compared should at least be approximated at the same granularity level.

The course of a reference segment determines directions at which other line segments are situated relative to the reference segment. However, further

investigations concern the involvement of line segment orientations and their relations to positional relations, as they have been used in the current work. In particular, it is of interest which properties there are that cannot be defined on positional relations alone. Moreover, the question arises whether the method can be refined so as to make it possible to distinguish small Mosel-like meanders, and large Fulda-like meanders. Eventually, it is of interest to thoroughly analyse how courses of different reference segments of the same polygon are related. For this purpose it might be useful to use the algebraic properties of the $\mathcal{BA}$ relations, as they have been defined in [7].

7   Summary

A qualitative shape representation has been described which allows linear objects to be characterised. This representation is based on arrangements of line segments in the two-dimensional plane, as they especially occur in polygons. The qualitative relations used distinguish both a left-right and a front-back dichotomy, including several intermediate relations such as *front-middle* and *front-overlap left*, for instance. Furthermore, an ordering of the relations is defined in accordance to the ordering of line segments in polygons.

A number of properties of linear objects have been defined on the basis of these relations for the purpose of distinguishing different grades of meanders of linear objects. Applying this method to several German streams shows that they can be ordered regarding their meanders, so that less complex meanders can be distinguished from more complex ones, resulting in an ordering which can be intuitively comprehended. In particular, small meanders and large meanders are both allowed for.

References

1. J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.
2. A. G. Cohn and S. M. Hazarika. Qualitative Spatial Representation and Reasoning: An Overview. *Fundamenta Informaticae*, 43:2–32, 2001.
3. M. Egenhofer. Query Processsing in Spatial-Query-by-Sketch. *Journal of Visual Languages and Computing*, 8:403–424, 1997.
4. M. Egenhofer and R. Franzosa. Point-set topological spatial relations. *International Journal of Geographical Information Systems*, 5(2):161–174, 1991.
5. C. Freksa. Temporal reasoning based on semi-intervals. *Artificial Intelligence*, 94:199–227, 1992.
6. B. Gottfried. Tripartite Line Tracks, Qualitative Curvature Information. In W. Kuhn, M. Worboys, and S. Timpf, editors, *Spatial Information Theory: Foundations of Geographic Information Science*, LNCS (2825), pages 101–117. Springer, 2003.
7. B. Gottfried. Reasoning about Intervals in Two Dimensions. In W. Thissen, P. Wieringa, M. Pantic, and M. Ludema, editors, *IEEE Int. Conf. on Systems, Man and Cybernetics*, pages 5324–5332. IEEE Press, 2004.

8. B. Gottfried. Singularities in Qualitative Reasoning. In B. Buchberger and J. Campbell, editors, *7th Int. Conf. on Artificial Intelligence and Symbolic Computation*, LNAI (3249), pages 276–280, Austria, 2004. Springer.

9. B. Gottfried. Global Feature Schemes for Qualitative Shape Descriptions. In H. W. Guesgen, C. Freksa, and G. Ligozat, editors, *IJCAI-05 WS on spatial and temporal reasoning*, 2005.

10. E. Jungert. Symbolic spatial reasoning on object shapes for qualitative matching. In A. U. Frank and L. Campari, editors, *COSIT 1993, Spatial Information Theory: A Theoretical Basis for GIS.*, LNCS 716, pages 444–462. Springer, 1993.

11. R. C. Meathrel and A. Galton. A Hierarchy of Boundary-based Shape Descriptors. In B. Nebel, editor, *IJCAI 2001*, pages 1359–1364, Seattle, Washington, USA, 2001. Morgan Kaufmann.

12. D. A. Mitzias and B. G. Mertzios. Shape Recognition With A Neural Classifier Based On A Fast Polygonal Approximation Technique. *Patt. Rec.*, 27:627–637, 1994.

13. D. A. Randell, Z. Cui, and A. G. Cohn. A spatial logic based on regions and connection. In *Proc 3rd Int. Conf. on Knowledge Representation and Reasoning*, pages 165–176, San Mateo, 1992. Morgan Kaufman.

14. C. Schlieder. Qualitative Shape Representation. In P. Burrough and A. M. Frank, editors, *Geographic objects with indeterminate boundaries*, pages 123–140, London, 1996. Taylor & Francis.

15. K. Zimmermann and C. Freksa. Qualitative Spatial Reasoning Using Orientation, Distance, and Path Knowledge. *Applied Intelligence*, 6:49–58, 1996.

Landmarks in OpenLS — A Data Structure for Cognitive Ergonomic Route Directions

Stefan Hansen[1], Kai-Florian Richter[1], and Alexander Klippel[2]

[1] Transregional Collaborative Research Center SFB/TR 8 Spatial Cognition
Universität Bremen, Germany
`{beeswax, richter}@sfbtr8.uni-bremen.de`
[2] Cooperative Research Centre for Spatial Information
Department of Geomatics
The University of Melbourne, Australia
`aklippel@unimelb.edu.au`

Abstract. Landmarks support the structuring of environmental information into cognitive conceptual units, they have the potential to identify uniquely pertinent intersections for route following, and they disambiguate spatial situations at complex intersections. Not using them in automatically generated route directions is a violation of cognitive ergonomics. While we have made great progress on the one hand in characterizing and on the other hand in mining potential landmarks, viable data structures that incorporate their cognitive conceptual functions in route directions are poorly developed. The present article closes this gap by providing a representation based on the OpenLS standard that allows for capturing the semantics of landmarks. In this data structure, the cognitive conceptual essence of a landmark is represented allowing for generating route directions automatically and imbuing street network data with cognitively meaningful elements.

1 Introduction

Traveling from place A to place B in an unfamiliar environment is a challenging task. To ease this task, human beings often rely on route information provided by external means, i.e. route directions [27,6]. Route directions that regard mental conceptualization processes involved in the human navigation are referred to as *cognitive ergonomic* route directions in this article. They try to reduce the cognitive load for the travelers and to enhance the travellers location awareness at the same time. Manifold research has been carried out on what constitutes good route directions, how to formalize cognitive aspects, and how route directions can be generated automatically.

A recurring theme in this research are landmarks: they turn out to be one of the most important environmental features in wayfinding and route following (e.g., [23,20,5]). They are a prime means to structure space; and they are widely used in route directions given by humans, as they easily allow linking actions during wayfinding to the environment. Many aspects of landmarks have been

examined in the past, but the main challenges for wayfinding assistance systems are the automatic definition and extraction of appropriate salient objects, which may function as landmarks, from the available data sets and their integration in automatically generated directions. In other words, how can the data that is available through various sources these days be transferred into meaningful units that allow for communicating knowledge rather than information?

Mining landmarks and attaching measures to saliency are well advanced research topics, but the semantic embedding of landmarks into data structures is yet unsolved. While a good understanding exists why and when people use landmarks in organizing spatial knowledge for the purpose of communication or memory, our abilities to formalize this knowledge and to integrate landmarks in current information technology, such as PDA navigation assistants, is still limited. A major difficulty is to sufficiently formalize the concept of a landmark itself; this is a necessary step to enable a service providing route directions to integrate landmarks into the generated instructions.

This paper closes the gap between approaches identifying landmarks and those generating cognitive ergonomic route directions that automatically integrate landmarks. It proposes a data model that allows to capture the required information for generating route directions. It relates the different parts of the information, which are communicated in the route directions, to each other and constrains their use in order to avoid generating instructions that are inadequate for humans. The corresponding data structure is used to extend the OpenLS standard [17,1], a server specification for location based services proposed by the Open Geospatial Consortium (OGC); the data structure is defined in a XML markup-language as used in the OpenLS specification. This standard enables a straight-forward integration of our data structure in concrete OpenLS-based applications. The data structure itself is not constrained by any limitations or requirements of a specific application and also independent from the underlying data set. Hence, it is well-suited to be the integrating brace for different approaches.

The paper is structured as follows: first we provide some pertinent details on landmarks in route following and route directions, and their integration in the automatic generation of route directions. Then we introduce a classification of landmarks which is used as formal basis of the data structure (Section 3), and the OpenLS standard, which is used to define the data structure (Section 4). In Section 5 we present our data structure and how it may be used to integrate landmarks into OpenLS. The paper concludes with an outlook on future work in Section 6.

2   Landmarks in Route Directions

Landmarks structure environments [8], for example, as anchor points for spatial knowledge [3]. In acquiring knowledge on a previously unknown environment, they are learned early on [23]. Accordingly, landmarks are important features of route directions [6]. They may be used for identifying origin and destination and

distinguishing decision points [19], and they are especially useful to link actions to be performed within the environment, i.e. with the decision point the action needs to be performed at [5]. In that, they work better than street signs [25]. Not surprisingly, humans use landmarks to communicate route knowledge in both verbal and graphical instructions [5, 28].

Given that landmarks are so important, the idea of integrating them in automatically generated route directions is straightforward. But here the question arises which parts of our environment can be used as landmarks? It exists a plethora of definitions of "landmark", with Lynch's qualitative description provided in his seminal work on elements of a city that structure knowledge of the environment being one of the first that still has great impact on today's research [15]. According to Presson and Montello [20], for example, everything that stands out of the background may serve as a landmark. In the context of route following even road intersections are possible landmarks [12, 18]. Its salience determines whether an object may serve as landmark. The more salient an object is, the better it is identifyable by a human, i.e. the more it differs from the background. A combination of different visual, structural, and semantic aspects determines salience [24]. It also depends on the conceptualization of a wayfinding event (cf., [13, 22]).

The work presented in this article is embedded in the endeavour of generating route directions automatically. There are approaches to identify objects suitable as landmarks, i.e. to extract salient objects from geographic data sets (e.g., [21, 7]). And several approaches aim at automatically generating route directions that take cognitive aspects including landmarks into account (e.g., [22, 4, 16]). Integration of these two kinds of approaches is still an open issue.

The representation proposed in this paper, which builds on open standards and is based on cognitive considerations, aims on narrowing the gap between them. The following are the necessary steps of the generation process with special focus on the integration of landmarks (cf. [7]).

1. The first step is to create a data base, which comprises all objects in the current environment that can potentially function as a landmark. This step is idenpendent from an actual route and can be done once before route calculation, and then be used for any route. It only has to be redone if the underlying data changes. GIS data [2] or even data extracted from the internet [26] may be used as underlying data set.
2. The actual route is calculated according to the user's request (e.g., shortest route, quickest route, via locations) and all potential landmarks for that route are identified.
3. The identified landmarks are rated, for example, according to their salience in the context of the route [30] and the most useful are selected to be integrated in the route directions [7].
4. Based on these landmarks other methods for improving the cognitive ergonomics of the instructions, like combining instructions for several decision points into a single instruction (termed *chunking* by Klippel et al. [10] or *segmentation* by Dale et al. [4]), can be applied to the data generated so far.

➡ At this point all necessary data for generating route directions is collected and the route is already structured according to the instructions to be generated later.

5. In a final step the actual route directions are produced. If the results of step four are applicable, these instructions can be personalized regarding user–specific requirements and preferences (e.g., mode of presentation).

In this article we focus on a representation used for encoding the results of the fourth step above. The information processed and collected in step 4 needs to be stored in order to be usable in step 5. The data structure preferably correlates the single parts of the data set according to their future use in route directions; the information is represented independent from the modality of the route directions to be used in step 5. The generation process of step 5 may produce route instructions which adapt to the specific needs and personal preferences of a user; such personalization may require adaptation of the previous steps. However, in this paper we focus on the integration of landmarks as an aspect of cognitive ergonomic route directions and leave the personalization aspect as future work.

3 Classification of Landmarks

We need a formal specification of how landmarks function in route directions in order to integrate them in automatically generated route instructions. This specification needs to identify all required information for referring to a landmark in route directions.

Therefore, we propose a classification of landmarks according to their function in route directions [14]. While this classification excludes a detailed discussion of which (visual) information to communicate in order to enable a wayfinder to identify a landmark, it is sufficient to derive the relation between route elements and landmarks and the function of landmarks within an instruction; this way it reflects cognitive ergonomic aspects of using landmarks. Hence, it is used as the basis for the data structure presented in this paper.

Landmarks are classified according to an eight–level taxonomy. Each level describes a different aspect of a landmark's function within an instruction; the first four levels of the taxonomy are shown in Figure 1. Note that not each level provides new information for each type of landmark. The core aspects of this taxonomy presented in [14] are outlined in the following.

Root Level. The basic concept addressed in this taxonomy is *landmark*. Accordingly, it is listed at the *root level*.

Functional Level. Landmarks used in route directions may serve different purposes which are reflected in different conceptualizations. Among other things, the function of a landmark depends on the number of route elements the landmark is utilized for. Route elements are either decision points or route segments: a route consist of a sequence of alternating decision points and route segments [9, 5]. On this level, two different groups of landmarks are
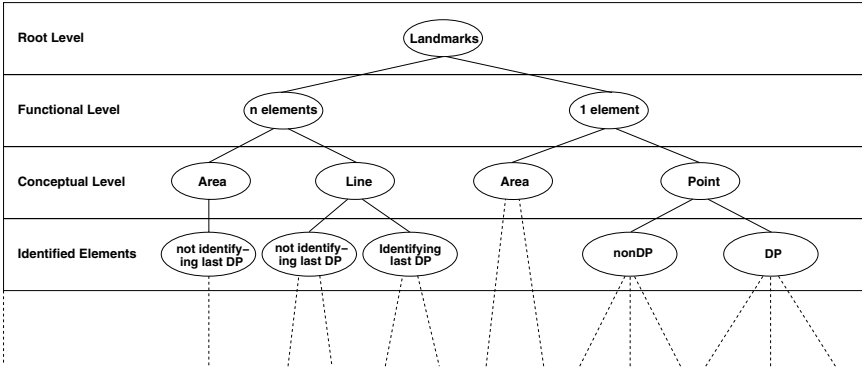
Fig. 1. The first four levels of the taxonomy of landmarks

distinguished. The distinctive feature is whether the landmark is relevant for one single route element (*1 Element*) or for a sequence of more than one route element (*n Elements*).

Conceptual Level. On the *Conceptual level* landmarks are categorised by the way humans conceptualize the functional role of their spatial extension. In the mental conceptualization they may function either as point–like (*Point*), line-like (*Line*) or area-like (*Area*) entity depending on their usage in the route following process. That means, geometrically higher–order landmarks (with $area \rightarrow line \rightarrow point$) can be conceptualized as a landmark of a lower geometrical order. For example, an areal or linear landmark can be referred to in an instruction as a point–like landmark (cf. Fig. 2).
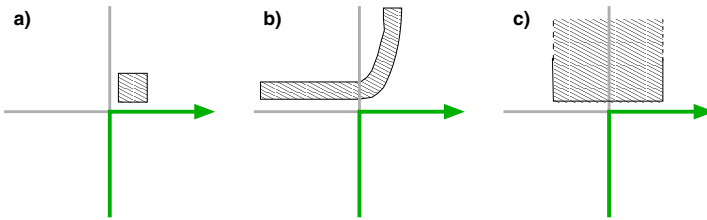
Fig. 2. Landmarks conceptualized as point–like with a) a point-like, b) a linear, and c) an areal geometry; the corresponding instruction is "turn right at the landmark"

Identified Elements. On the *Identified Elements level* landmarks are distinguished by the type of route element they relate to; landmarks can be used either to identify a route–segment or a decision point. 1–Element–landmarks located at decision points can be used either to identify the decision point or to more specifically describe the action the traveller has to perform at the corresponding intersection. The latter landmarks are categorised as *DP*;

the former as *nonDP*. n–Elements–landmarks are distinguished by whether they identifiy the last decision point they are utilized for (*identifying last DP*, e.g. "follow the tram tracks till they end.", or *not identifying last DP*, e.g. "follow the river until you reach the town hall.", respectively).

Object Class. This level is only relevant for point-landmarks located at decision points. Here, we distinguish between different categories of landmarks with a point-like function: *Street Name*, *Structure*, and general salient object (*GSO*). The category *Street Name* comprises the branches of an intersection identifiable by their name (e.g., by a street sign), which may serve as a landmark. The category *Structure* denotes intersections that are referred to because of their salient structure (e.g., a T–intersection). All other objects that may function as a point–landmark are grouped in the category *GSO*. It is necessary to distinguish between these three categories as their integration in route directions depends on different requirements for describing the landmark.

Turn. The *Turn level* refers to landmarks which relate to one particular decision point. In the case of n–Elements–landmarks, this decision point is the last the landmark is applicable for, but only if it is identified by the landmark itself (see *Identified Elements*). At this decision point, the traveller has either to perform a directional change or to go straight. Hence, we further distinguish between the categories DP^+, where a directional change is required, and DP^-, where the traveller has to go straight.

Geometrical Level. On the *Geometrical level*, landmarks are categorised according to their geometry. We distinguish *point–like*, *linear–like* and *area–like* landmarks. These categories are not based on the exact mathematical definition of the terms, but are used according to their common colloquial understanding.

In a 2–dimensional projection of the route and its surrounding environment, in a truly mathematical sense all landmarks would geometrically be areas. But humans conceptualize some as point–like (e.g., buildings), some as line–like (e.g., rivers), and others as area–like entities (e.g., forests)[1]— independent of their use in route directions. The categorization of a landmark's geometry depends, therefore, on the ratio of its width and its length and on its size proportional to the route itself.

The transition between these categories is vague and cannot be easily defined in a formal way: it is often not unambiguously clear whether a landmark is *point–like*, *linear–like* or *area-like*. Heuristics need to be applied in order to resolve this vagueness.

Spatial Relational Level. The last, most fine-grained level of the taxonomy is the *Spatial Relational Level*. Here, we differentiate the spatial relations that hold between landmarks and actions. Depending on the functional role of the landmark in that action, its geometry, and its position in the spatial configuration of the surrounding environment, we associate an appropriate

---

[1] The question whether a landmark is point–like or area–like is also often dependent on scale.

spatial relation with the landmark reflecting its conceptualization. These conceptualizations are identified by spatial terms (e.g., *before*, *after*, *at*, *in*, *around*, *along* as shown in Fig. 3 and 5) (cf. [9, 22]).
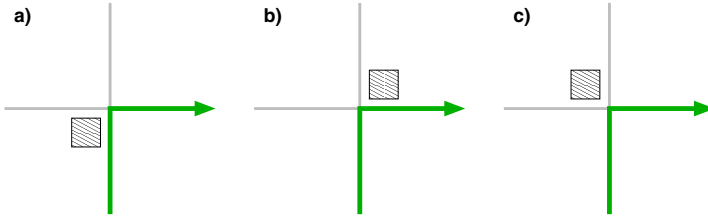
Fig. 3. a) Landmark located *before* the turning point, b) *after* the turning point, and c) not located at functional relevant branch (*at*)

We use this classification as basis to represent different types of landmarks in a system's internal, abstract data structure. This way, a landmark's function is already specified and we can constrain the final generation of instructions in order to adapt the directions to the users' requirements.

4   OpenLS

We use XML for Location Services (XLS), a XML based markup–language defined in the OpenLS specification, for defining a data structure capturing the classification of landmarks presented in the last section [17]. OpenLS describes the so called GeoMobility Server (GMS), an open platform for location-based services and its core services (directory service, gateway service, location utility service, presentation service, route service). It consists of a set of specifications of interfaces and (XML–)schemas, which define the access to the core services of such a server and the abstract data types used in the documents exchanged between server and client. OpenLS specifies primarily the interaction between client and server (request and response schemas) and the format in which the transfered data is encoded.

4.1   The Navigation Service

Additionally to the five core services, a sixth service, the Navigation Service, has been defined [1]. It is based on the Route Service and comprises the same functionality plus the option to provide the client with all information necessary to generate more elaborate route directions.

Answers to a navigation request do not contain complete route directions; instead the information necessary to generate them is transferred encoded in a special data structure defined by XLS. This then allows the client to produce route directions specific to its abilities. The data structure is supposed to capture

the information after the fourth processing step as described in Section 2. Thus, our focus of research is on this part of the generation process of route directions. Accordingly, XLS was chosen as basis for specifying the data structure proposed in this article.

The OpenLS data structure basically consists of a sequence of instructions. Each instruction describes the action the traveller has to perform at a decision point combined with information on the next route segment. The data structure originally used in OpenLS Navigation does not allow to store all information that is needed to generate cognitive ergonomic route directions. For example, even though landmarks may be integrated in the instructions, this is only possible in a very simple and restricted way, which is insufficient to describe all possible functions of landmarks within route directions.

4.2   Landmarks in OpenLS

Landmarks play an important role in the navigation process of humans and are ominpresent in human generated route directions. Therefore, it would be desirable to integrate them in a standard such as OpenLS.

The data structure for route directions in OpenLS explicitly provides an attribute for describing a landmark within an instruction. Each direction can be enriched by an advisory, that can serve different purposes, among others also describing a landmark. The offered attributes allow providing the name of the landmark and its location along the route (left or right side of the route or on both sides). Other infromation about the landmark cannot be encoded, i.e. the available data types do not allow for encoding information about a landmark in an extent necessary for the cognitive ergonomic use of landmarks. Therefore, the integration of additional data types into XLS is necessary in order to provide all information required for the use of landmarks within route directions.

5   Extending OpenLS to Include Landmarks

Our work aims at extending the data structure of the Navigation Service with several features necessary for generating cognitive ergonomic route directions. We introduce encoding of angular turning information and of the spatial structure of an intersection in order to enable the generation of precise route directions (cf. [11]); we extend the original data types with respect to spatial chunking [10] and hierarchical structuring of route directions [13]; and we enable integration of landmarks based on our classification of landmarks (see Section 3). Therefore, large parts of the original data types are replaced by new elements regarding the functionality required to generate cognitive ergonomic route directions. In the following, we describe how this integration of landmarks is implemented.

5.1   Data Type Representing Landmarks

All types of landmarks defined in our data structure are derived from an abstract parent type comprising all basic information about a landmark. With this basic

parent type we can use the different types of landmarks in a polymorphic way, i.e. use any type of landmark at the same place in an instruction without the need of specifing which concrete type of landmark to use beforehand.

A landmark type specification captures all information needed to identify the particular object: its geographical position, the spatial relation to the relevant route element, and, if needed, specific information necessary for the type of landmark at hand. Based on the abstract parent type, all other types are developed according to the classification presented in Section 3. Figure 4 shows the class-tree of the types of landmarks used.
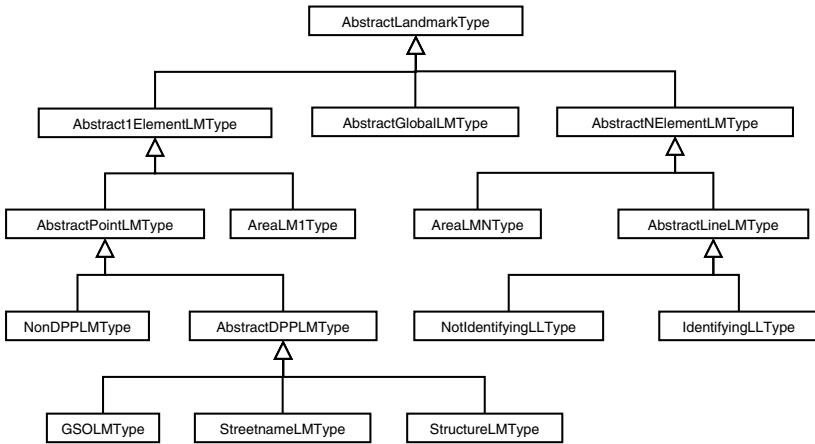
Fig. 4. Class diagram of landmarks defined in the OpenLS extension

Abstract1ElementLMType
The category of landmarks related to a single element of the route is divided into two subcategories: point-landmarks and area–landmarks. The proposed data structure contains a class for the category *1 element* with two child–classes for the two subcategories.

AbstractPointLMType. Landmarks of this type are categorised into landmarks located at a decision point and landmarks located at a route–segment. Since we distinguish three different kinds of salient objects, which all require a special and unique description to be used as a landmark at a decision point, again three sub–classes have to be defined.

StreetnameLMType. Streets indentifiable by their names may function as landmarks to identify an intersection and to mark the further direction to take. The type *StreetnameLMType* indicates explicitly that a street is used as a landmark. Since the location of such a landmark is already defined by the street's position in the intersection's spatial configuration, a further

spatial relation describing its location relative to the decision point is not necessary. In the route's specification, it is sufficient to indicate the street's appearance by its name and position.

StructureLMType. Intersections can potentially be identified by their spatial structure. Salient intersections like roundabouts or T–intersections can be used as landmarks [12, 18]. Similar to streets functioning as landmarks, intersections require a specific description: a specification of the intersection's spatial configuration is usually sufficient for its identification. Our data structure supports the most salient categories of intersections (roundabouts, complex intersections, T– and fork–intersections). A further spatial relation does not need to be provided, since it describes the spatial struture of the decision point itself.

GSOLMType. The appearance of general salient objects along a route needs to be specified by their position and the spatial relation used to describe their location relative to the route. This is required for generating an instruction referring to such an object.

Landmarks at a route segment (*NonDPPLMType*) do not require a further categorization in sub-classes. The provided information comprises a description, their geographic location, and the spatial relation used to refer to them.

AreaLM1Type. This type of landmarks does not require a further division in sub–types and, therefore, the class representing this kind of landmarks has no child–classes. Since for this type no additional information is required, an object of this type contains only the common class–variables like geographic location and a landmark's description. The spatial relation used with these landmarks is always IN, which is not explicitly represented in the XLS definition of this class.
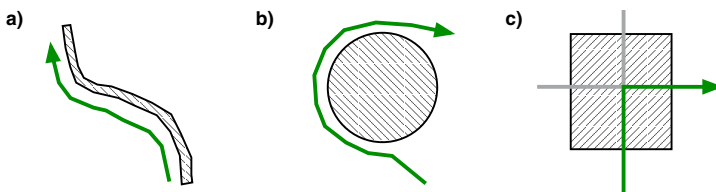
Fig. 5. a) A linear landmark located *along* the the route, b) *around* an areal landmark, and c) a decision point located *in* an areal landmark

AbstractNElementLMType

Landmarks related to more than one route element are represented by the abstract class *AbstractNElementLMType*. Similar to the class *Abstract1Element-LMType* two child-classes for the two required sub-categories are introduced. *AbstractLineLMType* stores information on landmarks conceptualized as linear and *AreaLMNType* those conceptualized as area-like.

AbstractLineLMType landmarks are divided into two different sub–classes. One comprises all landmarks that themselves identify the point where the course of the route ceases following the landmark (*IdentifyingLLType*), and the other comprises all landmarks which require additional information in form of a point–like landmark to identify this point (*NotIdentifyingLLType*).

AreaLMNType landmarks are represented by a non-abstract class since there exists no further categorization which requires child–classes. The provided information is the same as for area-landmarks identifying a single decision point. Again, there is only one possible spatial relation (AROUND), which is implicitly represented with this type.

5.2   Integration of Landmarks with Other Features of a Route

As mentioned above, route directions encoded in the proposed data structure consist in its simplest form of a sequence of route elements. In OpenLS, these are termed *maneuvers*; they describe the required action at a decision point and the previous route segment. Extending this, route directions can contain higher-order route elements subsuming other route elements; this way structuring the instructions. All these elements (decision points, segments and chunks) may be related to landmarks. Additionally, n-Elements-landmarks can be combined with other landmarks, as well. Figure 6 shows the structure of an example route. This route consists of a start- and an end-maneuver (which is common to all routes), a chunk that subsumes three elementary maneuvers, and two additional elementary maneuvers. The segment of *Maneuver 1*, the decision point of *Maneuver 5* and the *Chunk* are all related to landmarks. The implementation of these relations and of other possible relations between route elements and landmarks are explained in the following.

Landmarks at. . .

. . . chunks Chunks combine several instructions for single decision points into one single higher–order route element comprising a sequence of several decision points (cf. [10, 4]). Landmarks often are the structuring element of chunks. This purpose can be served by either line-like or point-like landmarks. Therefore, each chunk can contain an element of the type *AbstractLineLMType* or *AbstractPointLMType*.

. . . linear landmarks In the context of chunking route directions it is distinguished between line–like landmarks that are sufficient to specify the end of a chunk or line–like landmarks that are not. The latter type of landmarks requires additional information, which can be given in form of another landmark. Therefore, n-Elments-landmarks of type *NotIdentifyingLLType* or *AreaLMN-Type* contain an additional element that specifies the decision point where the route stops following the landmark. This can be marked by an element of type *Abstract1ElementLMType*.
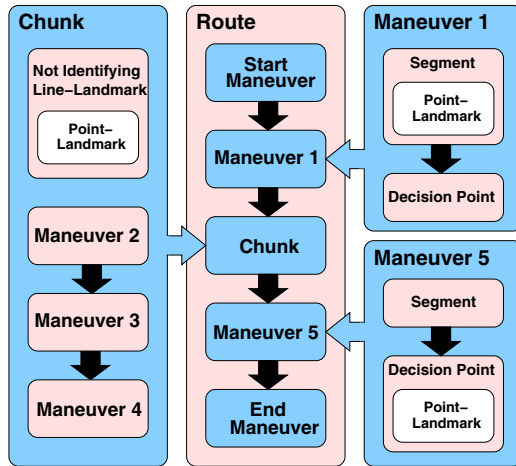
Fig. 6. Structure of an example route

. . . segments Objects used as a landmark at route segments need to be of category 1-Element-landmark. Elements representing a segment can contain objects of the type *Abstract1ElementLMType*. In principle, the number of landmarks related to a segment is not limited. If a segment contains more than one landmark, the decsion which one is used in the instructions is made in step 3 of the generation process for route directions described in Section 2, and may depend on a user's needs and preferences.

. . . decision points Intersections may be identified by landmarks with a point–like or an area–like function. Therefore, objects representing an instruction at a decision point may contain landmark objects of the three according types (*StreetNameLMType, StructureLMType, GSOLMType*). The number of landmarks related to an instruction at an intersection is not constrained, since several objects salient enough to function as a landmark may be located at an intersection. The most appropriate needs to be identified in the rating step of the generation process (step 3).

Figure 7 shows an example of a decision point identified by a point–landmark—a church—where the traveller has to perform the action of "*turn right*". The XLS–code encoding the corresponding maneuver for this example is shown in Figure 8. It contains information about the route segment leading to the decision point and about the decision point itself, including the landmark. An intersection is represented by its geographic position and by the streets crossing at this point, which are identified by their names and their angle relative to current movement direction. The landmark located at the decision point is a church, which functions as a general salient object used as a point landmark. It is, therefore, represented by an element of type *GSOLMType*. Since the traveller has to turn right *after* she passed the church, the spatial relation used here is termed AFTER.
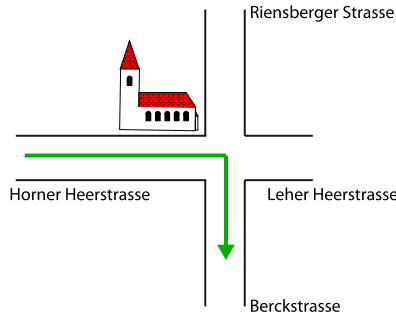
Fig. 7. Sketch of the spatial situation at the intersection used in the example

Based on this XLS–code an OpenLS-based client is able to generate a verbal instruction for this example that specifies the required action and uses the landmark to identify the intersection; for example, "turn right after the church". A pictorial representation, i.e., the sketch shown in 7, may also be reproduced with the provided information.

6   Conclusion and Outlook

Landmarks are omnipresent in any route following and wayfinding activity. They are very important in structuring spatial knowledge and a prime means to link actions in wayfinding to the environment. They are widely used in route directions given by human beings; not using them in automatically generated route directions is a violation of cognitive ergonomics.

In this article, we propose a representation of landmarks that may be used by systems generating route directions. Underlying this representation is a taxonomy that classifies landmarks according to their function in route following and their conceptualized geometric properties. This taxonomy is reflected in a corresponding data structure that captures all information requried to integrate landmarks in route directions. The data structure is based on the OGC OpenLS standard and provides a XML-based specification of actions to be performed in route following along with the structure these actions are performed in. This integration of landmarks in a well-defined standard that specifies route information independent from a concrete implementation or underlying data set is a well-suited brace between different approaches to deal with landmarks automatically. Hence, our approach closes the gap between approaches mining landmarks in data sets on the one hand and approaches employing landmarks in the automatic generation of route directions on the other hand. Furthermore, the function of landmarks is defined on the abstract, internal system's level. Thus, we can easily constrain their usage in an externalization presented to a user such that the instructions adapt to the users' requirements.

There are some desirable extensions left for future work. For example, landmarks used for orientating the traveller at the beginning of a route and for

```
<xls:XManeuver  xsi:type="xls:XManeuverType"  actionType="
    Turn" id="mn1" directionOfTurn="Right" junctionType="
    Intersection" >
  <xls:ManeuverPoint>
    <gml:pos>14.0  4.0</gml:pos>
  </xls:ManeuverPoint>
  <xls:JunctionCategory xsi:type="
      xls:StandardIntersectionType" TurnDirection="right"
      >
      <xls:RouteBranch Streetname="Berckstrasse">
        <xls:Angle uom="degree">90</xls:Angle>
      </xls:RouteBranch>
      <xls:NoRouteBranch Streetname="Riensberger_Strasse
          ">
        <xls:Angle uom="degree">270</xls:Angle>
      </xls:NoRouteBranch>
      <xls:NoRouteBranch Streetname="Leher_Heerstrasse">
        <xls:Angle uom="degree">0</xls:Angle>
      </xls:NoRouteBranch>
  </xls:JunctionCategory>
  <xls:OneElementLandmark xsi:type="xls:GSOLMType" Name=
      "Horner_Kirche" SpatialRelation="after" >
    <xls:Description xsi:type="
        xls:LMDescriptionExampleType"></xls:Description>
    <xls:PointPosition>
      <gml:pos>12.0  2.0</gml:pos>
    </xls:PointPosition>
  </xls:OneElementLandmark>
  <xls:PreviousSegment Streetname="Horner_Heerstrasse">
    <xls:Distance value="10"></xls:Distance>
    <xls:TravelTime>P0Y0M0DT0H0M2.3S</xls:TravelTime>
    <xls:BoundingBox>
      <gml:pos>2.0  4.0</gml:pos>
      <gml:pos>14.0  4.0</gml:pos>
    </xls:BoundingBox>
  </xls:PreviousSegment>
</xls:XManeuver>
```

Fig. 8. Specification of the maneuver depicted in Fig. 7

identifying the destination of a route require additional information (cf. [19]), which is not yet sufficiently covered by the proposed data structure. To easily and reliably identify the objects used as visual landmarks, a description of those

features determining the saliency of the landmark has to be provided. While the data types already offer means to store information for this purpose, determining exactly which information to store and how to organize it is an open question.

Other aspects of cognitive ergonomic route directions have to be regarded in the data structure, as well. We are currently finishing methods to perform chunking, i.e. subsuming instructions for several decision points into a single instruction (cf. [10, 4]) on the proposed data structure. This allows a more ergonomic, human-like structuring of route directions. Reducing the underspecification of cognitive situation models [31] (as instantiated by verbal route directions) by naming the spatial structures in which actions are embedded is an ongoing research effort.

Finally, our approach may be extended to include aspects of personalized route directions. This step would allow to regard a user's personal preferences (e.g., mode of presentation or familiarity with the environment) into the generation process, and, thus, to produce route directions that are truly adequate for an individual person.

Acknowledgements

References

1. Bychowski, T. (2003). OpenGIS Location Services (OpenLS): Part 6 – Navigation Service. OGC Implementation Specification 03-007r1 (Version 0.5.0). Open GIS Consortium Inc.
2. Brenner, C., Elias, B. (2003). Extracting landmarks for car navigation systems using existing GIS databases and laser scanning. Proc. Photogrammetric Image Analysis, International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXIV, Part 3/W8. München.
3. Couclelis, H., Golledge, R.G., Gale, N., Tobler, W. (1987). Exploring the anchor-point hypothesis of spatial cognition. Journal of Environmental Psychology, 7 (2):99-122.
4. Dale, R., Geldof, S., Prost, J.-P. (2003). CORAL: Using natural language generation for navigational assistance. In: Oudshoorn, M. (Ed.), Proceedings of the 26th Australasian Computer Science Conference (ACSC2003), Adelaide, Australia.
5. Denis, M. (1997). The description of routes: A cognitive approach to the production of spatial discourse. Cahiers de Psychologie Cognitive, 16:409-458.
6. Denis, M., Pazzaglia, F., C.Cornoldi & Bertolo, L. (1999). Spatial discourse and navigation: An analysis of route directions in the city of Venice. Applied Cognitive Psychology 13:145-174.

7. Elias, B. (2003) Extracting Landmarks with Data Mining Methods, in: Kuhn, W., Worboys, M.F. and Timpf, S., eds: "Spatial Information Theory: Foundations of Geographic Information Science", Vol. 2825, Lecture Notes in Computer Science, Berlin, Springer, P. 398-412
8. Hirtle, S.C., Jonides, J. (1985). Evidence of Hierarchies in Cognitive Maps. Memory & Cognition 3(13):208–217.
9. Klippel, A. (2003). Wayfinding Choremes. Conceptualizing Wayfinding and Route Direction Elements. Universität Bremen.
10. Klippel, A., Tappe, T., Habel, C. (2003). Pictorial representations of routes: Chunking route segments during comprehension. In C. Freksa, W. Brauer, C. Habel, and K.F. Wender (Eds.), Spatial Cognition III. Routes and Navigation, Human Memory and Learning, Spatial Representation and Spatial Learning (pp. 11-33). Springer, Berlin.
11. Klippel, A., Hansen, S., Davies, J., Winter, S. (2005). A High-Level Cognitive Framework For Route Directions. Proceedings of SSC 2005 Spatial Intelligence, Innovation and Praxis: The national biennial Conference of the Spatial Science Institute. September 2005. Melbourne:Spatial Science Institute. *ISBN 0-9581366-2-9*
12. Klippel, A., Richter, K.-F., Hansen, S. (2005). Structural salience as a landmark. Workshop Mobile Maps 2005, Salzburg, Austria.
13. Klippel, A., Tappe, T., Kulik, L., Lee, P.U. (2005). Wayfinding choremes - A language for modeling conceptual route knowledge. Journal of Visual Languages and Computing, 16(4):311-329.
14. Klippel, A., Richter, K.-F., Hansen, S. (in prep.). Conceptualization of landmarks – a functional-geometric ontology.
15. Lynch, K. (1960). The image of the city. MIT Press, Cambridge, MA.
16. Maaß, W. (1993). A cognitive model for the process of multimodal, incremental route descriptions. In: Frank, A.U., Campari, I. (Eds.), Spatial Information Theory: A Theoretical Basis for GIS (pp. 1–13), European Conference COSIT. Springer, Berlin.
17. Mabrouk, M. (2005). OpenGIS Location Services (OpenLS): Core Services. OGC Implementation Specification 05-016 Version 1.1. Open GIS Consortium Inc.
18. Mark, D. M. (1985). Finding simple routes: 'Ease of description' as an objective function in automated route selection. In: Proceedings, Second Symposium on Artificial Intelligence Applications (IEEE) (pp. 577–581), Miami Beach
19. Michon, P.-E., Denis, M. (2001). When and why are visual landmarks used in giving directions? In: Montello, D.R. (Ed.), Spatial Information Theory. Foundations of Geographic Information Science (pp. 292-305). International Conference, COSIT 2001. Springer, Berlin.
20. Presson, C.C., Montello, D.R. (1988). Points of reference in spatial cognition: Stalking the elusive landmark. British Journal of Developmental Psychology, 6:378-381.
21. Raubal, M., Winter, S (2002). Enriching wayfinding instructions with local landmarks. In: Egenhofer, M.J., Mark, D.M. (Eds.), Geographic Information Science (pp. 243-259). Springer, Berlin.
22. Richter, K.-F., Klippel, A. (2005). A model for context-specific route directions. In: Freksa, C., Knauff, M., Krieg-Brückner, B., Nebel, B., Barkowsky, T. (Eds.), Spatial Cognition IV. Reasoning, Action, and Interaction: International Conference Spatial Cognition 2004 (pp. 58-78). Springer, Berlin.
23. Siegel, A.W., White, S.H. (1975). The development of spatial representations of large–scale environments. In: Reese, H.W. (Ed.), Advances in Child Development and Behaviour (pp. 9–55), Academic Press, New York.

24. Sorrows, M. E. and Hirtle, S. C. (1999). The nature of landmarks for real and electronic spaces. In Freksa C. and Mark, D. M. (eds), Spatial Information Theory, number 1661 in Lecture Notes in Computer Science, Springer.
25. Tom, A., Denis, M. (2003). Referring to landmark or street information in route directions: What difference does it make?. In: Kuhn, W., Worboys, M., Timpf, S. (Eds.), Spatial Information Theory. Foundations of Geographic Information Science (pp. 362-374). International Conference, COSIT 2003. Springer, Berlin.
26. Tomko, M., Winter S. (2005). Reconstruction of scenes from geo-referenced web resources. In: Proceedings of SSC 2005 Spatial Intelligence, Innovation and Praxis: The National Biennial Conference of the Spatial Science Institute, Melbourne, Australia.
27. Tversky, B., Lee, P.U. (1998). How Space Structures Language. In: Freksa, C., Habel, C., Wender, K. F. (Eds.), Spatial Cognition: An Interdisciplinary Approach to Representing and Processing Spatial Knowledge. Springer, Berlin.
28. Tversky, B., Lee, P.U. (1999). Pictorial and verbal tools for conveying routes. In: Freksa, C., Mark, D.M. (Eds.), Spatial information theory. Cognitive and computational foundations of geographic information science (pp. 51-64). Springer, Berlin.
29. Waller, D., Montello, D.R., Richardson, A.E., Hegarty, M. (2002). Orientation specificity and spatial updating of memories for layouts. Journal of Experimental Psychology: Learning, Memory, & Cognition, 28:1051-1063.
30. Winter, S. (2003). Route adaptive selection of salient features. In: Kuhn, W., Worboys, M.F., Timpf, S. (Eds.), Spatial Information Theory: Foundations of Geographic Information Science (pp. 320-334). International Conference, COSIT 2003. Springer, Berlin.
31. Zwaan, R. A., Radvansky, G. A. (1998). Situation models in language comprehension and memory. In: Psychological Bulletin Vol. 123, (pp 162-185)

Status Functions, Collective Intentionality: Matters of Trust for Geospatial Information Sharing

Francis Harvey

Department of Geography, University of Minnesota, 414 Social Sciences, Minneapolis, MN, 55455 USA
`fharvey@umn.edu`

Abstract. A GIS coordinator for a U.S. city complains about data sharing metadata with a single word: "headaches." He underscores the function of metadata in supporting the process of finding a collective understanding of what the data can be reliably used for. In John Searle's philosophical work on social reality, this coordinator articulates the struggle to create and maintain the status functions and collective intentionality required for successful data sharing. Each technical solution to data sharing can fail to ensure that the meaning of the shared data is understood. Data sharing must move one rung higher and become information sharing, which requires the collective recognition of status functions and the creation of collective intentionality. Integral to successful information sharing is the creation of an environment of trust that ensures that function recognition occurs. In this paper I examine cases that highlight trust as a measure of status function recognition.

1 Theoretical Discussion

1.1 Concepts of Trust

The starting point for this paper is the suggestion that social science's concept of trust is under-theorized. Following Niklas Luhmann, a definition of trust in sociological literature can be summarized as follows: Trust is an indicator of people's willingness to place faith and power in relationships and institutions in which they have limited influence [6]. However, trust is more than a Wittgensteinian game. The examples later in this paper show the importance of creating a trusted institutional reality. Collective intentionality, following Searle the shared intent of a group [9], together with status functions, are indicated in individual's trust in the institutional reality. In this sense, trust is the satisfaction of conditions for an individual to accept the status functions that make up that institutional reality. Refining Luhmann's definition of trust, status functions provide the theoretical purchase for theorizing the institutional organization of geospatial information sharing.

As a wide range of philosophers, political scientists, and sociologists indicate [14, 6, 2], trust is a key issue in governance. While we commonly focus on trust as an important issue in citizen/government relationships, it is also fundamental to the

operation of government. This is apparent in smaller bureaucracies where the ability to "bend the rules" is much more commonplace than in large bureaucracies with strongly entrenched administrative procedures. Local government agency activities experience an inherently closer coupling with political representatives and with different agencies in both intra-municipal and inter-municipal activities. Granovetter's insights into the role of trust in economic relationships (embeddedness) is significant to this study of institutional trust [3]. What these points suggest is that trust is a blanket indicator of status functions that satisfy participants' institutional facts. Geospatial information sharing increases as the status functions connected to the information increasingly correspond to the collective intentionality of the sharing arrangements. The status functions of data sharing distinguish two types of trust. First, the type based on personal relations and, second, the type based on the exchange of impersonal objective data. The reliance on objective data can undermine personal trust relationships in some cases.

The historian Theodore Porter's *Trust in Numbers* [7] provides a starting point to begin understanding the role of trust in data sharing and coordination. His book demonstrates that the pursuit of scientific objectivity is integral to the development of trust in and between modern state institutions. In this regard, geographic information is integral to the process of "objective" public decision making and no different from the statistics and actuarial methods he describes. Trust in objectivity becomes a clear way of assuring a smoothly functioning bureaucratic government and aligning power relationships in the complex relationships required to assure the operation of the nation state in the light of social, political, and technological breakdowns [14]. These complex relationships involve a myriad of status functions, which can be typified as rules and regulations, procedures, and standards.

Porter highlights the role of standards in advancing trusted bureaucratic mechanisms. Standards play a key role in activities to build a national geographic information infrastructure and are a key issue in negotiations between different levels of government. Trust, in Porter's analysis of quantification [7], is fundamental to technologies of distance such as GIS [5]. As Porter points out, trust is also a key value underpinning bureaucratic cooperation. This is amply evident in technoscientific standards. Trust is part of the social interactions government agencies engage in to make decisions, but the reliance on standards diminishes its role in most organizational approaches [14]. The tension between trust in objective data and trust in political culture is very evident in local government adaptation and use of standards for geographic information sharing. As Ehrlich's work on human capital points out this dynamic process creates a new social reality involving high levels of human capital (experts) and requires a great deal of public trust in a symbiotic relationship, necessary for sustained growth.

1.2 Trust as a Measure of Status Function Recognition

John Searle's work on social reality and the collective intentionality offers a philosophical framework for approaching these questions. Expanding on the introductory section, this examination draws on Searle's account of social reality [10], and considers trust to be a measure of status function recognition. In one institutional reality a status function takes the form, for example, in an agency rule that "areas inundated at

least two weeks of the year are designated wetlands". Agencies have such rules to fulfill legal mandates entrusted them [1]. Different agencies may have responsibility for the same physical phenomena, but interpret the phenomena in incompatible ways. The institutional reality of another agency may express the status of a wetland as a system of agency-defined geochemical and biological characteristics [4]. Institutionally, these realities become the basis for a network of status functions that make up collective intentionality. Individuals working in either institution need only understand why certain characteristics are accepted to distinguish wetland from other land types and designate functions to land types identified as wetlands. These institutions and individuals may transfer data, but it is not meaningfully shared between autonomous institutional social realities without understanding the respective deontic relations.

The power of information sharing status functions arises out of a collective acceptance. The difference can be made between collective acceptance of institutional reality A and institutional reality B. In each institutional reality distinct status functions are connected to status indicators. For example, in institutional reality A, the continual presence of water may indicate that an area is a wetland. In institutional reality B, the presence of water must be accompanied by certain vegetation types if it is designated a wetland.

Successful geospatial information sharing involves developing status functions from another institutional reality. This could be directly compared to a process of learning but it differs in that the learnt meanings are first contextualized in the original institutional reality. Comparable to learning vocabulary in a new language, the common first step of learning is to recognize that the French word *eau* means the English word water. Learning the status functions of another institutional reality usually requires that the status functions be related to previously known status functions.

Ultimately, collective recognition of status functions from different institutional realities occurs, creating a new collective intentionality. The new institutional reality can serve as a bridge between other institutional realities, become a new, separate institutional reality, or supplement and possibly replace one or multiple institutional realities. The development of an institutional reality reflects changes in the recognition of the status and functions of deontic relations.

In summary, geospatial information sharing involves a process of function recognition and creation of a new institutional reality. Successful information sharing requires, at minimum, status functions that are accepted in multiple institutional realities. I would argue that, based on research evidence, a wholesale adoption of another social reality never takes place. Assigned functions are however considered individually and adapted for a particular social reality.

2   Case Studies

The two case studies are examples that illustrate that the tension between objective data and trust in personal relationship corresponds to tensions between institutional actors (here governmental agencies). Personal trust also carries over to institutional trust. The following two case studies illustrate two aspects of geospatial information sharing. The first aspect, status function discord, exemplifies the difficulties arising from a lack of trust between government agencies. This case study highlights the

importance of accepting status functions for successful geospatial information sharing. The second case study provides an example of a related process, namely that of stabilizing status functions that creates the institutional reality that satisfies the conditions of individuals in agencies participating in data sharing.

2.1 Status Function Discord

Geographic Information Systems (GIS) have been used by governments since the 1960s [11]. In Southern California, the city and county of San Diego, California (4,300 square miles with a population of 800,000) were one of the first users of GIS. Power companies there invested in GIS data collection and maintenance to support their operations. Municipalities, counties, state agencies, and federal agencies could get the San Diego Gas and Electric data for free, but had to pay royalties. Due to the increasing use of geographic information in 1984 the city and county started the Regional Urban Information System (RUIS) which was used for ten years. During these ten years more and more municipalities and consulting engineers in the area began to use GIS. The city and county reviewed the situation and decided in 1994 to pursue a new strategic plan with a focus on data preparation, distribution, and sales. First and foremost, the new system (called SanGIS) would focus on maintenance of land base data (geographic information combining land use and land cover information), providing public access, and marketing of geographic information. Through subscriptions to the land use data and minimal charges for public access, SanGIS, legally organized in 1997, hoped to achieve financial solvency and provide for the geospatial information needs of the area.

Even though SanGIS is managed by the city and county of San Diego, it is a public-private entity organized under a Joint Powers Agreement under California Government Code section 6500. The City Manager and County CAO are the sole two members of the board of directors. There is no advisory board, nor other forms of oversight. Other government agencies have no direct representation. For most intents and purposes, SanGIS functions as a private company, which offers it flexibility in negotiating contracts and in negotiations with public agencies and private companies. SanGIS has a staff of seven. Four staff members are county employees and three staff members are temporary agency employees.

Financial solvency was sought mainly through subscriptions. For a $10,000 yearly subscription a municipality could acquire all SanGIS data for their portion of the county. Even though the fees for public access were small in comparison (minimum of $10), little data was sold and few map creation service contracts signed. Marketing went somewhat better, but with few successes. With few subscriptions and limited sales, operating costs continued to be far higher than revenues. After these experiences, in 2002 SanGIS changed their subscription model, sharply reducing the yearly charge to $ 6000 for a north and south zone in the developed coastal areas and offering public administrations the data for $1000 less. Updated data from SanGIS comes quarterly and is very accurate.

What was the reason for the limited number of subscriptions and sales of accurate data under the first pricing model? Were costs the sole reason for very limited geospatial information sharing? Ultimately, the lack of subscriptions and sales led SanGIS to change their pricing scheme, but for municipalities and other public agencies this still

would have meant abrogating too much power and control over local data collection. The $10,000 year subscription fee was beyond the means for most municipalities and agencies. SanGIS defined the geospatial information sharing status functions without broader input from the existing and intended user community. Information cost was one of status functions that did not satisfy many individual agencies conditions for information sharing. The control over data collection and re-use dictated by SanGIS were more tangible status functions that other government agencies in the area also did not accept. Potential participants wanted a say in how and when data was collected, updated and maintained, especially if they were going to pay.

In response to the limited satisfaction of the desires to reduce costs and coordinate data sharing, area governmental agencies grasped an opportunity to craft an institutional reality that better fulfilled their collective understanding of geospatial information sharing status functions. This occurred through the activities of San Diego's regional planning agency, SANDAG, which is responsible for transportation planning in the region. The data it acquired for its planning purposes was made available for free to area government. Also included in the free data was a data set with rasterized land-use for the San Diego county area. SANDAG's land use data was not as accurate as the SanGIS land base and lacked much of the detailed information provided by SanGIS, but was useful enough for most planning purposes. The status functions were developed in participation and satisfied individual needs. The collective acceptance was promoted by the SANDAG emphasis on participation in administrative interactions among administrations, federal, and state bodies active in San Diego. It, and its predecessor the San Diego County Comprehensive Planning Organization, have used GIS since 1970 and it has actively supported a GIS coordinator group that meets regularly to discuss GIS development and exchange approaches for meeting changing regulatory mandates. Also organized under a joint powers agreement, SANDAG's board of directors is made up of nineteen cities and county government with SANDAG defining its role as "The Forum for regional decision-making" [8]. Each local administration has a formal agreement with SANDAG. Further, representatives of important federal agencies and the governments of the City of Tijuana and the state of Baja California, Mexico are advisory representatives. Interviewed administrative staff spoke of the importance of the regular GIS coordinator group meeting and other semi-formal and informal groups that had sprung up. Trust was further increased and human capital was enhanced by the creation of status functions in alignment with established institutional reality. New functions that local administrations took on were discussed in these interactions and resolved cooperatively. The geospatial information sharing status functions benefited from the existing collective intentionality.

In this example, discord arising as a result of one agencies approach to status functions, particularly contentions surrounding preparation and maintenance of information, led to the creation of an alternative collective intentionality that assured status functions for local governments. The availability of free, or low cost, geographic information, even if limited in accuracy, in a participatory 'forum' that permitted local administrative control and reinforced autonomy led many administrations to find most satisfactory status functions than SanGIS.

2.2 Stabilization of Status Functions

In Dane County, Wisconsin the use of geographic information technology offers some parallels to the example from San Diego, but shows that the introduction of geographic information technologies can also lead to tensions in the existing collective of government agencies involving existing status functions. Dane County is different from San Diego County because Wisconsin is one of the few states in the United States that has a funded county level position in each county whose sole purpose is the integration of land information in that county. Similarly to the actions of SanGIS, the Land Information Officer of Dane County destabilized an information sharing collective with strong personal trust among participants when new unsatisfactory conditions were imposed. This example illustrates the process of finding collective recognition of status functions.

The Wisconsin version of the geospatial information sharing strategy centers on county land information offices in each of the 72 counties. Each office has statutory responsibility for coordinating geographic information activities in that county and receives funds through a surcharge placed on property transfers. Each county has a Land Information Officer (LIO) and Land Records Modernization Plan [13]. In rural Wisconsin, the Land Information Officer readily integrates into the under-funded, overworked county administrations. Tensions with other administrative agencies arise, but can be mostly dealt with informally. The case is quite different in the main urbanized areas of the state. The City of Milwaukee has had long and tenuous relationships with surrounding counties in the state's largest metropolitan area and Dane County, location of the state capital, Madison, has also experienced many tenuous administrative relationships with the county and neighboring municipalities.

Clashes between status functions in Dane County are common. Since large areas of Dane County are still farmland, strong agriculture interests clash with urban and suburban interests regularly. The state and federal governments have local administrative offices that implement state and federal laws and regulations including soil conservation, natural resource protection, farmland preservation, etc. These laws and programs often follow existing administrative boundaries, but often follow boundaries defined solely for the purpose. The boundaries of a watershed will always coincide with a hill ridge, which may, or may not, also coincide with an administrative boundary. Finding satisfaction for the various conditions makes the creation of status functions difficult. Trust shows itself in the creation of an institutional reality that uses geospatial information that individuals accept in the fulfillment of their mandates.

The county land information officer play a key role by helping to facilitate data sharing by crafting status functions. S/he coordinates county data and partners with many state agencies [12]. However, the cities and federal agencies are independent and have their own institutional realities and status functions. In this environment, relationships between administrations were already tense and the county land information officer already has their hands full balancing conflicting status functions. Perhaps seeking a pragmatic way to collect aerial photographs of the county for clarifying these disputes, the county land information officer of Dane County decided to support the development of a partnership with a limited number of municipalities to collect aerial photographs and photogrammetric data for the county. This involved a cost-sharing arrangement with these municipalities and the county. However, the

aerial photographs were made available only to those municipalities and the county. If other agencies or municipalities wanted aerial photographs, they had to purchase a license.

Through this arrangement a new institutional reality was being crafted that stabilized around the function that financial contributors had access to the data, but others did not. This never found a collectively acceptable status. The reactions were positive among those with access and negative for the others. Some administrative staff justified the exclusionary licensing arrangement as necessary to assure that the data was collected by the agencies needing the data in a timely fashion. The complexity of negotiating a cost-sharing arrangement with all administrative agencies in the county was seen to be overwhelming. Other staff accused the county and 'partners' of creating geographic information 'haves' and 'have-nots' and destroying the principle of equitable opportunities for all parts of the county. Regardless of position, administrative staff agreed that the result reduced the willingness to share data in the county, but was beneficial for the project partners.

Stabilization involved the crafting of a new status function. A status function that reduced geospatial information sharing. The trust among project partners could now be distinguished from the distrust of others – the new institutional reality in Dane County imposed reasons for actions, which at the time of the interviews (2002) sought to create status functions, which reflected a broader agreement between governmental agencies.

3   Conclusion and Future Questions

3.1   Facets of Information Sharing

Trust is the satisfaction of conditions in order that an individual accepts a status function, thus enhancing the collective intentionality among institutions. These case studies illustrate the complex process through which status functions are developed and are integral the collective intentionality among information sharing partners. In other words, the success of geospatial information sharing is directly related to the acceptance of status functions. In the case of the San Diego area, the increase of trust indicated the recognition of an enhanced institutional reality and status functions in accord with the established collective intentionality. In Dane County, the decrease in trust indicated the creation and stabilization of a new institutional reality and status functions that was less satisfactory to individuals in the collective rationality. What these points suggest is that trust in an information sharing collective is a blanket indicator of satisfactory status functions.

3.2   Discord in Collective Intentionality

Changes to institutional reality occur, how do status functions develop? These case studies have taken a snap shot of developments and related factors to the establishment of institutional reality and status functions. Over time, the institutions change following the political process, changes in personal, changes in social values, etc. Do these changes result in discords in collective intentionality as multiple status functions

evolve? This interesting question should be explored through more extensive research.

Status functions may change as recognition of their status adapts to changes in values. How do we assess these changes? Corresponding to institutional changes, status functions themselves may change. These are further questions for more extensive research.

Several open research questions remain related to the issue that these institutions will change over time. How does change in conditions affect an analysis of geospatial information sharing using Searle's concepts? Two points highlight some of the theoretical issues future GIScience work should attend to.

Acknowledgements

References

[1] N. R. Chrisman: Design of geographic information systems based on social and cultural goals. Photogrammetric Engineering and Remote Sensing, 53 (1987) 1367-1370

[2] A. Giddens: The Consequences of Modernity. The Stanford University Press, Stanford, CA (1990)

[3] M. Granovetter: Economic action and social structure: The problem of embeddedness. American Journal of Sociology, 91 (1985) 481-510

[4] F. Harvey and N. R. Chrisman: Boundary objsects and the social construction of GIS technology. Environment and Planning A, 30 (1998) 1683-1694

[5] B. Latour: We Have Never Been Modern. Harvard University Press, Cambridge (1993)

[6] N. Luhmann: Trust and Power. John Wiley and Sons, New York (1979)

[7] T. M. Porter: Trust in Numbers. The Pursuit of Objectivity in Science and Public Life. Princeton University Press, Princeton, NJ (1995)

[8] SANDAG: Fact Sheet: The forum for regional decision making, SANDAG, San Diego (2002) 2

[9] J. Searle: Rationality in Action. MIT Press, Cambridge, MA (2001)

[10] J. R. Searle: The Construction of Social Reality. The Free Press, New York (1995)

[11] R. F. Tomlinson: A Geographic Information System for Regional Planning. In G. A. Stewart, (ed.: Symposium on Land Evaluation, Commonwealth Scientific and Industrial Research Organization, MacMillan of Australia., Melbourne (1968)

[12] D. L. Tulloch, D. Barnes, D. Bartholomew, D. Danielson and N. von Meyer: The Wisconsin Land Information Program: Supporting Community Land Information System Development. Surveying and Land Information Systems, 57 (1997) 241-248

[13] D. L. Tulloch and B. J. Niemann: Evaluating Innovation: The Wisconsin Land Information Program. Geo Info Systems, 6 (1996) 40-44

[14] M. E. Warren: ed., Democracy and Trust. Cambridge University Press, Cambridge (1999)

Pattern Recognition in Road Networks on the Example of Circular Road Detection

Frauke Heinzle, Karl-Heinrich Anders, and Monika Sester

Institute of Cartography and Geoinformatics, University of Hannover,
Appelstr. 9a, 30167 Hannover, Germany
{frauke.heinzle, karl-heinrich.anders,
monika.sester}@ikg.uni-hannover.de

Abstract. The paper will introduce into the subject of recognition of typical patterns in road networks. Especially we will describe the search for ring structures and its implementation in detail. Applications to detect these patterns and to use them for eliciting additional implicit knowledge in vector data are shown. We will familiarise the reader with different methods and approaches for the automatic detection of those patterns in vector data. The retrieval of implicit information in vector data can be very helpful for many tasks, ranging from generalisation of maps to the spatial analysis and enrichment of GIS data to make it searchable by search engines.

1 Introduction

Roads play an important role in the development of towns, regions and different land uses. "The 'transport land use' of public streetspace is contiguous and connects all other land uses. This contiguity is a basic topological property … Clearly, these patterns (of streets and land parcels) will be influenced by topography, land ownership, land value and other social, economic and physical factors. However, it does mean that close attention to the structural logic of the access network is important for understanding how existing cities are structured and how new ones may be designed." (Marshall 2005)

In GIS and neighbouring disciplines there is plenty of information stored in spatial data sets. They contain extensive knowledge: on the one hand there are the explicitly collected geometrical features and associated non-spatial attributes. On the other hand we can find much more knowledge, which is implicit given in topological and geometrical information, in terms of typical structures between features and relations of their attributes. In a road network much information is implicitly coded: the density gives an indication of populated places, junctions can be identified as motorway interchanges, road patterns show if a city has a medieval centre or belongs to a North American town.

We evaluate the existing road vector data by means of different types of graphs. Starting from the base graph representing the whole road network, we derive so called strokes and analyse their properties in the graph. Further typical patterns in the

network will be recognised, like loops, splitting of lanes, motorway interchanges, parallel streets, roundabouts and the like. In the next following step more significant structures will be found, like grids and stars. One of these significant higher level structures – namely ring-shaped configurations - will be described within this paper and the algorithm for their detection will be presented. For this purpose we analyse topological and geometrical properties of the graphs.

The extracted information regarding the structure of road networks and their typical characteristics provides a gain in knowledge which can be used for many applications: city planning, architecture, map generalisation, automatic map generation and other visualisation purposes, as well as information retrieval. With the knowledge of the most important and characteristic patterns of a town these structures can be preserved during the process of generalisation, e.g. the knowledge of such structures can support applications like typification (Anders 2005). Furthermore the knowledge recovered can be used to enrich and automatically annotate data bases, which is crucial for information retrieval in the internet (Jones et al. 2002). Spatial data sets are only accessible by search engines when their content is described by an ontology or descriptive metadata. Whereas metadata annotation is a very tedious task – automating this process has to rely on techniques that are able to automatically interpret the content.

The paper is composed of the following sections. After this introduction we will give a short overview of related topics. In section 3 a brief review of the already investigated patterns in former publications is given, while section 4 consists of the main part – the description of ring structures and the approach to detect them in road networks. The used methods and algorithms are specified and results are shown, which will be evaluated in section 5. Section 6 gives a brief conclusion.

2   Related Work

Pattern recognition is primarily used in the domain of interpreting digital images. Fundamental techniques for image interpretation have been used in many standard applications to recognise structures in pictures (e.g. Haralick and Shapiro 1992). Typical pattern recognition applications are classification of text, the automatic recognition of handwritten postal codes, or the automatic recognition of human faces.

In contrast, this paper deals with pattern recognition in vector data. In this case, however, there is no need to extract elementary low level features, because they already exist in data bases, but rather on structural pattern recognition. Vector data are often pure "Spaghetti" data in GIS, which means there are points, lines and polygons defined by coordinates but without topological or structural information. Even when annotated GIS data is available, still there is a considerable amount of implicit information in the data. Also for these data sets holds what is true for images: a visual representation of a digital data set (map) says more than 1000 words. E.g. the fact that a city has a medieval centre can be read from the structure of the network – even when it is not explicitly modelled as such. The usefulness and necessity of interpreting vector data therefore arises from the search for higher aggregations of topographic features. Even when annotations are available, they might not be usable as they are not machine readable or understandable: imagine e.g. a road data set from China.

Unless there is a semantic mapping function available that allows a transformation or an identification of classes, subclasses and attributes of Chinese land use data sets, annotations are practically of no use. Therefore, we rely on elementary geometric information and thus uses geometry and topology as a kind of "Lingua Franca".

Different strategies to discover implicit information have been developed dependent on the kind of knowledge to extract. There is the field of spatial data mining, which provides new technologies for knowledge discovery in large spatial data bases. Implicit information is extracted by using spatial rules. The aim is to find regularities in the data by applying those rules (Koperski and Han 1995, Lu et al. 1993). Another range of applications is the search for patterns in vector data. Especially the field of automated map generalisation is dealing with the problem of pattern recognition and structure modelling (Mackaness and Edwards 2002, Jiang and Claramunt 2004, Zhang 2004). The importance of pattern recognition in road networks for other disciplines like architecture, road construction or urban management is shown in (Marshall 2005). The taxonomy used by Marshall (2005) is composed of linear, treelike, radial, cellular and hybrid forms. Up to date we concentrated our investigations on some important representatives of these forms. These typical patterns are strokes as type of linear forms, grids as type of cellular forms, stars as type of radial forms and circular roads as type of cellular forms. The first three structures - strokes, grids and stars - were examined and their detection described in earlier publications (Heinzle et al. 2005). A brief summary is given in section 3.

3   Previous Work

The basic data of our investigation are the geometries of road networks. The only information we used is the knowledge about the location of roads and their junctions. Although most of the road databases available on the market are enriched with attributes like road classes or names, we did not consider the exploitation of this information. This was mainly for the reasons described above: on the one hand, the work was conducted within the project SPIRIT (2005), where the task was to interpret and annotate data of various origin and different countries. An overall ontology of all varying attribute tables and their specifications was not available. Therefore, our work contributed to generate such an ontology, which is independent of manual operation. The presented work uses topological and geometrical facts to transfer implicit information into semantics. On the other hand, we primarily looked for patterns which are typically not annotated in spatial data sets.

A graph based approach is used to search for patterns in road networks. A hierarchical structure of different graphs to reproduce different levels of detail of the network is introduced. The basic graph contains all nodes and lines of the network, with nodes representing the line intersections and edges corresponding to the lines themselves. The investigation of the topological structure of the graphs is the basis for the recognition of patterns. In the following some important patterns are briefly presented, which were described in more detail in (Heinzle et al. 2005).

Fig. 1. Strokes (left), grids (middle) and star (right) detected in different data sets (Heinzle et al., 2005)

As described before, there are a set of important and typical patterns in road structures:

- Strokes: The first pattern to mention here is the structure of strokes, namely long lines in the graph that proceed relatively straight with a continuous flow. The principle of the strokes (Thomson and Richardson 1999) is based on the assumption that traffic routes are built as curvature-poor as possible. Figure 1 left shows the main strokes in a data set. Strokes are basic patterns that can also be used for the search of other patterns as well as to the detection of ring-roads.
- Grids: Another important part of road networks are grids – a structure which is found very often in North American cities. A grid is characterised by a set of mostly parallel lines, which are crossed by a second set of parallel lines. Figure 1 middle shows detected grids in a data set.
- Stars: Starlike configurations are a common structure in road networks and are typical for concentrated urban areas where the infrastructure is dense. It is characterized by a fuzzy centre from which rays radiate. Figure 1 right exemplifies a star in a data set.
- Rings: Rings are elements of the road network that can be frequently observed in cities. They form closed loops and show up in different sizes: on the one hand there are small rings like roundabouts, on the other hand, there are large rings that typically surround a city centre.

4   Detection of Ring-Shaped Structures in Road Networks

The latest work in the context of this research is the detection of ring-shaped structures in road networks. First, we will give a short overview on the types and structure of rings.

Ring-shaped patterns occur in road networks in many styles. There are common ring roads, small circle-shaped pedestrian areas, streets along old town walls, roundabouts and big traffic circles of motorways in the vicinity of important towns. The size of such ring-shaped structures depends primarily on their functionality and varies from very small entities to larger elements.

A ring should approximate a circle with all the characteristic properties of a circle. In road networks the appearance of ring shaped patterns adapt to natural restrictions like courses of rivers and altitude differences or to human made conditions like town walls and existing building areas. Therefore ideal circles can be found rarely. Usually they are deformed and stretched. Often, they only represent a partial sector of a full circle. Correspondingly, the properties like the centre point, radius or perimeter also change. Figure 2 shows the ring shaped structure of a city in southern France. Although it is not a perfect circle, we as humans can identify it as circular structure.

Fig. 2. Ring structure of the city of Montbrison in southern France

4.1 Characterisation of Rings

In the following we concentrate on the detection of large ring roads enclosing a city centre, thus, we exclude small circles and roundabouts in this consideration. The typical characteristics of a circle-shaped ring can be described by many properties, e.g. the following:

- compactness:
  The compactness C is a measure of the jaggedness of the object contour. If the compactness refers to a circle then it is defined as $C = P^2 / 4\pi A$ where P is the perimeter and A the area of the object. In case of an ideal circle the compactness is optimal and has the value one. C is a measure for the roundness of an object.
- convexity:
  The convexity of an object is determined by the ratio of the area of the convex hull to the area of the object. In the optimal case of a circle it has the value one.
- centre and radius:
  The centre should be located near the centre of the city and the radius should be constant. In case of an ellipse-shaped ring the radius should adapt to the adequate ellipse properties.

-    size:
    The size of the ring depends primarily on the expansion of the settlement area
    and on the development of the city. The ring should enclose at least the inner
    city.
-    geometric moments:
    Geometric moments are shape descriptors, that are invariant to some transfor-
    mations, especially affine transformation. The moment M of order $p+q$ of an
    object B with the indicator function $f(x,y)$ is defined as

$$M_{p,q} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x,y)dxdy = \iint_B x^p y^q dxdy .$$

In the case of discrete coordinates it applies accordingly to the function $g(c,r)$

$$M_{p,q} = \sum_{c=-\infty}^{\infty} \sum_{r=-\infty}^{\infty} c^p r^q g(c,r) = \sum_{(c,r) \in B} c^p r^q g(c,r)$$

with $g(c,r) = 1$ inside the object
                $g(c,r) = 0$ outside the object.

Geometric moments can be adapted to raster as well as vector data. They de-
scribe the shape of an object. The accuracy depends on the order of the calcu-
lated moments. The shape will be better approximated the more moments with
higher order are calculated. The binary indicator function is defined object-
wise. The basic advantage for comparing real objects with the ideal shape of a
ring – a circle – is the invariance of the calculated moments to affine transfor-
mation. Therefore so called standard positions are determined. Different meth-
ods for computing the standard position are described in (Voss and Süße
1995). We used the iterative strategy, because it is the most robust one regard-
ing numeric instabilities. The method splits the affine transformation A into
three parts, x-shearing X, y-shearing Y and an anisotropic scaling S
($A = S \cdot Y \cdot X$).
    As a comparison two equivalence classes were used with the following
invariants:

1. circle equivalence class:
    All ellipses will be represented by a circle. The standard position describes
    a circle with $m_{0,0} = 3.544908$ and the invariant moments:
    $m_{p,q} = 0$ for $p+q = 3$
    $m_{4,0}=0.564190$ $m_{3,1}=0$ $m_{2,2}=0.188063$ $m_{1,3}=0$ $m_{0,4}=0.564190$
2. square equivalence class:
    All parallelograms will be represented by a square. The standard position
    describes a square with $m_{0,0} = 3.464102$ and the invariant moments:
    $m_{p,q} = 0$ for $p+q = 3$
    $m_{4,0}=0.692820$ $m_{3,1}=0$ $m_{2,2}=0.115470$ $m_{1,3}=0$ $m_{0,4}=0.692820$

    To find ring-shaped contours in a data set it is necessary to detect objects with a
compactness and a convexity close to one. Additionally we use the geometric

moments and the equivalence class of a circle to determine the patterns. Other constraints could be the size of the objects or the minimal number of the edges belonging to the ring. Trivial rings like small rectangles or circles are filtered out.

Our test data sets consist of a road network. In the current version we assume, that the city is located somewhere within the data set – however no further information as to where it is located exactly is needed.

4.2   The Approach to Detect Ring Roads

One of the primary characterisations of ring roads in our definition is the fact that they enclose the centre of the city. The size of a ring road is not known in advance, some cities have small inner rings, some towns are enclosed by ring roads rather outside, some even have several rings. Therefore the algorithm must be adaptive, however the precondition of enclosing the central city has to be set, otherwise it will be impossible to distinguish between ring roads and other closed loops in the network, in which we are not interested at this moment.

As we are concentrating on major central ring roads, the data has to be preprocessed to eliminate similar circular structures like small roundabouts. In contrast to such ring structures with a global character, roundabouts are local features. Roundabouts are characterized by the fact, that they describe a closed stroke with no features inside. Thus they can easily be detected by the stroke detection algorithm. Figure 3 shows all simple circles (roundabouts) in a data set.
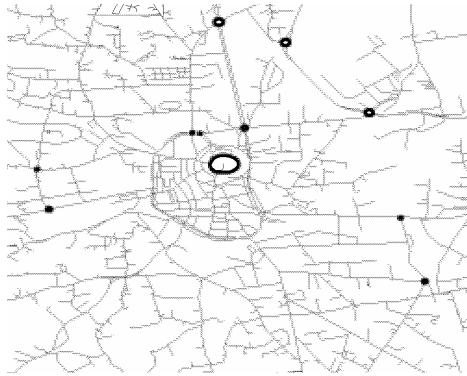
Fig. 3. Detected roundabouts which will be integrated in the strokes

The approach to detect large ring roads consists of several steps. These steps of the recognition process are based on the investigation of all road polygons. That means, the streets induce a tessellation (planar subdivision) of the plane into many subareas (meshes in the road network). We will call these subareas road polygons. The centrality of all road polygons is calculated and used in the further process.

The idea is that rings are located around the city centre and they are composed of strokes. They form an area that is circular shaped. Thus, first of all an approximate position for the city centre has to be determined (step 1). Then in step 2 neighbouring

road polygons are aggregated as long as they are not separated by a major stroke. In step 3 a combination of the aggregated polygons forming a circle are sought. This part is solved by taking some heuristics into account. The steps are described in more detail in the following.

Step 1: The centrality of the network features can be measured very well by using convex hull peeling techniques or halfspace medians (Aloupis 2005). Convex hull peeling algorithms are designed to measure the distribution of multidimensional data and the data depth. They are a quantitative measure of how central a point is with respect to a data set. It is a possibility to rank data points and to find the centre of the data cloud. We used this approach, which is also often called Tukey depth (Tukey 1975), to determine the central points of a data set. To simplify the calculation we used the centroids of all road polygons.

Fig. 4. Tukey depth: the deeper the centroids the bigger the point symbol is displayed

One of the advantages of this approach is the independence of the location of the city in the data set. It is not compulsory that the town is situated in the middle of the data set – the distribution of the road polygons and their centroids respectively is the only criterion. After computing the Tukey depth of all polygons we classify them into five categories, so we have more or less central polygons. Figure 4 shows an example, where the more central points are bigger and the points more outside are smaller displayed.

The next step of the algorithm is the aggregation of these road polygons (step 2). To this end we use the centrality measure and the area of the polygons. Small polygons and those ones in the outskirts (according to the different categories of centroid

depths) will be more aggregated than inner ones. In every category a predefined percentage rate of polygons has to be aggregated. The percentage varies from a low rate in the best category (inner polygons) to a higher rate in the category five (outer polygons). All polygons of one category are sorted according to their area and selected for aggregation until the percentage rate is reached.

Fig. 5. Aggregation of road polygons in an iterative process using strokes as border lines

The aggregation is based on the use of the already extracted strokes. If a polygon is selected to be aggregated with a neighbour, all edges of the polygon are examined regarding their adjacency to strokes. The edge which belongs to the shortest stroke is deleted leading to the fact that the polygon is merged with its neighbour. This iterative aggregation maintains the principal structure of the network. Two neighbouring

meshes are aggregated as long as they are not sharing an important stroke as common border. In this way, through the iterative aggregation potential parts of the inner city will "survive". The aggregation terminates when the number of polygons falls below a limit. The threshold in our examples was set to 18 polygons (Figure 5). The magic number of 18 is due to the problem of combinatorial explosion. In the further process we combine the remaining road polygons. The high number of possible combinations is limiting the number of polygons to assure a acceptable computational performance. Our experiments have shown, that the number of 18 polygons is a good compromise between performance and the quality of the ring road detection result.

The next challenge is the selection of a subset of these polygons that compose the ring road (step 3). Any combination of these polygons can be a candidate for the circular road. In order to generate possibilities, we proceed as follows: a graph is built with polygon centroids as nodes and their neighbourhoods as edges (Region Adjacency Graph). Figure 6 shows an example of such a graph.

Fig. 6. Region Adjacency Graph of the remaining polygons after aggregation

Subsequently all elements of the graph will be combined in a depth first search, looking for the combination of polygons that forms the best circular structure. Because of the combinatorial explosion we imposed two restrictions to a possible combination: 1. it should contain the polygon with the highest Tukey depth value and 2. it is not to be allowed to build holes. Still there are a plenty of resulting combinations, i.e. in the example shown in figure 6 where 15 polygons are left, 5718 possible combinations exist. The outline polygon of each combination is identified. Figure 7 shows five examples out of the 5718 possible ones.

Fig. 7. Five examples of possible polygon combinations

In order to determine if the aggregated shape is circular geometric moments are used (see section 4.1). The moments up to fourth order are calculated for each outline polygon. We implemented the calculation on the basis of rational numbers to avoid numerical impreciseness, which can lead to extremely different values. As a comparison we used two equivalence classes – circles and squares – as described in section 4.1.

The categorisation of all objects into these two shape categories with a nearest neighbour classification has its weakness. The real data possess a variety of shapes, which seem to be hard to classify. Most of the time the values of real ring roads are closer to the ideal values of a square. They have rough edges, bumps and missing sectors, therefore the moments differ much from the ideal reference values of circles.

An approach for solving this problem is the determination of decision criteria with the use of data mining. A supervised learning leads to a decision tree, i.e. the C45-algorithm (Witten and Frank 2000). With data mining it is possible to handle the scores of data and their diversity. Training data are introduced by giving a set of examples and counterexamples for polygons representing rings.

All eligible polygons, which were selected as ring candidates are assigned an attribute describing the probability to be a ring in percent. The best candidates are further evaluated regarding their centrality (the highest Tukey depth values should be as most central as possible inside the ring), their curvature and their convexity. These measure together with the percentage probability is summed up and yield the final score for being a circle. The result of the algorithm to detect ring shaped structures is a list of the ten best polygon candidates with a quality measure. The ring with the best properties, which has to satisfy experimental determined minimal criteria, is written out as result. If no candidate meets the conditions, the application returns a blank value. Figure 8 shows the detected ring road of the previous used example. Observe, that the algorithm is able to detect the imperfect circle which has an obviously missing sector.

Fig. 8. Detected circular road in the example data set

4.3 Examples

Figure 9 shows the road network of the city of Avignon in the south of France. As in all examples the roads are unclassified, there is no knowledge about importance, traffic flow or specific routes.
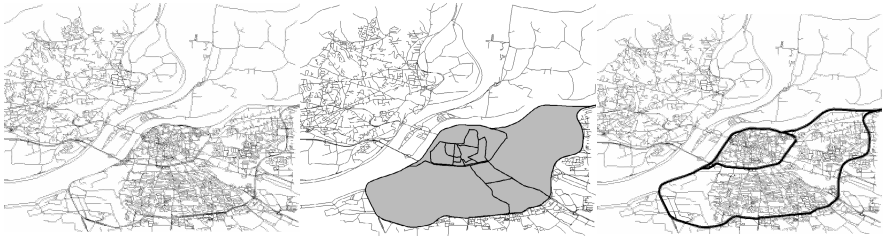
Fig. 9. The French city of Avignon (left), ring candidates (middle) and the two best ring-shaped polygons (right)

Out of the road network the grey coloured polygons are the ones left, which are considered in the process of combinatorial analysis as described in section 4.2. Out of the many candidates the two black outlined polygons are the ones, which values fit best to the decision criteria.

The next three figures show ring shaped structures of two other French cities (Figure 10 and Figure 11) and a German town (Figure 12) detected by the algorithm. The variety of appearances of ring roads can be seen very well, it ranges from shapes close to ideal circles to patterns, which look more like a rectangle with bumps or missing sectors or very elongated ellipse-like structures. Another important fact is shown by the figures as well, namely the size of the ring structures are quite different with respect to the settlement areas of the cities. Whereas the French town is completely 'ringlocked' (Figure 10), the ring road of the German one (Figure 12) is situated inside the town and surrounds only the city centre, while the settlement area has gone far beyond it.
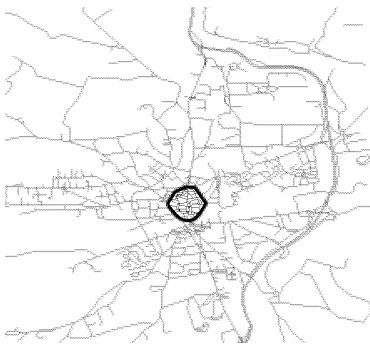
Fig. 10. The French city of Issoire and the detected ring road

Fig. 11. The French city of Montbrison and the detected ring

5   Evaluation of the Results

First comparisons of the found ring structures with the existing knowledge about real ring roads were satisfying. In the case of Brunswick (Figure 12) the names of the roads that are part of the determined ring can be checked since many of the road elements contain "Ring" in their name. The detected ring road consists of streets like "Altewiekring", "Hagenring", "Heinrich-Büssing-Ring", "Rebenring", "Wendenring", "Altstadtring" and "Neustadtring". What is, however, also true and does again motivate our research is the fact that the inverse is not true: The selection of all roads named "Ring" do not form a closed circle (Figure 13).
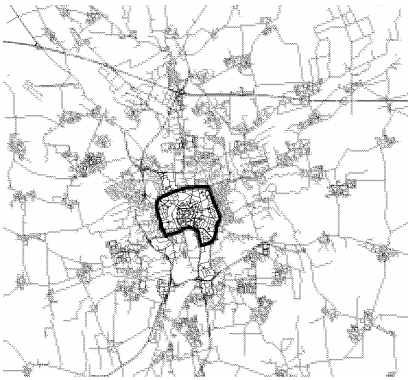
Fig. 12. The German city of Brunswick and the detected ring

Fig. 13. Selection of all roads named 'Ring' in Brunswick

The detected ring structures of other cities were compared with the intuitive interpretation of test persons. Apart from small discrepancies the results did not reveal structural faults. The algorithm therefore is a practicable method and a very good approach to extract such ring structures.

The algorithm allows to detect ring structures that may even deviate quite considerably from ideal circles. The examples show that it is possible to find elongated forms, rings that are missing certain sectors, and also rings within rings. As the criteria for rings are invariant to affine transformation, also different sizes of rings can be detected. The algorithm can be applied to any road data given in terms of a line set. It is flexible and needs no prior information by a human operator.

It is interesting to note that the patterns can occur at different scales. Our algorithms are able to take this in to account. In the case of rings, as shown in the examples, circular structures of arbitrary sizes can be detected. The only assumption is that the ring surrounds the city centre.

This is also subject to future work. In case a city has several centres, there might also be several ring roads. The extension of our approach is straightforward: after the first ring(s) have been detected, those parts of the city are excluded from the data set and the process is started anew.

6   Conclusion

The presented results together with the other structures shown in section 3 (grid, star, stroke) should be seen as parts of one overall topic - namely the detection of implicit information in vector data and the enrichment of data sets with new knowledge. The understanding of typical structures in settlements, road networks or land use boosts the potential to use data sets far in excess of the existing applications. The presented structures are important ones, there are more to be detected and especially to be exploited by combining them. However "the topic of 'structure' seemed to expand progressively under examination, revealing new layers of complex detail, almost like an unfolding fractal" (Marshall 2005). In future the stress will be laid on the investigation of applying those patterns shown in combination with other information to derive higher level knowledge. The benefit of this research is to provide answers to questions like 'the age of a town', 'the classification of cities into more or less important ones', 'the detection of a city centre', 'the possibilities of suburban or industrial settlement' or even 'the best or nice places to live'. Answers to such questions might seem trivial when assuming that annotated spatial data is available, however, this is rarely the case. On the one hand, this type of information might be too vague to define or too expensive to acquire. On the other hand, it may only be described in an unknown schema or ontology. Geometric properties can be extracted as basic structure from any kind of spatial data set. This "geometry language" can be applied to all data sets – independent of the potentially used ontology. Thus, via geometric properties, an ontology transfer can be possible (see e.g. also Volz, 2005).

Acknowledgement

References

1. Aloupis, G.: Geometric Measures of Data Depth. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, accepted in December 2005 (2005)
2. Anders, K.H.: Level of Detail Generation of 3D Building Groups by Aggregation and Typification. Proceedings of 22nd International Cartographic Conference, 9. - 16. July 2005, La Coruña/Spain (2005)
3. Haralick, R.M., Shapiro L.G.: Computer and Robot Vision. Addison-Wesley, Vol. I & II (1992)
4. Heinzle, F., Sester, M., Anders, K.H.: Graph-based Approach for Recognition of Patterns and Implicit Information in Road Networks. Proceedings of 22nd International Cartographic Conference, 9. - 16. July 2005, La Coruña/Spain (2005)
5. Jiang, B., Claramunt, C.: A Structural Approach to the Model Generalization of an Urban Street Network. GeoInformatica, 8:2, (2004) 157-171

6. Jones, C.B. et al. : Spatial Information Retrieval and Geographical Ontologies: An Overview of the SPIRIT project". Proc. of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Tampere, Finland, (2002) 387 – 388

7. Koperski, K., Han, J.: Discovery of Spatial Association Rules in Geographic Information Databases. In: Egenhofer MJ, Herring JR (eds), Advances in Spatial Databases, Vol. 951 of Lecture Notes in Computer Science, Springer Verlag, Heidelberg, (1995) 47–66

8. Lu, W., Han, J., Ooi, B.C.: Discovery of General Knowledge in Large Spatial Databases. Proc. Far East Workshop on Geographic Information Systems, Singapore, (1993) 275-289

9. Mackaness, W., Edwards, G.: The Importance of Modelling Pattern and Structure in Automated Map Generalisation. Joint Workshop on Multi-Scale Representations of Spatial Data, Ottawa, Canada (2002)

10. Marshall, S.: Streets & Patterns. Spon Press, Taylor & Francis Group, New York (2005)

11. SPIRIT (2005) – Spatially-Aware Information Retrieval on the Internet, http://www.geo-spirit.org/

12. Thomson, R., Richardson, D.: The 'good continuation' principle of perceptual organization applied to the generalization of road networks. Proceedings of the 19th International Cartographic Conference, Ottawa, (1999) 1215-1223

13. Tukey, J.: Mathematics and the picturing of data. Proceedings of the International Congress of Mathematicians, Vancouver, (1975) 523-531

14. Volz, S.: Data-Driven Matching of Geospatial Schemas. COSIT 2005, (2005) 115-132

15. Voss, K., Süße, H.: Adaptive Modelle und Invarianten für zweidimensionale Bilder. Verlag Shaker, Aachen (1995)

16. Witten, I.H., Frank, E.: Data Mining, Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann Publishers (2000)

17. Zhang, Q.: Modeling Structure and Patterns in Road Network Generalization. ICA Workshop on Generalisation and Multiple Representation, Leicester, UK (2004)

Implementing Anchoring

James Hood and Antony Galton

University of Exeter, UK
{J.M.Hood, A.P.Galton}@exeter.ac.uk

Abstract. In an earlier paper we introduced Anchoring, a new approach for handling indeterminate location in a spatial information system. Here we develop it further, showing how Anchoring can be made to work in real systems. In particular, we specify a new kind of locational component for a spatial data model. We then show how that component can be implemented into an object-relational database with extensions conforming to the OpenGIS Consortium's Simple Feature Model. We argue that Anchoring, in contrast to other formalisms for handling indeterminate location, is better suited to the needs of a spatial data provider who, in supplying data to different organisations, needs to be authoritative, and thus does not want to compromise data quality by representing indeterminate data with unwarranted precision. Anchoring, in providing new ways to describe spatiality, allows that we state only what we know for certain.

Introduction

In [9] we introduced Anchoring, a new approach for handling indeterminate location in a spatial information system. There our description of Anchoring was in terms of a high-level extension to a spatial information system. This led to a partial formalisation of Anchoring as a first-order logical theory. In this paper we drop down from those heights of abstraction and start to show how Anchoring can work in real systems. In particular, we present Anchoring as a new kind of locational component for a spatial data model. We then show how that component can be implemented into an object-relational database with extensions conforming to the OpenGIS Consortium's Simple Feature Model.

1   Anchoring in Brief

Before all that, though, let us introduce Anchoring. We do this by placing it in contrast to the 'classical' model of location that we hope to replace.

1.1   The Classical Model and Its Limitations

Consider Figure 1. We can think of this as presenting the structure of a data model in a typical spatial database. In it there are spatial objects (or *features*, as they sometimes called), which are things with 'whereness'. They are the objects with which the spatial database is concerned. The spatial objects populate

a part of the spatial data model—what we call the *information space*. The information space captures and records all the various non-spatial attributes that the spatial objects have and the non-spatial relations that they stand in. By contrast, spatial attributes and spatial relationships for a spatial object are not stored explicitly in information space. Rather, they are stored implicitly through the link between a spatial object and its spatial location. Spatial locations are the regions of what we call *precise space*, a mathematical representation of space with a coordinate reference system.[1] In a typical spatial database the precise space is the coordinatised Euclidean plane representing part of the Earth's surface. Spatial objects are associated with regions on the plane, and we consider the region to which a spatial object is tied as its spatial location. The key point to note here is this. In the classical model, there is just *one way* of tying the spatial objects to regions in the precise space. As we put it, since they must be coextensive with a region in the precise space, all spatial objects must have 'exact location'.
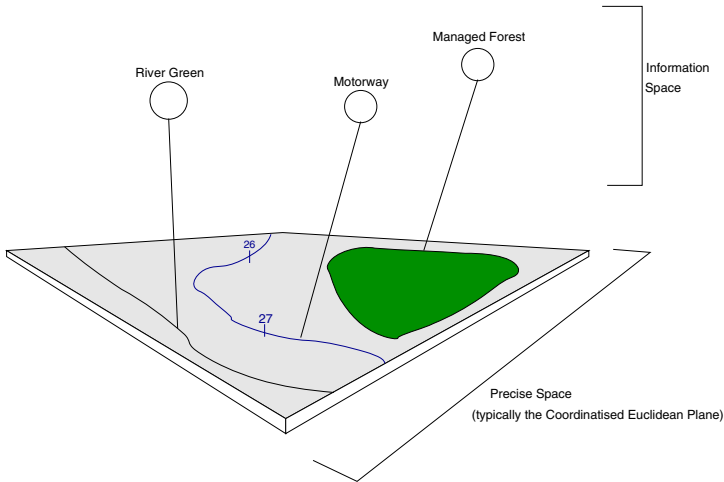
Fig. 1. Classic model of location using 'exact location'

This conclusion—that to be a 'spatial' object the object must have an exact location—means we must be able to locate the spatial object exactly on some region in a precise space. So those objects that cannot be tied to exact location cannot be represented in the data model. Hence there is no room for objects such as hills, valleys and other geomorphological features, whose location, either because of its inherent vagueness or of our lack of knowledge, cannot be specified precisely.

It may help at this point to recall some examples we presented in [9]. We will return to these later on in the paper.

---

[1] What we call a 'precise space' is usually just called a 'spatial reference system'.

Our first example is Greendale, in which runs the River Green. The river, as is shown by its representation in Figure 1, can be considered as having an exact location. That is, within the confines of a spatial data model, we can represent the River Green as a linear geometric construct. This, we might say, is 'good enough' for the purposes of our application. Greendale, by contrast, does not have exact location, as it is a geomorphological feature and part of the natural landscape. While a competent (human) map user would recognise such a feature not so much as an object but rather as implicit in a certain configuration of contour lines, the classical spatial data model is restricted to tying objects to regions. Consequently, Greendale, if it is to be represented at all, must be identified with some precise region. But since Greendale is lacking a precise boundary, any region we choose will invariably be a limited approximation.

Moving away from vagueness, our second example deals with uncertainty. The Highway Management Office in Devon County Council uses a spatial database to keep track of, among other things, road accidents. Road accidents are 'spatial' objects in virtue of their association with geometric points that represent accident locations. Using the classical model, though, accidents cannot be recorded in the database unless they can be given an exact location. But such information may not be available. For example, suppose we want to record information about an accident for which all we know about its location is that it happened 'somewhere' between junctions 26 and 27 of the M5 motorway. Such information cannot be entered into our database unless we can associate the accident with some region. Perhaps one way of overcoming this might be to relax the constraint that accidents must be located on points and allow them to be located on extended geometric regions instead. That way the accident could be located on the region occupied by the particular stretch of motorway, and so could be entered into the database. However, that would lead to some anomalous answers to queries since, to the information system, the accident would have to be considered as coextensive with that region. The accident might, for example, have other accidents located within it! Now we, as competent map users, knowing what accidents are, would understand what was really meant by having the accident associated to the region in this way. But the spatial database would not, since for it there is just one kind of locational relation between objects and regions.

Our final example features an odd mix of uncertainty and vagueness. It is loosely based on a real-life example. Consider an area of managed forest which falls within the confines of a national park. The managed forest is home to a rare species of butterfly. This population is, of course, located in space, but we can see that its spatiality is different from the kind possessed by other features in the spatial data model such as rivers, buildings and accidents (provided we do know where such accidents happen). Because observations have been made there, the population is definitely known to inhabit two particular areas in the forest. However, these are not the only places that the butterflies inhabit, and it is taken as given that the population is scattered all over the forest. What the users want is to loosely associate the location of the butterfly population with the entire

extent of the managed forest, but without thereby implying that every point within that extent is at every, or any, time occupied by a butterfly. As a make-shift compromise the users have added a textual 'note' onto the forest's database record which states the existence there of the butterfly population. Needless to say, though, such a note lacks any logical structure and so is unusable for the spatial information system.

1.2   Enter Anchoring . . .

Uncertainty and vagueness are very different. On the one hand, under a common sense view, vagueness seems to do with the entity we are trying to represent, while on the other, uncertainty seems to do with our (lack of) knowledge about the entity. That the location of Greendale, for example, is indeterminate seems to be something intrinsic to Greendale itself, and not to us; while the indeterminacy of the accident is to do with us and our lack of knowledge, and nothing to do with the accident. Despite their differences, though, vagueness and uncertainty have enough commonality in the representational problems they raise to justify our treating them together, at least in some respects. In both cases we find a kind of *indeterminacy* which prevents us from using the classical model—that is, we cannot record in the database information about the location of the spatial object. That fact is our motivation for Anchoring, a new kind of representational framework in which we treat vagueness and uncertainty together.

Anchoring gives us more than one way of relating objects in information space to regions in precise space. We spoke earlier of there being just one locational relation between the two spaces, namely exact location. Anchoring in effect supplements exact location with other locational relations. The two most important locational relations are **anchored in** and **anchored over**. Their meanings are as follows. We say

- a spatial object **o** is *anchored in* some region R if and only if, whatever may be the nature of **o**'s location, we can certainly say that **o** is located inside R.
- a spatial object **o** is *anchored over* some region R if and only if, whatever may be the nature of **o**'s location, we can certainly say that it contains the whole of R.

In addition, for compatibility with the classical model, we stipulate that an object has exact location (in the sense of the classical model) if and only if there is some region which the object is both anchored in and anchored over.

The result of modelling our examples with the anchoring relations is shown in Figure 2. In our accident example, we now say that the accident is 'anchored in' the stretch of motorway between junctions 26 and 27 of the M5. That is to say, though we know that the accident is indeed located exactly at some particular location, at present we don't know where that location is, only that it is somewhere within the stretch of motorway. Accidents may still be constrained

to be points, but, as it happens, the exact point on which this particular accident is located is unknown to us, and so the spatiality of the accident must be represented in the spatial information system in a different way.
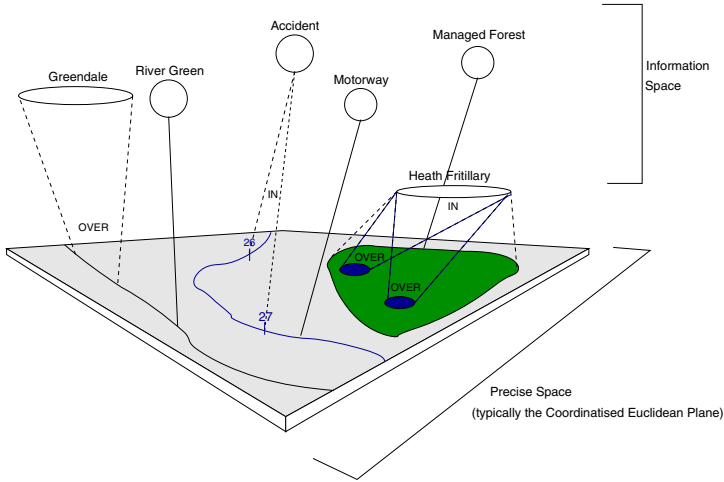
Fig. 2. Locational model using anchoring relations

The other two examples are handled similarly. Greendale is now 'anchored over' a designated stretch of Green River, while the butterfly population is now anchored in the region occupied by the national park and anchored over two specified areas where observations have been made. Each one of the anchorings provides a different piece of locational information, and thus contributes to the spatiality of the spatial object in question.

With our Anchoring approach, we are saying, in short, that there may be many ways in which we can consider objects in information space to be 'spatial'. Exact location, where we identify the location of an object with a precisely specified region, is just one way. Other ways are our two anchoring relations. In short, the job now is to set up a framework in which it is possible to specify a spatial object by a whole bag of different pieces of locational information. It is like describing the location of some object to someone we know. We might say, 'It's definitely within this bit', 'It's twenty miles north of so-and-so', or 'It covers such-and-such'. As we speak the listener is building up some idea of where we are talking about. Our task with Anchoring is to simulate such understanding in our spatial database.

2   Anchorings, Relative Locations and Location Sets

In this section we flesh out the Anchoring approach in terms of a conceptual specification for a new kind of locational component for a spatial data model.

The specification will be independent of any particular data model architecture (hence our use of 'conceptual'), but will nevertheless be geared towards actual implementation in real systems. (Indeed, in section 3 we demonstrate the feasibility of this assertion by implementing the approach as an extension to an object-relational data model.) In what follows, we write

  – bold lowercase letters (e.g., $\mathbf{a}$) to denote objects in information space.
  – italic uppercase letters (e.g., P) to denote elements of the precise space.

In addition, we let the elements of precise space be points or various kinds of sets of points (regions) which can function as precise spatial locations.

2.1   Location Sets

In the classical model, what makes an object 'spatial' is its being assigned an exact location in a precise space. We would like now to change this definition. An object is now a spatial object if and only if that object has a **location set**. For a given object $\mathbf{o}$ in information space, $loc(\mathbf{o})$ denotes its location set. $\mathbf{o}$ is therefore spatial if and only if $loc(\mathbf{o}) \neq \perp$.[2] A location set embodies all the information about the location of the object that has been entered into the database. Such locational information falls into two sorts: anchorings and relative locations.

2.2   Anchorings

An **anchoring** is an ordered pair (t, X), where X is a region in precise space and t is the type of anchoring (currently restricted to just IN and OVER). The anchorings in the location set $loc(\mathbf{o})$ of some object $\mathbf{o}$ must satisfy the following constraints.

  – Any region an object is anchored over is part of any region it is anchored in.

  T1 $(\text{IN}, X) \in loc(\mathbf{o}) \wedge (\text{OVER}, Y) \in loc(\mathbf{o}) \rightarrow Y \subseteq X$

  – No two regions that an object is anchored in are disjoint.

  T2 $(\text{IN}, X) \in loc(\mathbf{o}) \wedge (\text{IN}, Y) \in loc(\mathbf{o}) \rightarrow X \cap Y \neq \emptyset$

We are now in a position to describe our accident example in terms of anchorings and location sets. Let $\mathbf{a}$ be the object in information space representing the accident and M the region in precise space representing the location of the stretch of motorway between junctions 26 and 27. We describe our current knowledge about where the accident is in terms of its location set as follows:

$$loc(\mathbf{a}) = \{(\text{IN}, M)\}$$

---

[2] The location set may be empty, meaning that while the object has a location, we know nothing about it. This case, $loc(o) = \emptyset$, is to be distinguished from the case $loc(o) = \perp$, which means that no location set (not even an empty one) has been assigned.

2.3 Relative Locations

But what if we want in our location set to refer not to regions in precise space, but rather to other objects in information space? The region with which an object is associated may change, and so anchorings which are linked to that region may under particular circumstances also need to change. For example, consider once again our butterfly population **b** and the area of managed forest **f**. Recall that the population is definitely known to inhabit, and so is anchored over, the two regions in which it has been observed (let us call them S_1 and S_2), while it is definitely contained, and thus anchored in, the area of the managed forest. Let F be the region occupied by the managed forest (so **f** is exactly located on F). The location of b described in terms of its location set is therefore

$$loc(\mathbf{b}) = \{(\text{OVER}, S_1), (\text{OVER}, S_2), (\text{IN}, F)\}$$

But what if the region occupied by the managed forest changes? For example, suppose the local authority had dubious respect for the natural environment and allowed the development of a housing estate whose extent infringed directly upon the managed forest. Let F' denote the new region that the managed forest occupies after the development. Since the butterfly location will now also change, we need to update its location set too. After the update the new location of the butterfly population will be

$$loc(\mathbf{b}) = \{(\text{OVER}, S_1), (\text{OVER}, S_2), (\text{IN}, F')\}$$

In high-level representational frameworks of the kind we are describing there is usually no consideration about the practicalities of updating our knowledge. Such 'non-monotonic' aspects, if they are considered at all, are usually part of a dedicated theory. In designing a spatial data model, by contrast, such practicalities are of the utmost importance. So given that the Anchoring approach is intended for real implementation, we introduce now the notion of a **relative location**, which is the second type of locational information a location set may contain. Like an anchoring, a relative location is a pair whose first element is either IN or OVER. But whereas an anchoring is a relation between an object and a region, a relative location is a relation between two objects. In a sense a relative location in a object's location set relates that object to the location of another object *whatever the location happens to be at any given time*.[3] Furthermore, it does this without the introduction of any explicit temporal framework. Thus, we can describe the location of the butterfly population as follows:

$$loc(\mathbf{b}) = \{(\text{OVER}, S_1), (\text{OVER}, S_2), (\text{IN}, \mathbf{f})\}$$

(that is, by a location set comprising two anchorings and one relative location).

---

[3] A relative location is thus in some ways similar to Donnelly's notion of a 'relative place' [3].

2.4   Reasoning with Location Sets

As presented thus far, our representational framework is of limited use. We can capture location information pertaining to a particular spatial object in a location set, and have that information described in terms of either anchorings or relative locations. Anchorings, through their connection with a precise space, allow for drawing inferences about various regions in precise space. But relative locations allow for no such inference. Our anchoring rules, alluded to in the first section and introduced in our previous paper, provided a mechanism for deriving further information. The task now is to incorporate those rules into our framework.

At this point it is important to emphasise the distinction between a location set and the information which may be inferred from it. A location set, as stated earlier, embodies just the information about the location of the object that has been entered into the database. It is therefore not the same as the information that can be *inferred* from this data. The elements of a location set are the *givens*, the 'information atoms' that the database agent has to work with. Confronted with a query or an assertion, the database agent, like a competent map user, should therefore base its response on an understanding of how the relevant location sets relate the objects in question to each other, to other objects and to the spatial frame in general: in short, it should know what the location sets *mean*. To this end we now give rules specifying the meaning of location sets.

The first two rules may seem obvious, but nonetheless need to be stated. Together the rules state that if a location set contains an in-anchoring or an over-anchoring for a given region, then we can infer that the object is anchored in or anchored over that region, respectively. (As in [9], for the anchored-in relation we write '$\lhd$', and for the anchored-over relation we write '$\rhd$'.)

– If an object's location set contains an in-anchoring for some region, then we can infer that the object is anchored in that region.

A1 $$(\mathrm{IN}, R) \in loc(\mathbf{o}) \to \mathbf{o} \lhd R$$

– If an object's location set contains an over-anchoring for some region, then we can infer that the object is anchored over that region.

A2 $$(\mathrm{OVER}, R) \in loc(\mathbf{o}) \to \mathbf{o} \rhd R$$

The importance of these rules is in the use of the conditional, not the biconditional. That is, just because we could infer, in our chain of reasoning, that an object is anchored somehow to some region, we are not thereby licenced to infer that that information is explicitly contained in the location set. A location set is, after all, finite and contains just those 'information atoms' which have been recorded.

The next two rules extend our constraints about what anchorings a location set may or may not contain to more general constraints on the anchoring relation itself.

- Any region an object is anchored over is part of any region it is anchored in.

  A3 $\qquad\qquad\qquad \mathbf{o} \triangleleft R \wedge \mathbf{o} \triangleright S \rightarrow S \subseteq R$

- No two regions that an object is anchored in are disjoint.

  A4 $\qquad\qquad\qquad \mathbf{o} \triangleleft R \wedge \mathbf{o} \triangleleft S \rightarrow S \cap R \neq \emptyset$

In terms of a logical theory, the constraints T1 and T2 introduced above now emerge as consequences of the axioms A1–A4.

Next, rules A5 and A6 show how existing anchoring information leads to new anchoring information.

- An object anchored in region R is anchored in any region containing R.

  A5 $\qquad\qquad\qquad \mathbf{o} \triangleleft R \wedge R \subseteq S \rightarrow \mathbf{o} \triangleleft S$

- An object anchored over some region R is anchored over any part of R.

  A6 $\qquad\qquad\qquad \mathbf{o} \triangleright R \wedge S \subseteq R \rightarrow \mathbf{o} \triangleright S$

Together, rules A3–A6 characterise the primitive anchoring relations.

Notice so far that there has been no mention of relative location. The anchoring relations are 'primitive'—which is to say, their meaning is given not explicitly by definition, but implicitly through their function in the rules as previously stated. By contrast, we now going to explicitly define the relative location notion of 'located within'. We will then see that this leads to some interesting consequences, both practically and philosophically, for the relation and the Anchoring approach in general.

Let $\mathbf{o}$ and $\mathbf{p}$ be objects in information space. We read $\mathbf{o} \sqsubseteq \mathbf{p}$ as '$\mathbf{o}$ is located within $\mathbf{p}$'. The relation $\sqsubseteq$ is defined by the following rule.

- $\mathbf{o}$ is located within $\mathbf{p}$ if and only if there is some region which $\mathbf{o}$ is anchored in and $\mathbf{p}$ is anchored over.

  Def$_\sqsubseteq$ $\qquad\qquad \mathbf{o} \sqsubseteq \mathbf{p} =_{\text{def}} \exists R(\mathbf{o} \triangleleft R \wedge \mathbf{p} \triangleright R)$

By analogy with A1 and A2, we can state rules for relative location as follows.

  A7 $\qquad\qquad\qquad (\text{IN}, \mathbf{p}) \in loc(\mathbf{o}) \rightarrow \mathbf{o} \sqsubseteq \mathbf{p}$

  A8 $\qquad\qquad\qquad (\text{OVER}, \mathbf{p}) \in loc(\mathbf{o}) \rightarrow \mathbf{p} \sqsubseteq \mathbf{o}$

By these axioms we can now derive a series of theorems expressing constraints on what relative locations a location set may or may not contain.

- If $\mathbf{p}$ is located within $\mathbf{o}$, then any region that $\mathbf{p}$ is anchored in must overlap any region that $\mathbf{o}$ is anchored in.

  T3 $\quad (\text{IN}, \mathbf{o}) \in loc(\mathbf{p}) \wedge (\text{IN}, R) \in loc(\mathbf{p}) \wedge (\text{IN}, S) \in loc(\mathbf{o}) \rightarrow R \cap S \neq \emptyset$

– If $\mathbf{p}$ is located within $\mathbf{o}$, then any region that $\mathbf{p}$ is anchored over must be contained within any region that $\mathbf{o}$ is anchored in.

T4 $(\text{IN}, \mathbf{o}) \in loc(\mathbf{p}) \wedge (\text{OVER}, R) \in loc(\mathbf{p}) \wedge (\text{IN}, S) \in loc(\mathbf{o}) \rightarrow R \subseteq S$

T5 and T6 are similar to T3 and T4, but govern the relative location $(\text{OVER}, \mathbf{o})$, as opposed to $(\text{IN}, \mathbf{o})$.

T5 $(\text{OVER}, \mathbf{o}) \in loc(\mathbf{p}) \wedge (\text{IN}, R) \in loc(\mathbf{o}) \wedge (\text{IN}, S) \in loc(\mathbf{p}) \rightarrow R \cap S \neq \emptyset$

T6 $(\text{OVER}, \mathbf{o}) \in loc(\mathbf{p}) \wedge (\text{OVER}, R) \in loc(\mathbf{o}) \wedge (\text{IN}, S) \in loc(\mathbf{p}) \rightarrow R \subseteq S$

Some interesting further consequences follow from these rules. At first sight, $\sqsubseteq$ may seem to be a good candidate for the merological parthood relation. However, our hopes for having the full power of mereology are dashed as soon as we try to infer the first merological axiom, namely that parthood is reflexive (everything is part of itself). From Def$_\sqsubseteq$, reflexivity of $\sqsubseteq$ requires that, for every object $\mathbf{o}$,

$$\exists R(\mathbf{o} \lhd R \wedge \mathbf{o} \rhd R)$$

i.e., every object has some region which it is both anchored in and anchored over. But that implies that all objects must have exact location, which goes counter to the base tenet of the Anchoring approach. Hence making $\sqsubseteq$ reflexive precludes the representation of objects with indeterminate boundary, and so we cannot therefore stipulate such a property. Because it is not reflexive, $\sqsubseteq$ is not 'parthood' in the mereological sense.

The axiom of anti-symmetry also cannot be proved. However, attempting to prove the axiom brings to light some consequences for Anchoring in its relation to ostensibly similar approaches. Recall that parthood anti-symmetry states that, for any two objects, if they are both part of each other, then they are equal. Let $\mathbf{o}$ and $\mathbf{p}$ be spatial objects. Suppose $\mathbf{o}$ is located within $\mathbf{p}$ and $\mathbf{p}$ is located within $\mathbf{o}$. The definition of $\sqsubseteq$ implies the existence of regions R and S such that $\mathbf{o}$ is anchored in R and anchored over S, and $\mathbf{p}$ is anchored in S and anchored over R. By rule A3, it follows that $R \subseteq S$ and $S \subseteq R$, and so $R = S$. So $\mathbf{o}$ and $\mathbf{p}$ are each both anchored in and anchored over the same region S. But since that doesn't imply that $\mathbf{o} = \mathbf{p}$, anti-symmetry is not proved. However, from assuming that $\mathbf{o}$ and $\mathbf{p}$ are each located within the other, we have inferred that $\mathbf{o}$ and $\mathbf{p}$ are exactly located on one (and the same) region. Furthermore, in the case where $\mathbf{o}$ does equal $\mathbf{p}$, we conclude that if $\mathbf{o}$ is located within itself, then it possesses exact location.

This is an interesting conclusion, since vaguely located objects, like Green Dale, do not have exact location, and so do not have regions which they are simultaneously both anchored in and anchored over. So, from the result in the previous paragraph, we infer that such vaguely located objects cannot be located within themselves. However, far from this being damaging for the Anchoring approach, we are of the opinion that it demonstrates a virtue which is in fact absent from other frameworks.

Consider, for example, supervaluation semantics [6], which was proposed as a way of handling vague predicates in logical models of language. Recently it has been mooted as a possible way of handling indeterminate location in spatial information systems [1]. In supervaluation theory, a property may be ascribed to a vaguely described object just so long as that property is possessed under all possible *precisifications* of that description, that is, all precise descriptions of the object to which the vague description may be regarded as being in some sense an approximation. Given this, note that every precisely described object is indeed located within itself, and so, by the tenets of supervaluation theory, a vaguely described object must be located within itself too. This, however, really only makes sense on the assumption that a vaguely described object is in some way to be equated with the totality of its possible precisifications. This runs directly contrary to the spirit of Anchoring, which prefers to recognise a kind of irreducible vagueness for which the notion of precisification is simply inappropriate.

3   Anchoring in an Object-Relational Database

We return to earth now with an examination of the practicalities of incorporating Anchoring into spatial data models.

The OpenGIS Consortium (OGC)'s Simple Feature Specification for SQL [4] details how their conceptual spatial data model (what they call a 'Simple Features' model) can be implemented into a relational or object-relational database. As we mentioned in the introduction, the Anchoring approach stipulates only the locational component of a spatial data model and, as such, we want it not so much to replace but to extend existing spatial data models. In this section we demonstrate how we could integrate Anchoring into an object-relational data model conforming to the OGC specification.

Currently, with reference to our accident example, a specification for a table maintaining information about accidents may, in accordance with the OGC specification, look something like the following.[4]

```
CREATE TABLE Accidents (
id VarChar(20),
location Point );
```

Here accident location is declared to be of type Point, a subclass of type Geometry. In object-relational terms, Geometry is an example of a 'user-defined type', a custom data type which has an accompanying set of related operations. In declaring a field as Geometry, or one of its subclasses, the user then can make use of the various spatial operations that are provided for Geometry. This raises the level of abstraction in communicating with the database. For example, a

---

[4] This is not quite true. A real table specification would need to set up the necessary connections between the point object and a spatial reference system (what we call a precise space) which sits in its own row in another table. We have omitted such connections because they are not part of the substance of our argument. But in a real system the connections would need to be added.

query to return the ids of all accidents located within our now familiar stretch
of motorway between junctions 26 and 27 of the M5 may be written as

```
SELECT a.id
FROM Accidents a, Motorway-Segments m
WHERE within(a.location, m.location)
AND m.start = 26 AND m.end = 27 AND m.name = 'M5'
```

which, in virtue of the use of the within operation, should be decipherable even
for those not familiar with SQL. In implementing Anchoring into the object-
relation model, we will make full use of the benefits of user-defined types. Ide-
ally, after the work is complete, the changes required to have accident location
described in terms of location sets rather than points will be no more than is
shown in

```
CREATE TABLE Accidents (
id VarChar(20),
location LocationSet );
```

Furthermore, since data type operations can be 'overloaded', there will be no
need to make changes to queries such as the one above: in the case of our query,
for example, it will return the id of any accident located within the stretch of
motorway regardless of whether the location of the accident is described by a
location set or by a point.

The reminder of this section will go as follows. First, we look at various ways
that a location set can be structured as a user-defined type. Secondly, after
selecting the most appropriate, we then encode into the user-defined type the
constraints T1–T6 that we gave in §2.4. Finally, we extend the spatial opera-
tion 'within' so that it works with location sets or a mix of location sets and
Geometries.

3.1   Location Set as a User-Defined Type

Described below are three ways of constructing a location set as a user-defined
type. Our butterfly population example, because it includes both anchorings and
relative locations, is used to illustrate each of the three.

The result of a first attempt to place a location set in an object-relation
framework is depicted in Figure 3. As we see, the location of the butterfly, which
in a classical model would have had to be a geometric primitive or construct
therefrom, is now a location set. The location set is pretty much the same as
how we described it in §2.3—which is to say, it contains three anchorings and
one relative location. The only difference between how it was then and how it
is now is in the reference of the the relative location. In §2.3 the reference was
to the actual managed forest object. But now the reference is to the object's
location set. The reason for the change is the nature of the object-relational
model, specifically the fact that table rows do not have object identifiers and so
cannot have object reference. However, having the reference changed in this way
is easily handled: where before we read $\mathbf{b} \sqsubseteq \mathbf{f}$ as '$\mathbf{b}$ is located within $\mathbf{f}$', now we
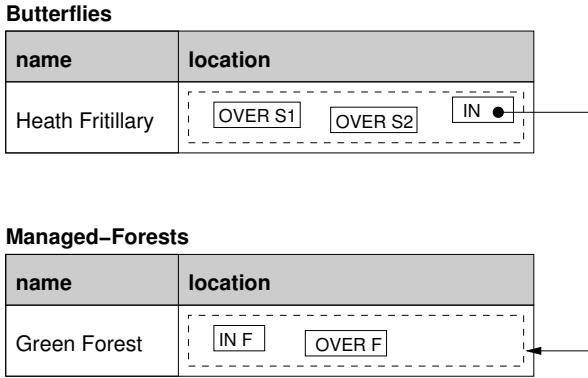
Fig. 3. Location set as user defined type (version 1)

just read it as '**b** is located within the location as described by **f**'s location set'. The change makes absolutely no difference to the workings of the theory.

Location sets are complex. They may contain arbitrary many instances of up to two different types of information and may also have to maintain strings of cross-references to other location sets. Though collection-based user-defined types are possible in an object-relation model (arrays, sets, multisets are all part of the SQL standard), the complexity of a location set may entail that we should maintain its content in separate tables outside of it, as it were. Such a possibility is illustrated in Figure 4. Here anchorings and relative locations are maintained in the tables Anchorings and Relative-Locations, while location set cross-referencing is maintained through object reference.

Fig. 4. Location set as user-defined type (version 2)

Our final implementation of a location set is illustrated in Figure 5. Here object references, which have played an important role thus far, have been removed and replaced with location set identifiers. We do this because object references, though stipulated since SQL 99, have not come into widespread use—e.g, they are still not implemented in the popular open-source database PostgreSQL. This final implementation is what we use below in specifying in SQL the location set constraints and the extended 'within' operation.

Relative–Locations

owner	type	object
1	IN	2

Butterflies

name	location
Heath Fritillary	id=1

Managed–Forests

name	location
Green Forest	id=2

Anchorings

owner	type	region
1	OVER	S1
1	OVER	S1
2	OVER	F
2	IN	F

Fig. 5. Location set as user-defined type (version 3)

3.2 Anchoring Constraints

In an object-relational database implementation of Anchoring, the constraints T1–T6 should be implemented at the table level so as to constrain the kinds of relative locations and anchorings that a location set may or may not contain.

First, the definition of the Anchorings table, which maintains the constraints T1 and T2, may look something like the following:

```
CREATE TABLE Anchorings (
owner LocationSetId NOT NULL,
type CHAR NOT NULL CHECK(type IN ('IN', 'OVER')),
region GEOMETRY NOT NULL,

CONSTRAINT T1
CHECK(NOT EXISTS(SELECT x.id
                 FROM Anchorings x, Anchorings y
                 WHERE x.owner = y.owner
                 AND x.type = 'IN' AND y.type = 'OVER'
                 AND NOT within(y.region, x.region))),
```

```
CONSTRAINT T2
CHECK(NOT EXISTS(SELECT x.id
                 FROM Anchorings x, Anchorings y
                 WHERE x.owner = y.owner
                 AND x.type = 'IN' AND y.type = 'IN'
                 AND disjoint(y.region, x.region))),
);
```

while the definition of Relative-Locations (omitting, for the sake of space, T5 and T6) will be

```
CREATE TABLE Relative-Locations (
owner LocationSetId NOT NULL,
type CHAR NOT NULL CHECK(type IN ('IN', 'OVER')),
object LocationSetId NOT NULL,

CONSTRAINT T3
CHECK(NOT EXISTS(SELECT o.id
                 FROM Relative-Locations o, Anchorings pa, Anchorings oa
                 WHERE o.type = 'IN'
                 AND pa.owner = o.object AND pa.type = 'IN'
                 AND oa.owner = o.owner AND oa.type = 'IN'
                 AND disjoint(oa.region, pa.region))),

CONSTRAINT T4
CHECK(NOT EXISTS(select o.id
                 FROM Relative-Locations o, Anchorings, pa, Anchorings oa
                 WHERE o.type = 'IN'
                 AND pa.owner = o.object AND pa.type = 'IN'
                 AND oa.owner = o.owner AND oa.type = 'OVER'
                 AND NOT within(oa.region, pa.region)))
);
```

3.3   An Extended 'Within'

Before we leave our discussion of the implementation of anchoring, we show an example of one of the extended spatial operations. The spatial operation we use is 'within', which, under [4], is defined as a two-place boolean operation on Geometry and its various subclasses. The operation returns True if the first Geometry is within the second; otherwise it returns False. We now extend this operation to take either two location sets (if both locations we want to relate are described by location sets) or a location set and a Geometry (if one location is described by a location, but the other, keeping to the classical model, is described by a Geometry).

The extended within taking two location sets is described immediately below. Note that we make use here of the (often confused) fact that a Boolean in SQL does in fact take one of three, not two, values. These are 1, 0 or NULL. Consequently we can use SQL to form a three-valued logic if we are careful. However,

since the interpretation of NULL is non-standard (does it mean 'unknown', 'not applicable', or something else), its use is inevitably problematic, and there are supporters and opponents of its use in this way. Here we do not put ourselves in either camp: we just acknowledge the fact that for a proper implementation of Anchoring, a three-valued logic is required.

```
CREATE FUNCTION within(lsA LocationSet, lsB LocationSet) returns Boolean AS
# Is the location described by lsA within the location described by lsB?
IF EXISTS(SELECT id FROM RelativeLocations
          WHERE type='IN' AND owner=lsB.id AND object=lsA.id) THEN
 RETURN 1; # yes

# Is there a region which the object described by lsA is anchored in
# and the object described by lsB is anchored over
ELSE IF exists(SELECT a.region
          FROM Anchorings a, Anchorings b
          WHERE a.type='IN' AND b.type='OVER'
          AND within(a.region, b.region) THEN
 RETURN 1; # yes

# Are the locations of the objects described by lsA and lsB disjoint or
# partially overlapping. Note: disjoint and poverlap will also have to have
# extended defintions.
ELSE IF disjoint(lsA, lsB) or poverlap(lsA, lsB) THEN
 RETURN 0; # no

ELSE
 RETURN NULL; # maybe
END IF;
```

The second extension of within (which follows immediately below) is here to support backwards compatibility with the classical model. After taking a Geometry and a location set, the function first simply casts the Geometry to a new location set whose contents are an in-anchoring and an over-anchoring on the region described by the Geometry. It then passes the two location sets as arguments to the previous function.

```
CREATE FUNCTION within(ls LocationSet, r Geometry) RETURNS Boolean AS
RETURN within(ls, LocationSet(('IN',r), ('OVER',r)))
```

4   Setting Anchoring in Context

Within the field of GIScience the issue of how to handle uncertainty and vagueness is not new. Fuzzy set theory, supervaluation semantics, rough set theory and logical calculi such as, for example, Cohn and Gotts' Egg Yolk theory [2] have all at various times been put forward as solutions to the problem. So why is Anchoring any different? What does it add that the other formalisms lack? Our answers to these questions have in some ways already been addressed in our

previous paper [9], in which we provided a critical analysis of these frameworks in the light of Anchoring. Though we shall not repeat that analysis here, we should note one common criticism, which can be levelled across all these other frameworks. In these other frameworks, there is too much emphasis on introducing *arbitrary* boundaries out of precisely delineated lines which, were it not for the use of the formalism, would not exist. Why, for instance, is one fuzzy membership function better than another? Who stipulates where the inner or outer parts of the egg yolk or rough set are placed? While there may be methods that provide a reasonable basis for drawing out such information (for example, [5,7]), it remains true that this information was not present in the original entity prior to its straitjacketing by the formalism. So, what we sought in Anchoring was a means to say just what we know *for certain* about the entity we seek to capture. Having the perspective in mind of the spatial data provider (such as the UK Ordnance Survey), who deals with many different organisations and users, we developed Anchoring accordingly to fit this picture. Other frameworks, we felt, because of their necessary forced precision, are just not appropriate for a spatial data provider who seeks to be authoritative.

But the discussion does not end here. Since writing our previous paper, we have come to the view that Anchoring is *complementary* to the other formalisms we have mentioned, not opposed to them. That this is the case comes, we think, from identifying two components of a spatial information system: (1) the database management system (DBMS) and (2) the modelling system. The DBMS sits at the heart of any spatial information system. It is here that a spatial data 'feed', such as Ordnance Survey's MasterMap, enters.[5] The various other applications which form part of a spatial information system depend on the DBMS for their data, posing queries and making assertions to it. The DBMS is a *repository* of information. The DBMS needs to store facts about spatial location, but at the moment all it has is the classical model. Replacing this with Anchoring provides more flexibility in the definition of spatiality.

Existing formalisms for handling vagueness and uncertainty operate, by contrast to Anchoring, in the modelling system. The modelling system is used to predict the future, explain the past, or to explore 'what if' scenarios [8]. The modelling system draws on the data in the DBMS and, say, after running experiments, updates that data too. In the modelling system, inference going beyond what we only know for certain is acceptable (and often necessary). Hence the usefulness of these other formalisms we have mentioned. But it follows that the DBMS would then need to have a different kind of locational model to accommodate this representation of vagueness and uncertainty. Perhaps this could have been provided by implementing any one of the other formalisms within the DBMS. However, having the perspective of a data provider in mind has alerted us to the full implications of approximating unnecessarily. A data provider such as Ordnance Survey simply must have the means to represent vagueness appropriately for its needs; to state only *what is known for certain*. Anchoring provides this.

---

[5] `www.ordnancesurvey.co.uk/mastermap`

5   Summary

In this paper we presented Anchoring on a number of different levels relating to its representation within a spatial information system. First, there was the conceptual idea of what anchoring is about in general. This is perhaps best seen in the pictures featuring the precise space and information space. Though not technical in content, the pictures give us a view of a spatial data model that highlights some major shortcomings of how location is handled in the classical model. Second, with the introduction of location sets, we showed how the locational element of the data model can be described by a series of *loc* equations: each one describing the location of an object in terms of the contents of the object's location set. Next, with the addition of the logical theory of §2.4, we saw how the contents of location sets and what information we can infer from that content can rest on a secure, mathematical foundation. Finally, in turning to extending the Simple Feature model of the OGC, we showed that Anchoring is a genuine possibility for real systems.

References

1. Brandon Bennett. What is a forest? On the vagueness of certain geographic concepts. *Topoi*, 20:189–201, 2001.
2. A. G. Cohn and N. M. Gotts. The 'egg-yolk' representation of regions with indeterminate boundaries. In Peter A. Burrough and Andrew U. Frank, editors, *Geographic Objects with Indeterminate Boundaries*, GISDATA 2, pages 171–187. Taylor & Francis, 1996.
3. Maureen Donnelly. Relative places. *Applied Ontology*, 1(1):55–75, 2005.
4. Keith Ryden (editor). OpenGIS implementation specification for geographic information — simple feature access — part 2: SQL option. Technical report, Open Geospatial Consortium Inc, 2005. Available for download at `http://www.opengeospatial.org/specs`.
5. FangjuWang and G. Brent Hall. Fuzzy representation of geographical boundaries in GIS. *International Journal of Geographical Information Systems*, 10(5):573–590, 1996.
6. Kit Fine. Vagueness, truth, and logic. *Synthese*, 30:263–300, 1975.
7. Peter Fisher, Tao Cheng, and Jo Wood. Where is Helvellyn? Multiscale morphometry and the mountains of the English Lake District. *Transactions of the Institute of British Geographers*, (29):106–128, 2004.
8. Antony Galton. Dynamic collectives and their collective dynamics. In A. G. Cohn and D. M. Mark, editors, *Spatial Information Theory*, Lecture Notes in Computer Science, pages 300–315, Berlin, 2005. Springer-Verlag. Proceedings of COSIT 2005.
9. Antony Galton and James Hood. Anchoring: A new approach to handling indeterminate location. In A. G. Cohn and D. M. Mark, editors, *Spatial Information Theory*, Lecture Notes in Computer Science, pages 1–13, Berlin, 2005. Springer-Verlag. Proceedings of COSIT 2005.

Generating Raster DEM from Mass Points Via TIN Streaming

Martin Isenburg[1], Yuanxin Liu[2], Jonathan Shewchuk[1],
Jack Snoeyink[2], and Tim Thirion[2]

[1] Computer Science Division, University of California at Berkeley
[2] Computer Science, University of North Carolina at Chapel Hill

Abstract. It is difficult to generate raster Digital Elevation Models
(DEMs) from terrain mass point data sets too large to fit into mem-
ory, such as those obtained by LIDAR. We describe prototype tools for
streaming DEM generation that use memory and disk I/O very effi-
ciently. From 500 million bare-earth LIDAR double precision points (11.2
GB) our tool can, in just over an hour on a standard laptop with two
hard drives, produce a 50,394 × 30,500 raster DEM with 20 foot post
spacing in 16 bit binary BIL format (3 GB), using less than 100 MB of
main memory and less than 300 MB of temporary disk space.

1 Introduction

Paradoxically, advances in computing capability can make processing more dif-
ficult because they enable the acquisition of larger and larger data sets. For
example, modern airborne laser range scanning technology (LIDAR) makes it
possible to sample large terrains with unprecedented speed and accuracy, obtain-
ing gigabytes of mass points that subsequently must be inspected, processed, and
analyzed [1]. Doing so with existing tools requires powerful computers with very
large memories. To make this data accessible to the wider audience with com-
modity computing equipment, we need algorithms that can process large data
sets without loading them entirely into memory.

Programs that process large data sets must avoid *thrashing*—spending so
much time moving data between disk and memory that computation slows to a
crawl. Algorithms have been designed that guarantee the optimal number of disk
I/O operations for specific problems [2,3], at the cost of greater programming
complexity. Data layouts have been proposed that are I/O-efficient when ac-
cessed with sufficient locality [4,5], at the cost of an expensive initial re-ordering
step. Instead, we advocate a streaming paradigm for processing large data sets,
wherein the order in which computations are performed is dictated by the data's
order [6,7]. We accomplish this by injecting "finalization tags" into a data stream.
These tags allow computations to complete early, results to be written to the
output immediately, and memory to be reused for incoming data.

For stream processing of triangle meshes, Isenburg and Lindstrom [8] propose
a *streaming mesh* format that interleaves vertices, triangles, and *vertex final-
ization tags*. Each finalization tag indicates that all the triangles referencing a

particular vertex have appeared in the stream, and no more will arrive in the future. A simple stream processing application that benefits from streaming mesh input is computing vertex normals: each incoming triangle adds its normal vector to the vectors for its three vertices. The normal vector for a vertex (the average of its adjoining triangles' normals) can be computed and output as soon as the vertex's finalization tag is read from the stream.

This topological method of finalizing mesh geometry has many applications, but does not provide a way to process a stream consisting solely of points. So that we may develop streaming point-processing algorithms, we propose in a companion paper [9] to enhance a stream of points with *spatial finalization tags.* A spatial finalization tag indicates that all the points inside a spatial region have appeared in the stream, and no more will arrive in the future. In Section 2 (and the companion paper), we show how equipping point streams with spatial finalization enables us to compute Delaunay triangulations of huge point sets in a streaming manner. Knowledge about finalized space makes it possible to certify final triangles early, output them, and reuse the memory they were occupying.

In this paper, we use both finalized point streams and streaming meshes to link several streaming software modules, which collectively read huge, LIDAR-generated sets of mass points, produce giant TINs, and generate high-resolution raster Digital Elevation Models (DEMs), with good I/O efficiency and the use of only a small memory footprint.

Our processing pipeline is illustrated in Figure 1. It consists of three concurrently running processes linked by pipes. A *finalizer* module, called `spfinalize`, reads a file of raw mass points and injects spatial finalization tags, producing a finalized point stream. Without touching the disk, this stream is piped into our streaming Delaunay triangulator `spdelaunay2d`, which writes a TIN in streaming mesh format. As the triangles stream out of the triangulator they are—without touching the disk—piped into a rasterization module called `tin2dem`, which uses the triangles and a user-selected interpolation method to estimate elevation values at the raster points of a user-specified elevation grid. Each raster point, with its elevation value, is immediately written into one of many temporary files that are kept open simultaneously; each temporary file represents a few rows of the raster DEM. Finally, `tin2dem` loads these temporary files one by one, sorts each one, and assembles the raster points into a single output DEM.

The streaming triangulator begins to output TIN data long before it has read all the mass points. Thus it continually frees memory for immediate reuse, so the memory footprint of the triangulator remains small, permitting us to create seamless TINs far larger than the computer's main memory. The rasterizer writes out raster points as early as possible and, like the triangulator, discards triangles and mass points that are no longer needed. For linear interpolation, it rasterizes and discards each triangle immediately upon reading it. For quintic interpolation, it rasterizes and discards each triangle as soon as all the triangles within a surrounding two-ring have arrived in the input stream (enabling the computation of second-order derivatives at the triangle's vertices, which are needed for interpolation). When the finalizer, triangulator, and rasterizer run
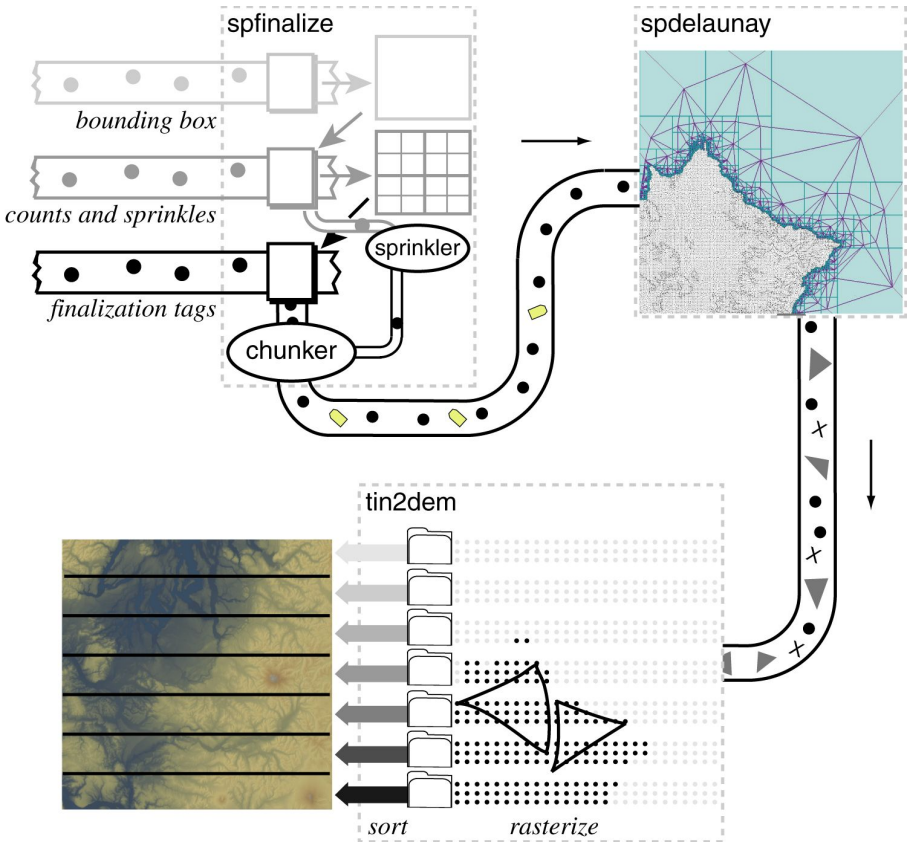
Fig. 1. Our pipeline for generating a raster DEM from mass points. A streaming finalizer performs three read passes over the file of raw mass points, and produces a finalized point stream. A streaming Delaunay triangulator reads the finalized mass points from a pipe and produces a streaming TIN mesh, as described in Section 2. A streaming rasterizer interpolates raster points and stores them to temporary disk files; then it reads them back, sorts the points, and produces a raster DEM, as described in Section 3.

simultaneously, they together occupy a memory footprint that is a small fraction of the sizes of the input and output streams.

The order in which the raster points are created depends on the order in which the triangles can be interpolated, which depends on the order of the triangles' appearance in the streaming TIN, which depends on the order in which regions of space are finalized, which finally depends on the original order of the mass points. For I/O efficiency, the rasterizer does not output raster elevation values directly into the final, row-ordered DEM file; instead it writes raster points (i.e. row and column) together with their corresponding elevation values into temporary files on disk. Each file receives several rows' worth of rasters.

After all the triangles are processed, the raster points in the temporary files are read into memory one file at a time. The points in each file are sorted into the correct row and column order, and the elevation values are written to the final DEM in binary BIL format. A special "no data" value is written for those parts of the DEM that did not receive any raster points. As only a few rows of raster points are in memory at any given time, the raster DEM we create can be far larger than the computer's main memory.

This technique can also be used for streaming creation of a DEM from an existing TIN, but requires converting the TIN to a streaming format [8].

2   Streaming Generation of TINs from Mass Points

In the introduction of the DEM Users Manual [1], David Maune says that raster DEMs can be obtained photogrammetrically,

> however, with most DEM technologies it is more common to first acquire a series of irregularly spaced elevation points from which uniformly spaced elevation points are interpolated. For example, to achieve a uniformly spaced DEM, data producers typically start with mass points and breaklines, produce a TIN, and then interpolate the TIN triangles to obtain elevations at the DEM's precalculated x/y coordinates. ... mass points could have been carefully compiled by a photogrammetrist to reflect key spot heights, for example. Alternatively one can imagine hundreds or thousands of similar mass points randomly acquired by a LIDAR or IFSAR sensor.

Our goal is to process hundreds of millions of mass points, which can be easily gathered by LIDAR and IFSAR sensors. The technology that enables us to do this is a new, *streaming* approach for computing Delaunay triangulations [9] that makes it possible to produce billion-triangle TINs on an ordinary laptop computer. A streaming computation reads data sequentially from a disk or a pipe and processes it in memory, confining its operations to the active elements of the stream maintained in a small memory buffer. To keep the size of this buffer small, a streaming computation must be able to *finalize* elements (e.g., triangles and points) that are no longer needed so they can be written to the output and freed. Unlike with most external memory algorithms, no information is temporarily written to disk; only the output is written. (Our `tin2dem` module makes a necessary exception to this rule; see Section 3.)

Our streaming Delaunay triangulation algorithm [9] is based on the idea of enhancing a point stream with *spatial finalization*. Finalization is a technique for improving the efficiency of streaming computation by including in the stream some information about the future of the stream. The triangulator's input stream is not just a list of points—it also contains *spatial finalization tags* that inform our Delaunay triangulator about regions of space for which the stream is free of future points. The purpose of these tags is to give our algorithm (or other point processing applications) guarantees that allow it to complete computation

in finalized regions, to output triangles (or other data) that can be certified to be part of the correct solution, and to reuse the memory those triangles were occupying—long before all the points in the input stream have been read.

We define a *spatially finalized point stream* to be a sequence of entities, of which all but one are points or finalization tags. The first entity in the stream is special: it specifies a subdivision of space into regions. We use a rectangular $2^k \times 2^k$ grid of cells as our subdivision. Hence, the stream begins by specifying a bounding box (which encloses all the points in the stream, and defines the grid of cells) and the integer k. The remainder of the stream is points and tags. Immediately after the last point that falls in a given cell of the grid, we insert a finalization tag that certifies that no subsequent point in the stream falls into that cell.

Our Delaunay triangulator uses the spatial finalization tags to certify triangles as *final*. A triangle is final when its circumcircle is completely inside the union of the finalized cells. When a triangle is final, it can be written to the output and its memory can be freed to make room for more data to stream in. A vertex is final when all the triangles that adjoin it are final, and likewise can then be written to the output and have its memory freed. The output is written in a streaming mesh format [8], described briefly in Section 1.

We have implemented streaming two- and three-dimensional Delaunay triangulators by taking existing sequential Delaunay triangulators, based on the well-known incremental insertion algorithm [10,11,12], and modifying them so that they retain in memory only the active (non-final) parts of the triangulations they are constructing. (We will discuss only the two-dimensional triangulator in this paper.) The streaming triangulators keep their memory footprints small throughout execution. This makes it possible to compute seamless TINs for huge point sets.

Where do we get a finalized point stream? Ideally, we advocate that programs that create huge point sets should include finalization tags in their outputs. We believe that this option will become increasingly attractive as huge data sets become more common, because it is usually easy to implement, and if it is not implemented, few users will be able to process those data sets with commodity computers. However, we also have written software called a *finalizer* that adds finalization tags to a point stream as a preprocessing step. An important feature of our finalizer is that its output can be piped directly to another application, notably our triangulator, so there is no need to store the finalized point stream on disk.

Our finalizer makes three read passes over a raw point file, and writes a finalized point stream to the output during the third pass. During the first pass, it computes a rectangular, axis-aligned bounding box of the points. The finalization regions are defined by the subdivision of this bounding box into a $2^k \times 2^k$ grid of uniform cells. The second pass counts how many input points lie in each cell. The third pass uses these counts to find the last point in each cell, and to insert a finalization tag immediately after it.

The finalizer also reorders the points, so that all the points that lie in one cell appear contiguously in the stream. Specifically, for each cell of the grid, the finalizer stores the points that lie in the cell in a memory buffer until all the cell's points have arrived. Then it outputs all the cell's points at once into the finalized point stream, followed by a finalization tag for the cell. One motivation for delaying points in the stream is the fact that each point inserted into a triangulation adds far more memory (for triangulation data structures) to the triangulator's memory footprint than the point would occupy in the finalizer's memory buffers. Therefore, we put off inserting a point until it has a chance to be final. Another motivation is to improve the speed of point location in the incremental Delaunay insertion algorithm, by keeping the number of triangles small.

We triangulate huge raw point sets by running our finalizer software, `spfina-lize`, and our Delaunay triangulator, `spdelaunay2d`, concurrently while piping the finalizer's output, a finalized point stream, directly into the triangulator (see Figure 1). The two processes together triangulate the 11.2 GB of bare-earth LIDAR data for the Neuse River Basin of North Carolina, which consists of over 500 million points (double-precision x, y, and elevation coordinates), in 48 minutes on an ordinary laptop computer, while using 70 MB of memory. This time includes the finalizer's three read passes over the raw input points stored on disk, and the triangulator's concurrent writing of a streaming mesh to disk. It is about a factor of twelve faster than the previous best out-of-core Delaunay triangulator, by Agarwal, Arge, and Yi [3]. We have constructed triangulations as large as nine billion triangles, in under seven hours using 166 MB of main memory.

3   Mass Points to DEM

In this section, we discuss our software pipeline for converting mass points into a DEM. We assume that the user has specified the output raster grid, usually by giving the northing, easting, and grid resolution. Our task is first to create a seamless TIN, then to interpolate elevation values at the grid points and create a single raster DEM for output. The first part of our pipeline is the streaming Delaunay triangulator described in the previous section, which creates the TIN. The second part is software that uses the TIN triangles to interpolate the elevation at the raster points defined by the raster grid, then collects the resulting raster points into a single global DEM in some standard order. We consider rasterizing and ordering in the following two subsections.

3.1   Rasterizing the Streaming TIN

We make the following choices for our implementation.

- We take in points with UTM coordinates in a single zone. It is a simple matter of programming to customize the software to user requirements such as generalizing to multiple zones or geographic (latitude/longitude) coordinate

systems. The selected interpolation function needs to match the coordinate system, of course.

– We reject triangles that have an edge longer than a user-defined threshold from being rasterized. This prevents interpolating the elevation across large triangles that can only exist in areas without LIDAR data. These areas are not LIDAR mapped but are inside the convex hull of the LIDAR points, including water surfaces where LIDAR does not give any returns.

– We use linear or quintic interpolants based on TIN triangles. Other forms of interpolation could also be supported, but these two are popular, and they demonstrate key concepts. The linear interpolant requires information from only a single triangle. It is primarily I/O-bound, so it shows the best improvement from streaming. The quintic interpolant [13] requires neighbor information to estimate first and second derivatives at each vertex. This requires more computation and more buffering, as we need two rings of surrounding triangles to compute estimates for the second derivative.

– We output the results to a single DEM in a binary format, primarily to show that our computation is seamless. Of course, a 12 gigabyte raster DEM stored as a single file in row-ordered BIL format is not useful for most applications that use DEM files, because they do not (yet) use streaming computation. We could instead produce tiled output that can be ingested by GIS systems, but we decided to make a single DEM as a proof of concept and I/O-efficiency. Writing tiles or other orderings as output is an easy modification.

3.2   Row-Ordering of Streaming Raster Points

The rasterizer produces the elevation values that make up a raster DEM, but does not produce them in the order that they need to appear in the file—instead, raster points are written out in the order triangles are interpolated, so that the triangles' memory can be freed up to make room for more incoming TIN data.

Since we know the file location where each raster point will ultimately go, we could write it there using random access to a file (or equivalently, using memory mapping and relying on the operating system to swap portions of the elevation data between memory and disk). This strategy is slow because the file is stored in a row-order layout and the interpolator accesses the rows in a random-access manner, entailing many distant disk seeks. Thrashing can be reduced (but not eliminated) by buffering several raster points per row before writing them to disk. The approach has other disadvantages too: it entails an initial pass over the entire file to initialize the entire DEM with "no data" raster points, and a final pass if the DEM is to be stored in a compressed format. Moreover, some computer systems limit main memory address space mapping and random file access (e.g. `fseek`) to under 2 GB, which is not quite enough to store a 32,678 × 32,768 grid with 16-bit elevation values. We want to create larger DEMs.

Agarwal *et al.* [14] use an alternative approach. They write each interpolated elevation value with its column and row address to a temporary file. After all the raster points are written, they use an external sorting algorithm to bring the raster points into their proper order. This strategy produces a temporary file

that is two or three times the size of the final DEM, as the column/row address of a raster point often requires more bytes of storage than the actual elevation value.

We found that a hybrid approach gives the best overall performance. We maintain many temporary files, each representing a few consecutive rows of the output DEM. In memory, we maintain for each temporary file a small buffer of raster points and write out a buffer whenever it fills up. To save temporary disk space, we devised a fast compression scheme that avoids redundant bits when writing a full buffer to disk: raster points are grouped by row; within each row, the raster points are sorted by column and runs of columns are combined (so that a column number does not need to be stored for every point); and elevations are encoded as differences.

The contents of one temporary file are small enough to fit uncompressed in main memory. After the rasterizer finishes processing the streaming TIN data, it decompresses the temporary files one at a time, sorts the raster points into the correct row and column ordering, and outputs them into the final DEM file while putting "no data" values wherever there are no raster points. Our use of multiple temporary files can be seen as a preliminary bucket sort step, which makes the subsequent sorting steps fit in memory.

4   Experimental Results

In Table 1 we report performance, in terms of computation time, main memory use, and temporary disk space, for our streaming pipeline as it generates raster DEMs of various resolutions from LIDAR-collected mass points. The measurements are taken on a Dell Inspiron 6000D laptop with a 2.13 GHz mobile Pentium processor and 1 GB of memory, running Windows XP. The point file is read from, and the DEM file is written to, an external LaCie 5,400 RPM firewire drive, whereas the temporary files are created on the 5,400 RPM local hard disk.

Each run takes as input a binary 11.2 GB file that contains 500,141,313 points in double precision floating-point format, and produces as output a raster DEM in binary BIL format with 16-bit elevations. As points are read from disk they are converted to single-precision floating-point format, which provides sufficient precision for our purpose. Each run constitutes a pipeline of three stream modules (see the command line in the table) that operate in four steps. The first two steps could be omitted if the input points were stored in a format that provides spatial finalization.

1st step. A read pass over the points by `spfinalize` allows it to compute the bounding box. This pass is I/O-limited as it takes 6 minutes to read 11.2 GB from disk.

2nd step. Another read pass over the points by `spfinalize` is used to count the mass points lying in each cell of a $2^9 \times 2^9$ grid derived from the bounding box. These counts are used in the next step for spatial finalization. This pass is also I/O-limited, and takes 6 minutes. So far, relatively little memory is used.

Table 1. Statistics for an input data set of 500 million LIDAR-collected mass points of the Neuse River Basin, from which we compute DEM grids with 40, 20, and 10 foot spacing using linear interpolation. We give an example of how three processes are piped together on the command line. We report time, memory use, and disk use for each step of the computation and each module in the pipeline, at each resolution.

number of points, file size		500,141,313	11.2 GB
LIDAR	$[\min_x; \max_x]$	[1,930,000;	2,937,860]
input mass points	$[\min_y; \max_y]$	[390,000;	1,000,000]
	$[\min_z; \max_z]$	[−129.792;	+886.443]

```
spfinalize -i neuse.raw | spdelaunay2d -ispb -osmb | tin2dem -ismb -o neuse.bil -xydim 20
```

		40 ft	20 ft	10 ft
DEM	grid spacing			
output grid	cols	25,198	50,394	100,788
	rows	15,250	30,500	61,000
	output file size	750 MB	3.0 GB	12.0 GB
Time (hours:minutes)	**total**	**0:53**	**1:07**	**1:20**
bounding box 1st step		0:06	0:06	0:06
finalization counters 2nd step		0:06	0:06	0:06
finalize, triangulate, raster, scratch 3rd step		0:40	0:43	0:48
read scratch, sort, write final 4th step		0:01	0:08	0:20
Memory (MB)	**maximum**	**55**	**64**	**91**
`spfinalize` (1st step)		.1	.1	.1
`spfinalize` (2nd step)		5	5	5
`spfinalize` (3rd step)		35	35	35
`spdelaunay2d` (3rd step)		10	10	10
`tin2dem` (3rd step)		10	19	35
`tin2dem` (4th step)		38	46	91
Scratch Files (MB)	**total size**	**82**	**270**	**868**
(compressed) number of files		60	239	477
rows per file		256	128	128
number of written raster points in millions		111	444	1,767
storage & I/O savings—raw : compressed		8.0 : 1	9.6 : 1	11.9 : 1

3rd step. During the third read pass over the points, a pipeline of three processes do the real work: `spfinalize` adds finalization tags to the stream of points and pipes space-finalized points to `spdelaunay2d`, which immediately begins to triangulate the points and pipes a streaming TIN onward to `tin2dem`, which begins to rasterize incoming triangles onto the DEM grid as early as possible, and writes raster points grouped by rows into temporary files on the second hard drive. This step is CPU-limited, with the three processes sharing the available CPU cycles. The proportion of the workload carried by `tin2dem` increases as the DEM grid resolution becomes finer, as more interpolation operations are performed per triangle.

The memory footprint is that of all three processes combined: `spfinalize` requires 35 MB, most of which goes toward delaying the points in a cell until all the cell's points arrive. `spdelaunay2d` requires 10 MB for streaming Delaunay triangulation. The memory requirements of `tin2dem` depend only

on how many raster points are accumulated per DEM row before they are written to a temporary file in compressed form. For the results we report in the table, each DEM row has its own memory buffer, whose contents are compressed and written to disk when 64 raster points accumulate. Each raster point occupies six bytes—two for the elevation and four for the column index—so the 10-foot resolution DEM with 61,000 rows occupies a 35 MB memory footprint. All three processes together occupy 104 MB.

4th step. In this step, only the `tin2dem` process is still running. It loads the temporary files into memory, one at a time; sorts the rasters into the correct row and column order; and outputs them to a binary file in BIL format, while filling in missing raster points with the number −9999, which represents "no data." This final step, like the first two, is I/O-limited, and the disk throughput is the bottleneck. We would significantly reduce the amount of data written to disk, and realize faster running times, if we replaced the uncompressed BIL format with a compressed format for DEMs.

The memory footprint of `tin2dem` is larger during the 4th step than during the 3rd, because it must simultaneously store in memory all the raster points from one temporary file. For the 10-foot resolution DEM, each temporary file contains 128 rows of up to 100,788 rasters each, so the memory footprint is 91 MB. (Some potential for memory reuse is left untapped.)
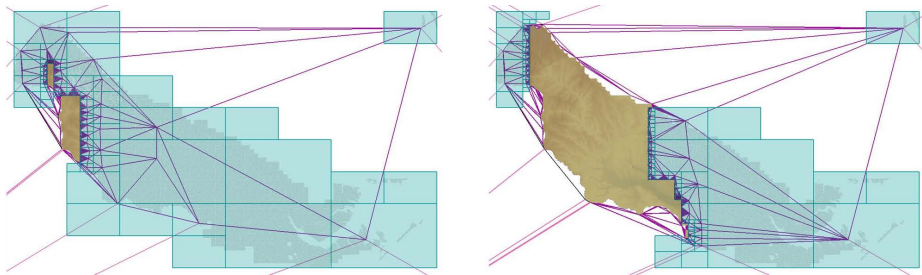
Fig. 2. Two snapshots during streaming construction of a DEM grid from the LIDAR-collected mass points of the Neuse River Basin. Blue cells are unfinalized space, violet triangles are non-final Delaunay triangles, color-shaded areas contain elevation rasters that have already been generated, and the grey LIDAR points inside the blue cells have not yet streamed into the triangulator.

Recent work by Agarwal *et al.* [14] also describes a method for constructing DEM grids from huge LIDAR data sets. They report an end-to-end running time of 53 hours for rasterizing the 20 foot DEM of the Neuse River Basin. This is much slower than the one hour and 16 minutes we report, but most of the difference is attributable to their much more complex interpolation method [15], which takes 86% (46 hours) of their computation time. Most of the remaining seven hours are spent finding neighboring mass points that are needed by the interpolator. This is where we believe our data-driven approach can shine: we

do not have to spend any time gathering neighbor data for our computations. The arrival of sufficient neighbor data is what determines when a computation is performed. However, we are not yet in a position to make a direct comparison between our DEM creation software and that of Agarwal et al.

5   Discussion

The prototype workflow described in this paper can take huge numbers of mass points, build monstrously large, seamless TINs, and output high-resolution raster DEMs. We achieve this with a streaming, data-driven paradigm of computation and an off-the-shelf laptop, using main memory resources that are just a fraction of the size of the input and the output. Our processing pipeline uses spatial finalization of points to enable streaming Delaunay computation, and vertex finalization in the resulting TINs to enable streaming interpolation of elevations and the production of streams of raster points.

Streaming algorithms can succeed only if streams have sufficient *spatial coherence*—a correlation between the proximity in space of geometric entities and the proximity of their representations in the stream. Fortunately, our experience is that huge real-world data sets usually do have sufficient spatial coherence. This is not surprising; if they didn't, the programs that created them would have bogged down due to thrashing. (Our finalizer also increases the stream's spatial coherence by reordering the points.) For data sets with no spatial coherence at all, however, we advocate an external sort of the points before finalizing them.

Our `tin2dem` tool can process pre-existing TIN data. First, however, each TIN needs to be converted to a streaming mesh format, with interleaved vertices, triangles, and vertex finalization tags. For large TINs whose vertices and triangles are ordered with little spatial coherence, this conversion can be expensive [8]. If a TIN is spatially coherent enough, however, a simple streaming conversion method is to make one pass over the TIN file. As soon as a vertex is surrounded by a complete star of triangles, we guess that the vertex can be finalized and inject a finalization tag for it into the stream.

Many other GIS processing tasks could be implemented as streaming computations, enabling them to handle huge data sets. Slope and aspect could be produced immediately by the linear or quintic interpolator. Mass points could come from photogrammetry, IFSAR, or contour lines, instead of from LIDAR. To support breaklines, we plan to extend our streaming Delaunay triangulator to construct *constrained* Delaunay triangulations. For this to remain efficient, we will need streaming input files that encode both points and breaklines in a spatially coherent order. We will also need to devise a method of spatial finalization that takes into account both points and breaklines.

Other forms of interpolation could be streamed too. Streaming natural neighbor (or area-stealing) interpolation [16,17,18] is straightforward to implement, as the Delaunay triangulation is already available. Interpolation methods that compute weighted averages of the k nearest neighbors (typically with $6 \leq k \leq 12$) can also be implemented in a streaming manner, but it takes some care to

devise an algorithm that ensures that the streaming interpolator does not hold too many points in memory, yet does not discard a point that might be a near neighbor of some future point in the stream.

Interpolation methods that use global radial basis functions are not suitable for streaming, or even for global use on large data sets. They can be made a little more suitable by building them into a quadtree, as is done for the regularized spline with tension (RST) [15] implemented in GRASS. However, Agarwal *et al.* [14] has an external memory implementation of a restricted variant of the RST, which uses a bitmap to enforce the locality of the computation, and they compare it to the RST implementation in GRASS. RSTs are not intended to provide a fast, turnkey interpolant, so it is no surprise that they do not work effectively with massive data sets. Their real advantage is having many parameters to tune, to help fit specific characteristics to portions of the landscape.

It will be more challenging to apply streaming processing to less local operations, such as hydrological enforcement [19]. Hydrological enforcement adjusts elevations to respect water flow, so that bridges and culverts do not look and behave like dams. We think that these adjustments could be performed in a streaming manner if summary information on water flow were collected and buffered.

We would like to hear from people who have a specific application and data sets that could benefit from streaming. Suitable applications are those whose operations have some spatial locality, although the degree of locality might be determined by the data.

References

1. Maune, D.F., ed.: Digital elevation model technologies and applications: The DEM users manual. ASPRS, Bethesda, MD (2001)
2. Vitter, J.S.: External memory algorithms and data structures: Dealing with MASSIVE data. ACM Computing Surveys **33**(2) (2001) 209–271
3. Agarwal, P.K., Arge, L., Yi, K.: I/O-efficient construction of constrained Delaunay triangulations. In: Proceedings of the Thirteenth European Symposium on Algorithms. Volume 3669 of LNCS., Mallorca, Spain, Springer Verlag (2005) 355–366
4. Cignoni, P., Montani, C., Rocchini, C., Scopigno, R.: External memory management and simplification of huge meshes. IEEE Transactions on Visualization and Computer Graphics **9**(4) (2003) 525–537
5. Yoon, S., Lindstrom, P., Pascucci, V., Manocha, D.: Cache-oblivious mesh layouts. ACM Transactions on Graphics **24**(3) (2005) 886–893
6. Isenburg, M., Gumhold, S.: Out-of-core compression for gigantic polygon meshes. ACM Transactions on Graphics **22**(3) (2003) 935–942
7. Isenburg, M., Lindstrom, P., Gumhold, S., Snoeyink, J.: Large mesh simplification using processing sequences. In: Visualization '03 Proceedings. (2003) 465–472
8. Isenburg, M., Lindstrom, P.: Streaming meshes. In: Visualization '05 Proceedings. (2005) 231–238
9. Isenburg, M., Liu, Y., Shewchuk, J., Snoeyink, J.: Streaming computation of Delaunay triangulations. ACM Transactions on Graphics **25**(3) (2006) Special issue on Proceedings of ACM SIGGRAPH 2006.

10. Lawson, C.L.: Software for C^1 Surface Interpolation. In Rice, J.R., ed.: Mathematical Software III. Academic Press, New York (1977) 161–194
11. Bowyer, A.: Computing Dirichlet Tessellations. Computer Journal **24**(2) (1981) 162–166
12. Watson, D.F.: Computing the n-dimensional Delaunay Tessellation with Application to Voronoi Polytopes. Computer Journal **24**(2) (1981) 167–172
13. Akima, H.: A method of bivariate interpolation and smooth surface fitting for irregularly distributed data points. ACM Transactions on Mathematical Software **4** (1978) 148–159
14. Agarwal, P.K., Arge, L., Danner, A.: From LIDAR to grid DEM: A scalable approach. In: Proc. International Symposium on Spatial Data Handling. (2006)
15. Mitasova, H., Mitas, L.: Interpolation by regularized spline with tension: I. Theory and implementation. Mathematical Geology **25** (1993) 641–655
16. Gold, C.M.: Surface interpolation, spatial adjacency and GIS. In Raper, J., ed.: Three Dimensional Applications in Geographic Information Systems. Taylor and Francis, London (1989) 21–35
17. Sibson, R.: A brief description of natural neighbour interpolation. In Barnett, V., ed.: Interpreting Multivariate Data. John Wiley & Sons, Chichester (1981) 21–36
18. Watson, D.F.: Natural neighbour sorting. Australian Computer Journal **17**(4) (1985) 189–193
19. Hutchinson, M.F.: Calculation of hydrologically sound digital elevation models. In: Proceedings of the Third International Symposium on Spatial Data Handling. (1988) 117–133

Towards a Similarity-Based Identity Assumption Service for Historical Places

Krzysztof Janowicz

Institute for Geoinformatics
University of Muenster, Germany
`janowicz@uni-muenster.de`

Abstract. Acquisition and semantic annotation of data are fundamental tasks within the domain of cultural heritage. With the increasing amount of available data and ad hoc cross linking between their providers and users (e.g. through web services), data integration and knowledge refinement becomes even more important. To integrate information from several sources it has to be guaranteed that objects of discourse (which may be artifacts, events, persons, places or periods) refer to the same real world phenomena within all involved data sources. Local (database) identifiers however only disambiguate internal data, but fail in establishing connections to/between external data, while global identifiers can only partially solve this problem. Software assistants should support users in establishing such connections by delivering identity assumptions, i.e. by estimating whether examined data actually concerns the same real word phenomenon. This paper points out how similarity measures can act as groundwork for such assistants by introducing a similarity-based identity assumption assistant for historical places to support scholars in establishing links between distributed historical knowledge.

1 Introduction and Motivation

This section describes both the motivation for writing this paper as well as an insight into the idea of and need for identity assumption services within the domain of cultural heritage.

1.1 Motivation

Within the last years similarity measurements gained credence as method for information retrieval and integration within the GIScience community. The research however is mostly focused on inter-concept measurements using several (incompatible and proprietary) knowledge representation formats. Therefore only few real world applications such as a similarity enabled pedestrian navigation service developed by Raubal [1] are discussed in the literature until now. Additionally most conceptualizations published on the web use more or less standardized (description) logic based formalization languages which results in a gap between available theories and available ontologies. Moreover it is claimed that similarity measurements are

closer to the human way of thinking than logic based reasoning services such as subsumption reasoning and therefore deliver more adequate results. Nevertheless there is no elaborate study supporting this theory (at least for GIScience information retrieval scenarios).

The main motivation underlying this paper is to give an insight into how similarity measurements can help in solving real world problems and to show (instead of contrasting both approaches) how similarity theories can interact with existing and well established reasoning services. The presented use case is therefore chosen in a way that it benefits from both, rigid logic based reasoning for knowledge extraction and discovery and flexible measurement theories to return ranked similarity assumptions back to the system user. The paper focuses on spatiotemporal relations, but also takes thematic and referential relations into account for similarity measurements.

The theory presented in this paper is focused on instance rather than concept similarity. It is adapted to fit the identity assumption use case and hence the chosen top-level ontology and knowledge representation format. First steps toward a full grown similarity measurement framework for high expressive description logics (aiming to close the mentioned gap), currently developed at the Muenster Semantic Interoperability Lab (MUSIL), are discussed in [2].

1.2 Introduction

The domain of cultural heritage is very heterogeneous in a sense that the themes or exhibits that museums and related institutions are concerned with range from history of science through all kinds of art and historical documents up to biodiversity. Accordingly the number and type of preserved exhibits rang from millions of collected organisms to a small number of valuable paintings.

Creating and maintaining metadata about historical facts and exhibits gets increasingly important for scholars and curators to structure, manage, and query their own data. As long as metadata is used for internal workflows only (such as the preparation of an exhibition), each institution may develop and maintain their own schema and representation format; however to refine and enrich the own knowledge base or to answer complex scientific questions, interchange with external sources is needed. To support these tasks the Committee on Documentation (CIDOC) provides a well established and standardized core ontology (called CIDOC CRM) [3] intended to annotate heterogeneous cultural heritage information to make it available in a machine readable (RDF) and reasonable way for knowledge integration, mediation and interchange. The vision is to make all annotated datasets available through web (or grid) services to enable automatic metadata harvesting [4] and to form a shared network of interlinked historical information. The CIDOC CRM ontology can be regarded as the underlying semantic level that provides meaning within the intended cultural heritage data infrastructure (which can bee seen analogous to an SDI) by delivering a common metadata schema.

To make use of external data sources, however, a common language is not enough. Moreover it has to be guaranteed that the collected metadata refers to the same real world phenomenon (which could be a historical place, person, event or object) as the local datasets. Global authorities (such as the Alexandria Digital Library Gazetteer

Server [5]) provide unique identifiers and annotated datasets for some common kinds of real world phenomena. Scholars can refer to these global identifiers in addition to (or instead of) their local identifiers and therefore reduce maintenance effort and redundancy on one hand and to enable data interchange on the other. If compared datasets refer to the same global identifier and one decides to trust the global authority as well as the external party that linked their dataset to the specific identifier, it can be assumed that the same real world phenomenon is meant.

Nevertheless until now most datasets do not refer to global authorities and scholars have to decide as the case arises if the harvested information is relevant for the own. This is for several reasons: First, our knowledge about historical places, which are of primary interest within this paper, is often vague and incomplete. Moreover the referring place names are ambiguous and may change during history (however Gazetteer services can be used to disambiguate common place names). The same is true for the geopolitical units the historical place belongs to. Imagine the Turkish city Istanbul, which was founded as Byzantium as part of the Greek Empire; conquered by the Persian Empire; renamed as Nova Roma and Constantinople (called Tsargrad by ancient Slavics) as the second capital of the Roman Empire; later acted as capital of the Ottoman Empire and finally lost the capital status and was renamed to Istanbul in the early 20th century as part of the Turkish Republic. While both the Alexandria Gazetteer and the Getty Thesaurus of Geographic Names [6] contain more than ten alternative (historical) names for Istanbul, the names themselves differ (e.g. Nova Roma is missing in the ADL Gazetteer while Stambul is missing in Getty). Moreover all entries, independent from the historical context connected with the certain place name, refer to the same geopolitical hierarchy (i.e. as part of Turkey). The impact of these shortcomings for the Gazetteer feature types used within the presented similarity measure is discussed later on. In addition to these problems many places are only referred to within historical documents by their role in certain historical events (such as the place where Admiral Nelson died after the Battle of Trafalgar [3] or the spot where a new species was found during an expedition). Such places are not necessarily referred to by spatial relations to other entities or even coordinates. Finally, the most significant reason why global identifiers provided by Gazetteers can only partially solve the problem of identity is that using Gazetteers to determine whether two datasets refer to the same real world place, presumes that all involved institutions have manually annotated millions of local datasets beforehand, which is not the case until now.

Therefore an identity assumption assistant should support scholars in analyzing the harvested metadata and returning promising datasets - promising in a way that the external datasets *probably* refer to the same real world place addressed by the own data. The identity assumption theory used by such an assistant should be non-rigid in a way that it returns a ranked list of estimations instead of trying to automatically conclude safe predictions out of vague historical data. This paper proposes a similarity-based theory that generates such ranked assumptions by comparing CIDOC CRM annotated information (sets of RDF-Triples) about historical places. The proposed theory will be introduced stepwise and elucidated by the scenario "Battle of Trafalgar", specifying the places, actors and events that are being compared.

2 Related Work

This section provides a brief overview about existing similarity measures focusing on those related to GIScience and the CIDOC conceptual reference model.

2.1 Similarity Measurements

The notion of distance is central to the idea of similarity measurements as it determines how close certain aspects of compared entities or classes are to each other. From this perspective, research about similarity is concerned with finding and combining distance metrics for kinds of aspects. Depending on the chosen knowledge representation approach these aspects can be: features, dimensions, transformations, and language constructors; virtually everything that is used to describe the compared classes or entities. In contrast to subsumption reasoning, similarity returns the degree of overlap and therefore is usually a function from compared classes or entities to numeric values (normalized to values between 0 and 1).

The idea of similarity measurement is widely applied across cognitive and information science. An overview about different theories (from cognitive science), their benefits and shortcomings is discussed in [7]. MDSM, a feature-based approach for lightweight ontologies, well established in GIscience is introduced in [8] and extended by thematic roles in [9]. Similarity theories based on conceptual spaces [10] are presented in [1, 11]. A hybrid model is discussed in [12]. A similarity theory for semantic web services is introduced in [13]. Measurements for similarity between different ontologies are discussed in [14, 15]. First steps towards a similarity theory for Description Logics are discussed in [2, 16].

2.2 The CIDOC Conceptual Reference Model

The CIDOC conceptual reference model [3] is a top-level ontology specifying the most fundamental concepts common to all fields of the cultural heritage community. To be that generic, CIDOC CRM does not provide conceptualizations for concrete kinds (such as kinds of exhibits), but defines a framework providing the base terminology necessary for annotation of and reasoning within historical information. While the classes and relationships, which are of major interest within the identity assumption theory discussed here, are introduced in the theory section below (see section 4), this section first gives a broad overview about some characteristics and design decisions underlying CIDOC CRM.

The current release of the CIDOC CRM (version 4.2) is structured into a class hierarchy (allowing for multiple inheritance) specifying 84 top-level entity classes and 141 relations (properties) between their instances (some of them also structured hierarchically). By convention, the names of classes always start with an **E** (entity) followed by a unique number, whereas properties are marked by a leading **P** (property), a unique number and (if an inverse property is defined) the letters **F** (forward) or **B** (backward). The properties are specified by restricting their domains and ranges and with regard to the classes by property quantification (cardinality restrictions). The CIDOC manual however explicitly points out that these quantifications should not be treated as implementation recommendations to allow

incomplete information within the knowledge base (difficulties for the similarity measurement arising from this semiformal kind of specification are discussed in section 4). The classes themselves are specified in an informal way as plain text description, except for their super/sub-class relations. Some of them are declared *abstract*, which means that they have no direct instances.

To adapt the generic CIDOC CRM framework to concrete annotation needs, each institution can define extensions (as long as they are consistent with the existing model) to the core model or use the *E55.Type* metaclass. This class, which's instances are in fact classes again, is intended to support the annotation of concrete types within metadata. In other words instances of *E55.Type* are categories, such as *naval engagement* or *war* for the class *E5.Event,* that are not specified within the CIDOC core model, but in external, application specific vocabularies. Difficulties arising from the extensive use of *E55.Type* are discussed in section 4.2.

A (partial) definition of the reference model is available as RDF schema in [3].

3   The Battle of Trafalgar

The scenario introduced within this section will later on be used to demonstrate certain aspects of the similarity-based identity assumption theory.

The Battle of Trafalgar is one of the most significant naval battles within the Napoleonic Wars and the 19th century. The battle took place during the Third Coalition War and prevented Napoleon's Invasion of Britain, establishing Royal Navy's position as the dominating naval power for more than a century.

Napoleon's strategy was to lure the Royal Navy away from the English Channel by attacking colonies in West Indies, then turn the fleet back to Europe, meet up with the allied Spanish fleet and jointly break the blockade at Brest to attack the remaining British fleet protecting the Channel, to establish a save passage for the French invasion troops. The responsible French Admiral de Villeneuve however ignored Napoleons strict order and sailed to the harbor of Cádiz near the Strait of Gibraltar. To permanently avoid a French invasion, the Royal Navy tried to block his fleet there, but instead of breaking out immediately, de Villeneuve hesitated and did not leave Cádiz until he was informed about Napoleons plan to replace him. The Royal Navy (under command of Horatio Nelson) was already waiting for this moment and attacked the disorganized Franco-Spanish fleet at Cape Trafalgar. The resulting battle was a great success for the British fleet because they destroyed or captured most of the enemies' ships without loosing one of their own. Admiral Nelson however was deadly wounded during the battle.

Of course our knowledge about the Battle of Trafalgar is very detailed and historically well documented; nevertheless the scenario satisfies our requirements as the following questions show: Which spatial relation holds between the naval battleground and the terrestrial cape of Trafalgar? Nelson was wounded during the battle, but did he die during or after it? As similarity measures the degree of overlap between assertions, we expect it to handle ambiguities arising from different perspectives or ontological modeling decisions. The cape itself is located at a strategically prominent position at the Strait of Gibraltar and is therefore relevant for the European history (Carthaginian Empire, Roman Empire, Napoleonic Wars) as

well as the African (Muslim Iberia). Hence the name and the geopolitical assignment to states, empires and provinces has changed over time (note that most gazetteers do not contain the Arabic name). Finally ships, which are of major interest for the Battle of Trafalgar, were frequently renamed (sometimes even several times within one year). Moreover the old names were reused for other ships during the same period. Therefore one cannot conclude from two datasets describing the participation of a ship, referenced by its name, within several historical events, that this particular ship actually sailed from one event to the other. In addition, three ships with similar names were involved in the Battle of Trafalgar called (H.M.S.) Neptune respectively Neptuno.

To measure similarity, all metadata concerning the battle itself and all entities linked to it have to be compared for overlap. To keep the scenario focused and concise, we limit the scenario to the Cape of Trafalgar and some selected, mostly spatiotemporal, relations to other places, events and actors.

4   Similarity-Based Identity Assumption Theory (SIAT)

Places referred to in historical sources (represented in CIDOC CRM as instances of *E53.Place*) probably refer to the identical real world place if they are related through the same or similar relationships to other instances, which themselves again refer to identical real world places, events, actors, or objects. These relations to other instances are annotated as RDF-triples. The more common triples two instances share, the more similar they are and the higher is the probability that both point to the same real world place. However, instances within a knowledge base always represent the approximated and partial knowledge an authority or museum has about a real world phenomenon. Hence, even if two instances share all triples, identity cannot be guaranteed. Therefore the identity assumption assistant delivers estimations rather than assertions. In other words, measuring similarity between real world places means to develop (or apply) distance metrics for their descriptions and to determine their overlap.

This section stepwise introduces the components forming the similarity theory and explains how they are jointly used for identity assumptions. Distance weightings (rephrased to similarity measures) for neighborhoods and hierarchies are discussed as well as inference rules generating new triples out of existing ones.

4.1   Recursive Similarity Function

This section elucidates how the similarity framework compares CIDOC CRM instances by recursively comparing their descriptions (RDF-triples) for overlap, while the concrete distance measures, prototypical expansion rules as well as the identity assumption itself are defined in later sections (see 4.2 and 4.3) .

Similarity Between Predicate-Object-Tuples
As CIDOC CRM annotated metadata is represented by RDF-triples, these triples have to be compared by similarity measurement. Each triple consists of three components: the described resource itself (called subject), a relation (called predicate) and another

resource (called object) linked to the first one by the chosen predicate. Note that the object itself may be the subject of another triple again. In other words a CIDOC CRM instance (subject) is described by its relations to other CIDOC CRM instances. Two RDF-triples are similar if their similar subjects are related by similar predicates to similar objects. Subject similarity however measures the overlap between all (predicate, object)-tuples describing the compared subjects.

Equation E1 defines the similarity (sim_t) for such tuples as the product of the predicate (p_1 and p_2) and object (o_1 and o_2) similarities. While the similarity between predicates (sim_p) is determined in notion of hierarchical or neighborhood distance (see section 4.2) the similarity between the involved objects is just the subject similarity (sim_s); reflecting the fact that those objects are again described by sets of (predicate, object)-tuples. Consequently, similarity does not only depend on the similarity of the referred objects, but also on the kind of this reference.

$$sim_t((p_1,o_1),(p_2,o_2)) = sim_p(p_1,p_2) * sim_s(o_1,o_2) \qquad \text{(E1)}$$

If new triples are generated out of existing ones by inference rules (see section 4.2) and similarity between instances is measured by measuring similarity to other instances, then a maximum search depth has to be specified. This search depth determines the maximum number of expansions (using inference rules) and recursion steps before the measurement terminates. On the one side a low search depth decreases computing time, but on the other side also the expressiveness of the measurement. If the maximum search depth is reached, only the Resource-URIs (values of *RDF:about*) are compared (see section 4.2). However, this is rather a theoretical problem than of practical relevance for most local (cultural heritage) knowledge bases, because they focus on the description of their local exhibits and information and use additional resources/entities only as a kind of reference point.

In terms of the Battle of Trafalgar scenario the knowledge bases would contain all locally available knowledge about the cape (the subject) such as the events that took place there as well as the actors and objects participated in these events and the broader geopolitical units (in other words the objects of interest). The second level knowledge, however, would not again be described in such detail but more generic, while the objects (third level knowledge) involved in those descriptions may be only referenced to by global identifiers. Therefore the local knowledge base would not store all historical knowledge about Spain or even Europe.

According to Equation E1 the similarity derived from comparing the RDF-triples R1 and R2 about *Cape Trafalgar*, is the product of the similarity sim_p between *P89F.falls_within* and *P121.overlaps_with* and the similarity sim_s between *E53.Place(Province Cádiz) and E53.Place(Cádiz)*.

$$\text{P89F.falls\_within(E53.Place(Cape Trafalgar), E53.Place(Province Cádiz))} \qquad \text{(R1)}$$

$$\text{P121.overlaps\_with(E53.Place(Cape Trafalgar), E53.Place(Cádiz))} \qquad \text{(R2)}$$

Even if the compared representations of *Province Cádiz* and *Cádiz* would be equal, the overall similarity for the compared RDF-triples is decreased by the fact, that they describe different spatial relation between *Cape Trafalgar* and (*Province*) *Cádiz*.

Similarity Between Subjects

Real world phenomena are not represented within knowledge bases as single RDF-triple, but as sets of them. Therefore the similarity between all RDF-triples that contain the intended instances as subject or object, have to bet taken into account. To avoid loops during comparison (see Equation E1) all predicates used in given and inferred triples have to be aligned in search direction before similarity is measured. In case of asymmetric relations this means that they have to be replaced with their counterparts. All triples with interchanged subjects and objects are removed from the set of compared triples and are therefore not taken into account for the similarity measurement if an "inverse" triple already exists.

Next the similarities sim_t (see Equation E1) between all (predicate, object)-tuples derived from the RDF-triples describing the local subject (called source) and those describing the compared-to subject (called target) have to be measured. In the following the resulting set of similarities (which is the Cartesian product of the sets of all RDF-triples from the source and the target subject with respect to their similarities) is stepwise processed so that the triples $((p_s, o_s), (p_t, o_t), sim_t)$ with the maximum similarity value for sim_t are saved for further processing and all triples containing either the involved source or target tuple are removed from the set of similarities.

The similarity between two compared subjects sim_s is just the normalized (to [0,1]) sum of these selected similarities (see Equation E2). Note that similarities involving the comparison of predicates between which a meaningful notion of distance cannot be defined (see section 4.2), as well as 'unused' tuples (if source and target are described by a different amount of RDF-triples), are not taken into account and are therefore not element of C (which moreover additionally decreases computing time).

$$sim_s(S_s, S_t) = \frac{1}{|C|} \sum_{i \in C} sim_{t_i} \text{ ; where C:=\{i| i is selected } sim_t \text{ similarity value\}} \tag{E2}$$

In terms of the Battle of Trafalgar scenario, if the source *Cape Trafalgar* is described by the RDF-triples R1, R3, R5 and the target cape is described by R2 and R4, the similarity between the source and target cape is:

$sim_s(S_s, S_t) = \frac{1}{2} *(sim_t((p_{R1}, o_{R1}), (p_{R2}, o_{R2})) + sim_t((p_{R3}, o_{R3}), (p_{R4}, o_{R4})))$

P8B.witnessed(E53.Place(Cape Trafalgar), E7.Activity(Battle of Trafalgar)) (R3)

P8F.took_ place_at(E7.Activity(Battle of Trafalgar), E53.Place(Cape Trafalgar)) (R4)

P53B.is_ former_or_current_location_of(E53.Place(Cape Trafalgar), (R5)
E24.Physical_Man_Made_Thing(HMS Victory))

The set of selected similarities C:={$sim_t((p_{R1}, o_{R1}), (p_{R2}, o_{R2}))$; $sim_t((p_{R3}, o_{R3}), (p_{R4}, o_{R4}))$} used for the computation of $sim_s(S_s, S_t)$ is derived from the Cartesian product of all potential similarities, however no other combinations are possible because distance cannot be measured between spatial and temporal predicates. Note that for demonstration purpose R4 was specified with P8F (forward) and has to be switched to its inverse predicate P8B (backward) before measurement. To avoid back-pointing references, the RDF-triples R3 and R4 are not used later on within the comparisons of the *Battle of Trafalgar* instances.

The RDF-triple R5 does not influence the similarity between the compared instances. Insofar, and in contrast to models such as MDSM that define similarity as the ratio between common and distinguishing features, the similarity theory underlying SIAT can be regarded as a so called common elements approach [7] however it (in contrast to MDSM) supports partial matches.

4.2 Similarity Measures and Their Application Within SIAT

While the previous section describes the similarity framework as such, this section introduces the underlying similarity measures derived from converting distance weightings for predicates, types and identifiers.

Similarity Within Hierarchies and Neighborhoods
The notion of distance (see also [17]) is used within SIAT as generic quantification for the relatedness between universals (predicates or types) arranged within neighborhoods or hierarchies. The similarity weightings discussed here are therefore inverse distance (dissimilarity) measures.

In contrast to theories assuming a constant distance within subsumption hierarchies, SIAT proposes a variable weighting depending on the hierarchy depth. This reflects the fact that abstract universals are less similar to each other than concrete ones, because those already share all features of their ancestors.

$$hsw(u_1,u_2) = \frac{depth(lub(u_1,u_2))}{depth(lub(u_1,u_2)) + edge\_distance(u_1,u_2)} \tag{E3}$$

In Equation E3 *hsw* is defined as the ratio of the hierarchical depth level of the least upper bound (*lub*) of the compared universals (u_1 ands u_2) and the sum of this depth and the edge distance between the those universals. The edge distance is the shortest path, in other words it is the number of edges to be passed from u_1 to u_2. This depth-weighted similarity is only applicable for subsumption hierarchies and used within SIAT to calculate the distance between hierarchically ordered CIDOC CRM predicates and types such as those derived from the ADL Gazetteer feature type hierarchy and the WordNet taxonomy.

The weighting *nsw*, specified in Equation E4, is used to calculate similarity between spatial and temporal CIDOC CRM predicates. Their graphs describe neighboring state changes (either in space or in time) instead of hierarchically ordered predicates with shared features and therefore one can not argue for a depth weighted measure.

$$nsw(u_1,u_2) = \frac{max\_distance - edge\_distance(u_1,u_2)}{max\_distance} \tag{E4}$$

In Equation E4 *max_distance* is defined as the maximal edge distance (longest path) within the neighborhood graph. With increasing graph depth the distance between adjacent nodes (here predicates) decreases, but is independent of the relative position of the nodes within the graph.

Topological Distance and Spatial Reasoning

In CIDOC CRM the class *E53.Place* represents extents in space in the pure sense of physics, independent of temporal and contextual constraints [3]. Scholars create instances of *E53.Place* within their documents to refer to a certain spatial extent on the surface of the earth that is of interest for some reason at a given time during history. The real world extent referred to is present over time while the point of interest may change its spatial disposition or even (temporally) disappear. The places can be identified by instances of *E44.Place_Appellation* which themselves are not necessarily stable over time and therefore may refer to several places. Moreover, historical knowledge is vague and incomplete and therefore the spatial extent described by an instance of *E53.Place* is not known exactly, but instead determined through its relation to other places within the SIAT approach. The points of interest are represented in CIDOC CRM by instance of *E18.Physical Thing* and its subclasses such as E27.Site.

Topological Distance

The CIDOC conceptual reference model distinguishes the following spatial relations between places (see [3] for disambiguation): *P88.consists_of (forms_part_of), P89.falls_within (contains), P121.overlaps_with* and *P122.borders_with*. To measure similarity by comparing the relations to other places, a topological distance between the CIDOC CRM predicates has to be defined. To achieve this, the predicates are mapped (see Fig.1) to those defined in the Closest-Topological-Relationship-Graph [18]. The similarity (n*sw*) is applied to the graph to generate the weightings needed to calculate sim_p for (predicate, object)-tuples involving spatial predicates (see section 4.1). Relations that are topologically close have a higher similarity value and therefore more impact on the similarity of the places they refer to. These places are again compared by taking into account their spatial relations to other places and so on.
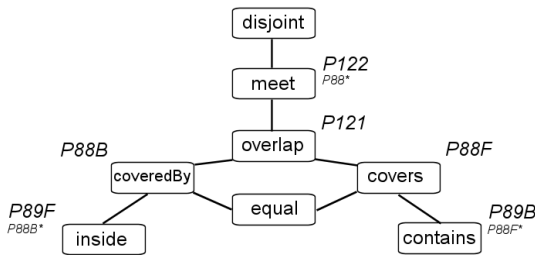

Fig. 1. The CICOC CRM spatial relations within the Closest-Topological-Relationship-Graph

The CIDOC conceptual reference model does not specify relation for *disjoint* and *equal* and therefore no mapping is possible. The properties *P88F.consists_of* and *P88B.forms_part_of* describe the fact that a place can be subdivided into one or more constituents (which are places themselves again), and implies spatial as well as contextual containment. This kind of composition cannot be clearly mapped to one of the topological relationships within the graph and therefore may be assigned to

covers/coveredBy and *meet* (or even *inside/contains*) as well. SIAT maps the *P88* relations to *covers/coveredBy* to refer to the idea of common boundaries (purely spatial) as well as the aspect of containment (spatial and contextual). However this decision is debatable and should be answered by empirical findings within complex real world applications (see section 5).

With regard to the Battle of Trafalgar scenario the similarity between the tuples derived from R1 and R2 about Cape Trafalgar is:

$sim_t(R1, R2)=0.5 * sim_s(E53.Place(Province Cádiz), E53.Place(Cádiz))$

Spatial Reasoning
Besides topological neighborhood, spatial inference is used within SIAT to increase the available amount of place information used for the similarity measurement (see section 4.1). These inferences are drawn from simplified reasoning rules based on the spatial relations introduced in CIDOC CRM and their combination. Each applied spatial reasoning rule generates one or more new RDF-triples. Two rules are illustrated here representative for spatial inference rules in general.

$$AND(P89F(E53(x), E53(y)), P89F(E53(y), E53(z))) \rightarrow P89F(E53(x), E53(z)) \quad \text{(S1F)}$$
$$AND(P89B(E53(x), E53(y)), P89B(E53(y), E53(z))) \rightarrow P89B(E53(x), E53(z)) \quad \text{(S1B)}$$

Rule S1F and S1B generate new triples based on the transitivity of *P89*.

$$AND(P121(E53(x), E53(y)), P89F(E53(y), E53(z))) \rightarrow P121(E53(x), E53(z)) \quad \text{(S2)}$$

Rule S2 infers from the triples (x overlaps_with y) and (y falls_within z) a new triple stating that x overlaps with z. In same cases x could also fall within z, but this could not be concluded for sure.

Temporal Distance and Temporal Reasoning
Besides the relationship to other places, historical events can act as reference points for identity assumptions. If two instances of *E53.Place* are related in a similar way to a certain event, they probably refer to the same real world place. However this implies that the identity of the event can be assumed out of its representation in the database. In CIDOC CRM *E2.Temporal_Entity* is defined as the abstract root class of all perdurants. Its direct subclass *E2.Period* describes (historical) periods as well as all kinds of events (*E5.Event*) which are further distinguished into instances of *E7.Activity*, *E63.Beginning_of_Existence* or *E64.End_of_Existence* (see [3] for further subtypes and disambiguation). All periods are related to at least one place (*E53.Place)* by the *P7.took_place_at (witnessed)* relation.

Temporal Distance
The following relations between temporal entities are distinguished (and related to Allen's temporal logic [19]) within the CIDOC CRM: *P114.is_equal_in_time_to*, *P115.finishes (is_ finished_by)*, *P116.starts (is_started_by)*, *P117.occurs_during (includes)*, *P118.overlaps_in_time_with (is_overlapped_in_time_by)*, *P119.meets_in_ time_with (is_met_in_time_by)*, and *P120.occures_before (occurs_after)*.

To determine the distance between these predicates, we use Freksa's conceptual neighborhood [20]. As our knowledge about historical periods is incomplete and often no crisp starting and ending points are defined, we assume that the 'temporal location' of events [20] is more or less fixed, while their duration varies. Therefore the C-Neighbor structure is chosen as model for temporal distance within SIAT (see Fig. 2).
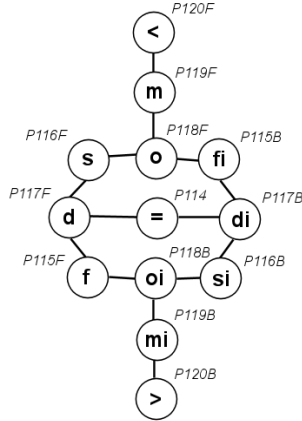

Fig. 2. The CICOC CRM temporal relations within the Freksa's [20] C-Neighborhood

In addition to this purely temporal relation, CIDOC CRM also introduces spatio-temporal relations such as *P9.consists_of (forms_part_of)*, *P10.falls_within (contains)* and *P132.overlaps_with* between periods. These properties cannot be integrated into the temporal conceptual neighborhood because they do not only assume a temporal relation between the periods, but also one between the places at which the periods took place [3].One may argue that these properties can be split up into their temporal and special aspects and then be integrated into the according graphs; however this is not possible for *P132.overlaps_with* because the overlapping relationship within the C-Neighborhood is not symmetric. Therefore the integration of this additional predicates is not discussed here in detail, but left for future work.

In the used scenario, R3 and R4 describe Cape Trafalgar by the fact that it is the place where the Battle of Trafalgar took place. This is the usual way how places are connected to events (and vice versa) in CIDOC CRM. However to determine whether the same battle is meant, the similarity measure has to compare not only the relating predicates (using *nsw*) but also the related-to objects (here temporal entities). If we assume that (beside others) the battle in the source annotation is described by R6 and the one in the target annotation by R7, the similarity is:

$sim_t(R6,R7)=0.63*sim_s(E5.Event(Trafalgar Campaign),E7.Activity(Battle of Cape Ortegal))$

P117F.occurs_during (E7.Activity(Battle of Trafalgar), E5.Event(Trafalgar Campaign))   (R6)

P114. is_finished_by(E5.Activity(Battle of Trafalgar), E5.Activity(Battle of Cape Ortegal))   (R7)

Note that the Battle of Cape Ortegal is one of the three battles of the Trafalgar Campaign.

Temporal Reasoning

The same kind of simplified inference rules discussed for spatial reasoning is also applied to generate new triples out of existing information. Two rules are introduced here representative for temporal inference rules in general.

$$AND(P117F(E4(x), E4(y)), P117F(E4(y), E4(z))) \rightarrow P117F(E4(x), E4(z)) \qquad \text{(T1F)}$$

$$AND(P117B(E4(x), E4(y)), P117B(E4(y), E4(z))) \rightarrow P117B(E4(x), E4(z)) \qquad \text{(T1B)}$$

Rule T1F and T1B generate new triples based on the transitivity of *P117*.

$$AND(P132(E 4(x), E4(y)), P10F(E4(y), E4(z))) \rightarrow P121(E4(x), E4(z)) \qquad \text{(T2)}$$

Rule T2 is the spatiotemporal equivalent to S1. Note that one could moreover infer purely temporal and purely spatial relations between the involved periods and places, which are not discussed here in detail.

Referential Relations

Besides spatial and temporal relationships to other places or events, referential information is of major importance for identity assumptions. These appellations include all kind of names, (structured) phrases, codes and marks intended to identify a certain instance of a given class in a known context [3].

Within SIAT these appellations (besides types) act as termination point for the recursive similarity function because concrete appellations are not compared by (predicate, object)-tuples again, but by external similarity (respectively distance) measures established for given kinds of appellations such as distance between spatial coordinates, notions of prototypical distances between postal codes as well as temporal distances between time points or spans and purely syntactical edit-distance measures between names. However this distances have to be normalized (and if necessary transformed and classified) to values between 0 and 1 before their integration into SIAT. For instance in cases of global identifiers, 0 should be returned for different and 1 for exactly the same global identifier because – at least in the context of cultural heritage – no meaningful notion of distance between identifiers could be defined.

The description of these measures for all available kinds of appellations is out of the scope of this paper and therefore not discussed here in detail. This is especially because we assume that only some well known (broader / upper level) entities such as countries or epochs are annotated by (unambiguous) appellations and focus on a notion of place identity based on vague knowledge about their spatial, temporal and thematic alignment within historical knowledge.

In general referential information can be obtained from Gazetteers[1], databases about historical events or actors, text corpora, several kinds of global authorities as well as from local knowledge or convention. Within CIDOC CRM it is represented by

---

[1] Note that the province Cádiz is represented as point-geometry in the ADL Gazetteer and Getty Thesaurus. The (lat/lon) coordinates between both vary about 50km: whereas those from Getty are the same as the coordinates of the city of Cádiz, the coordinates from ADL point to the center of the province. Comparing these coordinates by spatial distance measures (with Spain as reference) would therefore result in a similarity value interpreted to *nearby* instead of *equal* (even for tolerant equal-buffers).

instances of *E41.Appellation*. Place appellations (*E44.Place_Appellation*) are further distinguished in *E45.Address*, *E46.Section Definition*, *E47.Spatial_Coordinates* and *E48.Place_Name* while *E49.Time_Appellation* (and its subclass *E50.Date*) comprises all kinds of references to time-spans. Both the degree of precision and concrete format of the appellations are not specified or restricted by the CIDOC conceptual reference model. All appellations are related to the referred instances by *P1.is_identified_by (identifies)* and its subrelations.

Note that in CIDOC CRM annotated documents *RDF:about* is used for the concrete value of the appellation (in the sense of an identifier) as well as for a description of the resource itself and therefore SIAT has to interpret RDF:about as predicate (sim_p) and its value by the appropriate external similarity theory (string matching in the worst case) which may have strong influence on the quality of the similarity assessment[2]. However we do not claim that this is indented by the CIDOC model but seems to be usual annotation practice.

Actors and Physical Things

Relationships to prominent actors (*E39.Actor*) or physical things (*E18.Physical_ Thing*) can be applied the same way as the relations to events are used to generate assumptions about places. On behalf of hierarchically ordered relations in general, the participation of actors within events is discussed here briefly.

P12.occurred_in_the_presence_of (was_present_at) is the most generic relation-ship defined between persistent items (*E77.Persistent_Item*) and events within CICOD CRM. Its subproperty *P11.had_participant (participated_in)* restricts the range to actors and describes the active as well as passive participation in an event. To emphasize intentionally (and therefore active) participation in a certain activity (*E7.Activity*), the subproperty *P14.carried out by (performed)* is used. These predi-cates can be compared to each other using *hsw* for the inter-predicate similarity sim_p. Note that as no root relation is defined in CIDOC CRM, the distance between relations not ordered hierarchically (except those for which a neighbourhood is defined such as temporal and spatial relations) cannot be calculated and therefore sim_p is 0 by definition and the according (predicate, object)-tuple has no influence on the similarity between compared subjects.

In terms of the Battle of Trafalgar scenario the comparison of the triples R8 and R9 as part of the similarity measurement between the compared battles is calculated as follows and tends to 0:

$$sim_t(R8,R9)=0.33*sim_s(E21.Person(Nelson), E19.Physical\_Object(HMS Victory))$$

P14F.carried_out_by (E7.Activity(Battle of Trafalgar), E21.Person(Nelson))   (R8)

P12B.was_ present_at (E19.Physical_Object(HMS Victory),E7.Activity(Battle of Trafalgar))   (R9)

Distance Between Types

CIDOC CRM annotated documents make extensive use of the class *E55.Type* and the *P2.has_type (is_type_of)* relation to express all kinds of classifications not further

---

[2] In technical terms this involves constructs such as: <crm:E47.Spatial_Coordinates rdf:about= "Lat: 36.5333, Long: -6.3000">…(where the type of coordinate system can be specified by E55.Type) or <crm:E69.Death rdf:about="Death of Nelson on the deck of H.M.S. Victory">….

distinguished in the core model. Within SIAT we focus on types of *E4.Event* and *E53.Place*, but other instances of *E55.Type* can be compared accordingly.

The ADL Gazetteer [5] and the Getty Thesaurus [6] do not only deliver unique identifiers to unambiguously link place names to certain geophysical or geopolitical entities, but also deliver feature types for these entities. The ADL feature types are considered here because they are commonly used within the domain of cultural heritage and moreover are organized hierarchically. Since no comparable formal definition is available for the feature types themselves, the introduced *hsw* measure is applied to determine how close two types are related to each other. The ADL thesaurus has no common root element and defines six top types (administrative areas, hydrographic features, land parcels, manmade features, physiographic features, and regions) instead [5]. Therefore *hsw* returns the similarity value 0 for types that do not belong to a common top type, which expresses the fact that these types are fundamentally different.

For the scenario this means that the comparison of types specified in R10 and R11 similarity yields 0, because of a missing common super type for *administrative areas* and *regions*[3].

$$P2F.has\_type (E53.Place(Province\ Cádiz),\ E55.Type(administrative\ areas)) \qquad \text{(R10)}$$

$$P2F.has\_type (E53.Place(Andalusia),\ E55.Type(regions)) \qquad \text{(R11)}$$

To specify and compare types of events the, WordNet [21] hypernym/hyponym hierarchy is chosen and "event" (WordNet database location: {00028105}) is defined as top term. Again *hsw* is chosen as to determine the degree of similarity. The nodes within the WordNet taxonomy are not necessarily single terms but synsets (sets of synonym terms). The edge_distance within a synset is set to 0.

WordNet (and *hsw*) can also be used to compare types of *E70.Thing* and *E39.Actor*. This is, however, not discussed here in detail.

4.3 Identity Assumptions

Similarity is measured between instances, whereas identity is assumed for real world phenomena. A high similarity in general indicates that the compared instances are closely related together in terms of the comparable parts of their descriptions. In SIAT this similarity is primarily measured by comparing the spatial disposition and temporal witness [3] of instances representing real world places. Following the law that set cardinality is decreasing with an increasing amount of membership restrictions, even if different real world places share the same topological relations to other common real world places and even if the same real world event took place at different locations at the same time, it is the more improbable to find such real world places, the more common relationships they need to share. However, to draw this

---

[3] The example was chosen intentionally to point out difficulties concerning uncertainty within and between global authorities: While ADL describes Andalusia as *region* and the province Cádiz as *administrative area*; Getty marks Andalusia as *autonomous community* and *first level subdivision* (and therefore as administrative record type) and the province Cádiz as *province* and *second level subdivision*. Note that the ADL Feature Thesaurus also specifies types as *countries, 1st order divisions*, but they are not applied to the examined places.

conclusion, two additional parameters are needed: on the one hand the number (n_t) of compared (predicate, object)-tuples has to be taken into account (for sim_s) and on the other hand a measure has to be specified that allows to consider the information value of the returned similarity. Within SIAT this value (s), that could be compared to the notion of variance, is just the difference between $sim_s(S_s,S_t)$ received from E2 and the similarity obtained by taking into account also those tuples that yield 0 because no meaningful distance between them could be specified.

   Therefore in fact SIAT does not deliver assumptions about identity, but returns a triple (IA in Equation E5) describing how unlikely the compared instances refer to different places. The term unlikely is used here intentionally instead of improbably to point out the vague nature of such assumptions.

$$IA \;=\; \langle sim_s(S_s,S_t),\, n_t,\, s \rangle \tag{E5}$$

Promising identity assumptions are those where the overall similarity sim_s as well as the number of compared tuples n_t are high and the number of tuples that could not be compared and therefore have no impact on the identity assumption is low (reflected by a small s value). However the last named parameter should not be interpreted in a way that high s values automatically exclude an identity assumption.

   In terms of the Battle of Trafalgar scenario, RDF-triples stored in local knowledge bases describe the information about the cape and the battle from the perspective and standard of knowledge of the examined historical sources and therefore may differ in their granularity, perspective and historical reference frame. A document describing the importance of Cape Trafalgar for the history of Spain shares some information about the relation to other places, events and actors with a British view on the Battle of Trafalgar, but contains additional knowledge and focus on other participants.

5   Discussion and Future Extensions

The theory introduced in this paper compares CIDOC CRM instances by extracting their relations to other instances and recursively comparing both, the relations and the related-to instances. Each resulting tuple from the source instance is compared to the most similar of the target instance whereas each tuple is only used once. The process terminates when all instances (each object of a RDF-triple may be the subject of the next similarity measurement step) are examined and only primitive values are left (primitive in a sense that they are the values of *RDF:about* and could not be further decomposed within the CIDOC framework). The similarity between these primitives, describing all kinds of appellations or types, is determined by using external (not recursive) measures such as distances within type hierarchies or between spatial coordinates. If a high similarity value is obtained from a sufficient number of compared tuples, it is possible, but improbable, to find more than one place that meets the required description (which is independent from the question whether the annotations and the historical knowledge reflect 'true' information about the compared real world phenomena or not).

   Promising results (IA-triples) are reported back to the user for further examination. Whenever a scholar decides to trust a certain assumption, the information provided by the external data source can be used to validate or enrich the local knowledge about

the referred real world place. Moreover it becomes possible to establish a persistent link between both data sources used for complex queries. Such queries across multiple databases can provide solutions to scientific questions that could not be answered before.

In contrast to many existing similarity theories the measurement framework presented here focuses on the integration of classical reasoning tasks (to make hidden knowledge explicit and therefore increase the number of comparable tuples) and similarity (to return vague assumptions). Moreover the theory supports partial matches (not possible in models such as MDSM [8]) and integrates spatial, temporal and thematic aspects within one similarity framework.

Nevertheless a lot of work remains to be done and should focus on the application in complex real world scenarios on the one hand and the refinement (supported by empirical finding) of the measurement on the other hand. A fixed set of inference rules should be established for a concrete implementation of the assistant. It has to be examined whether contradicting information expressed in compared RDF-triples should decrease similarity taking into account the vague and incomplete character of historical knowledge. Moreover a theory of trust has to be defined and integrated into the identity assumption theory to indicate how much the user trusts a certain authority. More work is also needed to answer the question how the identity assumptions have to be presented to the user and what kind of additional information is necessary to support scholars in verifying them and their quality. The questions how many compared tuples are sufficient for a precise assumption and whether the similarity of predicates should be weighted differently from the similarity between the related-to objects, cannot be answered beforehand, but only through extensive applications in real world scenarios. Additionally we do not claim that similarity is the only strategy to create identity assumptions from RDF-triples, other approaches have to be examined and integrated into an overall theory.

Acknowledgement

References

1. Raubal, M., *Formalizing Conceptual Spaces*, in *Formal Ontology in Information Systems, Proceedings of the Third International Conference (FOIS 2004)*, A. Varzi and L. Vieu, Editors. 2004, IOS Press: Amsterdam, NL. p. 153-164.
2. Janowicz, K., *SIM-DL: Towards a Similarity Measurement Theory for Description Logics in GIScience*. under review 2006.
3. Crofts, N., et al., *Definition of the CIDOC Conceptual Reference Model (version 4.2)*. 2005.
4. *The Open Archives Initiative Protocol for Metadata Harvesting (Version 2.0): http://www.openarchives.org/OAI/openarchivesprotocol.html*

5. *Alexandria Digital Library Gazetteer: http://middleware.alexandria.ucsb.edu.*

6. *Getty Thesaurus of Geographic Names: http://www.getty.edu/vow/TGNFullDisplay*

7. Goldstone, R. and J. Son, *Similarity*, in *Cambridge Handbook of Thinking and Reasoning*, K. Holyoak and R. Morrison, Editors. 2004, Cambridge University Press

8. Rodríguez, A.M. and M.J. Egenhofer, *Comparing Geospatial Entity Classes: An Asymmetric and Context-Dependent Similarity Measure.* International Journal of Geographical Information Science, 2004. **18**(3): p. 229-256.

9. Janowicz, K., *Extending Semantic Similarity Measurement by Thematic Roles*, in *First International Conference on GeoSpatial Semantics, GeoS 2005, Mexico City, November 29-30, 2005.* 2005, Springer Verlag: Berlin. p. 137-152.

10. Gärdenfors, P., *Conceptual Spaces - The Geometry of Thought*. 2000, Cambridge, MA: Bradford Books, MIT Press.

11. Schwering, A. and M. Raubal, *Measuring Semantic Similarity between Geospatial Conceptual Regions*, in *First International Conference on GeoSpatial Semantics, GeoS 2005, Mexico City, November 29-30, 2005.* 2005, Springer-Verlag: Berlin. p. 90-106.

12. Schwering, A. *Hybrid Model for Semantic Similarity Measurement. 4th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE05).* 2005. Agia Napa, Cyprus: Springer Verlag: Berlin. p.1449-1465.

13. Hau, J., W. Lee, and J. Darlington. *A Semantic Similarity Measure for Semantic Web Services*. in *Web Service Semantics Workshop 2005 at WWW2005*. 2005. Japan.

14. Maedche, A. and S. Staab. *Measuring Similarity between Ontologies*. in *European Conference on Knowledge Acquisition and Management (EKAW)*. 2002. Spain.

15. Ehrig, M., et al. *Similarity for Ontologies - A Comprehensive Framework*. in *13th European Conference on Information Systems*. 2005. Germany.

16. Borgida, A., T.J. Walsh, and H. Hirsh. *Towards Measuring Similarity in Description Logics*. in *International Workshop on Description Logics (DL2005)*. 2005. Scotland.

17. Rada, R., et al., *Development and Application of a Metric on Semantic Nets.* IEEE Transaction on Systems, Man, and Cybernetics, 1989. **19**(1): p. 17-30.

18. Egenhofer, M. and K. Al-Taha, *Reasoning About Gradual Changes of Topological Relationships*, in *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, A. Frank, I. Campari, and U. Formentini, Editors. 1992, Springer Verlag: Berlin. p. 196-219.

19. Allen, *Maintaining Knowledge about Temporal Intervals.* Communications of the ACM, 1983. **26**: p. 832-843.

20. Freksa, C., *Temporal Reasoning Based on Semi-Intervals.* Artificial Intelligence, 1992. **54**(1): p. 199-227.

21. WordNet: *http://wordnet.princeton.edu/.*

Coupling Bayesian Networks with GIS-Based Cellular Automata for Modeling Land Use Change

Verda Kocabas and Suzana Dragicevic

Spatial Analysis and Modeling Laboratory, Department of Geography,
Simon Fraser University, 8888 University Drive, Burnaby BC, V5A 1S6, Canada
{verdak, suzanad}@sfu.ca

Abstract. Complex systems theory and Cellular Automata (CA) are widely used in geospatial modeling. However, existing models have been limited by challenges such as handling of multiple datasets, parameter definition and the calibration procedures in the modeling process. Bayesian network (BN) formalisms provide an alternative method to address the drawbacks of these existing models. This study proposes a hybrid model that integrates BNs, CA and Geographic Information Systems (GIS) to model land use change. The transition rules of the CA model are generated from a graphical formalism where the key land use drivers are represented by nodes and the dependencies between them are expressed by conditional probabilities extracted from historical spatial datasets. The results indicate that the proposed model is able to realistically simulate and forecast spatio-temporal process of land use change. Further, it forms the basis for new synergies in CA model design that can lead to improved model outcomes.

1 Introduction

Geographic Information Systems (GIS) are well established as tools for the storage, handling, analyzing, and visualizing of spatial data [1, 2]. Despite these advantages, GIS is not well developed for handling the temporal component of data [3]. The current GIS have a limited capacity to handle short time step iterations needed for modeling dynamic spatial phenomena. There is, therefore, an urgent need to understand the spatio-temporal process itself, how to formalize the process and the problem as a geographic model, and the method to solve the problem within GIS frameworks.

The development of GIS based procedures that can handle the dynamics of geographic phenomena through the integration of complex systems theory is an important area of GIScience research. Complexity theory has been used extensively to model land use change processes together with cities and their evolution. Land use change and urban growth are characterized by a large number of interacting components and have several key signatures such as fractal dimensionality, self-similarity, self-organization and emergence that make them suitable to be modeled as complex systems [4-6]. For more than two decades, complex systems theory and cellular automata (CA) have been used to handle the dynamics, complexity, and self-organizing properties of land use change processes. The spatial complexity and dynamics of land use

change is represented by selecting various configurations of the basic elements of the CA model design - cell space, cell size, neighborhood size and type, transition rules and temporal increments [7-9]. Transition rules are the most important element of the CA as they control the behavior of the cells and their evolution into the future states. Recently, there has been a concerted effort to improve the transition rules [10, 11] or combine other approaches such as multi-criteria evaluation, principal component analysis and neural networks among others [12-14] to advance GIS-based CA.

Bayesian Networks (BNs) provides an alternative approach based on artificial intelligence that can overcome some of the challenges of CA models. In the late 1970s and 1980s, rule-based approaches were common in artificial intelligence (AI). Although neural networks were popularized later, they have failed when sufficient data are not available for data learning. In the late 1980s, BNs were seen as an efficient way to deal with data uncertainty [15, 16]. The early applications of BN were in medical diagnosis and genetics, but recently their use have expanded to areas such as environmental studies [17] and geographic information systems [18].

The Bayesian Networks are considered as probabilistic network-graphical models that use probability and graph theory in their implementation [19]. The advantage of probabilistic networks is that they provide explicit representations of dependencies or independencies between variables without scientific numeric or functional details [20]. As a probabilistic network, the BN representation was originally designed to model the uncertain knowledge of an expert to deal with complex systems and data uncertainty. In BN models, simple parts (such as land use drivers) of the complex system (such as land use change process) are constructed using graph theory. The parts are then combined to each other using probability theory [21]. BNs are used for probabilistic inference and based on Bayes' Theorem, which is a mathematical formula to calculate probabilities among several variables that are causally related [22].

The objective of this study is to improve existing CA models by proposing a hybrid GIS-based Bayesian network cellular automata model. The theoretical framework for this study is the integration of cellular automata with Bayesian networks since the formulation of cellular automata transition rules should depend on how driving factors of land use change are perceived. Thus, land use change is seen as the result of the interplay between land use drivers that mimics the complexity of a spatial process. Sensitivity analysis was used to test the model for changes occurring in the model outcomes when the number of nodes and land use classes are changed.

2   The BN-CA Model

Cellular automata are discrete spatial models [9]. They consist of an array of cells, each of which has cell states. The state of a cell at consecutive time *t+1* is a function of its state, its neighbourhood, and set of transition rules at an initial time *t*. Using this function, transition rules are applied to each cell to determine what state it should

change to during a time transition. This step is repeated over the whole cell array. The cell state can be mathematically represented as:

$$S_i(t+1) = f(N_i(t), S_i(t), T) \tag{1}$$

where $S_i(t)$ and $S_i(t+1)$ are the states of cell i at the initial and consecutive time t and $t+1$ respectively; $N_i(t)$ is the neighborhood state of cell i at time t; T are the transition rules. In general, the transition rules are in the form of <IF, THEN, ELSE> statements. For example, IF an event occurs in the neighborhood of a cell, THEN some-other-event occurs to the cell [23].

In land use change modeling, these rules represent how change occurs in the real world. Since there is no standard procedure or method for defining transition rules, they are reformulated in the spatial modeling literature using probabilistic expressions, accessibility algorithms, logistic regression, linguistic variables, multi-criteria evaluation methods, and neural network structure among others. However, when processes and changes in the urban area are difficult to describe (e.g. large number of factors affecting land use change), then most of these methods fail to adequately capture the land use change process. Neural Networks (NN) [14] have been used to improve the capability of CA models to deal with multiple land uses. With the NN-CA models, the parameters required for the simulation are determined by a training procedure and no transition rules are required. However, they are black-box models when incorporated in the CA structure. Explicit knowledge about the modeled land use change process is not provided. In addition, the optimal structure for the numbers of network layers and neurons is still unclear for a specific application [24]. Hence, there is a need to make the design and implementation of existing CA models more explicit.

The alternative method of Bayesian Networks is proposed to simulate land use change in cellular automata spatial models. A Bayesian network is the pair *(G, P)* where G is a Directed Acyclic Graph (DAG) where nodes represent variables, arcs between nodes represent probabilistic dependencies, and *P* is a multivariate probability distribution defined on variables that correspond to the nodes of *G*. A graph is called *directed* if the graph links have directions. A directed graph is *acyclic* if the graph contains no directed cycles.

The elements of BN are [19]:

1. Variables: A set of *variables* and a set of *directed edges* between variables,
2. States: Each set contains a finite set of mutually exclusive states,
3. Structure: The variables coupled with the directed edges form a DAG,
4. CPT: Each variable *A* with parents *B*1, *B*2… *Bn* has a Conditional Probability Table (CPT) which includes *P* (*A* | *B*1, *B*2… *Bn*).

If directed edges (arcs) in the DAG are assumed to represent causality, then BNs are sometimes called causal networks. However, when building BN models, users and experts do not need to see the links as causal relationships. Rather they should ensure

that links correspond to qualitative relationships. In order to explain the basic BN ideas, consider the example in Figure 1. The circles are nodes and they are the variables. They represent the most important factors about the particular phenomenon. They are linked so that a change in one will result in a chain reaction of impacts on all the linked variables in the direction of the links, assuming that the links represent causality. Each variable is probabilistically independent of its non-parents given its parents. Therefore, the absence of a direct link between A and G means that the influence of A on G results from other variables (e.g. C, D, E). The design of the network such as deciding which factors link to each other is based on how the phenomenon being modeled is perceived. For simplicity, assume that all the nodes are binary variables that take a value of either true (T) or false (F). While the states of the variables can be discrete, they can also be real valued, integer valued, or multivariate [25]. For each node, there is a conditional probability function that relates this node to its parents. For instance, the probabilistic relationship between D and its parent C is the *conditional probability distribution of D given C*. This is expressed in the conditional probability table shown in Figure 1. In the table, P_{00} means probability of D being false given C is false, that is $P(D = false \mid C = false)$.

The important characteristic of Bayesian Networks is their explicit representation of the conditional dependence and independence between variables [26]. The probability of every possible event as defined by the values of all the variables is called the joint probability distribution. It has been shown [19] that the joint probability distribution induced by the DAG can be factorized into the conditional distribution of each variable with respect to its parents (Equation 3):

$$P(X_1 ... X_n) = \prod_1^n P(X_i \mid pa_i) \tag{2}$$

where pa_i is the set of direct parents for variable X_i. Therefore, the probability distribution represented by the example network from Figure 1 is:

$$P(A,B,...,G) = P(A)P(B)P(C \mid A,B)P(D \mid C)P(E \mid C)P(F \mid D,E)P(G \mid C,D,E) \tag{3}$$

2.1 Model Framework

In the proposed model, the transition rules T in the equation (1) are equal to *(G, P)* and the cell states are defined as:

$$S_i(t+1) = f(S_i(t), T(G,P)), \tag{4}$$

which implied that the cell state at time *t+1* depends on the current cell state and transition rules defined by *G* and *P*. The proposed model eliminates the use of neighborhood type and size in the transition rules since it is proved that CA model outcomes are sensitive to the changing neighborhood size and type [27]. In the next section, the details of the model building process will be elaborated.
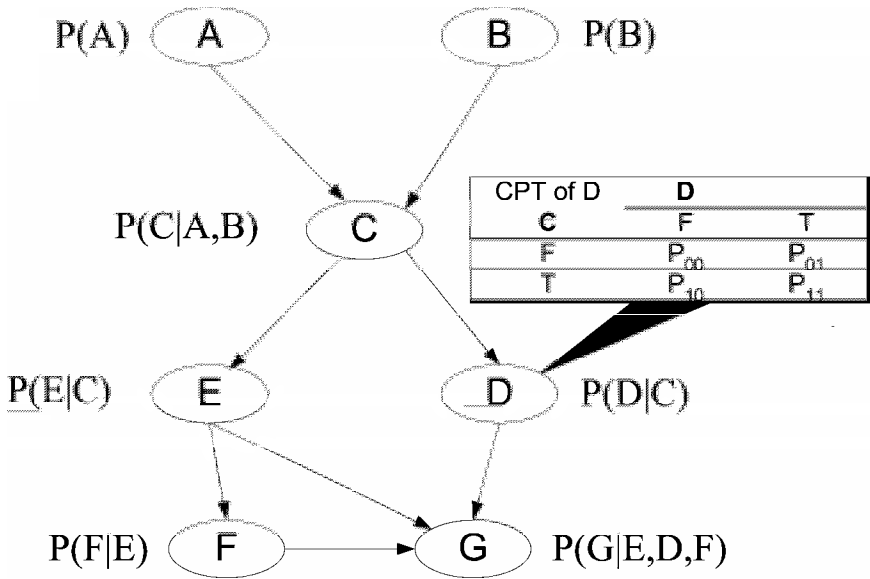
Fig. 1. A simple Bayesian network structure containing seven variables

2.2 Building the BN-CA Model

Defining Input Raster GIS Layers and Bayesian Network Nodes. Multiple GIS layers provide the data input to this model and they represent the key land use drivers that affect the land use change. The GIS operations such as reclassification, distance calculations, suitability evaluation, buffering, and overlay are used to derive suitable areas and constraints as input layers. These input GIS layers are defined as nodes in the BN structure. In addition to land use driver nodes, future land use state is defined as a node in the BN structure. Consequently, probabilities of each future land use state conditional on land use drivers are calculated. The values of the nodes are represented as discrete or continuous or both.

Structuring the Bayesian Network. Once the variables and their values are identified, the next step is to identify the structure of the BN and how the nodes are connected together. The structure should capture the relationships between variables. Nodes should be linked such that if one affects or causes the other, there should be an arc between them. The direction of the arc represents the causation.

The experts can construct the structure and quantify probabilities in the network by using their knowledge. Nevertheless, some researchers [28] have argued that instead of trusting experts, the BN should be constructed from the observed data in a learning process. This provides a better way to interpret the existing (observed) data in an accurate manner and to better understand the phenomenon that is being modeled. In land use change modeling, this gives an indication of what variables are the main factors in the land use change process.

Learning in BN depends on whether the network structure is known and whether the variables are all observable [29, 30]. In this study, the BN structure is extracted using the data at two temporal snapshots of the study site to find the factors underlying the land use change process. Figure 2 depicts how the BN sub-model is conceptualized in terms of observed data and predicted data. The data of the site are observed but the BN structure is not known. In this case, given the set of variables, the links must be found and then parameters as the values in the Conditional Probability Table (CPT) must be estimated using the observed data (data at time t and $t+\Delta t$ of the site).
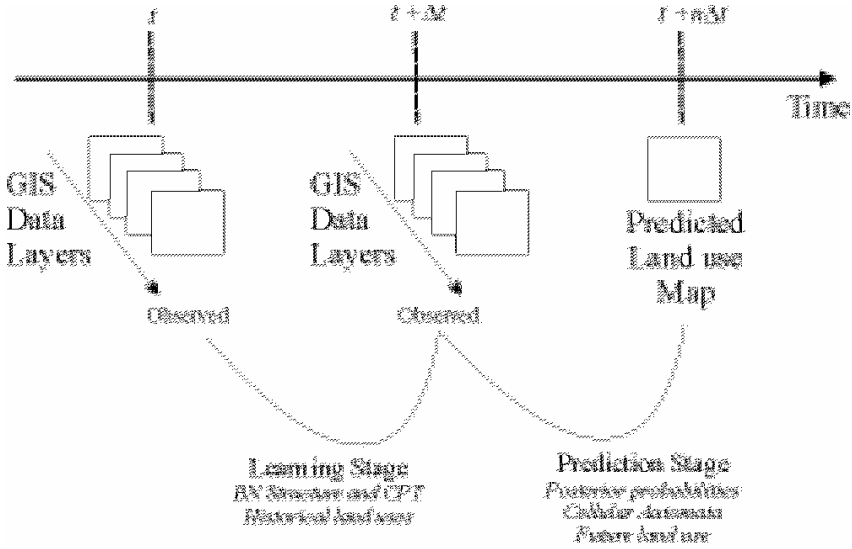

Fig. 2. GIS based BN-CA model: conceptual framework

The K2 algorithm [31] which uses Bayesian scoring method was used in this study to learn the BN structure from two observed land use datasets. The K2 algorithm provides an efficient way to structure learning from complete observable data [32]. The algorithm starts with a network with no links; then for each node, it incrementally adds parents whose addition increases the score of the resulting structure until the addition of parents does not increase the score.

Estimating Conditional Probability Table (CPT) in the BN Structure. In the process of BN structure learning, there are several possible ways to obtain estimates for the conditional probabilities in the CPT. It is possible to use subjective probabilities, usually encoded from expert knowledge when the data available for a particular variable are limited or non-existent. The alternative to subjective probabilities is learning, which is achieved by calculating the conditional probability table values using estimation techniques such as Maximum Likelihood Estimation (MLE) and Bayesian estimation.

In this study, since the network structure was learned from observed data in the previous step, the structure is known and complete data are available to derive the probabilities. This is the most studied case of learning BN in the research literature. MLE estimation was applied to compute the probabilities.

Inference - Predicting Future Land Use. The last step in building the BN sub-model of the BN-CA hybrid model is the prediction done by inference. The computation of a probability of interest given a model is called probabilistic inference [33]. In this step, observed variable states are entered as evidence in the BN to calculate the revised probabilities of interest given the evidence. In this study, the Junction Tree Algorithm was used to calculate the inference in the BN [34]. As shown in Figure 2, the observed data at time $t+\Delta t$ are entered into the BN as evidences and the probabilities of each land use state at time $t+n\Delta t$ are obtained.

For each cell, the BN inference is used to obtain the probabilities of each predicted land use state. Then, the transition rule of the CA changes each cell's land use state to one with the highest potential. Each cell is subject to inference and transition rule in each iteration.

3   Model Implementation and Simulation Results

3.1   Study Site

A hypothetical study site was created at 25m spatial resolution with 900x1363 cells. The two hypothetical snapshots of this site with the temporal interval of 10 years $(\Delta t=10years, n=2)$ were used in the simulations of the proposed model. Figure 3 depicts the new residential development that was created along the undeveloped land. In these land use maps, there are ten common land use classes, namely residential, industrial, commercial, office, schools, recreation, green areas, agriculture, undeveloped land and others (such as water bodies, roads).

3.2   Data Preparation

In the hybrid model, the first step is to identify the key variables-land use drivers that affect the land use change process since changes in the types of land use are induced by these drivers. A total of ten spatial variables were identified as key factors that affect future land use change. These variables are: distance to education facilities, distance to existing transportation network, distance to commercial centers, distance to employment centers, distance to recreational facilities, distance to green areas, constraints (water areas, steep sloped, flood areas), land use policies, land ownership and current land use (Table 1). These layers as variables were obtained by using raster GIS analyses. The Euclidean distances in the ArcGIS software were used to calculate the distance variables and to classify them into three main classes: 'good', 'medium' and 'low' accessibility to the land use driver being considered.
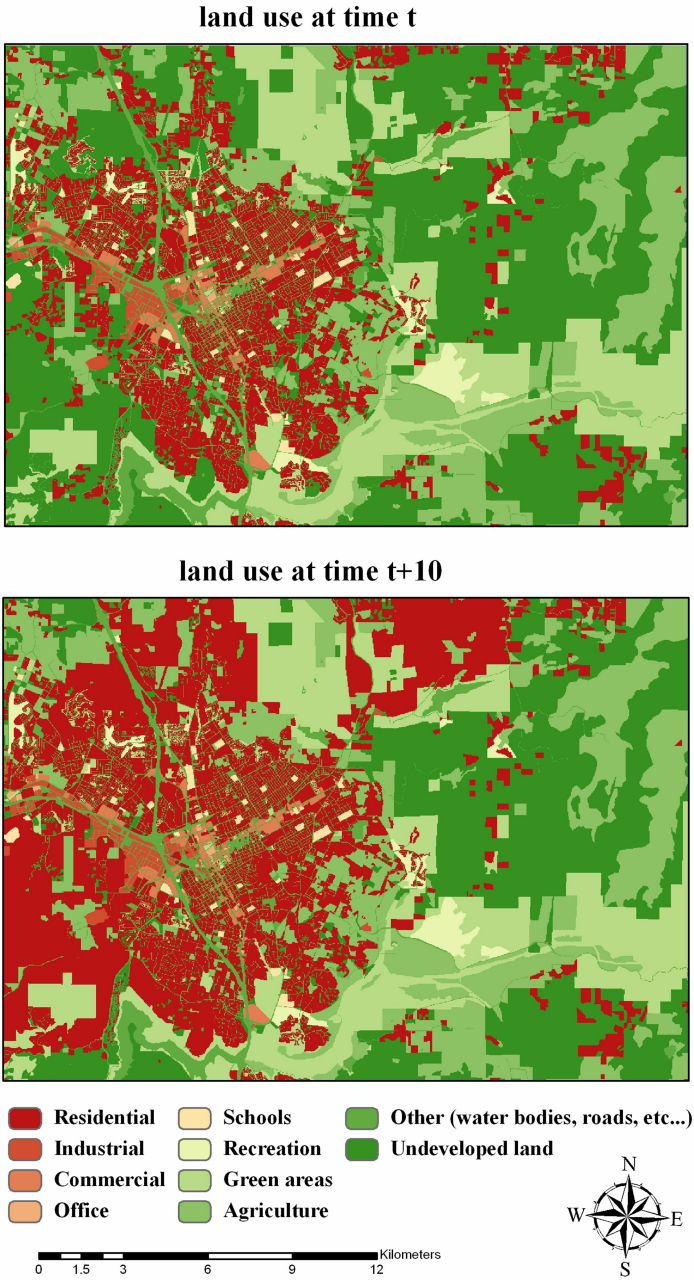
land use at time t

land use at time t+10

Fig. 3. Hypothetical site with land use classes at time *t* and *t+10* years

Table 1. Input variables of the proposed model

Land use drivers/variables	Raster GIS calculations
Distance to education	Euclidean distance from every cell to the nearest schools, universities, colleges.
Distance to existing transportation network	Euclidean distance from every cell to the nearest street network
Distance to commercial centers	Euclidean distance from every cell to the nearest commercial areas, such as shopping malls
Distance to employment centers	Euclidean distance from every cell to the nearest office spaces and business areas
Distance to recreational facilities	Euclidean distance from every cell to the nearest recreational attractions, such as tourist attraction sites, stadiums, etc
Distance to green areas	Euclidean distance from every cell to the nearest green areas, such as residential parks, hillsides, etc
Constraints	Physical and policy constraints to the development such as conservation areas, flood plains, steep slopes, etc
Land use policies	Government policy and plans on future land use
Land ownership	Public land ownership (land owned by schools, states, forest service, etc) and private land ownership
Current land use	Current land use

3.3 Simulations

The proposed BN-CA model was applied to the hypothetical study site with a spatial resolution of 25m and temporal resolution of 10 years. The study site data at time t and $t+10$ years were input to the model and the land use map at $t+20$ years was predicted $(\Delta t=10$ years, $n=2$ iterations that were generated). The algorithm of the proposed model was coded in the Matlab software using some of the functions of the Bayes Net Toolbox [35]. The proposed BN-CA model uses a loose coupling architecture with the ArcGIS and Matlab software.

All the variables, namely land use drivers were incorporated into the BN as nodes. In addition to those, as a last node, predicted land use was defined in the BN structure to obtain the probabilities of each future land use state conditional on the others. The distance variables have three states (good, medium, and low accessibility), constraints and ownership have two states (public and private ownerships), and policy and land use variables have ten states representing the land use classes.

Bayesian Network employs two important operations: *explanation* and *prediction.* The explanation part was accomplished by a BN learning procedure in which the structure of the network was constructed and values of the CPT were estimated from the observed data at initial time t and consecutive time $t+10$ years. Although all cells could be used for structure and parameter learning, 1000 cells from the raster GIS layers at time t were chosen randomly for the learning algorithms. When the number of cells required for learning increases, the sample complexity increases and creates computational complexity. Also, the performance of the learning improves with increasing the number of observed cells. The result of the structure learning procedure is shown in Figure 4, which explains the underlying processes and interactions in the
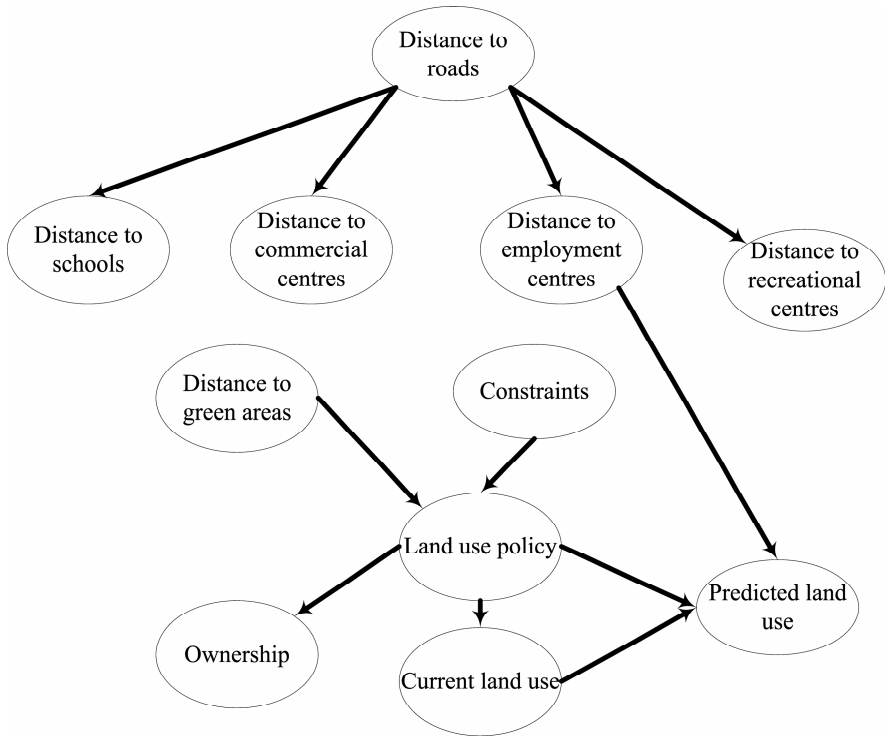
Fig. 4. Learned BN structure

study site. Distance to existing transportation network affects other distance variables, except green areas. In addition, distance to employment centers, land use policies and current land use directly affects the future land use change.

Prediction was employed by the probabilistic inference and cellular automata transition rules. With the inference algorithm, the probabilities of each future land use state were calculated. Then, transition rule decides the change from one land use state to another depending on the highest probability. Figure 5 illustrates the land use states predicted for 20 years, at *t+20* generated by the proposed model. The results imply that the predicted land use generated by the model show similar growth to the current land use change trend where new residential areas appeared on the undeveloped land.

4   Model Sensitivity

Sensitivity Analysis is useful in spatial modeling to identify what parts of the model are critical and which ones are less likely to be important to the results [27, 36]. The proposed model's sensitivity was tested by running the model with different configurations. There are two sensitivity areas of the model: *node sensitivity* and *classification sensitivity*.
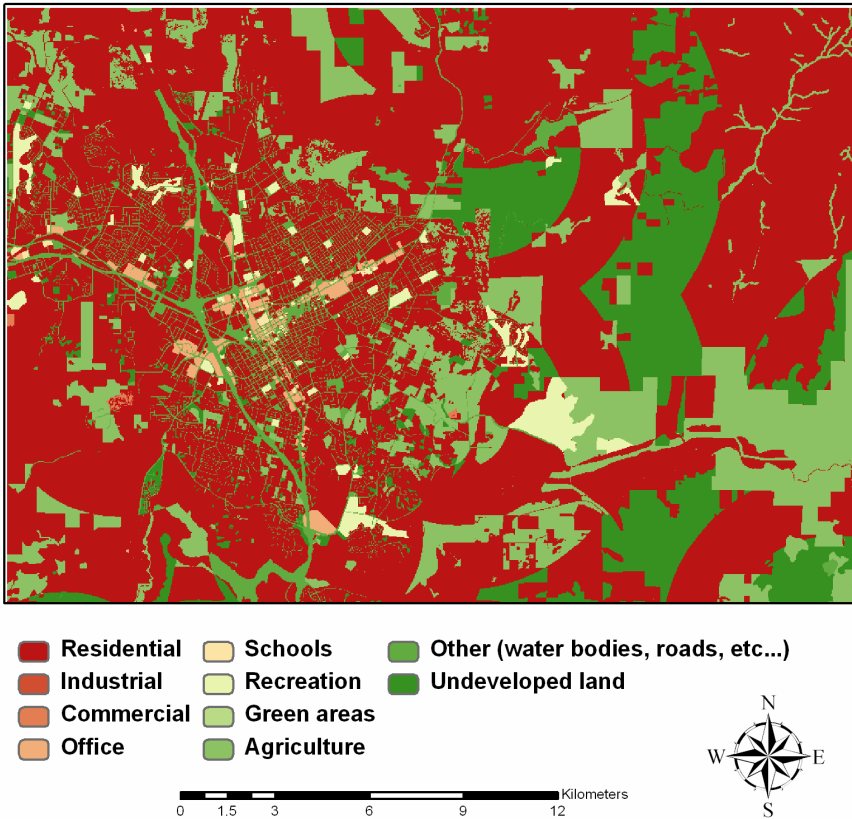
Fig. 5. Predicted land use generated by the proposed hybrid model at time *t+20* years with using eleven nodes

In testing the node sensitivity of the model, different number of nodes was incorporated into the BN sub-model. As a result, different simulation results were obtained. For example, Figure 6 shows the predicted land use map when seven nodes (only six distance nodes and future land use node) were employed. It can be seen from the figure that the resultant map has only seven land use classes and the model did not generate residential growth. In addition, large school areas were generated by the model. This shows that model is very sensitive to the number of nodes. During the modeling process, nodes should be chosen meaningfully. The key is to determine the most important nodes. It should be noted that the number of the nodes ought to be pruned due to computational complexity that can arise from large number of nodes. This is because as the number of nodes increases, the joint probability distribution grows exponentially and creates computational complexity in the calculation of probabilities.

Apart from the node sensitivity, there is also classification sensitivity on how land use categorization and classification affect the land use change models [37]. In this
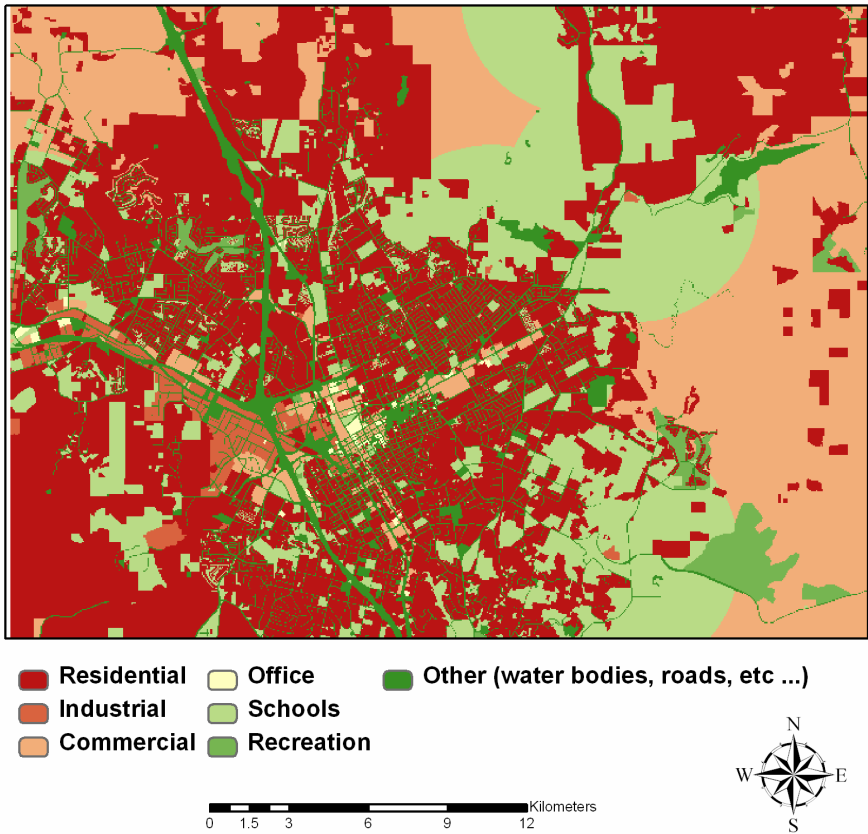
Fig. 6. Predicted land use at time *t+20* years when seven nodes are used in the model

study, simulations were performed with a variety of land use classes; from three to fourteen, and the outcomes of the model with different land use classes were analyzed. Figure 7 depicts two of the outcomes, using three land use classes (Figure 7a) and using fourteen classes (Figure 7b). Visual comparison of these maps and the simulation obtained for ten classes (Figure 5) illustrates that land use classification affects the residential growth since the generated directions and shapes are different.

In addition to the land use classification sensitivity examination, the model's sensitivity to the number of distance classes was investigated. Initial simulation was employed with three and five class distance layers (Figure 5 and 8 respectively). The simulation outcomes were compared and it can be seen that the generated pattern is different. Figure 8 depicts that large residential areas are created at the expense of green areas in the inner core of the urban area.

The results of sensitivity analysis emphasize the importance of the node selection and the land use classification in the modeling process. The GIS based BN-CA model is sensitive to the changes in the number of nodes, the number of distance classes, and the number of land use classes.

Fig. 7. Predicted land use at time *t+20* years with: a) three and b) fourteen land use classes
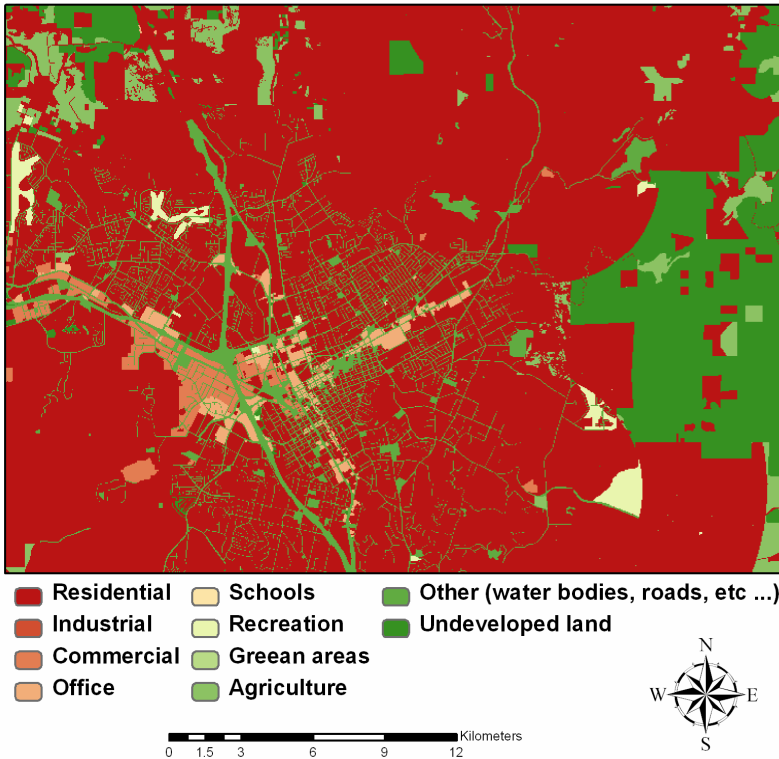
Fig. 8. Predicted land use at time *t+20* years when five class distance layers were used

5   Conclusion

This study developed a novel hybrid model that incorporated Bayesian Networks, Cellular Automata and GIS to predict dynamic land use changes in an urban environment. The developed model addressed some drawbacks of existing GIS-based CA models. First, the developed model is a dynamic GIS model that uses spatial complexity theory. Second, it makes explicit the information about the process of land use change. Third, the transition rule definition is relatively simple. Fourth, the model is explanatory and predictive which makes it useful for land use change modeling. Fifth, the model handles datasets with a large number of variables and relationships. In most probabilistic models, each variable in the model directly affects the outcome of the model. However, in the real world it is possible that one variable can affect another directly or indirectly through several variables such as in a Bayesian Networks. This characteristic is important in land use change modeling as it is difficult to define model parameters and transition rules when many variables affect the change. Finally, the model allows ease in calibration due to the learning procedure. Causal relationships are learned from available data by using Bayesian Networks.

In the developed model, the CA transition rules are represented by a specific Bayesian Network. Land use drivers are represented by nodes and dependencies

between them are represented by conditional probabilities. After the probabilities of the future land use state given the other variables states are calculated, the decision whether the current cell state will be converted or not to the other state is made. The developed model was applied to a hypothetical study site. The results showed that Bayesian networks are suitable for deriving the complex relationships between land use drivers that cause land use change. Moreover, the developed model is capable of producing various scenarios of land use change. With different scenarios through the inclusion of planning policies in the BN structure, it can be used to make predictions in response to policy changes. Further, when some of the variables change (e.g. new roads, new commercial centers), the model can be updated by recalculating the probabilities instead of re-running the model. This emphasizes the ability of the GIS-based BN-CA model to represent and respond to changing configurations.

Sensitivity analysis allowed model testing and the evaluation of the changes occurring in the model outcomes when the number of nodes and land use classes were changed. The results of the sensitivity analysis indicate that the model is sensitive to the changes in the number of nodes and land use classes. Due to the use of hypothetical datasets, validation of the proposed model is not accomplished. This implies that detailed calibration and validation procedures have to be established in GIS-based BN-CA model applications, and this forms the basis of ongoing research in the development of hybrid GIS BN-CA models.

Acknowledgements

References

1. Goodchild, M.E.: Geographic information science and systems for environmental management. Annu Rev Env Resour 28 (2003) 493-519
2. Longley, P.: Geographical information systems and science. Wiley, Chichester (2005)
3. Dragicevic, S., Marceau, D.J.: A fuzzy set approach for modeling time in GIS. International Journal of Geographical Information Science 14 (2000) 225-245
4. Allen, P.M.: Cities and regions as evolutionary complex systems. Geographical Systems 4 (1997) 103-130
5. Batty, M., Longley, P.: Fractal cities : a geometry of form and function. Academic Press, London ; San Diego (1994)
6. Portugali, J.: Self-organization and the city. Springer, Berlin ; New York (2000)
7. Yeh, A.G., Li, X.: A constrained CA model for the simulation and planning of sustainable urban forms by using GIS. Environment and Planning B-Planning & Design 28 (2001) 733-753
8. Torrens, P.M., O'Sullivan, D.: Cities, cells, and complexity: developing a research agenda for urban geocomputation. In: B.H.Carlisle, R.J.A.a. (ed.): 5th International Conference on GeoComputation. "GeoComputation CD-ROM". University of Greenwich, UK (2000)

9. White, R., Engelen, G.: High-resolution integrated modelling of the spatial dynamics of urban and regional systems. Computers, Environment and Urban Systems 24 (2000) 383-400

10. Batty, M.: Urban evolution on the desktop: simulation with the use of extended cellular automata. Environment and Planning A 30 (1998) 1943-1967

11. O'Sullivan, D., Torrens, P.M.: Cellular models of urban systems. University College London, The Centre for Advanced Spatial Analysis, London, UK (2000)

12. Wu, F., Webster, C.J.: Simulation of land development through the integration of cellular automata and multicriteria evaluation. Environment and Planning B-Planning & Design 25 (1998) 103-126

13. Li, X., Yeh, A.G.O.: Urban simulation using principal components analysis and cellular automata for land-use planning. Photogrammetric Engineering and Remote Sensing 68 (2002) 341-351

14. Yeh, A.G.O., Li, X.: Simulation of development alternatives using neural networks, cellular automata, and GIS for urban planning. Photogrammetric Engineering and Remote Sensing 69 (2003) 1043-1052

15. Charniak, E.: Bayesian Networks without Tears. Ai Mag 12 (1991) 50-63

16. Heckerman, D., Mamdani, A., Wellman, M.P.: Real-World Applications of Bayesian Networks - Introduction. Commun Acm 38 (1995) 24-26

17. Little, L.R., Kuikka, S., Punt, A.E., Pantus, F., Davies, C.R., Mapstone, B.D.: Information flow among fishing vessels modelled using a Bayesian network. Environmental Modelling & Software 19 (2004) 27-34

18. Stassopoulou, A., Petrou, M., Kittler, J.: Application of a Bayesian network in a GIS based decision making system. International Journal of Geographical Information Science 12 (1998) 23-45

19. Pearl, J.: Probabilistic reasoning in intelligent systems : networks of plausible inference. Morgan Kaufmann Publishers, San Mateo, Calif. (1988)

20. Buntine, W.L.: A guide to the literature on learning probabilistic networks from data. Ieee T Knowl Data En 8 (1996) 195-210

21. Jordan, M.I., Sejnowski, T.J.: Graphical models : foundations of neural computation. MIT Press, Cambridge, Mass. (2001)

22. Bayes, T., Price, R., Canton, J., Deming, W.E., Molina, E.C.: Facsimiles of two papers by Bayes I. An essay toward solving a problem in the doctrine of chances, with Richard Price's forward and discussion; Phil. Trans. Royal Soc., pp.370-418, 1763. With a commentary by Edward C. Molina. II. A letter on asymptotic series from Bayes to John Canton; pp.269-271 of the same volume. With a commentary by W. Edwards Deming. Hafner Pub. Co., New York (1963)

23. Batty, M.: Cellular automata and urban form: a primer. Journal of American Planning Association 63 (1997) 266-274

24. Zhou, J., Civco, D.L.: Using genetic learning neural networks for spatial decision making in GIS. Photogrammetric Engineering and Remote Sensing 62 (1996) 1287-1295

25. Neapolitan, R.E.: Learning Bayesian networks. Prentice Hall, Harlow (2003)

26. Varis, O.: A belief network approach to optimization and parameter estimation: application to resource and environmental management. Artif Intell 101 (1998) 135-163

27. Kocabas, V., Dragicevic, S.: Assessing cellular automata model behaviour using sensitivity analysis approach. Computers, Environment and Urban Systems (In Press)

28. Sanguesa, R., Burrell, P.: Application of Bayesian Network learning methods to Waste Water Treatment Plants. Appl Intell 13 (2000) 19-40

29. Buntine, W.L.: Operations for learning with graphical models. Journal of artificial intelligence research 2 (1994) 159-225

30. Heckerman, D., Geiger, D., Chickering, D.M.: Learning Bayesian Networks - the Combination of Knowledge and Statistical-Data. Mach Learn 20 (1995) 197-243

31. Cooper, G.F., Herskovits, E.: A Bayesian Method for the Induction of Probabilistic Networks from Data. Mach Learn 9 (1992) 309-347
32. Cheng, J., Greiner, R., Kelly, J., Bell, D., Liu, W.R.: Learning Bayesian networks from data: An information-theory based approach. Artif Intell 137 (2002) 43-90
33. Heckerman, D.: A tutorial on learning with Bayesian networks. In: Jordan, M.I. (ed.): Learning in graphical models. MIT Press, Cambridge, Mass. (1999) 301-354
34. Jensen, F.V., Lauritzen, S., Olesen, K.G.: Bayesian updating in causal probabilistic networks by local computations. Computational Statistics Quarterly 4 (1990) 269-282
35. Murphy, K.: The Bayes Net Toolbox for Matlab. Computing Science and Statistics 33 (2001)
36. Menard, A., Marceau, D.J.: Exploration of spatial scale sensitivity in geographic cellular automata. Environment and Planning B-Planning & Design 32 (2005) 693-714
37. Dietzel, C., Clarke, K.: The effect of disaggregating land use categories in cellular automata during model calibration and forecasting. Computers, Environment and Urban Systems 30 (2006) 78-101

Orientation Calculi and Route Graphs: Towards Semantic Representations for Route Descriptions

Bernd Krieg-Brückner and Hui Shi

Universität Bremen and DFKI-Lab Bremen, Germany
{bkb, shi}@informatik.uni-bremen.de

Abstract. We are aiming for semantic representations of route descriptions for dialogues between a driver and the Bremen intelligent wheelchair ROLLAND, integrating qualitative orientation calculi with Route Graphs. Relative orientations, and the algebraic properties of the inverse and full complement operations, are the basis for specifying properties of orientations between directed edges between locations. 8 orientations at the entry and the exit of an edge are then used to define the relations of variants of the Double-Cross Calculus with 8 and 12 orientations, resp. With an additional predicate "at" a location, we then define all 15 relations. Edges are related to route segments with orientation functions at entries and exits. The inherent origin orientation at a place is then used to solve the place integration problem when joining individual routes into Route Graphs. Finally, some abstract predicates for route descriptions such as "via", "pass by", etc., are defined in terms of these calculi.

1 Motivation

Our work is rooted in the Collaborative Research Center SFB/TR 8 SPATIAL COGNITION.[1] The major research aim here is to develop effective and natural communication between humans and robots about spatial issues, e.g. dialogues between a driver and the Bremen semi-autonomous wheelchair ROLLAND [12,13], which we use as a primary experimental platform to investigate how humans interact with a robot linguistically. A particular goal is the resolution of shared-control problems by clarification dialogues; a number of communication problems may occur because of mismatches between the robot's knowledge and the user's linguistic presentation [25].

According to Kuipers' Spatial Semantic Hierarchy (SSH) [11], knowledge about navigation space consists of multiple interacting representations, both qualitative and quantitative. The SSH model is intended to serve as a framework to model human cognitive knowledge and as a method for a robot to deal with spatial problems. The multi-level representations abstract an agent's spatial

---

knowledge away from the details of its environment and its sensory and motor devices. The SSH has been used in cognitive science and psychology for solving spatial problems through high-level reasoning.

Following the main ideas of the SSH we introduce two distinct spatial representations in the human-robot interaction on route navigation. The "robot level" represents the spatial knowledge of a robot; and the "conceptual level" the spatial knowledge of a user. The robot's spatial knowledge is represented as a Voronoi diagram based topological map with detailed metric information about navigable space [10,14].

In this context we are aiming for semantic representations of route descriptions, to model the users' spatial knowledge about route navigation, and to obtain a semantically well-founded interface to the robot's internal spatial representation, such that an efficient mapping between them can be realized.

At the conceptual level, knowledge is represented by formalised conceptual ontologies (linked to linguistic ontologies), integrating qualitative orientation calculi, in particular Freksa's Double-Cross Calculus, with Route Graphs. There are several reasons for this decision. First, Freksa's qualitative spatial calculus is well-defined [7,8,31] and provides an efficient way to represent orientation information and to reason about causal actions and topological relations. The Route Graph model [30,10] provides a general structure with standard concepts such as places, route segments, routes, and spatial relations like those in the SSH, see Fig. 1 for an example. Moreover, our empirical studies show that the combination is powerful enough to cover almost all users' knowledge in communication with a robot about spatial navigation tasks [5,25]. In practice, this representation provides a well-defined semantic interface to the robot's spatial knowledge representation and supports an efficient mapping between the two levels. As an example, we would like to give semantics to the route description in our office building, cf. Fig. 2.
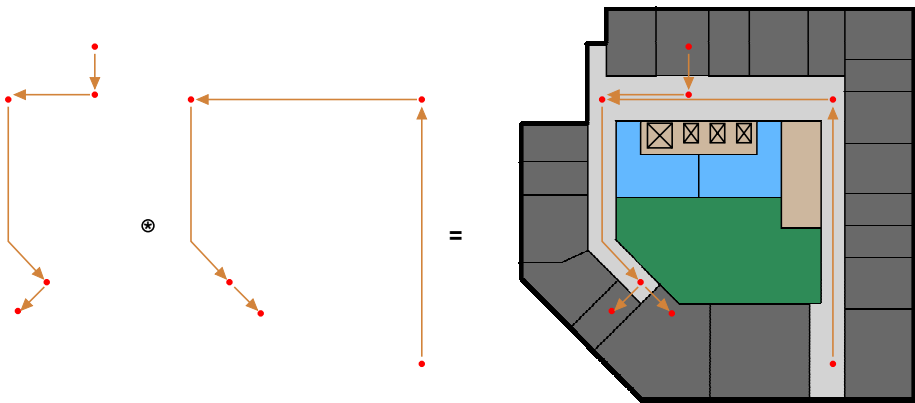

Fig. 1. Route integration resulting in a Route Graph

While a companion paper [24] describes the motivating empirical studies and the use of the spatial calculi and the Route Graph for route descriptions and dialogues in practice, we will concentrate on the theoretical foundations here. We hope that they may serve in other applications of qualitative spatial reasoning and navigation in the area of geographical information processing.

This paper is structured as follows: We introduce qualitative spatial calculi and Route Graphs in Sect. 2, the Route Graph model in Sect. 2.1, and qualitative spatial reasoning using orientation information in Sect. 2.2. Particular orientation calculi are then defined in Sect. 3: the Ego Orientation Calculus, and algebraic properties of the inverse and full complement operations, in Sect. 3.1; edges between locations, and algebraic properties, in Sect. 3.2; finally the Double-Cross Calculus in Sect. 3.3: 8 orientations at the entry and the exit of an edge are used to define the relations of variants of the Double-Cross Calculus with 8 and 12 orientations, resp. With additional predicates "at" a location, we then define all 15 relations. Edges are related to route segments with orientation functions at entries and exits, and route segments are concatenated to routes; the inherent origin orientation at a place is then used to solve the place integration problem when joining individual routes into Route Graphs in Sect. 4. Finally, some abstract predicates for route descriptions such as "via", "pass by", etc., are defined in terms of these calculi, which provide the desired foundational semantics, see Sect. 5. We conclude in Sect. 6.

2 Qualitative Spatial Calculi and Route Graphs

It is well-known that humans act and interact with many different spaces. The space of navigation, one that is highly relevant to our research here, is a mental construction that is schematized [29,28]. Certain information, such as exact metric data, is often systematically simplified and even distorted. Several empirical studies show that the critical elements of the space of navigation are landmarks and paths, links and nodes [26,4,25], thus topological maps are often used to describe such graph-like structures founded in human route descriptions or depictions (e.g., [11,29,30]). In [10], Krieg-Brückner et al. give a standard ontological definition of the Route Graph for navigation by various agents in different applications; we will use it to represent the structures of the navigation space, such that structural relations of route navigation can be inferred.

On the other hand qualitative spatial representation and reasoning is now a mature research area of Artificial Intelligence, and tries to model and to abstract common sense knowledge about space, such that an automation could model, explain and inform about a human's behaviour in space, even if precise quantitative information is not available or is too complicated to be computed [3]. A variety of approaches to qualitative spatial representation and reasoning has been proposed, e.g., Allen's qualitative temporal reasoning [1]; Region Connection Calculi (e.g., [3]); Schlieder [23] investigates the properties of projections from 2-D to 1-D; and Frank [6] uses orientation grids for spatial reasoning.

Considering the empirical evidence collected from our empirical experiments [5,25], we chose Freksa's qualitative spatial calculus using orientation information [7,8,31] to represent and to reason about the user's cognitive spatial knowledge. This calculus is perception-based and cognitively motivated. Information about relative spatial orientations is available through perception, thus these are used in most cases to give route directions by users.

2.1   The Route Graph Model

*RouteGraph*s have been introduced as a general concept for navigation by various agents in a variety of scenarios [30,10]. They can be used as metrical maps with sensory input and additional metrical computation models to control the navigation of robotic agents in the environment, or be used at the cognitive level to model abstractly humans' topological knowledge while they act in the space. The most important character of Route Graphs is that different routes can be integrated into a graph-like structure in which the information concerning these routes are composed.
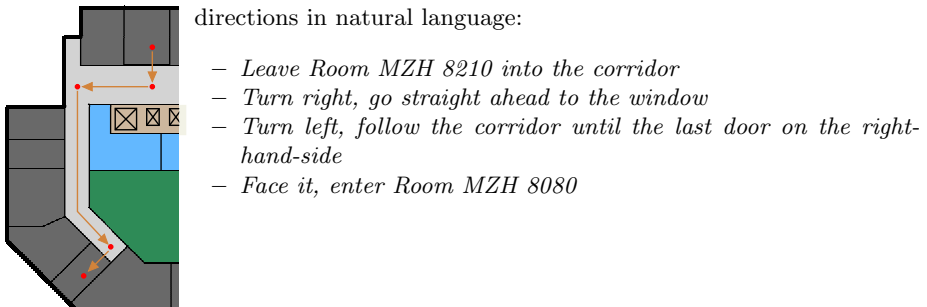

directions in natural language:

- − *Leave Room MZH 8210 into the corridor*
- − *Turn right, go straight ahead to the window*
- − *Turn left, follow the corridor until the last door on the right-hand-side*
- − *Face it, enter Room MZH 8080*

Fig. 2. Sample Route: to the Secretary's Office

Route Graphs are a special class of graphs. A *Node* of a Route Graph, called a *Place*, has a particular position and has its own "reference system"; it may, but need not, be rooted in a (more) global reference system, such as a 3D geo-system. An *Edge* of a Route Graph is directed from a source node to a target node. We call an edge a *RouteSegment*; it always has three additional attributes: an *Entry*, a *Course* and an *Exit*. Exactly what information is associated with these attributes is specially defined for each Route Graph instantiation. For example, an entry or an exit in a Route Graph at the metrical level can be an angle, measured in degrees, with respect to a global 2-D geometric system, the course is then characterized by metrical data for length and width; while an entry/exit at the cognitive level may contain qualitative orientation information (e.g., to the left/right), the course is then the path between two reorientations. Consider the example in Fig. 2; Fig. 3 shows the conceptual level, to be formalised below.

– *Leave Room MZH 8210*

mapped to a *RouteSegment*:

– *Source*: room MZH 8210
– *Entry*: turn towards the door
– *Course*: go through the door
– *Exit*: [turn to face the lift]
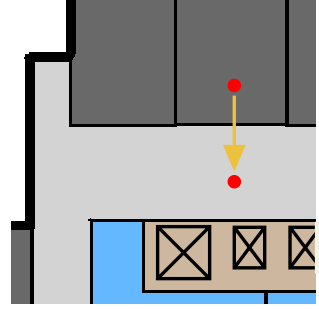– *Target*: corridor, facing the lift

Fig. 3. A Route Segment and its Components

A *Route* is a concatenation of a sequence of route segments from one place to another. For more information please see [30,10].

Moreover, important topological relations from the SSH can be redefined in the Route Graph, such as *at* (the view is seen at a place), or *on* (a place is on a path). However, in the Route Graph model so far, spatial relations between spatial objects are not yet explicitly defined, neither relations between regions, nor orientation relations between objects.

2.2   Qualitative Spatial Reasoning Using Orientation Information

Freksa puts forward the *Double-Cross Calculus* for qualitative spatial representation and reasoning using orientation information [7,8,31], for external qualitative spatial knowledge. In this calculus, point locations are used as basic entities, such that the properties of points and their relations are valid in the entire spatial domain. Moreover, in route descriptions humans often make decisions with respect to a certain point in space, if some restricted space of navigation is considered, such as in a building; while dimensions and shapes are important for the differentiation of similar objects (e.g., the higher building, the round tower).

In the Double-Cross Calculus, the concept *orientation grid* is introduced to represent qualitative orientation information. The grid is aligned to the orientation determined by two points in 2-dimensional space, the start point and the end point of a movement (represented as an arrow denoting the edge between these points here). Combining the front/back and the left/right dichotomy, the Double-Cross Calculus may distinguish 15 meaningful disjoint orientation relations, see Fig. 4.

The edge (between the two points a and b that are to be related) gives rise to the orientations *front* and *back* on this line, also *leftAtExit*, *rightAtExit*, *leftAtEntry*, and *rightAtEntry*, whose terminology we have borrowed from Route Graphs in this paper; *left* and *right* correspond to two areas w.r.t. the edge, similarly *leftFront*, *rightFront*, *leftBack*, and *rightBack*; finally, the special positions *atEntry*, *atExit*, and the predicate *onCourse* are introduced.
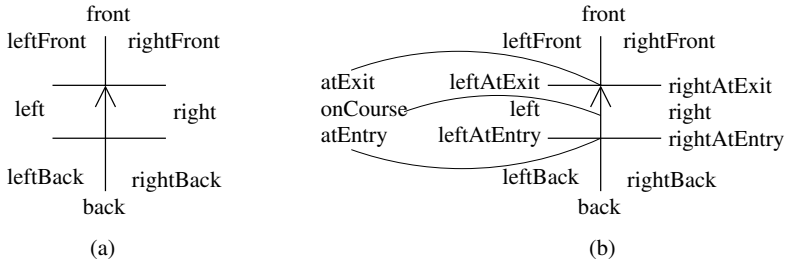
Fig. 4. Orientation grid with 8 (a) and 15 (b) relations for the Double-Cross Calculus

A simple example from [8] shows an application of orientation-based qualitative spatial reasoning, which is the process of determining a location in space on the basis of our own location and other locations we know. Suppose, we walk from start location a to location c and we have reached the intermediate location b. We can describe the orientation of location c qualitatively with reference to the segment between location a and b as the oriented line, denoted as *edge* **ab**, i.e., we compare the edge **bc** to **ab** with respect to their orientation (e.g., **bc** *rightFront* **ab**). At position c, we can compare the next edge **cd** with the previous stretch, the section **bc** (e.g., **cd** *leftFront* **bc**). The inference step then determines the goal location d with respect to the initial edge **ab**, in our case d *rightFront*, *front* or *leftFront* **ab**, see Fig. 5 (a). In [8] the composition table for qualitative orientation-based reasoning is given.
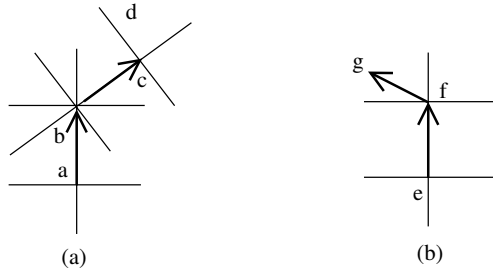
Fig. 5. Inferring the orientation of the goal location d

Supposing location b and f in Fig. 5 are the same. Now the question is "what is the orientation of location d with respect to **ef**?". To answer such questions the Route Graph Model specified in Sect. 4 provides a method for integrating common locations of different routes and recomputing the entries and exits of related route segments according to the reference orientations at the common locations. In the above example, the orientation between **ef** and **bc** will be computed, such that the orientation relation between d and **ef** can then be inferred.

3   Orientation Calculi

In this section we will formalise several orientation calculi, with a view to combining them with the Route Graph in Sect. 4. Before we turn to our major focus, the Double-Cross Calculus, we will introduce the Ego Orientation Calculus with its algebraic properties, then define locations, edges, and orientations between them as a separate theory, and finally use both to define the Double-Cross Calculus. With this stepwise approach, each algebraic structure can be considered separately, properties of each can be generalised beyond the particular application here, and each can potentially be used in a variety of other application contexts. We are progressing in this spirit in other projects within the SFB/TR 8 SPATIAL COGNITION to relate spatial theories and (constraint) calculi, in order to eventually arrive at a uniform theory of space (and time), rather a generic body of interlinked theories (as a "toolbox", in analogy to the CASL libraries [22]) that can advantageously be instantiated and configured to a particular application.

All the theories below are specified in the Common Algebraic Specification Language, CASL, which has been defined by the Common Framework Initiative, CoFI, and approved by WG 1.3 (Foundations of System Specification) of the International Federation of Information Processing, IFIP, as a de-facto standard, see [2,9,19,20]. CASL is supported by tools in the Heterogeneous Specification framework, HETS [16,17,18], for syntactic and semantic checking, consistency checking, verification, and deduction, e.g. during a dialogue using constraint checking systems.

3.1   The Ego Orientation Calculus

We start with the ego orientation calculus; cf. Fig. 6: the fat arrow denotes the ego orientation as a basis, and the fine arrow the orientation of some other object, in relation to the ego orientation. Depending on the application, we may wish to distinguish 2, 4, and 8 orientations (or more); the finer we split the star, the finer our statements become, but we generate more complexity this way, both in terms of computation and "cognitive load" on the user. In this paper we follow Freksa who claims (based on psychological studies) that 8 orientations are cognitively adequate.

The CASL specification in Fig. 7 is intentionally structured: At first, two orientations, *front* and *back* w.r.t. the ego orientation, are introduced, and two operations, inverse ($\sim o$) and complement ($\overline{o}$); moreover, plus ($+$) and minus ($-$), denoting addition and subtraction of orientations. ($Orientation, front, +$) constitutes a commutative monoid, i.e. $+$ is associative, commutative and has $front$ as a unit operation ($a + front = a$); the other algebraic properties are given as universally quantified axioms. These properties hold generally for all orientations, no matter how fine they are split.

Note that the structuring operator **then** introduces a new section of the specification that extends the previous one. In the case of **then %implies**, the
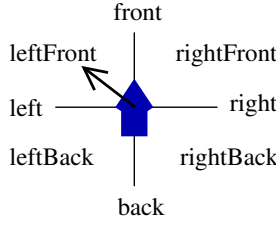
Fig. 6. Ego 8 orientations

spec EGOORIENTATION =
    free type *Orientation2* ::= *front* | *back*
    sort   *Orientation2* < *Orientation*
    ops $\sim$__ : *Orientation* → *Orientation*;%% inverse / converse
            $\overline{}$: *Orientation* → *Orientation*; %% full complement
            __+__ : *Orientation* × *Orientation* → *Orientation*, *assoc*,
            comm, *unit front*;
            __−__ : *Orientation* × *Orientation* → *Orientation*
    ∀ *a, b* : *Orientation*
    • $\overline{front}$ = *front*   • $\overline{back}$ = *back*
    • $\overline{a}$ = *front* − *a*   • $\overline{\overline{a}}$ = *a*   • *a* + $\overline{a}$ = *front*
    • $\sim$ *a* = *back* + *a*   • $\sim \sim$ *a* = *a*   • *a* − *b* = *a* + $\overline{b}$
then %implies
    • $\sim$ *front* = *back*   • *back* + *back* = *front*   • *a* + *front* = *a*
then
    free type *Orientation4* ::= *sort Orientation2* | *right* | *left*
    sort   *Orientation4* < *Orientation*
    • *left* = $\sim$ *right*   • *left* = $\overline{right}$
then %implies
    • *right* + *right* = *back*   • *left* + *right* = *front*
then
    free type *Orientation8* ::= *sort Orientation4* |
                       rightFront | *rightBack* | *leftFront* | *leftBack*
    sort   *Orientation8* < *Orientation*
    • *rightFront* + *rightFront* = *right*
    • *leftFront* = $\overline{rightFront}$ • *leftBack* = $\overline{rightBack}$
    • *rightFront* = $\sim$ *leftBack* • *leftFront* = $\sim$ *rightBack*
then %implies
    • *rightFront* = $\overline{leftFront}$ • *rightBack* = $\overline{leftBack}$
end

Fig. 7. Specification of ego orientation with 2, 4, and 8 orientations

following formulae are not axioms, but meant to be deducible; they constitute a proof obligation.

Now the specification is extended by another two orientations, *left* and *right*, and their properties; subsequently with the orientations *leftFront*, *rightFront*, *leftBack*, and *rightBack*, resp.; thus we arrive at 8 orientations.

Other authors have introduced a yet finer splitting and qualitative distances, cf. e.g. [21,15]; such additions should be easy extensions following the pattern introduced here.

3.2   Locations and Edges

Let us move to the definition of locations and edges, cf. Fig. 8. Edges denote directed connections between locations; they are a kind of "qualitative vectors", there is no reference system or quantitative information (such as length or angle in polar coordinates) associated with them. Similarly, locations are bare of additional attributes for the moment.

spec Locations = Identities **then**
    sort   *Location*
    op   *id* : *Location* → *Id*   %% *Id* is defined in Identities
end

spec Edges = EgoOrientation **and** Locations **then**
    free type *LEdge* ::= __$\xrightarrow{\circ}$__(*source, target* : *Location*)
    sort   *Edge* = {*v* : *LEdge* • ¬ *source*(*v*) = *target*(*v*)}
    op   __⟶ __ : *Location* × *Location* →? *Edge*
    ∀ *x, y* : *Location*
    • *def* *x* ⟶ *y* ⇔ *x* $\xrightarrow{\circ}$ *y* ∈ *Edge*
    • *x* ⟶ *y* = *x* $\xrightarrow{\circ}$ *y if def x* ⟶ *y*
    ops   −__ : *Edge* → *Edge*; %% inverse
          __∠__ : *Edge* × *Edge* → *Orientation*
    ∀ *v, w* : *Edge*
    • *v* ∠ *w* = ∼ (*v* ∠ − *w*)   • *v* ∠ − *w* = − *v* ∠ *w*
    • *v* ∠ *w* = ∼ (*w* ∠ *v*)   • − − *v* = *v*
then %implies
    ∀ *v, w* : *Edge*
    • *v* ∠ *w* = − *v* ∠ − *w*   • − *v* ∠ *w* = ∼ (*v* ∠ *w*)
end

Fig. 8. Specification of locations, edges, and orientations between them

A (proper) edge has a source and a (distinct) target location, it may be constructed by an arrow operation, as in *a* ⟶ *b*; ⟶ is a partial constructor operation here, denoted by "→?" in *Location* × *Location* →? *Edge*, since the restrictive predicate of *Edge* may not be fulfilled. An edge that might be a loop, i.e. *a* and *b* need not be distinct, may be constructed using a special arrow as in *a* $\xrightarrow{\circ}$ *b*.

The operation − inverts the direction of an edge, the operation ∠ yields an orientation relation between two edges (at the moment, *Orientation* is interpreted in the calculus above, but see below). The algebraic properties nicely relate the orientation relation between edges (∠), edge inversion (−), and orientation inversion (∼).

3.3   The Double-Cross Calculus

We now want to define the Double-Cross Calculus in terms of the Ego Calculus. Note that this is not the original Double-Cross Calculus, but a variant that expresses the ternary relations of the original Double-Cross Calculus, e.g. the orientation relation between the two edges **ab** and **bc** is expressed as $a \longrightarrow b \angle b \longrightarrow c \triangleright o$ below in Fig. 10.
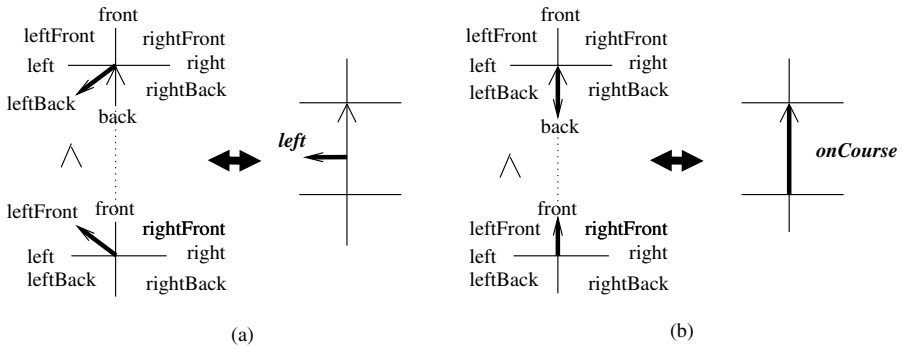
Fig. 9. Relation between 2 ego orientation grids and corresponding Double-Cross orientations

We start with the Double-Cross with only 8 orientation relations, as depicted in Fig. 4 on the left. Fig. 9 shows the principle of defining each relation of the Double-Cross Calculus as a conjunction of two relations in the Ego Calculus (cf. Fig. 6), a pattern that will repeat in the examples below: the basic edge ("vector") of the Double-Cross on the right links two ego orientation grids on the left, corresponding to the orientations at the entry and the exit, by the logical conjunction ∧.

Example (a) in Fig. 9 shows the definition of *left*; the other 7 relations follow the same pattern, cf. the specification in Fig. 10. The pattern is still the same when we extend the Double-Cross with relations at the entry and exit, resp., yielding 12 relations.

Now we turn to the remaining relations of the Double-Cross Calculus. For the predicate *onCourse* we only need two ego orientatin grids in the same pattern, see Fig. 9, example (b). Both the predicates *atExit* and *atEntry* can be defined using the standard construction. The definition of *atExit* deviates a little from the standard pattern seems natural; its style owes to the algebraic properties of the Ego Calculus.

The specification of the Double-Cross Calculus in Fig. 10 follows the same sequential structure as the informal presentation above: 8, 12, and 15 orientations are successively defined.

spec DoubleCrossCalculus = EgoOrientation **and** Edges
then

    sorts $Orientation8 < OrientationDCC13$;
           $OrientationDCC13 < OrientationDCC$
    ops $\sim\_ : OrientationDCC \to OrientationDCC$; **%%** inverse
           $\overline{\_} : OrientationDCC \to OrientationDCC$ **%%** complement
    pred $\_\angle\_\triangleright\_ : Edge \times Edge \times OrientationDCC$
    $\forall\ a, b, c, d : Location;\ o : OrientationDCC13$
    ● $a \longrightarrow b \angle c \longrightarrow d \triangleright o \Rightarrow a = c \lor b = c \lor a = d \lor b = d$
    $\forall\ entry, exit, c : Location;\ u, v, w : Edge$
    ● $v = entry \longrightarrow exit \land w = exit \longrightarrow c \land u = entry \longrightarrow c \Rightarrow$

$(v \angle w \triangleright leftFront$	$\Leftrightarrow v \angle w = leftFront$	$\land\ v \angle u = leftFront)$ $\land$
$(v \angle w \triangleright left$	$\Leftrightarrow v \angle w = leftBack$	$\land\ v \angle u = leftFront)$ $\land$
$(v \angle w \triangleright leftBack$	$\Leftrightarrow v \angle w = leftBack$	$\land\ v \angle u = leftBack)$ $\land$
$(v \angle w \triangleright front$	$\Leftrightarrow v \angle w = front$	$\land\ v \angle u = front)$ $\land$
$(v \angle w \triangleright back$	$\Leftrightarrow v \angle w = back$	$\land\ v \angle u = back)$ $\land$
$(v \angle w \triangleright rightFront$	$\Leftrightarrow v \angle w = rightFront$	$\land\ v \angle u = rightFront)$ $\land$
$(v \angle w \triangleright right$	$\Leftrightarrow v \angle w = rightBack$	$\land\ v \angle u = rightFront)$ $\land$
$(v \angle w \triangleright rightBack$	$\Leftrightarrow v \angle w = rightBack$	$\land\ v \angle u = rightBack)$

then

    free type $OrientationDCC13 ::= sort\ Orientation8\ |$
             $leftAtEntry\ |\ rightAtEntry\ |\ leftAtExit\ |\ rightAtExit\ |\ onCourse$
    ● $leftAtExit = \sim rightAtEntry$ ● $leftAtEntry = \sim rightAtExit$
    ● $leftAtExit = \overline{rightAtExit}$ ● $leftAtEntry = \overline{rightAtEntry}$
    ● $onCourse = \sim onCourse$ ● $onCourse = \overline{onCourse}$
    $\forall\ entry, exit, c : Location;\ u, v, w : Edge$
    ● $v = entry \longrightarrow exit \land w = exit \longrightarrow c \land u = entry \longrightarrow c \Rightarrow$

$(v \angle w \triangleright leftAtExit$	$\Leftrightarrow v \angle w = left$	$\land\ v \angle u = leftFront)$ $\land$
$(v \angle w \triangleright leftAtEntry$	$\Leftrightarrow v \angle w = leftBack$	$\land\ v \angle u = left)$ $\land$
$(v \angle w \triangleright rightAtExit$	$\Leftrightarrow v \angle w = right$	$\land\ v \angle u = rightFront)$ $\land$
$(v \angle w \triangleright rightAtEntry$	$\Leftrightarrow v \angle w = rightBack$	$\land\ v \angle u = right)$ $\land$
$(v \angle w \triangleright onCourse$	$\Leftrightarrow v \angle w = back$	$\land\ v \angle u = front)$

then

    free type $OrientationDCC ::=$
             $sort\ OrientationDCC13\ |\ atEntry\ |\ atExit$
    ● $atExit = \sim atEntry$ ● $atExit = \overline{atExit}$ ● $atEntry = \overline{atEntry}$
    $\forall\ entry, exit : Location$
    ● $entry \longrightarrow exit \angle entry \longrightarrow exit \triangleright atExit$
    ● $entry \longrightarrow exit \angle exit \longrightarrow entry \triangleright atEntry$
end

Fig. 10. The Double-Cross Calculus with 8, 12, and 15 orientations

Note that this extended variant of the Double-Cross Calculus enjoys the additional algebraic properties inherited from the ego calculus and relations between edges. In particular, inversion and complement is defined naturally; also, edges may be inverted, and Double-Cross Calculus relations are defined for commutations of edges.

spec PLACES = LOCATIONS **and** EDGES **then**
    sort   *Place* < *Location*
    op   *origin* : *Place* → *Edge*
end

spec SEGMENTS = PLACES **then**
    sort   *Segment* = $\{s : Edge \bullet source(s) \in Place \wedge target(s) \in Place\}$
    ops   *oEntry, oExit* : *Segment* → *Orientation*;
          $\_\_ \longrightarrow \_\_$: *Place* × *Place* →? *Segment*
    ∀ *x, entry, exit* : *Place*
    • $source(origin(x)) = x$
    • $oEntry(entry \longrightarrow exit) = origin(entry) \angle entry \longrightarrow exit$
    • $oExit(entry \longrightarrow exit) = entry \longrightarrow exit \angle origin(exit)$
then %% recomputation of entry or exit
    ∀ *a, b, c, d* : *Place*
    • $a \longrightarrow b \angle b \longrightarrow c = a \longrightarrow b \angle b \longrightarrow d + b \longrightarrow d \angle d \longrightarrow c$
    • $a \longrightarrow b \angle b \longrightarrow c = oExit(a \longrightarrow b) + oEntry(b \longrightarrow c)$
    • $oExit(a \longrightarrow b) = front \Leftrightarrow a \longrightarrow b \angle b \longrightarrow c = oEntry(b \longrightarrow c)$
    • $oEntry(b \longrightarrow c) = front \Leftrightarrow a \longrightarrow b \angle b \longrightarrow c = oExit(a \longrightarrow b)$
    • $id(b) = id(d) \wedge a \longrightarrow b \angle b \longrightarrow c = a \longrightarrow d \angle d \longrightarrow c$
      $\Rightarrow oExit(a \longrightarrow b) + oEntry(b \longrightarrow c) = oExit(a \longrightarrow d) + oEntry(d \longrightarrow c)$
    • $oExit(a \longrightarrow b) + oEntry(b \longrightarrow c) = oExit(a \longrightarrow d) + oEntry(d \longrightarrow c)$
      $\Rightarrow oEntry(b \longrightarrow c) = origin(b) \angle origin(d) + oEntry(d \longrightarrow c)$
    • $oExit(a \longrightarrow b) + oEntry(b \longrightarrow c) = oExit(a \longrightarrow d) + oEntry(d \longrightarrow c)$
      $\Rightarrow oExit(a \longrightarrow d) = oExit(a \longrightarrow b) - origin(d) \angle origin(b)$

Fig. 11. Specification of places, route segments, and place integration

4 Route Graphs, Integration

In Sect. 3, we defined several types, such as *Orientation*, *Location* and *Edge*, and their operations to specify various orientation calculi. To introduce Route Graphs we are interested in special kinds of locations and edges to define concepts like places and route segments. A *place* is a (uniquely identifiable) location and has its own local reference system, while a *route segment* is an edge from one place to another. A route segment always has an entry and an exit. In the context of route descriptions, an entry or an exit is in fact represented as an orientation with respect to a given origin at this place. Fig. 11 presents a specification of places and route segments.

An *origin* defines the "local reference system", i.e. the pose at a place, as an edge to an external location (e.g. a landmark). The origin "vector" may also be used to ground (the origin of) a place in a global reference system. In Fig. 6, Fig. 12, or Fig. 13 the fat arrow shows the origin as the "ego" orientation.

The operation *oEntry* returns the orientation of a segment with respect to the origin at its entry, and *oExit* the orientation of the origin at its exit with respect to the segment, see the first part of Fig. 11.

The second part of Fig. 11 deals with place and route integration. The representation of places and route segments corresponds to the mathematical notion of a directed graph with a set of nodes (places) and edges (segments between places), enriched with entries and exits.

The union of different routes into route graphs is often non-trivial and needs to extend the union of the corresponding sets of nodes and edges of directed graphs (cf. [10]). A significant step to integrate two different routes is to identify their *common places*, i.e., places with same identifiers (in our simplified case), say b and d, which are meant to be the "same" (need to be integrated). In order to integrate two route segments at a common place, we should take the origin of one route segment at that place and calculate the entry or exit of the other route segment with respect to the chosen origin, according to the axioms given in Fig. 11. Fig. 12 shows an example of recomputing the entry of $b \longrightarrow c$ with respect to $origin(b)$; while Fig. 13 shows an example of recomputing the exit of $a \longrightarrow d$ with respect to $origin(d)$.
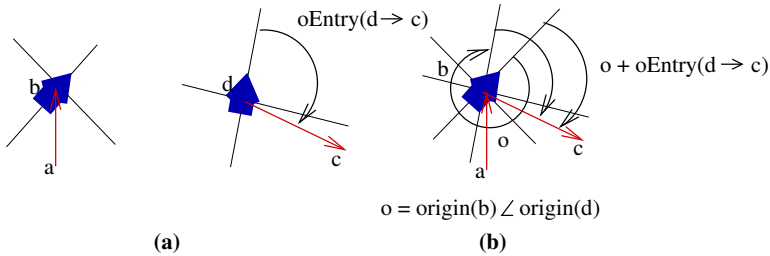
Fig. 12. Integration route segments: recomputation of an entry orientation
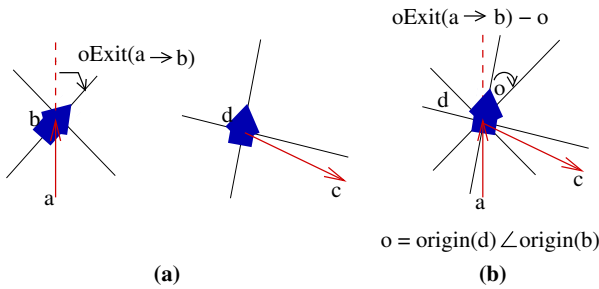
Fig. 13. Integration route segments: recomputation of an exit orientation

5   Route Descriptions

In the empirical experiments carried out in our office building [25], the users primarily used two operations in their route descriptions: travels along a path, and turns. In addition, users often use spatial relations and relational movements in their route descriptions. For example, "the Stuga-Raum is on the left" just

indicates the spatial relation between the "Stuga-Raum" and the current *position* (i.e., the current place with its origin pose); "through the door" includes a movement and a relation: after the movement the door is behind you. Instead of *turns* and *travels* with angle and distance information at the causal level of the SSH, users' route descriptions frequently contain only qualitative information to represent angles and distances, while metrical information is frequently inaccurate and not helpful.

spec ROUTEPREDICATES = SEGMENTS **and** DOUBLECROSSCALCULUS **then**
  pred $\_via\_$: *Segment × Location*
  $\forall$ *a, b, c : Place* • *a* $\longrightarrow$ *b via c* $\Leftrightarrow$ *a* $\longrightarrow$ *b* $\angle$ *b* $\longrightarrow$ *c* $\triangleright$ *onCourse*
then
  sort   *LROrientation* = {*o : OrientationDCC* • *o* = *left* $\vee$ *o* = *right*}
  preds $\_passBy\_$: *Segment × Location*;
       $\_pass\_on\_$, $\_along\_on\_$: *Segment × Location × OrientationDCC*
  $\forall$ *a, b, c : Place*; *lr : LROrientation*
 • *a* $\longrightarrow$ *b pass c on lr* $\Leftrightarrow$
    ($\exists$ *d : Location* • *a* $\longrightarrow$ *b via d* $\wedge$ *a* $\longrightarrow$ *d* $\angle$ *d* $\longrightarrow$ *c* $\triangleright$ *lr*)
  • *a* $\longrightarrow$ *b passBy c* $\Leftrightarrow$
    a $\longrightarrow$ *b pass c on left* $\vee$ *a* $\longrightarrow$ *b pass c on right*
  • *a* $\longrightarrow$ *b along c on lr* $\Leftrightarrow$
    ($\forall$ *d : Location* • *a* $\longrightarrow$ *b via d* $\Rightarrow$ *a* $\longrightarrow$ *d* $\angle$ *d* $\longrightarrow$ *c* $\triangleright$ *lr*)
end

spec LANDMARKS = LOCATIONS **then**
  sort   *Landmark* < *Location*
end

spec SECRETARYSOFFICEEXAMPLE = ROUTEPREDICATES **and** LANDMARKS **then**
  ops   *start, p1, p2, p3, p4, room8080, door8210, door8080 : Place*;
       lifts, window,   room8080, door8210, door8080 : Landmark
  • *oEntry*(*start* $\longrightarrow$ *door8210*) = *back*
  • *start* $\longrightarrow$ *p1 via door8210*
  • *start* $\longrightarrow$ *p1* $\angle$ *p1* $\longrightarrow$ *lifts* $\triangleright$ *front*
  • *oEntry*(*p1* $\longrightarrow$ *p2*) = *right*
  • *p1* $\longrightarrow$ *p2* $\angle$ *p2* $\longrightarrow$ *window* $\triangleright$ *front*
  • *oEntry*(*p2* $\longrightarrow$ *p3*) = *left*
  • *oEntry*(*p3* $\longrightarrow$ *p4*) = *leftFront*
  • *p3* $\longrightarrow$ *p4* $\angle$ *p4* $\longrightarrow$ *door8080* $\triangleright$ *right*
  • *p4* $\longrightarrow$ *room8080 via door8080*
end

Fig. 14. Route predicates and example route: to the secretary's office

Thus, in the specification in Fig. 14, we introduce predicates such as $\_via\_$ in *a* $\longrightarrow$ *b via c* to represent the relation "a location *c* is on route segment (path) *a* $\longrightarrow$ *b*"; $\_pass\_on\_$ in *a* $\longrightarrow$ *b pass c on lr* to represent "a location *c* is in the

orientation lr w.r.t. a route segment $a \longrightarrow b$", where lr is either *left* or *right*; and $\_passBy\_$ in $a \longrightarrow b$ *passBy* c to represent "a location c is either on the left or on the right hand side of a route segment $a \longrightarrow b$".

Considering our sample route description "to the secretary's office" presented in Fig. 2, we give a corresponding route specification using the above predicates in Fig. 14.

6   Conclusion

In this paper, we have presented an approach that combines qualitative spatial calculi using orientation information and the Route Graph as a topological map for the representation of and reasoning about route descriptions in human-robot interaction for navigation. It relates to the Spatial Semantic Hierarchy and provides a simple and effective model for representing and reasoning about human route descriptions. Moreover, using the Route Graph as a topological model, the mapping between the human's knowledge representation and the Rolland robot representation can be efficiently established. Inconsistencies in a route description can be caught before they are sent off to the robot for execution, and a clarification dialogue with the user can be generated.

We concentrated on orientation relations here; it will be relevant for future work to also consider information about regions, to cover descriptions such as "along the cafeteria", "along the river", etc. However, it is still not clear whether complex region calculi are really needed in such situations.

While we have focused on qualitative route descriptions, there are still situations in which users give quantitative information in their route descriptions, for example "ungefähr 25 Meter" (*about 25 meters*) or "ca. 30 Grad nach links" (*about 30 degrees to the left*). Such quantitative data are often redundant, such as "ungefähr 20 bis 30 Sekunden, dann am Ende des Ganges" (*about 20 to 30 seconds, then at the end of the corridor*); if the place "the end of the corridor" is uniquely defined in the environment, then the information "about 20 to 30 seconds" is unimportant for finding a route representation. However, some quantitative data are relevant, for example *about 30 degrees to the left* represents *leftFront*, and is more precise. The model above may be extended to treat such measurements in a qualitative (cf. [15]) or even quantitative way.

We hope that the rich algebraic theories may be useful in different application contexts. We have presented the theories in a structured way such that they may be easily reused. Others (e.g. [27]) have used Route Graphs in the GIS area, and we hope that there may be other uses, e.g. for context-based services, in particular, when dialogic interaction is concerned.

References

1. J. F. Allen. Maintaining knowledge about temporal intervals. *CACM*, 26(11):832–843, 1983.
2. E. Astesiano, M. Bidoit, B. Krieg-Brückner, H. Kirchner, P. Mosses, D. Sannella, and A. Tarlecki, editors. *CASL - the Common Algebraic Specification Language*. Number Special Issue. Springer-Verlag, 2003.
3. A. G. Cohn, B. Bennett, J. Gooday, and N. M. Gotts. Qualitative spatial representation and reasoning with the region connection calculus. *Ceoinformatics*, 1:1–44, 1997.
4. M. Denis. The description of routes: A cognitive approach to the production of spatial discourse. *Cahiers de Psychologie Cognitive*, 16:409–458, 1997.
5. K. Fischer. Linguistic methods for investigating concepts in use. *Methodologie in der Linguistik*, 2003.
6. A. U. Frank. Qualitative spatial reasoning with cardinal directions. In *Proc. Seventh Austrian Conference on Artificial Intelligence*. Springer, 1991.
7. C. Freksa. Qualitative spatial reasoning. In D. M. Mark and A. U. Frank, editors, *Cognitive and Linguistic Aspects of Geographic Space*. Kluwer, 1991.
8. C. Freksa. Using orientation information for qualitative spatial reasoning. In *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, volume 639 of *LNCS*, pages 162–178. Springer-Verlag, 1992.
9. C. L. D. Group, B. Krieg-Brückner, and P. Mosses. CASL Summary. In P. Mosses, editor, *CASL Reference Manual*, volume 2960 of *Lecture Notes in Computer Science*. Springer-Verlag Heidelberg, 2004.
10. B. Krieg-Brückner, U. Frese, K. Lüttich, C. Mandel, T. Mossakowski, and R. J. Ross. Specification of route graphs via an ontology. In C. Freksa, B. Nebel, B. Krieg-Brückner, M. Knauff, and T. Barkowsky, editors, *In Proceedings of Spatial Cognition IV, Chiemsee, Germany*, volume 3343 of *Lecture Notes in Artificial Intelligence*, pages 989–995. Springer, 2004.
11. B. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119:191–233, 2000.
12. A. Lankenau and T. Röfer. A safe and versatile mobility assistant. *IEEE Robotics and Automation Magazine*, 1:29–37, 2001.
13. A. Lankenau and T. Röfer. Mobile robot self-localization in large-scale environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1359–1364, 2002.
14. C. Mandel, K. Huebner, and T. Vierhuff. Towards an Autonomous Wheelchair: Cognitive Aspects in Service Robotics. In *Towards Autonomous Robotic Systems (TAROS 2005), Proceedings*, 2005 (to appear).
15. R. Moratz and M. Ragni. Qualitative spatial reasoning about relative point position. *Journal of Visual Languages & Computing*, 2006 (to appear).
16. T. Mossakowski. CASL: From semantics to tools. In S. Graf and M. Schwartzbach, editors, *TACAS 2000*, volume 1785 of *Lecture Notes in Computer Science*, pages 93–108. Springer, 2000.
17. T. Mossakowski. Heterogeneous tool set (Hets) Web Site. http://www.informatik.uni-bremen.de/cofi/hets, 2004.
18. T. Mossakowski. Heterogeneous specification and the heterogeneous tool set. Technical report, Universitaet Bremen, 2005. Habilitation thesis.
19. P. D. Mosses, editor. CASL *Reference Manual*, volume 2960 of *Lecture Notes in Computer Science*. Springer, 2004.

20. P. D. Mosses and M. Bidoit. Casl — *the Common Algebraic Specification Language: User Manual*, volume 2960 of *Lecture Notes in Computer Science*. Springer, 2004.
21. J. Renz and D. Mitra. Qualitative direction calculi with arbitrary granularity. In *Proceedings PRICAI 2004 (LNAI 3157)*, pages 65–74, Auckland, New Zealand, Sept. 2004. Springer.
22. M. Roggenbach, T. Mossakowski, and L. Schröder. Libraries. In P. Mosses, editor, Casl *Reference Manual*. [19], Part VI.
23. C. Schlieder. Anordnung: Eine Fallstudie zur Semantik bildhafter Repräsentation. In C. Freksa and C. Habel, editors, *Repräsentation und Verarbeitung räumlichen Wissens*, pages 129–142. Springer, 1990.
24. H. Shi and B. Krieg-Brückner. Modelling human route descriptions using qualitative spatial calculi and route graphs. 2006. (submitted).
25. H. Shi and T. Tenbrink. Telling Rolland where to go: HRI dialogues on route navigation. In *Proc. WoSLaD Workshop on Spatial Language and Dialogue, October 23-25, 2005*, 2005.
26. L. Talmy. How language structures space. In H. L. Pick and L. P. Acredolo, editors, *Spatial Orientation: Theory, Research and Application*. Plenum, NY, 1983.
27. S. Timpf. Making sense of space – the legibility of public transport stations. *Environment and Planning*, submitted.
28. B. Tversky. Structures of mental spaces – how people think about space. *Environment and Behavior*, Vol.35, No.1:66–80, 2003.
29. B. Tversky and P. Lee. How space structures language. In C. Freksa, C. Habel, and K. Wender, editors, *Spatial Cognition: An interdisciplinary Approach to Representation and Processing of Spatial Knowledge*, volume 1404 of *Lecture Notes in Artificial Intelligence*, pages 157–175. Springer-Verlag, 1998.
30. S. Werner, B. Krieg-Brückner, and T. Herrmann. Modelling Navigational Knowledge by Route Graphs. In C. Freksa, C. Habel, and K. Wender, editors, *Spatial Cognition II*, volume 1849 of *Lecture Notes in Artificial Intelligence*, pages 295–317. Springer-Verlag, 2000.
31. K. Zimmermann and C. Freksa. Qualitative spatial reasoning using orientation, distance, and path knowledge. *Applied Intelligence*, 6:49–58, 1996.

Incremental Rank Updates for Moving Query Points

Lars Kulik and Egemen Tanin

Department of Computer Science and Software Engineering
NICTA Victoria Laboratory, University of Melbourne, Victoria 3010, Australia
{lars, egemen}@csse.unimelb.edu.au

Abstract. The query for retrieving the rank of all neighbors of a moving object at any given time, a continuous rank query, is an important case of *continuous nearest neighbor* (CNN) queries. An application for ranking queries is given by an ambulance driver who needs to keep track of the closest hospitals at all times. We present a set of *incremental* algorithms that facilitate efficient rank updates for some or all neighbors of a moving query point. The proposed algorithms allow us not only to maintain the exact rank of all n neighbors at any given time but also to track the rank of a subset of all neighbors. We show that updates for these continuous rank queries can be performed in linear time for arbitrary polygonal curves in two dimensions and in logarithmic time for movements along a fixed direction. Instead of using Voronoi diagrams, our algorithms are based on small subsets of all bisectors between neighbors. We prove that it is sufficient to keep track of only $n - 1$ bisectors for all n neighbors. The algorithms for maintaining the rank only require minimal incremental updates on the bisector sets.

1 Introduction

Nearest neighbor (NN) queries are a well investigated research topic. With the ubiquitous availability of location information for mobile devices, continuous nearest neighbor (CNN) queries have become a major research focus. An important variant of CNNs are continuous ranking queries that retrieve the rank of k neighbors of a moving object. *Continuous ranking queries* have a wide application potential. For example, an ambulance driver may wish to continuously keep track of some of the hospitals in a city ranked by their distance using a mobile device. Alternatively, the driver of a car might want to track a set of gas stations or hotels with respect to their distance. In this paper, we propose a class of algorithms that facilitate efficient rank updates for a moving query point in order to continuously query the rank of the neighbors.

Algorithms retrieving the neighbors of a query point in the order of their distance relative to the query point are called *ranking* algorithms, because they rank spatial objects of a given database for that query point [6]. If only a subset of objects has to be retrieved from a database, the queries are called kNN queries, where k is the number of retrieved objects. The number of objects k can be less than or equal to the total number n of objects in the database. If the order of the neighbors is irrelevant, the query is called an *order insensitive kNN query* [7]. In this paper, we study *order sensitive kNN* queries, i.e., queries that retrieve objects in the order of their distance.

A number of algorithms have been developed to efficiently find the nearest neighbors for a static query point. One way to derive an algorithm for a CNN query is to use a

classical NN query algorithm: for each predetermined time or location update interval, a new NN query for a moving query point can be executed using one of the classical NN query algorithms. However, a repeated execution of the classical algorithms will not take advantage of the steps and the results of previously run NN queries. In addition, many of the objects of a previously run NN query may not have changed their status. Continuous requerying also might miss some of the rank updates if the frequency of update queries is set too low for a given application domain.

In this paper, we present a set of incremental algorithms that track the rank of the neighbors of a moving query point continuously. The positions of the neighbors are assumed to be fixed. We show that updates for such continuous rank queries can be performed in linear time for any arbitrary polygonal curve without any prior knowledge of the curve itself. For polygonal curves that largely consist of straight line segments with a limited number of turns, a modified algorithm is presented that achieves logarithmic complexity for movements along the straight line segments.

Our algorithms use a small subset of all bisectors between neighbors of the moving query point, and only need to maintain a list of $n-1$ bisectors for all n neighbors at any given time. We show that the bisector list is sufficient for a moving query point in order to (i) detect when rank updates are needed and (ii) which ranks need to be updated once the need arises. We finally show the connection of our work with higher-order Voronoi diagrams [1,11].

We believe that incremental algorithms for continuous queries will gain significant importance with the increasing use of wireless and mobile systems as wireless applications are typically constrained by limited communication and computation resources. The remainder of the paper is organized as follows: Current approaches for CNN queries are presented in Section 2. In Section 3, we demonstrate the algorithm for continuous rank updates in the one-dimensional case in order to motivate the underlying ideas of our approach. Section 4 presents the algorithm for ranking all neighbors of a moving query point in the two-dimensional case. In Section 5, we modify our algorithm for continuous rank updates to capture travel patterns that contain only a few directional changes. Section 6 develops algorithms for two variants of continuous rank updates: how to maintain a single rank and how to maintain multiple ranks by possibly different agents. Section 7 concludes our work and outlines future work.

2   Related Work

NN queries for static query points have been investigated by many researchers [4,5,6,13]. Different types of branch-and-bound algorithms have been studied on a number of spatial data structures, in particular R-trees and quadtrees [15,16,17]. Roussopoulos et al. [13] propose a depth-first branch-and-bound algorithm on R-trees, while Hjaltason and Samet [5] take a best-first approach. These algorithms are designed to efficiently find static nearest neighbors for a static query point in various applications.

For efficient indexing and querying of moving objects Saltenis et al. [14] introduce the Time-Parameterized R-tree (TPR-tree). TPR-trees are based on the assumption that the trajectory information is available and the location of each moving object can be represented as a linear function of time. Benetis et al. [2] build on the TPR-tree and

propose algorithms for NN and reverse NN queries. Reverse NN queries are used to find the objects that have a query point as their nearest neighbor. Recent approaches also use a grid-based model [3,9,22] rather than extending the R-tree concept. Our work differs from the moving object indexing research as we assume that the objects are static but the query point is in motion. In addition, many moving object indexing methods, such as the TPR-tree-based schemes, assume that the trajectories are known to a certain extent. Finally, these indexing methods commonly focus on spatial *snapshot* queries while our work focuses on continuous queries.

Tao and Papadias [19] introduce time-parameterized queries in spatio-temporal data that extend the snapshot queries with the *expiry time* concept. A time-parameterized query returns an expiry time along with a set of results. Other researchers have extended this work. For example, Iwerks et al. [8] build on this concept to allow for changes on pending update events for enabling truly continuous spatial queries. However, these papers neither focus on maintaining the rank information for the spatial objects nor on static objects with a moving query point.

Zheng and Lee [24], and Song and Roussopoulos [18] developed the first approaches to address queries where the objects are assumed to be static but the query point is in motion. Zheng and Lee use a Voronoi diagram-based approach to answer 1-NN queries. Song and Roussopoulos address kNN queries, where k can be greater than one, and use sampling-based methods to avoid brute-force resubmissions of new queries for a moving query point. A sampling-based approach, however, may miss certain updates or become computationally expensive for frequent updates.

Tao et al. [20,21] introduce CNN queries for moving objects with *a priori* known trajectories. Given a line segment, their result set contains a set of (*object*, *interval*) tuples. Each tuple represents the NN for a given interval on the line segment. A kNN version of the algorithm is also available. This work can be extended to trajectories consisting of several line segments.

Zhang et al. [23] introduce a location-based querying approach that is closely related to our work. They define *validity regions* for which the result of a query remains the same. Hence, clients do not need to resubmit queries that will not change the result set. In contrast to our work, they do not consider order sensitive ranking queries. They acknowledge the necessity of incremental algorithms that reduce redundant computations and communication, a requirement which we directly address in our approach.

Extensions to compute multiple CNN queries more efficiently are given in [7,10]. In [7] *safe regions* (similar to validity regions) are defined, where the output of multiple, registered CNN queries cannot change for a database of moving objects. They use the concept of safe regions to reduce the communication overhead in wireless applications. In [10] *conceptual partitioning* is used to ignore updates from objects that are far away from a query to reduce runtime costs. Such techniques can be used in tandem with our work to handle multiple CNN queries.

3   One-Dimensional Queries

We first present our approach in the case where all bisectors can be ordered along one axis to motivate the basic ideas. As a simplified notation, we call this case the

one-dimensional case and present our approach on the x-axis. Many properties carry over to CNN queries on higher dimensions.

3.1   Notation and Definitions

Assume n sites $\{S_1, S_2, \ldots, S_n\}$ are ordered on a straight line, i.e., a moving query point Q will first hit S_1, then S_2, and so forth until Q finally hits the last site S_n. A perpendicular bisector between two sites S_i and S_j, is called $B(S_i, S_j) =_{\text{def}} B_{ij}$ and is defined as the straight line that is equidistant from the sites S_i and S_j:

$$B(S_i, S_j) =_{\text{def}} \{P \mid d(P, S_i) = d(P, S_j)\}$$

where d is the usual Euclidean distance. Since $B_{ij} = B_{ji}$ by definition, we can assume without loss of generality that $i < j$ for B_{ij} throughout the rest of this paper. Let d_i denote the distance from the query point Q to a site S_i, i.e., $d_i =_{\text{def}} d(Q, S_i)$. In addition, we introduce a precedence relation $\prec$ for bisectors: $B_{ij} \quad \prec \quad B_{kl}$, if $d(Q, B_{ij}) < d(Q, B_{kl})$. Given these definitions, a set of basic properties that we use for our algorithms is summarized in the following observation.

Observation 1. *If the sites* $\{S_1, S_2, \ldots, S_n\}$ *are all on a single straight line with* $d_i < d_{i+1} \; \forall i \in \{1, \ldots, n-1\}$, *then*

1. $\forall i, j, k : i < j \Rightarrow B_{ik} \prec B_{jk}$
2. $\forall i, j, k : j < k \Rightarrow B_{ij} \prec B_{ik}$
3. $\forall i, j, k, l : i < k \wedge j < l \Rightarrow B_{ij} \prec B_{kl}$

These properties follow from the fact that $d_i < d_j$ if $i < j$ and the triangle inequality for distances.

3.2   A Base Algorithm

We explain the algorithm in the one-dimensional case with an example. At each step we maintain two sorted lists $\mathcal{S}$ and $\mathcal{B}$. $\mathcal{S}$ contains the sites ordered in terms of their distance to the query point, i.e., their rank order. We also maintain $\mathcal{B}$, a list of bisectors ordered by increasing x coordinates. We use the bisector list to detect when the next rank update will occur. It suffices to keep only those $n-1$ bisectors in the list $\mathcal{B}$ that are of the form B_{ii+1}. This means we only keep those bisectors between sites that are neighbored in $\mathcal{S}$. We show in Section 4.1 for the two-dimensional case, why we do not need to track all bisectors (i.e., $\mathcal{O}(n^2)$ bisectors).

Let $\{\ldots, S_i, S_j, S_k, S_l, \ldots\}$ be an ordered list for the sites. If the query point crosses a bisector B_{jk} then S_k is now closer to the query point than the site S_j. This also means that the list $\mathcal{S}$ changes to $\{\ldots, S_i, S_k, S_j, S_l, \ldots\}$. Since we only keep track of the bisectors between neighboring sites, our list $\{\ldots, B_{ij}, B_{jk}, B_{kl}, \ldots\}$ changes to $\{\ldots, B_{ik}, B_{jk}, B_{jl}, \ldots\}$. More precisely, we have to delete at most two bisectors from $\mathcal{B}$ and correspondingly insert at most two new ones. Note that these bisectors need not to be located next to each other on the straight line. They will be neighbors to each other if they are kept under the $\prec$ order rather than the order given by the x-axis. For each

rank update, we need to update the list of neighboring bisectors in order to detect the next rank update. In our example of a query point that moves from S_1 to S_5 (Figure 1), the algorithm will update the two lists as indicated below. At each step, the position of the query point with respect to the bisectors is shown with a '*'. We also show which bisectors are introduced and removed from $\mathcal{B}$.

$$S = \{S_1, S_2, S_3, S_4, S_5\}, \tag{1}$$
$$B = \{*, B_{12}, B_{23}, B_{34}, B_{45}\},$$

$$S = \{S_2, S_1, S_3, S_4, S_5\}, \tag{2}$$
$$B = \{B_{12}, *, B_{13}, B_{34}, B_{45}\}, \qquad \text{in}: B_{13}, \qquad\qquad \text{out}: B_{23}$$

$$S = \{S_2, S_3, S_1, S_4, S_5\}, \tag{3}$$
$$B = \{B_{13}, *, B_{14}, B_{23}, B_{45}\}, \qquad \text{in}: B_{23}, B_{14}, \qquad \text{out}: B_{12}, B_{34}$$

$$S = \{S_2, S_3, S_4, S_1, S_5\}, \tag{4}$$
$$B = \{B_{14}, *, B_{23}, B_{15}, B_{34}\}, \qquad \text{in}: B_{34}, B_{15}, \qquad \text{out}: B_{13}, B_{45}$$

$$S = \{S_3, S_2, S_4, S_1, S_5\}, \tag{5}$$
$$B = \{B_{14}, B_{23}, *, B_{15}, B_{24}\}, \qquad \text{in}: B_{24}, \qquad\qquad \text{out}: B_{34}$$

$$S = \{S_3, S_2, S_4, S_5, S_1\}, \tag{6}$$
$$B = \{B_{23}, B_{15}, *, B_{24}, B_{45}\}, \qquad \text{in}: B_{45}, \qquad\qquad \text{out}: B_{14}$$

$$S = \{S_3, S_4, S_2, S_5, S_1\}, \tag{7}$$
$$B = \{B_{15}, B_{24}, *, B_{25}, B_{34}\}, \qquad \text{in}: B_{34}, B_{25}, \qquad \text{out}: B_{23}, B_{45}$$

$$S = \{S_3, S_4, S_5, S_2, S_1\}, \tag{8}$$
$$B = \{B_{12}, B_{25}, *, B_{34}, B_{45}\}, \qquad \text{in}: B_{45}, B_{12}, \qquad \text{out}: B_{24}, B_{15}$$

$$S = \{S_4, S_3, S_5, S_2, S_1\}, \tag{9}$$
$$B = \{B_{12}, B_{25}, B_{34}, *, B_{35}\}, \qquad \text{in}: B_{35}, \qquad\qquad \text{out}: B_{45}$$

$$S = \{S_4, S_5, S_3, S_2, S_1\}, \tag{10}$$
$$B = \{B_{12}, B_{23}, B_{35}, *, B_{45}\}, \qquad \text{in}: B_{45}, B_{23}, \qquad \text{out}: B_{34}, B_{25}$$

$$S = \{S_5, S_4, S_3, S_2, S_1\}, \tag{11}$$
$$B = \{B_{12}, B_{23}, B_{34}, B_{45}, *\}, \qquad \text{in}: B_{34}, \qquad\qquad \text{out}: B_{35}$$

After an initial $\mathcal{O}(n \log n)$ sort to rank the sites, by using a doubly-linked list, in tandem with an array of the sites for direct access in their natural order, one can decrease the cost of each rank update on the sorted site list to $\mathcal{O}(1)$. Unfortunately, for the bisector list, we can show that for every bisector crossed, we have to perform $\mathcal{O}(\log n)$ steps for n sites. We can use a balanced tree over the bisector order $\prec$ to achieve this.

The existence of a natural order between the sites in one dimension significantly improves the runtime behavior of the basic algorithm stated above. We will see that the moving query point will take $\mathcal{O}(n)$ steps in the two-dimensional case to locate the next bisector to be intersected.
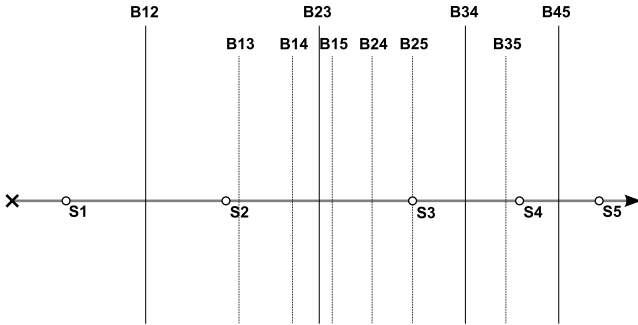
Fig. 1. A straight line, with five sites and a moving query point (marked with a cross). The long bisectors form the original bisector list when the query point is on the left of the site S_1. The shorter bisectors are the new bisectors that are introduced when the moving query point crosses an existing bisector.

This simple one-dimensional version of our algorithm shows: first, we only need to make incremental updates to keep our site ranks at all times; second, the updates are only required when the moving query point crosses a bisector, which avoids unnecessary query resubmissions at any other time.

4   Multi-dimensional Queries

For higher dimensional spaces, the main idea of the base algorithm does not change. In this section, we present the algorithm in the two-dimensional space but the approach can be applied to higher dimensional spaces. The main difference between the one-dimensional case and the higher dimensional case is the lack of a natural order for an arbitrary set of lines, i.e., the bisectors in our case (or hyper-planes for higher dimensions).

As in the one-dimensional case, we maintain for n sites $n - 1$ bisectors in two dimensions. Using these bisectors, we define a region where the ranks of the sites do not change. In fact, this region maps to a cell in the *ordered order-k Voronoi diagram* [11] of these n sites where $k = n$. We only keep track of one cell at any time and only $n - 1$ bisectors, and thus, we do not require the precomputation of this Voronoi diagram. As long as the query point stays in this cell, the ranks of the sites do not change, which renders a query resubmission redundant. Once the query point crosses a bisector, a rank update is required.

Crossing a bisector implies that the order for the two corresponding sites of this bisector needs to be altered. As in one dimension, we alter the rank of these sites first. As a side effect, we also have to update the $n - 1$ bisectors. Similar to the one-dimensional case only a small subset of bisectors needs to be updated when a rank update occurs. The updated bisector list then defines a new ordered order-k Voronoi cell ($k = n$), which we can use to detect the next bisector crossing.

We first introduce the basic mathematical concepts for our algorithm and state the relevance of our work to ordered order-k Voronoi diagrams. We then present the final algorithm and its runtime complexity.

4.1   Rank Maintenance Using Regions

At the heart of our algorithm lies the observation that we only need $n - 1$ neighboring bisectors for the n sites in order to determine the next rank update. We prove this observation in this subsection.

Three straight lines that are not pairwise parallel, generally have three distinct intersection points. Bisectors, however, have an important property that does not hold for *general line arrangements*: *concurrency*. Three lines are *concurrent* if they have a common intersection point P. Because two straight lines can only intersect in at most one point, the point P is the only point in which the three bisectors intersect. Given two points Q and R, we use the notation $\overline{QR}$ for the segment connecting those two points. The following theorem states that there are always triplets of bisectors for sites in *general position* (i.e., no three points are collinear) that are concurrent.

Theorem 1. *Let S_i, S_j, and S_k be sites in general position and B_{ij}, B_{jk}, B_{ik} their bisectors. If P is an intersection point of B_{ij} and B_{jk}, i.e., $B_{ij} \cap B_{jk} = \{P\}$, then B_{ik} also intersects B_{ij} and B_{jk} in P, i.e., $P \in B_{ik}$.*

Proof. Let P be an intersection of B_{ij} and B_{jk} (e.g., as in Figure 2). Because B_{ij} is a bisector, it follows $|\overline{S_i P_1}| = |\overline{S_j P_1}|$. Applying Pythagoras' Theorem we get $|\overline{S_i P}| = |\overline{S_j P}|$. Similarly, we obtain $|\overline{S_j P}| = |\overline{S_k P}|$ due to $|\overline{S_j P_2}| = |\overline{S_k P_2}|$. Hence, we get $|\overline{S_i P}| = |\overline{S_k P}|$, which means that $P \in B_{ik}$.

Therefore, the set of all bisectors does not constitute a *simple arrangement* [12] of lines (a line arrangement is simple if there are no concurrent intersection points). Theorem 1 allows us to introduce a precedence relation on bisectors. In order to motivate the general definition, we will first introduce the precedence relation for two bisectors.

Let $H(S_i, S_j) =_{\text{def}} H_{ij} =_{\text{def}} \{P \mid d(P, S_i) \leq d(P, S_j)\}$ denote the *closed half plane* that includes S_i but not S_j. This half plane is also called the *dominance region* of S_i over S_j. The intersection of two half planes is a *closed sector*. We obtain from Theorem 1 that three bisectors generate exactly six sectors (see Figure 2). A segment that connects a point in a sector with a point outside of this sector always first intersects its bounding bisectors before it intersects the third bisector. We record this observation in the following lemma. We use the notation $\beta(P, Q, R)$ if the point Q is between the points P and R, i.e.:

$$\beta(P, Q, R) \Leftrightarrow_{\text{def}} \exists \lambda \in [0, 1] : Q = \lambda P + (1 - \lambda)R.$$

It is important to note that this notation also captures the case where concurrent intersection points occur.
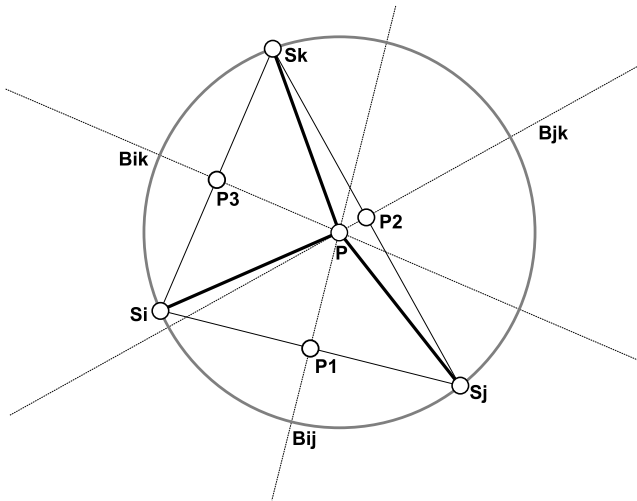
Fig. 2. Three bisectors for three sites intersecting at one point

Lemma 1. *Let S_i, S_j, and S_k be sites in general position, and B_{ij}, B_{jk}, and B_{ik} be their bisectors, then:*

$$Q \in H_{ij} \cap H_{jk}:$$
$$\forall Q', R\,[R \in \overline{QQ'} \wedge R \in B_{ik} \Rightarrow$$
$$\exists R'\,[\beta(Q, R', R)] \wedge (R' \in B_{ij} \vee R' \in B_{jk})].$$

On the basis of Lemma 1, we can formally define a precedence relation for bisectors:

Definition 1. *Let S_{i_1}, S_{i_2}, ..., S_{i_k} be k sites in general position, $\mathcal{B}$ be the set $\left\{B_{i_1 i_2}, B_{i_2 i_3}, \dots, B_{i_{k-1} i_k}\right\}$ of the $k-1$ corresponding bisectors, and B_{lm} another bisector not in $\mathcal{B}$. We say that the bisector set $\mathcal{B}$ precedes the bisector B_{lm} with respect to an arbitrary set $\mathcal{A}$, written as $\mathcal{B} \preccurlyeq_{\mathcal{A}} B_{lm}$, if and only if:*

$$\forall Q \in \mathcal{A}: \forall Q', R \left[R \in \overline{QQ'} \wedge R \in B_{lm} \Rightarrow \exists R'\, \beta(Q, R', R) \wedge \left(\bigvee_{j=1}^{k-1} R' \in B_{i_j i_{j+1}} \right) \right].$$

An immediate consequence of the definition of the precedence relation is the following corollary.

Corollary 1. *For every bisector B_{lm} holds:*

$$\mathcal{B} \preccurlyeq_{\mathcal{A}} B_{lm} \wedge \mathcal{B}' \preccurlyeq_{\mathcal{A}'} B_{lm} \Rightarrow \mathcal{B} \cup \mathcal{B}' \preccurlyeq_{\mathcal{A} \cap \mathcal{A}'} B_{lm}.$$

Proof. Since $\mathcal{A} \cap \mathcal{A}' \subset \mathcal{A}, \mathcal{A}'$ and $\mathcal{B}, \mathcal{B}' \subset \mathcal{B} \cup \mathcal{B}'$, the corollary follows from the definition of the precedence relation (if the disjunction is true for the bisectors in the set $\mathcal{B}$ and $\mathcal{B}'$ then it is true a fortiori for $\mathcal{B} \cup \mathcal{B}'$).

Our goal is to rank the sites for a moving query point at all times. We previously stated that two sites change their rank for a query point if the query point crosses their bisector. On the other hand, as long as a query point is in one half plane, or more generally in the intersection of a set of half planes, the ranks of the sites do not change. This motivates the concept of a *fixed-rank* region. A fixed-rank region is related to the ordered order-k Voronoi cell concept [11]. The fixed-rank region $H^{i_1 i_2 \cdots i_k}$ of order k is the set of all points such that the sites $S_{i_1}, S_{i_2}, \ldots, S_{i_k}$ are ranked by their distance[1]. More formally:

Definition 2

$$H^{i_1 i_2 \cdots i_k} =_{\text{def}} \bigcap_{j=1}^{k-1} H_{i_j i_{j+1}}$$

If all indices are consecutive then we use the following notation:

$$H^{l,m} =_{\text{def}} \bigcap_{j=l}^{m-1} H_{jj+1}.$$

The triangle in Figure 3 is the fixed-rank region $H^{1,4}$. The figure illustrates that the set of bisectors B_{12}, B_{23}, and B_{34} precede the bisectors B_{14} and B_{24} with respect to the gray triangle. Note that the bisector B_{24} has to be concurrent with the bisectors B_{23} and B_{34} according to Theorem 1 (which means that B_{24} intersects the boundary of the gray triangle at a single point). In addition, using B_{45} will introduce another rank for the site 5, i.e., $H^{1,5}$.

Using the precedence definition, we can rewrite Lemma 1 as $\{B_{ij}, B_{jk}\} \preccurlyeq_{H^{ijk}} B_{ik}$. If a set of bisectors $\mathcal{B}$ precedes another bisector B_{lm}, we also say that B_{lm} succeeds $\mathcal{B}$. The next theorem states that every bisector B_{lm} succeeds a set of neighboring bisectors with respect to the set $H^{l,m}$.

Theorem 2. *For every bisector B_{lm} holds:*

$$\bigcup_{i=l}^{m-1} \{B_{ii+1}\} \preccurlyeq_{H^{l,m}} B_{lm}.$$

Proof. We prove this theorem via induction over $k = m - l$. If $k = 1$, i.e., $m = l + 1$, then $H^{l,l+1} = H_{ll+1}$, and $B_{ll+1} \preccurlyeq_{H^{l,l+1}} B_{ll+1}$ is trivially true. Assume now the theorem holds for $k - 1$, i.e., $k - 1 = m - l - 1$, and $m > l + 1$. We can apply Lemma 1 and get:

$$\{B_{ll+1}, B_{l+1m}\} \preccurlyeq_{H^{l,m}} B_{lm}.$$

According to the induction assumption the theorem is true for $k - 1$ so that we can apply it to B_{l+1m} and obtain:

$$\bigcup_{i=l+1}^{m-1} \{B_{ii+1}\} \preccurlyeq_{H^{l+1,m}} B_{l+1m}.$$

---

[1] It is also possible to define a partial-fixed-rank region, i.e., a more general concept where only a subset of the ranks are relevant, but we omit the details for this concept here since they are not relevant to our discussion.
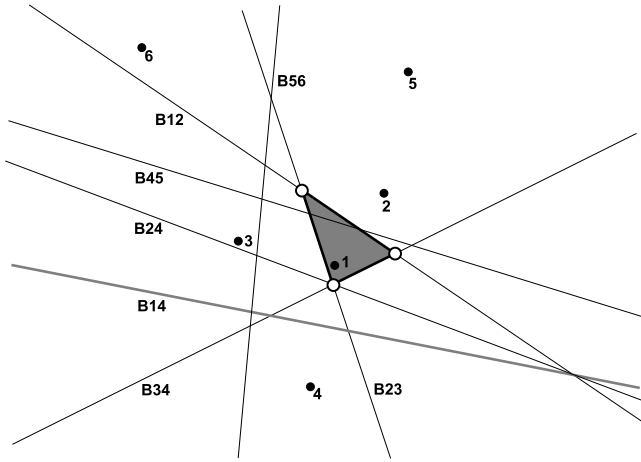
Fig. 3. A set of sites and some bisectors and a fixed-rank region. The ranks of the sites are used for naming the sites and the bisectors. The moving query point is initially located in the grey triangle.

Due to Corollary 1 and $H^{l,m} \subset H^{l+1,m}$ and two precedence relations above, the theorem follows.

Since $H^{1,n} \subset H^{l,m}$, the following corollary is an immediate consequence of Corollary 1 and Theorem 2.

Corollary 2. *For every bisector B_{lm} holds:*

$$\bigcup_{i=1}^{n-1} \{B_{ii+1}\} \preccurlyeq_{H^{1,n}} B_{lm}.$$

4.2   Incremental Rank Maintenance

According to Corollary 2, the set $\mathcal{B}$ of the $n-1$ bisectors $B_{ii+1}, i = 1..n-1$, precedes all other bisectors. Therefore, it is sufficient that the algorithm only keeps a list of the bisectors in $\mathcal{B}$. Our approach is to partition the set $\mathcal{B}$ of bisectors and keep two sub-lists of them: (1) a minimal set $\mathcal{M}$ of those k bisectors that enclose a query point (this list is cyclically sorted), and (2) an unsorted list $\mathcal{R}$ of the remaining $n - k - 1$ bisectors. This partition improves the performance of the spatial tests on the fixed-rank region condition, i.e., whether or not the moving query point has exited the current region. Other improvements, such as keeping a simple cache for the bisector toward which the motion of the query point is currently directed, can also be used to improve the complexity of the tests but are not the focus of this paper.

As it was the case in one dimension, the algorithm keeps track of the ranking of the sites in a sorted list, i.e., $\mathcal{S}$. $\mathcal{S}$ sorts the sites with respect to their distance. Once the query point crosses one of the bisectors of $\mathcal{M}$, the two corresponding sites change their

ranks in $\mathcal{S}$, i.e., the site further away is now closer and vice versa. Hence, the $\mathcal{S}$ list is updated. Updating this list implies there are some bisectors that are now redundant in our lists. These have to be deleted. Then, new bisectors need to be inserted. These steps are again similar to the one-dimensional case. As we alter the ranks of only two sites, and we only have to maintain the neighboring bisectors $\mathcal{B}$, it follows that the number of bisectors that need to be deleted is at most two. Similarly, the number of bisectors that need to be added is also at most two.

Assume $\mathcal{S} = \{S_{i_1}, S_{i_2}, \ldots, S_{i_n}\}$, then a rank update consists of the following two events:

1. If the query point crosses the bisector $B_{i_1 i_2}$ ($B_{i_{n-1} i_n}$), then we have to delete the bisector $B_{i_2 i_3}$ ($B_{i_{n-2} i_{n-1}}$) and insert the bisector $B_{i_1 i_3}$ ($B_{i_{n-2} i_n}$). If the query point crosses any other bisector, for example $B_{i_j i_{j+1}}$, we have to delete two bisectors, $B_{i_{j-1} i_j}$ and $B_{i_{j+1} i_{j+2}}$, and then insert the two new bisectors $B_{i_{j-1} i_{j+1}}$ and $B_{i_j i_{j+2}}$.
2. Finally, we have to check if any of the bisectors in $\mathcal{R}$ are now in $\mathcal{M}$. This is required as the new fixed-rank region could now be defined by the inclusion of some of the bisectors that were not in the previous version of the list $\mathcal{M}$.

In the case that the query point crosses two (or in general m) bisectors at the same time, i.e., the the crossing point is the intersection of at least two bisectors, then we have to update the ranks and lists simultaneously. The algorithm, however, will work without any modifications, if each bisector crossing is treated independently. The aggregated update can be decomposed into m separate updates that can be treated independently of each other and without any modifications required from our algorithm. Obviously, for scenarios in which many bisectors are crossed simultaneously, more efficient methods, which update multiple bisectors at the same time, could be introduced. We do not focus on this improvement in this paper but leave it for future work.

Finally, we illustrate the algorithm in an example below (Figure 4). We show in the example how the sets $\mathcal{S}$, $\mathcal{M}$, and $\mathcal{R}$ have to updated in the illustrated configuration.

$$\mathcal{S} = \{S_1, S_2, S_3, S_4, S_5, S_6\},$$
$$\mathcal{M} = \{B_{23}, B_{34}, B_{12}, B_{45}\}, \qquad \mathcal{R} = \{B_{56}\},$$
$$\mathcal{S} = \{S_2, S_1, S_3, S_4, S_5, S_6\}, \qquad \text{in} : B_{13}, \text{ out} : B_{23}$$
$$\mathcal{M} = \{B_{12}, B_{34}, B_{45}\}, \qquad \mathcal{R} = \{B_{13}, B_{56}\},$$
$$\mathcal{S} = \{S_2, S_1, S_4, S_3, S_5, S_6\}, \qquad \text{in} : B_{14}, B_{35}, \text{ out} : B_{13}, B_{45}$$
$$\mathcal{M} = \{B_{34}, B_{12}, B_{14}, B_{35}\}, \qquad \mathcal{R} = \{B_{56}\},$$
$$\mathcal{S} = \{S_2, S_4, S_1, S_3, S_5, S_6\}, \qquad \text{in} : B_{13}, B_{24}, \text{ out} : B_{12}, B_{34}$$
$$\mathcal{M} = \{B_{14}, B_{24}, B_{56}, B_{35}\}, \qquad \mathcal{R} = \{B_{13}\},$$
$$\mathcal{S} = \{S_4, S_2, S_1, S_3, S_5, S_6\}, \qquad \text{in} : B_{12}, \text{ out} : B_{14}$$
$$\mathcal{M} = \{B_{24}, B_{12}, B_{13}, B_{56}\}, \qquad \mathcal{R} = \{B_{35}\},$$
$$\mathcal{S} = \{S_4, S_2, S_1, S_3, S_6, S_5\}, \qquad \text{in} : B_{36}, \text{ out} : B_{35}$$
$$\mathcal{M} = \{B_{56}, B_{13}, B_{36}, B_{24}\}, \qquad \mathcal{R} = \{B_{12}\}$$
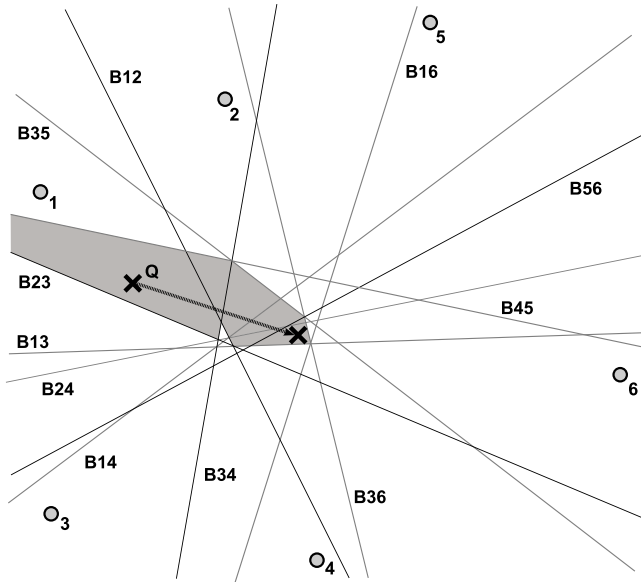
Fig. 4. A section of the trajectory of a moving query point. Sites are labeled with their ranks (i.e., when the query point was at its starting position). The grey areas are the fixed rank regions.

4.3   Complexity

As in the one-dimensional case, the initial ranking process can be computed in $\mathcal{O}(n \log n)$ time for n sites. For each crossed bisector we need to make an $\mathcal{O}(n)$ pass on the bisector lists to keep them up-to-date, because, in contrast to the one-dimensional case, there is no natural order of bisectors in two or higher dimensions. Thus, we make a full pass over the $\mathcal{R}$ to construct the new $\mathcal{M}$. In Section 5, we will see that given a direction of travel (an optimistic approach that assumes that this direction will be followed by the moving query point for a long time period), we can reduce the update complexity significantly. On the other hand, for trajectories that are arbitrary polygonal curves (note that in any case we do not have prior knowledge about the trajectory itself), it is an open question whether the bisectors can be maintained in $\mathcal{O}(\log n)$ time. This is a future research direction: the neighboring bisectors do not necessarily form an arbitrary set of lines, but may involve relationships similar to the ones presented previously in this paper.

We now show that the worst case complexity of the algorithm is $\mathcal{O}(n^3)$ for a movement of the query point that crosses a field of sites in its entirety.

Remark 1. If n sites are in general position and a query point Q moves on an arbitrary polygonal curve, then each line segment s of the polygonal curve can intersect up to $n(n-1)/2$ bisectors. The total number of updates on that line segment for our algorithm is then $\mathcal{O}(n^3)$.

Proof. Note that two sites can change their rank with respect to Q if and only if Q crosses their corresponding bisector. Each bisector can be written as a two-subset of

the n sites. Thus, there are at most $\binom{n}{2} = n(n-1)/2$ bisectors and each bisector corresponds to exactly one rank update. Each rank update is $\mathcal{O}(n)$, because the list $\mathcal{R}$ maintained in the algorithm is an unsorted list. Therefore, the total number of updates is at most $\mathcal{O}(n^3)$. This bound is tight (see Figure 5 for an example where $n = 6$). If all sites lie on a half-circle above the diameter of the circle aligned with the x-axis and Q moves along a line segment that is parallel to the x-axis and the line segment is above the intersection point of all bisectors and below the intersection points of B_{12} and B_{n-1n} with the circle, then Q has to intersect all the bisectors during its travel over the segment. It is important to note that along the path of the moving query point the bisectors are always introduced in front.
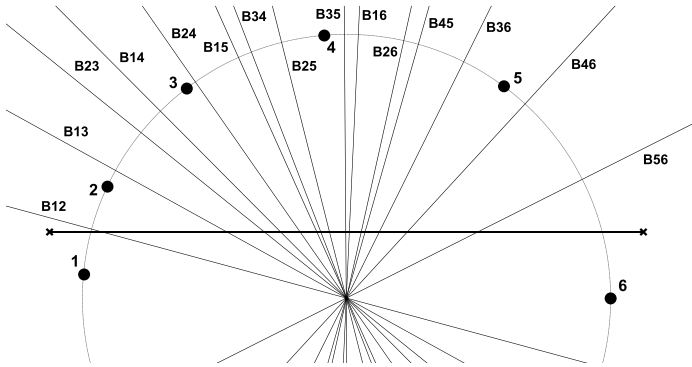

Fig. 5. All sites lie on a circle. A trajectory of a query point that will cross all the bisectors.

Note that the configuration in Figure 5 is not the only configuration that has the worst case complexity of $\mathcal{O}(n^3)$. Even if all sites do not lie on a circle but, for example, have slightly different positions such that the lines are not intersecting in a single point, the horizontal line in Figure 5 will still intersect the $\mathcal{O}(n^2)$ bisectors.

A simple solution to the rank maintenance problem that one can easily introduce is the precomputation of all the bisectors for n sites. In this case, for $\mathcal{O}(n^2)$ bisectors, we have to maintain our list $\mathcal{M}$ to detect the next update point. Using a similar worst case analysis, we can see that this leads to an $\mathcal{O}(n^4)$ cost, which is more expensive than our approach. This simple solution also has the disadvantage of precomputing all the bisectors even if the moving query point may lead to only a few updates in $\mathcal{S}$. Hence, this approach always performs worse than our algorithm.

Another simple solution would be to precompute the ordered order-k Voronoi diagram of n sites for $k = n$. This diagram will have $\mathcal{O}(n^4)$ faces in the worst case. The following example shows that the total number of faces is indeed $\mathcal{O}(n^4)$.

Remark 2. If $n/2$ (assuming n is even) sites are arranged parallel to the x-axis with increasing distances and the remaining $n/2$ sites are placed parallel to the y-axis (again with increasing distances), then there are

$$\underbrace{\frac{n}{2} - 1}_{\text{for } S_1} + \underbrace{\frac{n}{2} - 2}_{\text{for } S_2} + \ldots + \underbrace{1}_{\text{for } S_{n-1}}$$

bisectors along each axis. This generates $\mathcal{O}(n^4)$ faces.

One can link the neighboring cells of this diagram to form a data structure and enable rank maintenance by visiting sites on a trajectory of a moving query point. This approach, given a similar worst case scenario as above, will again perform poorly in comparison to our work due to the obvious precomputation overhead. In addition, it ignores the cases where only a few rank updates may be required on a trajectory, e.g., due to a short trajectory. On the other hand, for long trajectories that may include circular patterns, the precomputation of the ordered order-k Voronoi diagram may be quite beneficial. While our algorithm continues to invest $\mathcal{O}(n)$ time for maintaining the list $\mathcal{M}$ for every bisector crossing, the precomputation will only lead to an $\mathcal{O}(1)$ behavior for each update.

5   Multi-dimensional Queries with Directions

A common travel pattern that occurs in many real-life scenarios is a trajectory consisting of long straight line segments (e.g., Figure 6). In this case, an optimistic heuristic that gives a better performance than our original algorithm is to use balanced trees to store the list $\mathcal{R}$. This will reduce the cost of updating $\mathcal{M}$ from $\mathcal{R}$ to $\mathcal{O}(\log n)$ for n sites. Hence, the overall worst case behavior of the algorithm reduces to $\mathcal{O}(n^2 \log n)$ along a line segment.

This approach assumes that the number of line segments of the trajectory is much smaller than the number of bisectors to be crossed.

Given a direction we sort the n sites in $\mathcal{O}(n \log n)$ time. The list $\mathcal{R}$ of bisectors is stored in a balanced tree rather than an unsorted list. The cost of the creation of the balanced tree is also $\mathcal{O}(n \log n)$. The intersection points of the bisectors of $\mathcal{R}$ with the line segment can be used to store and find each bisector in this tree. Hence, bisector insertion and deletion operations can now be done in $\mathcal{O}(\log n)$ time as well as finding the bisector that the moving query point can intersect next after $\mathcal{M}$ is invalidated with a bisector crossing. Thus, redefinition of $\mathcal{M}$ can also be done in $\mathcal{O}(\log n)$ time.

In the worst case a moving query point crosses all $\mathcal{O}(n^2)$ bisectors along the straight line segment leading to computational complexity of $\mathcal{O}(n^2 \log n)$. Once the moving query point changes its direction, the tree that has been constructed for the old travel direction becomes invalid. The tree reconstruction requires an additional $\mathcal{O}(n \log n)$ cost for each direction change.

One marginal scenario that can occur is when many bisectors intersect at one point and a line segment from the trajectory of the moving query point also intersects with this point. In this case, many lines may fall under the same leaf node of the tree structure. For updating $\mathcal{M}$, this does not have an effect. Any of the rank updates can occur in any order. On the other hand, insertions and deletions to the tree may require an additional data structure. As we cannot differentiate these bisectors with their intersection points, we have to use their designations (e.g., k and l for B_{kl}). However, this does not change
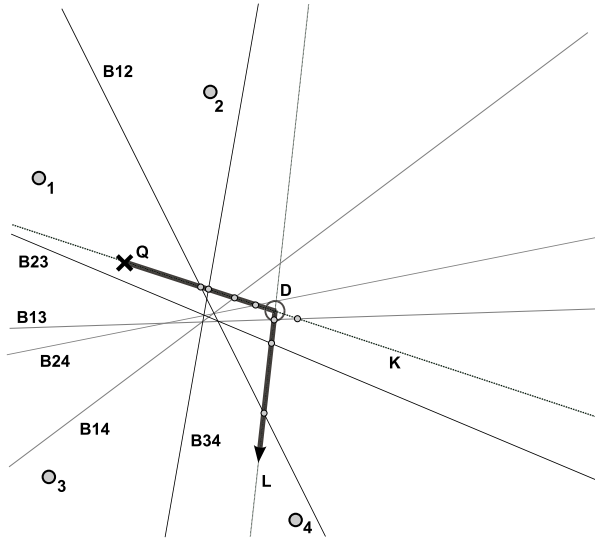
Fig. 6. A moving query point Q, four sites, and all of the bisectors for this setting. The query point moves from the starting point illustrated by a cross on K and then changes its direction at point D and moves on L. The order of the bisectors to be crossed does not change given a direction.

the space or time complexity of the algorithm as designations can also be used with a balanced tree structure.

6   General Rank Maintenance

We have introduced in Definition 2 the concept of a fixed-rank region. We developed two algorithms for incrementally maintaining the ranks of all sites. Our algorithms were based on the assumption that all sites are ranked consecutively. In this section we relax this assumption.

To motivate why non-consecutive ranks are important, we consider the following scenario: instead of a single agent ranking n sites, multiple agents are co-located and distribute their load by ranking the n sites jointly. Each agent only keeps track of a certain number of ranks. For example, if $\mathcal{S}$ consists of 9 sites, one agent keeps track of the first 3 ranks, a second agent the next 3 ranks and a third agent the last 3 ranks. Alternatively, the first agent could track the ranks 1, 5, 9, the second agent the ranks 2, 6, 8, and the last agent the ranks 3, 4, 7.

First, we present an algorithm for tracking a single rank k. The algorithm determines, for a given k, which site is of rank k for a query point at any position along its trajectory. This problem is related to the kth-nearest neighbor Voronoi diagram concept. A kth-nearest neighbor Voronoi region consists of all points for a given site S_i such that S_i is always the k-th closest site.

The algorithm for maintaining a single rank k works as follows (initially assume all sites are already ranked according to their distance). As the first step, the algorithm

selects the k-th site, called S_{1_k}. Then, at every step i, the algorithm keeps, for S_{i_k}, the list of all bisectors of S_{i_k}, because the rank of S_{i_k} can only change when one of those bisectors is crossed. More precisely, at each step we keep an unsorted list $\mathcal{B}_i = \bigcup_{j=1, j \neq k}^{n} \{B_{i_j i_k}\}$, and as long as the query point does not intersect any of those bisectors of $\mathcal{B}_i$, the rank of S_{i_k} does not change. Once the moving query point intersects one of the bisectors, this bisector will determine the new site S_{i+1_k} and $\mathcal{B}_{i+1}$ will be computed. The cost to keep $\mathcal{B}_i$ at every step is $\mathcal{O}(n)$. If m sites are visited as being the k-th neighbor, the total complexity is $\mathcal{O}(mn)$.

This algorithm can be easily extended to keep track of l ranks. In the first step, the algorithm selects the l sites according to their rank and stores them as an l tuple $(S_{1_{k_1}}, \ldots, S_{1_{k_l}})$. As in the algorithm for a single rank, we keep l lists $\mathcal{B}_{i_1}, \ldots, \mathcal{B}_{i_l}$ for the l ranks. The total cost of the algorithm is $\mathcal{O}(ln)$ at each step, which leads to a total complexity of $\mathcal{O}(mln)$ for m rank updates. If j agents keep track of n ranks and the ranks form a partition of the n ranks so that $\sum_{i=1}^{j} l_i = n$, then the total complexity for all j agents is $\sum_{i=1}^{j} O(ml_i n) = O(mn^2)$.

7   Conclusions and Future Work

In this paper, we introduced the continuous rank maintenance problem for moving query points. In contrast to previous work, the CNN queries investigated in this work are order sensitive. For a moving query point, we introduced algorithms to track the ranks of sites without any prior knowledge about the trajectory of the query point. We defined a fixed-rank region that determines the set of all those points for which the ranks of the sites do not change. This reduces the need for any query reprocessing and communication because in a fixed-rank region no updates are required. When one of the bisectors of the fixed-rank region is crossed, the ranks of all the sites are updated incrementally. We showed that our algorithms only need to keep track of $n - 1$ bisectors, from a set of $n(n - 1)/2$ bisectors, for all n sites. We introduced a more general concept for rank maintenance that allows us to track subsets of ranks. We developed algorithms that are able to track a single rank k as well as a subset of ranks, possibly by multiple agents. This enables a distributed approach for rank maintenance.

We introduced several extensions to our base algorithm. In the general case, in which a query point moves along an arbitrary polygonal curve, we showed that for each bisector crossing, we need an $\mathcal{O}(n)$ pass over the bisectors to keep the fixed-rank region up-to-date. In the special case, in which the trajectory consists of long straight line segments, we proved that only $\mathcal{O}(\log n)$ steps are required for updating the ranks of neighbors.

In our future work, we will investigate methods that reduce the number of updates for a moving query point that travels along an arbitrary polygonal curve in less than $\mathcal{O}(n)$ time. We plan to take advantage of the insight that the $n-1$ neighboring bisectors do not form an arbitrary arrangement of lines but may exhibit spatial relationships similar to the ones presented with this paper. Second, we will update our algorithms for situations where we can assume minimal prior knowledge about the trajectory of the query point. For example, a speed limitation may imply that the point cannot reach some segments of the bisectors given a period of time. Third, we will evaluate those cases where a query

point intersects concurrent bisectors. In the long term, we aim to adapt the algoritms to cope with dynamic situations in which the sites as well as the query point are in motion.

References

1. F. Aurenhammer. Voronoi diagrams: A survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, September 1991.
2. R. Benetis, C. S. Jensen, G. Karciauskas, and S. Saltenis. Nearest neighbor and reverse nearest neighbor queries for moving objects. In *Proceedings of the IEEE IDEAS*, pages 44–53, Edmonton, Canada, July 2002.
3. H. D. Chon, D. Agrawal, and A. El Abbadi. Range and KNN query processing for moving objects in grid model. *Mobile Networks and Applications*, 8(4):401–412, August 2003.
4. G. Hjaltason and H. Samet. Ranking in spatial databases. In *Proceedings of the SSD*, pages 83–95, Portland, ME, August 1995.
5. G. Hjaltason and H. Samet. Distance browsing in spatial databases. *ACM Transactions on Database Systems*, 24(2):265–318, June 1999.
6. G. Hjaltason and H. Samet. Index-driven similarity search in metric spaces. *ACM Transactions on Database Systems*, 28(4):517–580, December 2003.
7. H. Hu, J. Xu, and D. L. Lee. A generic framework for monitoring continuous spatial queries over moving objects. In *Proceedings of the ACM SIGMOD*, pages 479–490, Baltimore, MD, June 2005.
8. G. S. Iwerks, H. Samet, and K. Smith. Continuous k-nearest neighbor queries for continuously moving points with updates. In *Proceedings of the VLDB*, pages 512–523, Berlin, Germany, September 2003.
9. M. F. Mokbel. Continuous query processing in spatio-temporal databases. In *Proceedings of the EDBT (Workshops)*, pages 100–111, Heraklion, Greece, March 2004.
10. K. Mouratidis, M. Hadjieleftheriou, and D. Papadias. Conceptual partitioning: An efficient method for continuous nearest neighbor monitoring. In *Proceedings of the ACM SIGMOD*, pages 634–645, Baltimore, MD, June 2005.
11. A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Wiley, Chichester, NY, second edition, 2000.
12. J. O'Rourke. *Computational Geometry in C*. Cambridge University, Cambridge, UK, 1994.
13. N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In *Proceedings of the ACM SIGMOD*, pages 71–79, San Jose, CA, May 1995.
14. S. Saltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez. Indexing the positions of continuously moving objects. In *Proceedings of ACM SIGMOD*, pages 331–342, Dallas, TX, May 2000.
15. H. Samet. *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*. Addison-Wesley, Reading, MA, 1990.
16. H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, Reading, MA, 1990.
17. H. Samet. *Foundations of Multidimensional Data Structures*. Morgan Kaufmann, San Francisco, CA, 2006.
18. Z. Song and N. Roussopoulos. K-nearest neighbor search for moving query point. In *Proceedings of the SSTD*, pages 79–96, Redondo Beach, CA, July 2001.
19. Y. Tao and D. Papadias. Time-parameterized queries in spatio-temporal databases. In *Proceedings of the ACM SIGMOD*, pages 334–345, Madison, WI, June 2002.

20. Y. Tao and D. Papadias. Spatial queries in dynamic environments. *ACM Transactions on Database Systems*, 28(2):101–139, June 2003.

21. Y. Tao, D. Papadias, and Q. Shen. Continuous nearest neighbor search. In *Proceedings of the VLDB*, pages 287–298, Hong Kong, China, August 2002.

22. X. Xiong, M. F. Mokbel, and W. G. Aref. SEA-CNN: Scalable processing of continuous k-nearest neighbor queries in spatio-temporal databases. In *Proceedings of the IEEE ICDE*, pages 643–654, Tokyo, Japan, April 2005.

23. J. Zhang, M. Zhu, D. Papadias, Y. Tao, and D. Lee. Location-based spatial queries. In *Proceeding of the ACM SIGMOD*, pages 443–453, San Diego, CA, June 2003.

24. B. Zheng and D. L. Lee. Semantic caching in location-dependent query processing. In *Proceedings of the SSTD*, pages 97–116, Redondo Beach, CA, July 2001.

The Head-Body-Tail Intersection for Spatial Relations Between Directed Line Segments

Yohei Kurata and Max J. Egenhofer

National Center for Geographic Information and Analysis
and
Department of Spatial Information Science and Engineering
University of Maine
Boardman Hall, Orono, ME 04469-5711, USA
{yohei, max}@spatial.maine.edu

Abstract. Directed line segments are fundamental geometric elements used to model through their spatial relations such concepts as divergence, confluence, and interference. A new model is developed that captures spatial relations between pairs of directed line segments through the intersections of the segments' heads, bodies, and tails. This *head-body-tail intersection* identifies 68 classes of topological relations between two directed line segments highlighting two equal-sized subsets of corresponding relations that differ only by their empty and non-empty body-body intersections. The relations' conceptual neighborhood graph takes the shape of a torus inside a torus, one for each subset. Another 12 classes of topological relation classes are distinguished if the segments' exteriors are considered as well, lining up such that their conceptual neighborhood graph forms another torus that contains the other two tori. These conceptual neighborhoods as well as the relations' composition table enable spatial inferences and similarity assessments in a consistent and reasoned manner.

1 Introduction

Directed line segments (called *DL segments*) are fundamental geometric elements used in geographic databases to represent directed linear entities, such as one-way roads and watercourses. People also often employ arrow symbols, an expressive form of DL segments, to display dynamic phenomena, for instance, movement, interaction, causality, and change (Kurata and Egenhofer 2005b), and their spatial relations illustrate such scenarios as divergence, confluence, and interference (Kurata and Egenhofer 2006). Geographic databases have a tradition of processing spatial queries over line segments (Hoel and Samet 1991) and spatio-temporal database systems have been increasingly adding corresponding relations into query languages to enable the retrieval and analysis of spatio-temporal configurations at a high level of abstraction (Erwig and Schneider 1999). To further advance spatial query languages for directed linear element, and to supply a basis for the interpretation of the semantics of route

graphs (Krieg-Brückner and Shi, 2005) to communicate route descriptions in a meaningful way, formal models of spatial relations between DL segments and comprehensive reasoning mechanisms about those relations are needed. In comparison to the well-established methods for topological relations for regions, however, models of relations among DL segments have attracted only little attention. Since DL segments are 1-dimensional objects embedded in a 2-dimensional space, they have more degrees of freedom than regions in the same space, which is a property of DL segments that makes reasoning with them more intricate. This paper's scope is on those spatial relations that are invariant under topological transformations of the embedding space. Metric properties can be used subsequently to provide refinements of the resulting relations in analogy to the way line-line relations have been enhanced metrically (Nedas *et al.* in press).

Throughout the rest of this paper DL segments are illustrated as arrow symbols, which are essentially DL segments that are constrained such that they typically refer to (i.e., originate from, traverse, and point to) other elements in a diagram, thereby establishing the semantics of these related elements (Kurata and Egenhofer 2005a), whereas generic DL segments are often used independently. We focus on the direct relations between DL segments, which are formed by their mutual intersections, and do not consider indirect relations, which are materialized by other elements. Each DL segment is further assumed to be embedded in a 2-dimensional space and not to intersect with itself.

The remainder of this paper is structured as follows: Section 2 reviews models for line-line relations. Section 3 classifies the topological relations between two DL segments based on the intersections between the lines' heads, bodies, and tails (called the hbt-intersection), yielding 68 classes of DL relations. Section 4 schematizes these 68 classes by their conceptual neighborhood graph, thereby arranging the relations according to their similarities. Section 5 develops the relations' composition table, forming a formal foundation for qualitative reasoning about DL segments. Section 6 analyzes the effect of considering—in addition to the DL segments' heads, bodies, and tails—also their exteriors, rendering a DL relation model that identifies another twelve classes. Section 7 concludes with a discussion of future work items.

2   Models of Line-Line Relations

Binary relations between line segments in $\mathbb{R}^2$ (and their lower-dimensional relatives, temporal intervals in $\mathbb{R}^1$) have been studied extensively in artificial intelligence and spatio-temporal databases. Allen (1983) identified thirteen order relations between two temporal intervals embedded in $\mathbb{R}^1$. The *4-intersection* (Egenhofer and Franzosa 1991) captures the topological relations between two objects based on the existence/non-existence of geometric intersections between the objects' interiors and boundaries. For non-directed line segments in $\mathbb{R}^1$, it identifies eight topological relations (Pullar and Egenhofer 1988), essentially the same as for two regions in $\mathbb{R}^2$,

and distinguishes sixteen binary relations between two lines in $\mathbb{R}^2$, without capturing an equivalence relation (Hadzilacos and Tryfona 1992). The dimension-extended method of the 4-intersection (Clementini *et al*. 1993) finds eighteen line-line relations. The *9-intersection* (Egenhofer and Herring 1991) extends the 4-intersection by considering also the intersections with respect to the objects' exteriors, which gives rise to distinguishing formally 33 topological relations between two non-directed line segments in $\mathbb{R}^2$ (Egenhofer 1994). Another variation of the 4-intersection distinguishes explicitly the two boundaries (i.e., endpoints) of a line segment, identifying sixteen relations between two intervals in a temporal cycle, which are equivalent to topological relations between uni-directed line segments embedded in a cyclic 1-dimensional space (Hornsby *et al*. 1999). Models for detailed topological relations, capturing for non-empty intersections such properties as sequences of crossing types, intersection dimensions, and orientations of common segments, have been developed for region-region relations (Egenhofer and Franzosa 1995) and line-line relations (Clementini and di Felice, 1998), yielding a set of *topological invariants*. Such additional invariants of component intersections have been known to be germane to distinguishing even basic line-line relations, such as touching from crossing (Herring 1991). Nedas *et al*. (in press) provided metric refinements for detailed topological relations between line segments by incorporating two metric measures, *splitting ratios* and *closeness measures*, into the 9-intersection matrix and the topological invariants.

Conceptual neighborhood graphs, which establish links among relations to capture similarities among them, have been developed for a series of spatial and temporal relations (Egenhofer and Al-Taha 1992, Freska 1992a, Egenhofer and Mark 1995, Papadias *et al*. 1995, Schlieder 1995, Hornsby *et al*. 1999, Egenhofer 2005) to support qualitative analyses of change and to provide the foundations for computational similarity models over spatial relations (Egenhofer 1997) as well as a rationale for modeling the semantics of natural-language spatial predicates (Mark and Egenhofer 1994, Shariff *et al.* 1998). Several increments towards the identification of a conceptual neighborhood graph of line-line relations have been made, starting with Freksa's (1992a) neighborhood graphs of Allen's thirteen interval relations. A fragment of a conceptual neighborhood graph for line-line relations in $\mathbb{R}^2$ (Egenhofer *et al*. 1993) was recently incremented (Reis *et al*. 2005), but lacks the comprehensive treatment found for other relations. Apparently the degree of freedom for lines in a higher-dimensional space leads to an increased complexity for identifying those pairs of relations for which atomic transformations in the spirit of the conceptual neighborhood similarity are feasible.

In addition to topological relations, spatial relations of line segments have been categorized based on order and direction. The *double-cross* (Freksa 1992b) introduces left/right and front/back dichotomies to capture qualitative directions, yielding 15 direction relations from an observer's point to a target. Another approach, based on Allen's interval relations, identifies 63 order relations (essentially directional relations) between two straight DL segments in $\mathbb{R}^2$ (Schlieder 1995). The *dipole calculus* (Moratz *et al*. 2000) is based on 24 directional relations between two straight DL segments in $\mathbb{R}^2$, which fulfill the constraints of a relation algebra. Likewise, 26

order relations between two directed intervals in $\mathbb{R}^1$ form the *directed interval algebra* (Rentz 2001). The *Direction-Relation Matrix* provides an overall framework for describing direction relations between any pair of extended objects in $\mathbb{R}^2$, including arbitrarily shaped lines (Goyal and Egenhofer 2000).

3 Classes of Topological Relation Between DL Segments

A *DL segment* consists of two distinct points, a non-self-intersecting, continuous line that connects the two points, and an orientation imposed on the line, which categorizes the two points as start and end points. Following principles from algebraic topology (Alexandroff 1961), the two points form the *boundary* of a DL segment, whereas the connection between the points forms the *interior*. Due to the geometric similarity of DL segments and arrow symbols, the binary topological relations between DL segments are captured like the topological relations between two arrow symbols (Kurata and Egenhofer 2006), that is, based on the intersections between their interiors (the *body* X°) and the two ordered boundaries (*head* $\partial_{head}X$ and *tail* $\partial_{tail}X$). These nine intersections yield the DL segments' *head-body-tail-intersection* (*hbt-intersection*), which is concisely represented by a 3×3 matrix, called the *hbt-matrix* for DL segment relations (Eqn. 1).

$$M_I(A,B)=\begin{pmatrix} \partial_{tail}A\cap\partial_{tail}B & \partial_{tail}A\cap B^\circ & \partial_{tail}A\cap\partial_{head}B \\ A^\circ\cap\partial_{tail}B & A^\circ\cap B^\circ & A^\circ\cap\partial_{head}B \\ \partial_{head}A\cap\partial_{tail}B & \partial_{head}A\cap B^\circ & \partial_{head}A\cap\partial_{head}B \end{pmatrix} \qquad \textbf{(1)}$$

Like for line-line relations, each intersection could be analyzed for various topological invariants, such as its dimension, the number of intersections, and the type of intersection, but we choose the most fundamental property, that is, whether each intersection is empty (ϕ) or non-empty ($\neg\phi$). The empty/non-empty entries of the hbt-matrix are constrained among each other since the first and third column, as well as the first and third row, have at most one non-empty entry, because the head or tail of a DL segment is a point, which cannot intersect with more than one part of another DL segment. Among the $2^9 = 512$ configurations of a 3×3 matrix with empty or non-empty entries, only 68 cases satisfy this constraint (Tables 1 and 2). Each of these 68 configurations corresponds to a different set of topological relations (Fig. 1), while the remaining 444 configurations have no valid geometric interpretations. In this way,
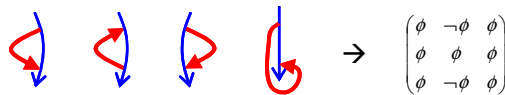

Fig. 1. Five different topological relations between two DL segments with the same hbt-matrix, which implies that the hbt-intersection groups this set of DL relations into the same TR-class

the hbt-intersection classifies topological relations between two DL segments into 68 topological relation classes (*TR-classes*). Among the 68 TR-classes, 34 classes (Table 1) are TR-classes without body-body intersections ($A° \cap B° = \phi$), whereas the other half (Table 2) has body-body intersections ($A° \cap B° = \neg \phi$).

Each TR-class is given a name, based on the name primitives *split* (*sp*), *diverge* (*dv*), *precede* (*pr*), *divergedBy* (*dvB*), *cross/touch* (*ct*), *mergedBy* (*mgB*), *follow* (*fl*), *merge* (*mg*), and *meet* (*mt*) that are assigned to the nine types of intersections (Eqn. 2).

$$\begin{pmatrix} split\ (sp) & diverge\ (dv) & preceed\ (pr) \\ divergedBy\ (dvB) & cross/touch\ (ct) & mergedBy\ (mgB) \\ follow\ (fl) & merge\ (mg) & meet\ (mt) \end{pmatrix} \tag{2}$$

The names of each TR-class are then described as a set of these name primitives, like *split-cross/touch-meet*, thereby indicating the intersections that establish each TR-class. The sequence in such a compound name, however, does not reflect the sequence of the intersection types that one might observe by following one segment form head to tail, but follows a standardized naming convention (row-order across Equation 2, i.e., *sp-dv-pr-dvB-ct-mgB-fl-mg-mt*). To more concisely refer to TR-classes, we also give each TR-class a numeric label. Tables 1 and 2 show the mappings from names to numeric labels.

Properties of the hbt-matrices reflect certain algebraic properties of the corresponding relations. For example, if an hbt-matrix N can be obtained by transposing another hbt-matrix M along its main diagonal (i.e., $N = M^T$), then the two corresponding TR-classes form a pair of converse relations. Likewise, if $M = M^T$, then the corresponding relation is symmetric. In total, there are twenty symmetric TR-classes, and each of the remaining 48 TR-classes has a converse TR-class within these 48 TR-classes, forming 24 pairs of converse TR-classes.

The TR-class whose matrix has non-empty entries along the main diagonal, but empty entries otherwise (i.e., #44: *split-cross/touch-meet*), deserves special attention, as it includes the situation in which two DL segments completely coincide, which would correspond to the equality of two DL segments and imply this class's transitive property (Fig. 2a). This coincidence is, however, not the only configuration that matches the hbt-matrix specification, because other topological configurations than equal (Fig. 2b) are also covered by the same hbt-matrix of non-empty values for the three intersections of the head-head, body-body, and tail-tail pairs. As a consequence, this TR-class is not a transitive relation (Fig. 2c). This issue is revisited in Section 6.
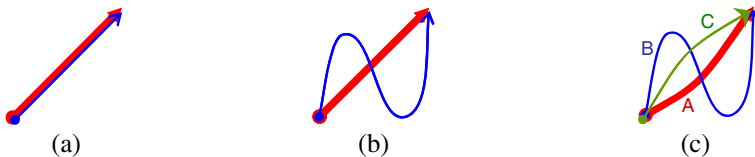

Fig. 2. Two configurations (a) and (b) covered by TR-class #44, and (c) an example that this TR-class is not transitive

Table 1. The 34 TR-classes without body-body intersections (#1 through #34) with examples of geometric prototypes

	symmetric TR-classes		converse pairs of asymmetric TR-classes		

0

#1 *disjoint*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

1

#2 *split*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

#3 *meet*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$

#4 *precede*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \phi & \phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$
↑↓
#5 *follow*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \phi & \phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$

#6 *diverge*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$
↑↓
#7 *divergedBy*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

#8 *mergedBy*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$
↑↓
#9 *merge*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

2

#10 *split-meet*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$

#11 *precede-follow*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \phi & \phi & \phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$

#12 *diverge-divergedBy*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

#13 *mergedBy-merge*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#14 *diverge-merge*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$
↑↓
#15 *divergedBy-mergedBy*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$

#16 *diverge-mergedBy*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$
↑↓
#17 *divergedBy-merge*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#18 *split-mergedBy*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$
↑↓
#19 *split-merge*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#20 *divergedBy-meet*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$
↑↓
#21 *diverge-meet*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$

#22 *diverge-follow*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \phi & \phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$
↑↓
#23 *precede-divergedBy*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \neg\phi & \phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

#24 *precede-merge*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \phi & \phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$
↑↓
#25 *mergedBy-follow*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \phi & \neg\phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$

3

#26 *split-mergedBy-merge*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#27 *diverge-divergedBy-meet*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$

#28 *precede-divergedBy-merge*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \neg\phi & \phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$
↑↓
#29 *diverge-mergedBy-follow*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \phi & \neg\phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$

#30 *diverge-divergedBy-mergedBy*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$
↑↓
#31 *diverge-divergedBy-merge*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#32 *divergedBy-mergedBy-merge*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$
↑↓
#33 *diverge-mergedBy-merge*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

4

#34 *diverge-divergedBy-mergedBy-merge*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

number of intersections

Table 2. The 34 TR-classes with body-body intersections (#35 through #68) with examples of geometric prototypes

	symmetric TR-classes	converse pairs of asymmetric TR-classes		

number of intersections

1

#35 *cross/touch*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \neg\phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

2

#36 *split-cross/touch*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \neg\phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

#37 *cross/touch-meet*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \neg\phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$

#38 *precede-cross/touch*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \phi & \neg\phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$
⇅
#39 *cross/touch-follow*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \neg\phi & \phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$

#40 *diverge-cross/touch*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$
⇅
#41 *divergedBy-cross/touch*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \neg\phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

#42 *cross/touch-mergedBy*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \neg\phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$
⇅
#43 *cross/touch-merge*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

3

#44 *split-cross/touch-meet*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \neg\phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$

#45 *precede-cross/touch-follow*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \phi & \neg\phi & \phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$

#46 *diverge-divergedBy-cross/touch*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \neg\phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

#47 *cross/touch-mergedBy-merge*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \neg\phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#48 *diverge-cross/touch-merge*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$
⇅
#49 *divergedBy-cross/touch-mergedBy*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \neg\phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$

#50 *divergedBy-cross/touch-mergedBy*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \neg\phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$
⇅
#51 *divergedBy-cross/touch-merge*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#52 *split-cross/touch-mergedBy*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \neg\phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$
⇅
#53 *split-cross/touch-merge*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#54 *divergedBy-cross/touch-meet*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \neg\phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$
⇅
#55 *diverge-cross/touch-meet*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$

#56 *diverge-cross/touch-follow*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$
⇅
#57 *precede-divergedBy-cross/touch*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \neg\phi & \neg\phi & \phi \\ \phi & \phi & \phi \end{pmatrix}$$

#58 *precede-cross/touch-merge*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$
⇅
#59 *cross/touch-mergedBy-follow*
$$\begin{pmatrix} \phi & \phi & \phi \\ \phi & \neg\phi & \neg\phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$

4

#60 *split-cross/touch-mergedBy-merge*
$$\begin{pmatrix} \neg\phi & \phi & \phi \\ \phi & \neg\phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#61 *diverge-divergedBy-cross/touch-meet*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \neg\phi & \phi \\ \phi & \phi & \neg\phi \end{pmatrix}$$

#62 *precede-divergedBy-cross/touch-merge*
$$\begin{pmatrix} \phi & \phi & \neg\phi \\ \neg\phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$
⇅
#63 *diverge-cross/touch-mergedBy-follow*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \neg\phi & \neg\phi \\ \neg\phi & \phi & \phi \end{pmatrix}$$

#64 *diverge-divergedBy-cross/touch-mergedBy*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \neg\phi & \neg\phi \\ \phi & \phi & \phi \end{pmatrix}$$
⇅
#65 *diverge-divergedBy-cross/touch-merge*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \neg\phi & \phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

#66 *divergedBy-cross/touch-mergedBy-merge*
$$\begin{pmatrix} \phi & \phi & \phi \\ \neg\phi & \neg\phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$
⇅
#67 *diverge-cross/touch-mergedBy-merge*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \phi & \neg\phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

5

#68 *diverge-divergedBy-cross/touch-mergedBy-merge*
$$\begin{pmatrix} \phi & \neg\phi & \phi \\ \neg\phi & \neg\phi & \neg\phi \\ \phi & \neg\phi & \phi \end{pmatrix}$$

4   Conceptual Neighborhoods of the 68 TR-Classes

If a continuous transformation can be performed between two spatial relations without having to go through a third relation, then this establishes a *direct transition*. These two relations have been called *conceptual neighbors* (Freksa 1992a). For example, the two TR-classes *split* (#2) and *split-mergedBy* (#18) are conceptual neighbors, because moving the head of the first DL segment from the other segment's exterior to its body is a direct transition. On the other hand, TR-classes *spilt* (#2) and *meet* (#3) are not neighbors, because disconnecting the head-tail intersection of *split* and establishing the head-head intersection of *meet* would need to go through either *split-meet* (#10) or *disjoint* (#1). We focus here on the equivalents of type-A neighbors for 1-dimensional intervals (Freksa 1992a), that is, those pairs of relations that can be transformed into each other by changing in their hbt-matrices one intersection from empty to non-empty, or vice-versa, but not several of them simultaneously. Equivalents of Type-B neighbors would require an entire DL segment to move—establishing, for instance, a link between *disjoint* (#1) and *diverge-cross/touch-mergedBy* (#50)—whereas the equivalents of Type-C neighbors are such that both boundary points of a DL segment would need to be moved—for example, turning *disjoint* (#1) and *diverge-merge* (#14) into neighbors.

The conceptual neighborhood graph of the 68 TR-classes is derived computationally from the relations' hbt-matrices, counting the number of differences in corresponding matrix cells with regards to empty/non-empty values (Egenhofer and Al-Taha 1992). Pairs of TR-classes with no differences in corresponding matrix cells would mean that the two TR-classes are identical, whereas pairs of TR-classes with a single difference across their hbt-matrices reflect an atomic change, dissolving either an intersection of two boundary elements, or an intersection between a boundary element and a body, or an intersection between two bodies. Pairs of TR-classes with a difference of 1 are called *conceptual 1-neighbors*.

The 1-neighbors among TR-classes #1 through #34 (as well among TR-classes #35 through #68) have the following countable properties: One TR-class has eight 1-neighbors; four TR-classes have six 1-neighbors and another four TR-classes have five 1-neighbors; eleven TR-classes have four 1-neighbors; twelve TR-classes have three 1-neighbors; and two TR-classes have two 1-neigbors. Across the two groups of TR-classes, each TR-class has exactly one 1-neighbor such that #n and #(n+34) are neighbors (1≤n≤34). This means each TR-class has, in addition to the 1-neighbors within its group, another 1-neighbor that extends to the other group, increasing the neighbor count by one.

In the conceptual neighborhood graph, each TR-class has a node and each 1-neighbor of two TR-classes maps onto an edge that links the TR-classes' two nodes; therefore, the number of 1-neighbors of a TR-class corresponds to the node's degree within the neighborhood graph. These properties bring forth a conceptual neighborhood graph of one layer for TR-classes #1 through #34 (Fig. 3), and another one for TR-classes #35 through #68.
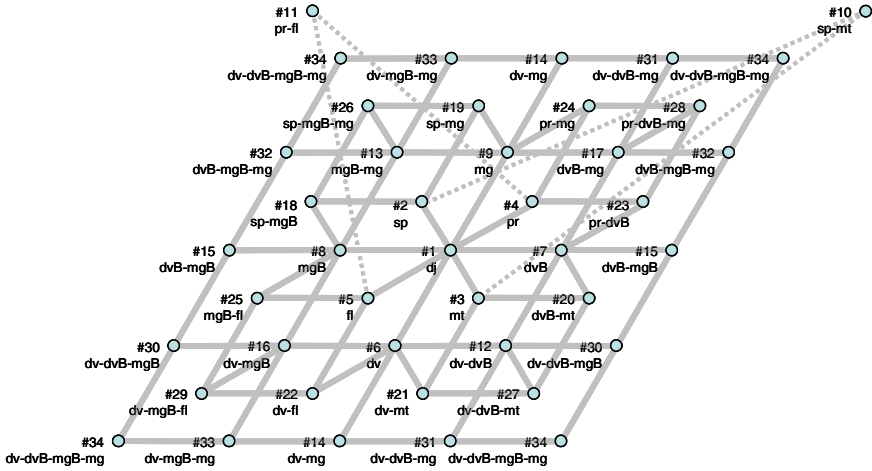
Fig. 3. The flattened conceptual neighborhood graph of the TR-classes #1 through #34. It displays more than 34 nodes in order to highlight some of the regularities of the neighborhoods by repeating the nodes in the front and back row as well as in the left and right column

The graph highlights the special status of the TR-classes *split-meet* (#10) and *precede-follow* (#11) as the only TR-classes with two conceptual neighbors among their 34 companions without body-body intersections. A visual inspection might suggest that this graph misses some links (e.g., along the diagonal from #5 to #16 or from #29 to #34), but their hbt-matrices confirm that such pairs cannot qualify as 1-neighbors, because any transition among them would require moving through another TR-class (e.g., through #6 or #8 to get from #5 to #16).

This graph of 1-neighbors also keeps its role as a reference framework when other kinds of neighbors would be considered (e.g., 2-neighbors whose matrix difference would be 2). In such a case, shortcuts along this graph would be introduced (e.g., another eight links among the eight TR-classes #2 thought #9 with exactly one non-empty intersection). Subsequently, only the graph derived from the 1-neighbors is analyzed. The conceptual neighborhood graphs of TR-classes #1 through #34, as well as TR-classes #35 through #68, have the following characteristics:

- TR-classes with fewer intersections are located closer to the center. This property is, however, one of choice, because different elements than TR-class #1 could have been selected as the central reference point.
- TR-classes located along the diagonal from top-left to bottom-right are symmetric (Fig. 4a).
- Pairs of TR-classes that are located symmetrically across the diagonal from top-left to bottom-right are converse, that is, the same matrices are obtained by transposing the matrices along their main diagonals (Fig. 4a).
- The conceptual neighborhood graph can be decomposed into four subgraphs with a horizontal and vertical mirror axis such that the same matrices are obtained by reversing the direction of one DL segment (Fig. 4b).
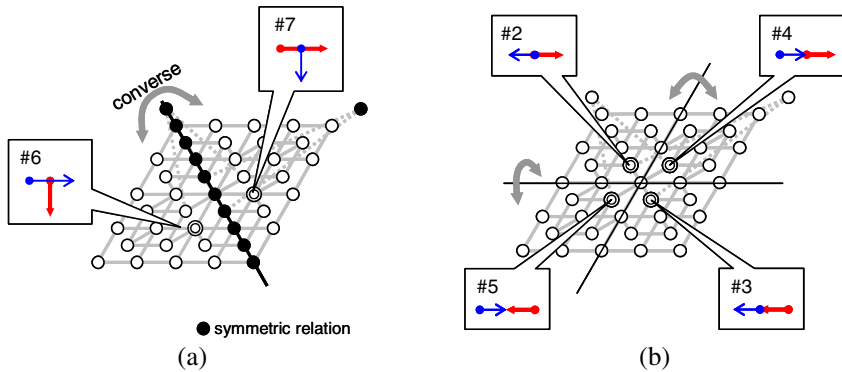
Fig. 4. Characteristics of the flattened conceptual neighborhood graph of TR-classes #1 through #34: (a) symmetric and converse TR-classes and (b) the four subgraphs that are obtained by reversing the direction of a DL segment when mirroring the TR-class along the graph's horizontal or vertical axis, as highlighted by TR-classes #2-#5

Gluing the flattened graph's front and back rows, and then the leftmost and rightmost columns, yields a non-redundant configuration in which the graph extends over the surface of a torus (Fig. 5b). TR-classes #10 and #11, which placed irregularly above the flattened graph (Fig. 3), are now outside the torus (Fig. 5b), highlighting through their irregular locations again the two nodes' unique properties of degree 2.
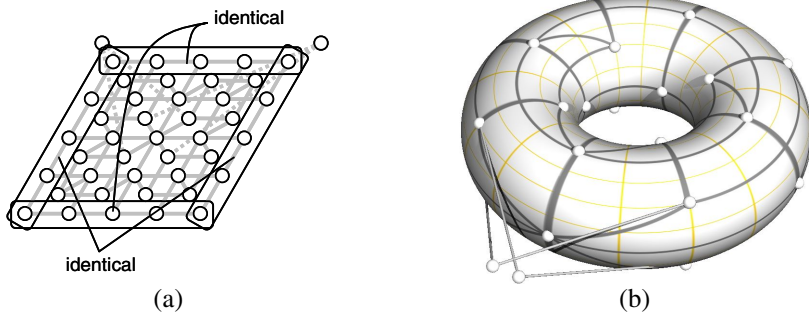
Fig. 5. The transition from (a) the flattened conceptual neighborhood graph of TR-classes #1 through #34 with repeated columns and rows to (b) a graph displayed on the surface of a torus, which is obtained by gluing together the repeated rows and columns along the fringes of the flattened graph.

The integrated conceptual neighborhood graph of TR-classes #1 through #68 has a two-layered structure, where each layer contains a homeomorphic conceptual neighborhood graph of #1 through #34 or #35 through #68, and each node in one of these layers is linked to one node in another layer, thereby representing the neighbor relation between #n and #(n+34). Consequently, the conceptual neighborhood graph of the TR-classes #1 through #68 is represented in a 3-dimensional space where nodes are aligned on two parallel planes (Fig. 6a). When the flattened structure is connected

through the redundant nodes at its fringes, the graph forms a shape that extends over the surfaces of two tori, one inside the other, with links across the two tori to establish the neighborhoods of #*n* and #(*n*+34) with 1≤*n*≤34 (Fig. 6b).
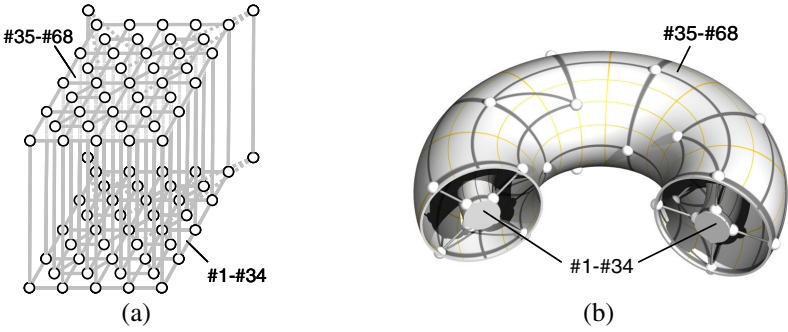

Fig. 6. Two structures of the conceptual neighborhood graph of TR-classes #1 through #68, where nodes are aligned on (a) two parallel planes and (b) the surfaces of two nested tori

5 Inferences About the 68 TR-Classes

The 68 DL segment relations allow us to capture the topological relations between these segments in a simple, consistent, and qualitative way that is suitable for processing queries about their relations recorded in a database. Queries may, however also refer to relations that have not been recorded explicitly so that it may be necessary to deduce result candidates from a logical combination of available information about several relations. In support of such queries involving implied relations, this section develops the *composition table* of the 68 TR-classes. Given three DL segments, *A*, *B*, and *C*, whose TR-classes are denoted by R_{AB}, R_{BC}, and R_{AC}, the composition of R_{AB} and R_{BC}, denoted by R_{AB} ; R_{BC}, is the set of all possible TR-classes between *A* and *C*. The actual value of R_{AC} is always included in R_{AB} ; R_{BC}.

5.1 Constraints on the Hbt-Matrix for Composed TR-Classes

Let the TR-classes R_{AB}, R_{BC}, and R_{AC} correspond to the respective hbt-matrices M_{AB}, M_{BC}, and M_{AC} (Eqn. 3).

$$M_{AB} = \begin{pmatrix} I_{AB}^{tt} & I_{AB}^{tb} & I_{AB}^{th} \\ I_{AB}^{bt} & I_{AB}^{bb} & I_{AB}^{bh} \\ I_{AB}^{ht} & I_{AB}^{hb} & I_{AB}^{hh} \end{pmatrix} \quad M_{BC} = \begin{pmatrix} I_{BC}^{tt} & I_{BC}^{tb} & I_{BC}^{th} \\ I_{BC}^{bt} & I_{BC}^{bb} & I_{BC}^{bh} \\ I_{BC}^{ht} & I_{BC}^{hb} & I_{BC}^{hh} \end{pmatrix} \quad M_{AC} = \begin{pmatrix} I_{AC}^{tt} & I_{AC}^{tb} & I_{AC}^{th} \\ I_{AC}^{bt} & I_{AC}^{bb} & I_{AC}^{bh} \\ I_{AC}^{ht} & I_{AC}^{hb} & I_{AC}^{hh} \end{pmatrix} \quad (3)$$

Symbols p_A, p_B, p_C denote arbitrary parts of *A*, *B*, and *C* (i.e., either tail or body or head). If *B*'s tail intersects with both p_A and p_C, then p_A and p_C must intersect, since *B*'s tail is a point on which both p_A and p_C are partly or wholly located. Similarly, if *B*'s head intersects with both p_A and p_C, then p_A and p_C must intersect as well. These geometric constraints lead to the constraint on M_{AC} that p_A and p_C intersect if *B*'s tail or head intersects with both p_A and p_C (Eqn. 4).

For $p_A, p_C \in \{t,b,h\}$

$$\left(I_{AB}^{p_A t} = \neg\phi \quad \wedge \quad I_{BC}^{tp_C} = \neg\phi\right) \quad \vee \quad \left(I_{AB}^{p_A h} = \neg\phi \quad \wedge \quad I_{BC}^{hp_C} = \neg\phi\right) \quad \rightarrow \quad I_{AC}^{p_A p_C} = \neg\phi \tag{4}$$

The intersection between A's tail and p_B, if it exists, is totally included in p_B, since A's tail is a point. Thus, if p_B intersects with A's tail, but not p_C, A's tail and p_C do not intersect. Similarly,

- A's head and p_C do not intersect if p_B intersects with A's head but not p_C,
- C's tail and p_A do not intersect if p_B intersects with C's tail but not p_A, and
- C's head and p_A do not intersect if p_B intersects with C's head but not p_A.

These four conditions imply another constraint on M_{AC} (Eqn. 5).

$$\text{For } p_A, p_B, p_C \in \{t,b,h\}$$

$$\begin{aligned}
I_{AB}^{tp_B} = \neg\phi \quad &\wedge \quad I_{BC}^{p_B p_C} = \phi \quad \rightarrow \quad I_{AC}^{tp_C} = \phi \\
I_{AB}^{hp_B} = \neg\phi \quad &\wedge \quad I_{BC}^{p_B p_C} = \phi \quad \rightarrow \quad I_{AC}^{hp_C} = \phi \\
I_{AB}^{p_A p_B} = \phi \quad &\wedge \quad I_{BC}^{p_B t} = \neg\phi \quad \rightarrow \quad I_{AC}^{p_A t} = \phi \\
I_{AB}^{p_A p_B} = \phi \quad &\wedge \quad I_{BC}^{p_B h} = \neg\phi \quad \rightarrow \quad I_{AC}^{p_A h} = \phi
\end{aligned} \tag{5}$$

Among the 68 configurations of the hbt-intersection (Tables 1 and 2), the configurations that satisfy the two constraints (Eqs. 4 and 5) are the candidates for M_{AC} and, therefore, the TR-classes that correspond to these candidates of M_{AC} configurations are the compositions of TR-classes R_{AB} and R_{BC}.

5.2 Composition Table for TR-Classes of DL Segment Relations

The compositions of the TR-classes were determined from the hbt-matrices (Tables 1 and 2) and the constraints on the hbt-matrix of the composition (Eqs. 4 and 5). The resulting composition table for the TR-classes between DL segments contains 68×68 = 4,624 composition elements, of which Table 3 shows a small subset for TR-classes #7, #9, #10, #19, and #44.

The validity of the composition table is demonstrated, although not proved formally, by the following observations:

- Any composition is connected in the conceptual neighborhood graph.
- Any composition satisfies the following constraint:

$$\forall X \quad R_{AA} = sp\text{-}ct\text{-}mt \in R_{AX} ; R_{XA} \tag{6}$$

Eqs. 4 and 5 do not contain the constraint on the non-existence of a body-body intersection. Accordingly, if a composition contains a TR-class without a body-body intersection (say R^-), then the composition also contains the TR-class that adds a body-body intersection to R^-. This means that if a TR-class in the lower layer is in a composition, then the vertical neighbor of this TR-class (located in the upper layer) is also in that composition (Table 3). For example, the composition *split-merge*; *split-meet*, contains *split* and *split*'s vertical neighbor, that is, *split-cross/touch*.

Table 3. A 5×5 subset of the composition table of the 68 TR-classes between DL segments
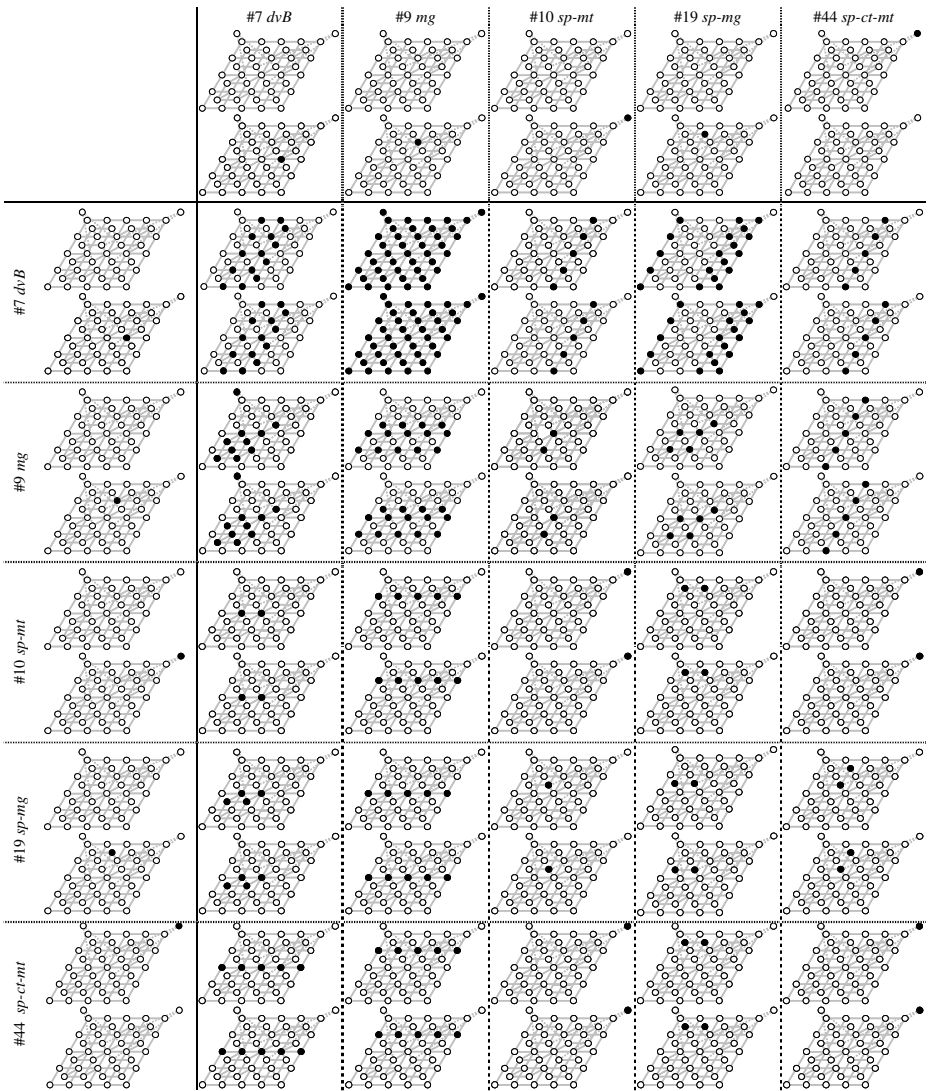
5.3   A Reasoning Example

To demonstrate the use of the composition table, consider a snow mountain with four skiing trails, *A*, *B*, *C*, and *D* (Fig. 7). A skier went down on *B* and found that:

- Trail *A* shares the same start point with *B*, takes a different course, and eventually ends at a *B*'s midpoint.
- Trail *C* starts at a *B*'s midpoint and does not intersect with B after that point.

The inference scenario is to determine symbolically (i.e., in a non-graphical way) what are the possible relations between trails *A* and *C*.
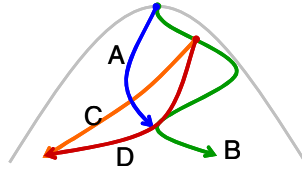

Fig. 7. A snow mountain with four skiing trails

The TR-class between *A* and *B* is *split-merge* (#19) and the TR-class between *B* and *C* is *divergedBy* (#7). According to Table 2, the composition of *split-merge* and *divergedBy* is *disjoint* (#1), *follow* (#5), *mergedBy* (#8), *mergedBy-follow* (#25), *cross/touch* (#35), *cross/touch-follow* (#39), *cross/touch-mergedBy* (#42), and *cross/touch-mergedBy-follow* (#59). Thus, these eight TR-classes are the candidates of the relation between *A* and *C*.

When the skier went down on trail *D*, she found that:

- Trail *A* ends at *D*'s midpoint and does not intersect before that point.
- Trail *C* shares the same start point with trail *D*, takes a different course, and eventually ends at the same destination with trail *D*.

This additional information further influences the inference about the possible relations between trails *A* and *C*. The TR-class between *A* and *D* is *merge* (#9) and the TR-class between *D* and *C* is *split-meet* (#10). According to Table 2, the composition of *merge* and *split-merge* is *disjoint* (#1), *diverge* (#6), *cross/touch* (#35), and *diverge-cross/touch* (#40). These four TR-classes are also the candidates of the relation between *A* and *C*. By integrating this result with the previous result, the possible TR-classes between *A* and *C* are narrowed down to *disjoint* and *cross/touch*. This indicates that the richer knowledge about the network reduces the ambiguity of its unrecorded relations.

6   The HBT$^+$-Intersection

Although the hbt-intersection features nine types of intersections, it resembles the 4-intersection (Egenhofer and Franzosa 1991) more than the 9-intersection (Egenhofer and Herring 1991), because both the hbt-intersection and the 4-interesection assess the intersections between the interiors and boundaries, but not the intersections with exteriors. Since the 4-intersection cannot distinguish some topological relations between line segments that the 9-intersection can distinguish (Egenhofer 1994), it could be expected that the hbt-intersection cannot distinguish some topological relations between DL segments as well. The observation about TR-class *split-cross/touch-meet* (#44) already gave evidence for an hbt-intersection that captures at least two significantly different topological relations (Figs. 2a-b). By adding to the hbt-intersection the intersections with the DL-segments' exteriors, however, the

difference between these two configurations could be captured: Fig. 2a's bodies have empty intersections with the opposite DL-segment's exteriors, whereas Fig. 2b's bodies have non-empty intersections with the opposite DL-segments' exteriors. The extension by considering the exteriors of DL segments in addition to their heads, bodies, and tails requires a total of 16 types of intersections, which are represented by the *hbt⁺-matrix,* a 4×4 matrix whose top-left 3×3 submatrix is identical to the configuration's hbt-matrix.

Among the 2^{16} = 65,536 configurations with empty and non-empty values for the hbt⁺-matrix, only 80 configurations have geometric interpretations. This number includes the 68 configurations obtained with the hbt-intersection plus another twelve configurations, each of which is a more constrained version of an hbt-intersection. These twelve configurations are shown in Table 4. The TR-classes that are captured by the hbt⁺-matrix include a specification of the equivalence relations for DL segments as a refinement of TR-class #44, constraining the two bodies to coincide, and none of the two bodies to extend through the other DL-segment's exterior.

The addition of these twelve TR-classes also has an impact on the conceptual neighborhood graph. Since twelve TR-classes in the hbt-intersection are subdivided into 24 TR-classes in the hbt⁺-intersection, the layer containing these twelve TR-classes (i.e., the upper layer in Fig. 6a) is split into two layers: one for the twelve collapsed TR-classes and another for 34 non-collapsed TR-classes with body-body intersections. Consequently, the conceptual neighborhood graph's structural framework consists of three parallel planes stacked on top of each other in the flattened version (Fig. 8a), or three tori nested transitively into each other (Fig. 8b).

Table 4. Collapsed topological relations and non-collapsed topological relations that can be distinguish by the hbt⁺-intersection, but not by the hbt-intersection

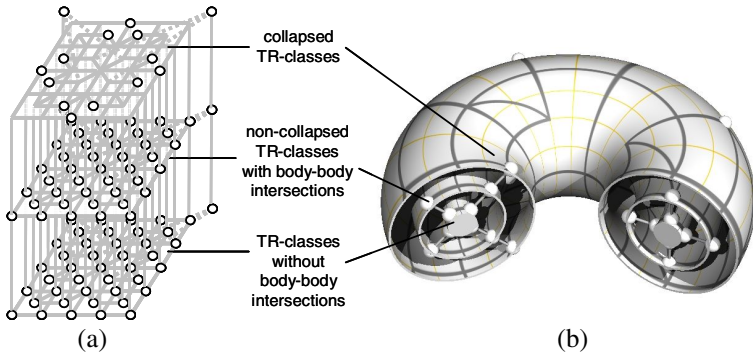|	Collapsed	Non-collapsed		Collapsed	Non-collapsed
#44	$\begin{pmatrix}\neg\phi & \phi & \phi & \phi\\ \phi & \neg\phi & \phi & \phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\neg\phi & \phi & \phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	#45	$\begin{pmatrix}\phi & \phi & \neg\phi & \phi\\ \phi & \neg\phi & \phi & \phi\\ \neg\phi & \phi & \phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\phi & \phi & \neg\phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\\ \neg\phi & \phi & \phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\end{pmatrix}$
#48	$\begin{pmatrix}\phi & \neg\phi & \phi & \phi\\ \phi & \neg\phi & \phi & \phi\\ \phi & \phi & \neg\phi & \phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\phi & \neg\phi & \phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\end{pmatrix}$	#49	$\begin{pmatrix}\phi & \phi & \phi & \neg\phi\\ \neg\phi & \neg\phi & \neg\phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\phi & \phi & \phi & \neg\phi\\ \neg\phi & \neg\phi & \neg\phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\end{pmatrix}$
#52	$\begin{pmatrix}\neg\phi & \phi & \phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\neg\phi & \phi & \phi & \phi\\ \phi & \neg\phi & \neg\phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	#53	$\begin{pmatrix}\neg\phi & \phi & \phi & \phi\\ \phi & \neg\phi & \phi & \phi\\ \phi & \phi & \neg\phi & \phi\\ \neg\phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\neg\phi & \phi & \phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \neg\phi & \phi & \phi & \neg\phi\end{pmatrix}$
#54	$\begin{pmatrix}\neg\phi & \phi & \phi & \neg\phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\neg\phi & \phi & \phi & \neg\phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	#55	$\begin{pmatrix}\phi & \neg\phi & \phi & \phi\\ \phi & \neg\phi & \phi & \phi\\ \phi & \phi & \neg\phi & \phi\\ \neg\phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\phi & \neg\phi & \phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \neg\phi & \phi & \phi & \neg\phi\end{pmatrix}$
#56	$\begin{pmatrix}\phi & \neg\phi & \phi & \phi\\ \phi & \neg\phi & \phi & \phi\\ \neg\phi & \phi & \phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\phi & \neg\phi & \phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\\ \neg\phi & \phi & \phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	#57	$\begin{pmatrix}\phi & \phi & \neg\phi & \phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\phi & \phi & \neg\phi & \phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \neg\phi & \phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$
#58	$\begin{pmatrix}\phi & \phi & \neg\phi & \phi\\ \phi & \neg\phi & \phi & \phi\\ \phi & \neg\phi & \phi & \phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\phi & \phi & \neg\phi & \phi\\ \phi & \neg\phi & \phi & \neg\phi\\ \phi & \neg\phi & \phi & \phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\end{pmatrix}$	#59	$\begin{pmatrix}\phi & \phi & \phi & \neg\phi\\ \phi & \neg\phi & \phi & \phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\\ \phi & \phi & \phi & \neg\phi\end{pmatrix}$	$\begin{pmatrix}\phi & \phi & \phi & \neg\phi\\ \phi & \neg\phi & \phi & \neg\phi\\ \neg\phi & \neg\phi & \phi & \neg\phi\\ \phi & \neg\phi & \phi & \neg\phi\end{pmatrix}$

Fig. 8. Structures of the conceptual neighborhood graph of the 80 TR-classes obtained by the hbt$^+$-intersection, where nodes are aligned on (a) three parallel planes or (b) the surfaces of three tori nested transitively in each other.

7   Conclusions

This paper classified the topological relations between two directed line segments into 80 classes, 68 of them were identified by the head-body-tail intersection, whereas intersections with the line segments' exteriors were needed to distinguish the remaining 12 cases. The 80 relations' conceptual neighborhood graph revealed a structure of three parallel planes or three nested tori. For the first time, such a comprehensive account of a conceptual neighborhood graph involving DL segment relations in a 2-dimensional space has been determined.

Future work about directed line relations will address the expressive power of the matrices' empty and non-empty intersections. If the hbt-matrix and hbt$^+$-matrix continue to record the existence/non-existence of the intersections, then the corresponding intersection models cannot distinguish some critical topological relations (Fig. 9). The development of more detailed model of the topological relations between DL segments will contribute to an improvement of the reasoning power about directed line segments.
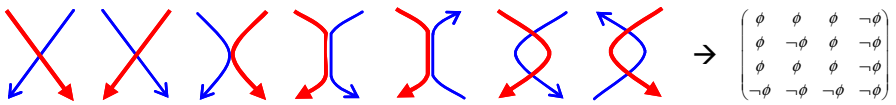

Fig. 9. Sets of DL segments, whose topological relations cannot be distinguished by the hbt-intersections or hbt$^+$-intersections

Acknowledgments

References

Alexandroff, P. (1961) *Elementary Concepts of Topology*. Dover: Mineola, NY.

Allen, J. (1983) Maintaining Knowledge About Temporal Intervals. *Communications of the ACM*. 26(11):832-843.

Clementini, E. and di Felice, P. (1998) Topological Invariants for Lines. *IEEE Transactions on Knowledge and Data Engineering*. 10(1):38-54.

Clementini, E., di Felice, P., and van Oosterom, P. (1993) A Small Set of Formal Topological Relationships Suitable for End-User Interaction. in: Abel, D. and Ooi, B.-C. (eds.) *Advances in Spatial Databases, Third International Symposium, SSD'93*, Singapore, Lecture Notes in Computer Science, 692, 277-295.

Egenhofer, M. (1994) Definitions of Line-Line Relations for Geographic Databases. *IEEE Data Engineering Bulletin*. 16(3):40-45.

Egenhofer, M. (1997) Query Processing in Spatial-Query-by-Sketch. *Journal of Visual Languages and Computing*. 8(4):403-424.

Egenhofer, M. (2005) Spherical Topological Relations. *Journal on Data Semantics* III.25-49.

Egenhofer, M. and Al-Taha, K. (1992) Reasoning About Gradual Changes of Topological Relationships. in: Frank, A., Campari, I. and Formentini, U. (eds.) *Theories and Methods of Spatio-temporal Reasoning in Geographic Space*, Pisa, Italy, Lecture Notes in Computer Science, 639, 196-219.

Egenhofer, M. and Franzosa, R. (1991) Point-Set Topological Spatial Relations. *International Journal of Geographical Information Systems*. 5(2):161-174.

Egenhofer, M. and R. Franzosa, R. (1995) On the Equivalence of Topological Relations. *International Journal of Geographical Information Systems*. 9(2): 133-152, 1995.

Egenhofer, M. and Herring, J. (1991) Categorizing Binary Topological Relationships between Regions, Lines and Points in Geographic Databases. in: Egenhofer, M., Herring, J., Smith, T. and Park, K. (eds.) *A Framework for the Definitions of Topological Relationships and an Algebraic Approach to Spatial Reasoning within This Framework*, NCGIA Technical Reports 91-7. National Center for Geographic Information and Analysis: Santa Barbara, CA.

Egenhofer, M. and Mark, D. (1995) Modeling Conceptual Neighborhoods of Topological Line-Region Relations. *International Journal of Geographical Information Systems*. 9(5):555-565.

Egenhofer, M., Sharma, J. and Mark D. (1993) A Critical Comparison of the 4-Intersection and the 9-Intersection Models for Spatial Relations: Formal Analysis. in: McMaster, R. and Armstrong, M. (eds.) *Autocarto 11*, Minneapolis, MD, 63-71.

Erwig, M. and Schneider, M. (1999) Developments in Spatio-Temporal Query Languages. in: *10th International Workshop on Database & Expert Systems Application*, Florence, Italy, IEEE Computer Society, 441-449.

Freksa, C. (1992a) Temporal Reasoning Based on Semi-Intervals. *Artificial Intelligence*. 54:199-227.

Freksa, C. (1992b) Using Orientation Information for Qualitative Spatial Reasoning. in: Frank, A., Campari, I. and Formentini, U. (eds.) *The International Conference GIS—From Space to Territory: Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, New York, NY, Lecture Notes in Computer Science, 639, 162-178.

Goyal, R. and Egenhofer, M. (2000) Consistent Queries over Cardinal Directions across Different Levels of Detail. in: Tjoa, A.M., Wagner, R., and Al-Zobaidie, A. (eds.), *11th International Workshop on Database and Expert Systems Applications*, Greenwich, U.K. 876-880.

Hadzilacos, T. and Tryfona, N. (1992) Model for Expressing Topological Integrity Constraints in Geographic Databases. in: Frank, A., Campari, I. and Formentini, U. (eds.) *The International Conference GIS—From Space to Territory: Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, New York, NY, Lecture Notes in Computer Science, 639, 252-268.

Herring, J. (1991) The Mathematical Modeling of Spatial and Non-Spatial Information in Geographic Information Systems. in: Mark, D. and Frank A. (eds.) *Cognitive and Linguistic Aspects of Geographic Space*, Kluwer: Dordrecht, Netherlands, 313-350.

Hoel, E. and Samet, H. (1991) Efficient Processing of Spatial Queries in Line Segment Databases. in: Gunther, O. and Schek, H.-J. (eds.) *Advances in Spatial Databases—2nd Symposium, SSD'91*, New York, NY, Lecture Notes in Computer Science, 525, 237-256.

Hornsby, K., Egenhofer, M. and Hayes, P. (1999) Modeling Cyclic Change. in: Chen, P., Embley, D., Kouloumdjian, J., Liddle, S. and Roddick, J. (eds.) *Advances in Conceptual Modeling*, Paris, France, Lecture Notes in Computer Science, 1227, 98-109.

Krieg-Brückner, B. and Shi, H. (2005) Orientation Calculi + Route Graphs: Towards Semantic Representations of Route Descriptions. *Dagstuhl Seminar on Spatial Cognition: Specialization and Integration*, http://www.dagstuhl.de/files/Materials/05/05491/ 05491. KriegBruecknerBernd.Slides.pdf

Kurata, Y. and Egenhofer, M. (2005a) Semantics of Simple Arrow Diagrams. in: Barkowsky, T., Freksa, C., Hegarty, M. and Lowe, R. (eds.) *AAAI Spring Symposium on Reasoning with Mental and External Diagram: Computational Modeling and Spatial Assistance*, Stanford, CA, 101-104.

Kurata, Y. and Egenhofer, M. (2005b) Structure and Semantics of Arrow Diagrams. in: Cohn, A. and Mark, D. (eds.) *COSIT '05*, Ellicottville, NY, Lecture Notes in Computer Science, 3693, 232-250.

Kurata, Y. and Egenhofer, M. (2006) Topological Relations of Arrow Symbols in Complex Diagrams. in: D. Barker-Plummer, R. Cox, and N. Swoboda (eds.), *Diagrams 2006: Fourth International Conference on the Theory and Application of Diagrams*, Stanford, CA, Lecture Notes in Artificial Intelligence, 4045, 112-126.

Mark, D. and Egenhofer, M. (1994) Modeling Spatial Relations between Lines and Regions: Combining Formal Methods and Human Subjects Testing, *Cartography and Geographic Information Systems*. 21(4):195-212.

Moratz, R., Renz, J. and Wolter, D. (2000) Qualitative Spatial Reasoning About Line Segments. in: Horn, W. (ed.) *14th European Conference on Artificial Intelligence*, Berlin, Germany, 234-238.

Nedas, K., Egenhofer, M. and Wilmsen, D. (in press) Metric Details of Topological Line-Line Relations. *International Journal of Geographical Information Science*.

Papadias, D., Theodoridis, Y., Sellis, T., and Egenhofer, M. (1995) Topological Relations in the World of Minimum Bounding Rectangles: A Study with R-Trees. In: Carey, M. and Schneider, D. (eds.) *SIGMOD RECORD* 24 (2): 92-103.

Pullar, D. and Egenhofer, M. (1988) Towards Formal Definitions of Topological Relations among Spatial Objects. *Third International Symposium on Spatial Data Handling*, Sydney, Australia, 225-241.

Renz, J. (2001) A Spatial Odyssey of the Interval Algebra: 1. Directed Intervals. in: Nebel, B. (ed.) *International Joint Conference on Artificial Intelligence 2001*, Seattle, WA, 51-56.

Reis, R., Egenhofer, M., and Mattos, J. (2005) Grafos de Vinzihanças Conceptuais para Relações Topológicas entre Linhas em R^2. *Fourth National Conference for Cartography and Geodesy*, Lisbon, Portugal, http://www.igeo.pt/Igeo/portugues/Novidades_eventos/ eventos/Lisboa/ivcncg_reis.pdf.

Schlieder, C. (1995) Reasoning About Ordering. in: Frank, A. and Kuhn, W. (eds.) *COSIT '95*, Lecture Notes in Computer Science, 988, 341-349.

Shariff, A., Egenhofer, M. and Mark, D. (1998) Natural-Language Spatial Relations between Linear and Areal Objects: The Topology and Metric of English-Language Terms. *International Journal of Geographical Information Science*. 12(3):215-246.

A GIS-Based Approach for Urban Multi-criteria Quasi Optimized Route Guidance by Considering Unspecified Site Satisfaction

Parham Pahlavani, Farhad Samadzadegan, and Mahmood Reza Delavar

Dept of Geomatics Engineering, Disaster Management Center of Excellence
University of Tehran, Tehran, Iran
`pahlavani@mas-rd-co.com, {samadz, mdelavar}@ut.ac.ir`

Abstract. Urban multi-criteria optimized route guidance by considering unspecified site satisfaction, an extended type of urban multi-objective optimized route selection, called as both NP-Hard problems and one of the branches of multi-criteria shortest path problems (MSPP). It is not only suggests a route based on route guidance principles and optimized due to routing criteria but also passes through all unspecified site(s) such as gas stations, banks determined by drivers. By proposing a novel approach on the bases of route guidance navigation system principles, virus theory (viral infection and local/site infection) and by GIS and GA utilization, this paper is come up to rate of search improvement in urban multi-criteria optimized route guidance by considering unspecified site satisfaction on real network with multiple dependent criteria. Tests of route selection for a part of north-west of Tehran traffic network are conducted and the results show the efficiency of the algorithm and support our analyses.

Keywords: GIS, Network analysis, Multi-criteria shortest path problem, Multi objective Genetic algorithm.

1 Introduction

Decisions are often evaluated based on quality of the processes behind. Decision making itself, however, is broadly defined to include any choice or selection of alternative course of action, and is therefore of importance in many fields in both the social and natural sciences, including geospatial information sciences. It is in this context that geospatial information systems (GIS) and spatial decision support systems (SDSS) increasingly are being used to generate alternatives to aid decision-makers in their deliberations. Often representation of more than one criterion or objective in models of many real-world spatial planning and management problems is desired and give rise to GIS-based multi-criteria decision making.

Network analysis in GIS provides strong decision support for users in searching shortest path, finding the nearest facility and determining the service area. Shortest path selection is known as one of the most important functions in GIS network analysis. It is classified to two types; one is single-criterion shortest path problems and the other is multi-criteria shortest path problems (MSPP). The current GIS usually

confront the lack of powerful tools in spatial analysis for MSPP [5], so the suitable solution for MSPP is one of the current major needs of GIS.

Urban multi-criteria optimized routing problems are called as both NP-Hard problems and one of the branches of MSPP [8]. Algorithmic and approximation schemes are available to solve MSPP; unless they meet some problems such as the complexity of these approaches often prohibits their implementation in large urban route networks. They are limited to a few numbers of criteria, each criterion is independent with no weight of importance and it is needed to select one solution from Pareto-optimal solutions. These categories of optimization problems cannot be solved practically and therefore an approximation of the general optimum has to be considered [28].

Urban multi-criteria optimized route guidance by considering unspecified site(s) satisfaction, an extended type of urban multi-objective optimized route selection, is not only suggests a route based on route guidance principles and optimized due to routing criteria but also passes through all unspecified sites such as gas stations, banks determined by drivers.

We have already outlined a GIS-based novel approach for using a genetic algorithm for urban multi-objective optimized route selection in static environment [20]. In this paper we describe a method that extends the previous novel approach in order to include driver's unspecified sites.

In section 2, related works are exhibited. In section 3, it is represented basic strategies for urban multi-criteria optimized route guidance by considering unspecified site satisfaction, in section 4; it will be represented a proposed method. Experiments are exhibited in section 5 and finally it is introduced the proper results in section 6.

2   Related Works

The work of Fernandez et al. [7], Mandow and Perez de la Cruz [15] on robot navigation uses lexicographical goal satisfaction to solve robot navigation problems for three independent criteria. Skriver and Anderson [25] also employ a label correcting approach to a bicriterion SPP. Their algorithm stores only label-sets and Pareto paths are recovered at the termination of the algorithm but are appeared to be limited to relatively small networks. In Martins et al. [17] a tree-based algorithm for ranking optimal paths in MSPP is outlined for combinations of two criteria. In Martins and dos Santos [16] a labeling algorithm for MSPP is outlined. This labeling search tree approach outlined works well in theory but in practice the memory costs of this approach are prohibitive to its implementation.

Hallam et al. [11] outlines an approximation algorithm for MSPP supplying heuristic information (like that for A*) to the algorithm. Pareto paths are selected based on their selection function value, which contains heuristic, and constraint information. Nepal and Park [19] combine heuristic labeling and exhaustive search algorithms. In the work presented by Roy et al. [23] the authors provide a clear and very useful application of GA to a problem that is multi-criteria by its definition. The primary goal of Quality of Service routing is to efficiently allocate wireless resources to satisfy these quality of service requirements.

Many examples appear in the literature on application of evolutionary computation (EC) approaches to problems in GIS. We do not explicitly consider the advantages and disadvantages of any particular approach but discuss these applications as a strong indication of the suitability of EC to multi-criteria problems in GIS. The work of Bennett et al. [3] uses an evolutionary approach to help cartographers create optimal shapes for the geographical and statistical characteristics of choropleth maps. For resolution of conflicts between the locations of objects resulting from scale reduction on maps Wilson et al. [27] use GA to search for optimal generalization. GAs are also used by Chemin et al. [6] to estimate pixel-based water/plan parameters in the study of crop productivity indicators from remote sensing data. The multi-criteria nature of environmental land-use planning requires the generation of many alternative candidate solutions that optimize criteria such as spatial allocation, operations costs, and environmental impacts.

In the field of utility multi-objective site selection, Li and Yeh [14] used GA in a GIS environment. They believed that the obtained results of this research show the capability of GA among the other basic conventional approaches for this specific problem. Huang et al. [12] have recommended further researches in GA and GIS utilization in transportation environments.

3   Strategy

First it is proposed the reasons of GA utilization in bellow and then it is introduced the main strategies for the problem in hand.

3.1   Why GA?

Genetic algorithms are a computational model simulating the process of genetic selection and natural elimination in biologic evolution. During the last two decades, genetic algorithms have received considerable attention regarding their potential as a novel approach to multi-objective optimization problems, known as evolutionary or genetic multi-objective optimization problems [28]. The inherent characteristics of genetic algorithms demonstrate why genetic search may be well suited to multiple-objective optimization problems. Genetic algorithms do not have many mathematical requirements and can handle all types of objective functions and constraints. Because of their evolutionary nature, genetic algorithms can be used to search for solutions without regard to the specific inner workings of the problem. Therefore, it is hoped that many more problems that are complex can be solved using genetic algorithms than using conventional methods.

As a high efficient search strategy for global optimization, genetic algorithms demonstrate favorable performance on solving the combinatorial optimization problems. With comparing to traditional search algorithms, genetic algorithms are able to automatically acquire and accumulate the necessary knowledge about the search space during its search process and self-adaptively control the entire search process through random optimization technique [10].

Because genetic algorithms, as a kind of metaheuristics, provide us with great flexibility to incorporate conventional methods into the main framework, we can

exploit the advantages of both genetic algorithms and conventional methods to establish much more efficient implementations to problems. Therefore, it is more likely to obtain the global optimal solution without encountering the trouble of combinatorial explosion caused by disregarding the inherent knowledge within the search space. It has been used to solve combinatorial optimization problems and non-linear problems with complicated constraints or non-differentiable objective functions. It necessitates the application of genetic algorithm into GIS route finding algorithms by considering unspecified site(s) satisfaction.

As a matter of fact, GA is a general search method [10]. It uses analogs of genetic operators on a population of states in a search space to find those states that have high fitness values. In optimization problems, genetic operators and coding methods are designed in advance so that the individuals may satisfy the constraints. In contrast, the objective of constraint satisfaction problems is to find an individual that satisfied constraints as the fitness value [13]. Search in usual GAs, based on neo-Darwinian evolutionary theory is conducted by crossover and mutation. Crossover is generally considered a robust search means. Offspring may inherit partial solutions without conflict from their parents; however, no information to decide which genes are partial solutions is available. From a search-strategic point of view this means that variables are randomly selected and then values which will be assigned to them are also randomly selected from genes contained within a population. Therefore search by crossover in GAs can not be considered to be efficient. Furthermore, mutation is a random search method itself. When we think about efficiency of global search that is the most important characteristic of GAs; we must admit that the rate of search is low, because GAs stresses random search rather than directional search. It can be considered that the above problem is directly inherited from problems of the neo-Darwinian evolutionary theory.

3.2   Main Strategies

Due to the advantages and disadvantages mentioned in the previous section, we plan to improve the rate of search by giving direction to evolution that is search using crossover and virus theory of evolution. Nakahara et al. [18] reported the mechanism of organic evolution by viral infection and its superiority to mutation. Anderson [1] pointed out the evolutionary significance of viral infection.

For this reason, domain specific knowledge, as partial solutions of problems, is usually introduced into an initial population. Partial solutions are considered to be viruses, and a population of viruses is created as well as a population of individuals. Our solution using the GA is based on the following main strategies:

- Any segment of an arterial road is regarded as a main virus, and a part of a route with a given site is regarded as a site virus. We generate a population of viruses (site viruses as well as main viruses) from each traffic zone to the others in addition to a population of routes (preprocessing step).
- Only routes that include viruses are generated as the initial population. If two of the same routes are produced in the population, one is removed.
- Minimal generation gap (MGG) model is used as an alternation of generations. Mutation does not occur in the proposed method.

- Infection by main and site viruses determine the optimal combination of viruses.
- Local/site infection is used to perform local exploitation around chromosome.
- The constraints on this particular problem can be grouped into the following two categories:
    - o Fixed constraints are always enforced and irrespective of the composition of the current solution. For example, passing through all unspecified site(s), selecting arterial and/or wide roads to decrease the number of turns and so on.
    - o Dynamic constraints are enforced based on the composition of the current solution. For example, selecting the roads where the traffic restrictions are not imposed and selecting the roads not congested with traffic and so on.
- Penalties for violations are defined for each constraint. We regard the route with the lowest number of penalties as the best one for drivers.

4   Proposed Method

In this section, first it is introduced the specific approach used in measuring each route criteria, then it is proposed the quality metric for evaluating the best recommended compromise solution and finally, it is represented the designed algorithm for the optimized route guidance by considering unspecified site.

4.1   The Specific Approach Used in Measuring Each Route Criterion

A route cost is a characteristic of a route, which is used by a driver as an assessment criterion in route selection. As there is a wide variety of costs describing each route, it is convenient to classify them. The primary costs are defined as travel distance, travel time, congestion, degree of difficulty, toll and scenery [22]. These six costs are called primary costs because they are primary in the sense that they are the important costs and are widely-used by most drivers in the assessment of a route. This classification will ease the subsequent development of an intelligent route selection system.

Given a set of origin-destination pair, there could be many possible routes for a driver. Each of these candidate routes has different values in their primary costs. One route may have a high value in one cost (e.g. shortest distance) but a low value in another cost (e.g. heavy congested route).

A detailed discussion of each of the primary costs is given bellow. Each cost is designed to have a score between zero and one. Because toll and scenery criteria are unimportant in this research, they were omitted in modeling.

4.1.1   Travel Distance and Travel Time
Let x_1^k be used to describe this cost. Let route j has the shortest distance among the set of route candidates ever of all populations. Then x_1 will be given a score of one. In other words, for this cost, a score of 1 is used to denote that this route has the shortest travel distance among the particular set of route candidates. For other candidate routes, the score of this cost is defined as follow:

$$x_1^k = length\_ratio_k = \frac{\max(length\_value) - length\_value_k}{\max(length\_value) - \min(length\_value)} \tag{1}$$

min(length_value) = best length value in all populations
max(length_value) = worst length value in all populations

"Travel time" cost can be developed in a way similar to travel distance:

$$x_2^k = time\_ratio_k = \frac{\max(time\_value) - time\_value_k}{\max(time\_value) - \min(time\_value)} \tag{2}$$

min(time_value) = best time value in all populations
max(time_value) = worst time value in all populations

4.1.2 Congestion

Congestion can be said to be recurring when it is triggered by a daily or periodic event. It can also be nonrecurring and caused by an accident on a street. Anyway, congestion is a situation which most drivers would like to avoid.

The work of Pahlavani and Samadzadegan [20] on a GIS-based decision supporting system for dynamic congestion modeling in real urban traffic network uses a discrete time dynamic network assignment procedure that predicts network flow at detailed temporal resolutions. Based on the predicted dynamic traffic flows, speed of each road can be reached by using Bureau of Public Roads functions defined for each type of road in urban traffic network in Iran. Inter vehicle distance is also obtained from inductive loop detectors. By considering and modeling speed of each road and inter-vehicle distance in particular period of time Pahlavani and Samadzadegan [20] have designed and implemented a fuzzy model for measuring levels of congestion for different types of urban roads in a range of [0,1]. So if c_p (congestion value for each segment of road) will be in a range of [0,1] then a score of 0 corresponds to on-road segment disclosure whereas a score of 1.0 corresponds to most severe level in on-road segment closure.

4.1.3 Degree of Difficulty

This cost can be computed as a function of the type and nature of the road such as the narrowness, winding, slope, number of traffic lights, and number of stop signs. A simpler way would be to assign different values for different types of road. For example, the following table can be used as a guideline for determining the degree of difficulty of the route [21].

Table 1. Guideline for determining degree of difficulty

Road type	Penalty for Degree of difficulty
Exp. Way with a negative slope	0
Exp. Way without a slope	0.1
Exp. Way with a positive slope	0.2
Major Arterial (outside the central district)	0.4
Major Arterial (inside the central district)	0.5
Minor Arterial	0.6
Collector-Feeder	0.8
Local street	1

4.1.4 Overall Score of Congestion and Degree of Difficulty for Search Candidate Route

For each candidate route in the proposed genetic population, the travel distance and travel time are supposed to be known and a score in the range [0,1] can be calculated for each of these costs. However, different road segments for each candidate route to be included in the proposed genetic population would have different cost scores for congestion and degree of difficulty. In order to calculate the complete set of primary costs of the route candidate, a method is needed to combine the cost score of different road segments into an overall score for congestion and degree of difficulty. This is based on the assumption that the traffic spatial database already has the score of the primary costs of each section. A method is developed for this calculation, which is described below [21]:

Let n be the number of road segments of the route candidate. Let d_p and c_p be the distance and congestion cost scores of each road segment, respectively (p = 1 to n). Let w_p be a weight of each road segment, which is defined as:

$$w_p = \frac{d_p}{\sum_{p=1}^{n} d_p} \tag{3}$$

Then, the overall score c for candidate route is calculated as:

$$congestion\_value_k = \sum_{p=1}^{n} w_p * c_p \tag{4}$$

Note that $\sum_{p=1}^{n} w_p = 1$ and $0 \le c_p \le 1$. The final overall score $congestion\_value_k$ is also in the range [0,1].

Let x_3^k be used to describe this cost then the score of this cost is defined as follow:

$$x_3^k = congestion\_ratio_k = \frac{\max(congestion\_value) - congestion\_value_k}{\max(congestion\_value) - \min(congestion\_value)} \tag{5}$$

min(congestion_value) = best congestion value in all populations
max(congestion_value) = worst congestion value in all populations

A similar method can be used for computing the overall cost score for degree of difficulty (x_4^k).

4.1.5 Types of Primary Costs

It is perceived that some costs of a candidate route are dynamic while some can be considered as static. The dynamic ones are travel time and degree of congestion. The static ones are travel distance. The degree of difficulty can be either static or dynamic depending on the method of determining the value of that cost.

4.2 Quality Metric

Many quality indicators or metrics require knowledge of ideal solution(s) for the problem in hand in order to measure how close the current approximation is to the

optimal solution(s). To generate ideal solution for the problem in hand, a pair of nodes (s,t) are chosen. Then the Dijkstra's algorithm in order to optimize separately on each of the criterion is implemented. So the ideal solution (route) will be its descriptive vector calling zi* which is obtained through independent implementation of Dijkstra algorithm for each criterion:

$$zi*=(min(length\_value),min(time\_value),min(congestion\_value),min(DoD\_value)) \quad (6)$$

If the descriptive vector of the best compromise solution (route) is called z_i,the regret of the best compromise solution (z_i) is calculated instead of the ideal solution (z_i*) according to a scale range independent weighted Lp-norm [9].

$$z_i = (x_1^k, x_2^k, x_3^k, x_4^k) \qquad (7)$$

$$r(z; p, w) = (\sum_{i=1}^{q} w_i^p \left[\frac{\left| z_i^* - z_i \right|}{z_i^*} \right]^p)^{1/p} \qquad (8)$$

weighted vector $w = (w_1, w_2, ..., w_q)$ which maintains the importance of each criterion z_i is introduced by driver.

The parameter p ($p \geq 1$) is used to reflect the emphasis of decision makers. It is used $p = 1$ because the sum of regrets of all objectives is important in this research.

$$r(z; 1, w) = \sum_{i=1}^{4} w_i \left[\frac{\left| z_i^* - z_i \right|}{z_i^*} \right] \qquad (9)$$

In this regard the obtained distance between z_i and z_i* from the equation 9 indicates how close is z_i from its ideal z_i*. It means if $z_i = z_i*$ then the distance becomes zero.

4.3 Algorithm

Figure 1 shows the general steps of the algorithm. Details are given in the following sections.

```
1. input map and map database
2. input origin and destination
3. initialize a population of viruses
4. initialize a population of individuals (routes). Set
Generation= 1
5. for generation = 1 to Number of Generation (repeat
until meeting deadline){
6.     calculate fitness values of individuals (Eq. 10)
7.     select two individuals at random
8.     n-point crossover (which n is the number of common
intersections between individuals)
9.     MGG
10.    viral infection
11.    local/site infection}
```

Fig. 1. Algorithm of the proposed method

4.3.1 Coding and Population Initialization

Variable-length chromosomes (routes) and their genes (intersections) have been used for encoding the problem. A chromosome of the proposed GA consists of sequences of positive integers that represent the IDs of nodes through which a routing path passes. In the other hand, we regard each route from an origin to a destination as an individual for the GA and express it as a sequence of intersections.

In general, there are two issues to be considered for population initialization of GA: the initial population size and the procedure to initialize population [10]. It was felt that the population size needed to increase exponentially with the complexity of the problem (i.e., the length of the chromosome) in order to generate good solutions. Recent studies have shown, however, that satisfactory results can be obtained with a much smaller population size. To summarize, a large population is quite useful, however, it demands excessive costs in terms of both cost and time [10]. As would be expected, deciding adequate population size is crucial for efficiency.

There are two ways to generate the initial population: heuristic initialization and random initialization [10].

To generate the initial population, a main virus as well as a site virus are randomly selected from the population of viruses and are controlled for dynamic constraints. If the viruses satisfy the dynamic constraints then, the routes from the origin to the main virus/site virus and the route from the main virus/site virus to site virus/main virus and the route form main virus/site virus to the destination, are randomly generated by using a modified Dijkstra algorithm based on d-Heap's structure with d=2 is capable for memory cost management in large networks [2].

In which by using GIS capabilities in huge spatial data base management, each link existing in network are weighting randomly through uniform distribution before a modified Dijkstra algorithm based on d-Heap's structure with d=2 is run in each route generation [21].

Finally, the route that combines these three routes with the viruses becomes an individual. This procedure is repeated up to initial population size needed for GA.

4.3.2 Fitness

The sum of weighted global ratios (SWGR) method [4] is used to construct the fitness function. In the multi-objective optimization, the SWGR method is used to obtain a compromise solution. In the genetic algorithms, the SWGR method is used primarily to adjust genetic search toward the Pareto frontier. Weights are readjusted adaptively along with the evolutionary process. Therefore, a good weighting vector is not a mandatory precondition to make genetic algorithms work. In addition, the drawbacks exhibited in the multi-objective optimization can be compensated by the powers of population-based search and evolutionary search.

The fitness of a route with the purpose of minimization case is evaluated by using the length of the route (x_1^k), the time required for a car to travel along it (x_2^k), overall congestion of the route (x_3^k) and the overall degree of difficulty of the route (x_4^k) in a range of [0,1]. So if the fitness value for each route will be in a range of [0,1] then a score of 0 corresponds to the worst route fitness value whereas a score of 1.0 corresponds to the best route fitness value. The fitness function of a route (k) can be expressed as:

$$f(R_k) = w_1 * x_1^k + w_2 * x_2^k + w_3 * x_3^k + w_4 * x_4^k \tag{10}$$

$$\sum_{i=1}^{4} w_i = 1 \tag{11}$$

where w_i are constants chosen by the driver due to their importance for him/her and these constants have a range from zero to one.

4.3.3 Selection and Crossover

The selection (reproduction) operator is intended to improve the average quality of the population by giving the high-quality chromosomes a better chance to get copied into the next generation [10].

Crossover examines the current solutions in order to find better ones [10]. Physically, crossover in the routing problems plays the role of exchanging each partial route of the two chosen chromosomes in such a manner that the offspring produced by the crossover will only be one route. This dictates selection of n-point crossover [26] which n is the number of common intersections between routes as a good candidate scheme for the proposed GA. However, the mechanism of the crossover is not the same as the conventional n-point crossover in which there is no need for common intersections to be in the same positions.

In the proposed scheme, we use the minimal generation gap (MGG) model [24] for alternation of generations. In this model, two routes are replaced by crossover with each new generation and so two evaluations of fitness are required between generations. Mutation is not used in this method (Figure 2).
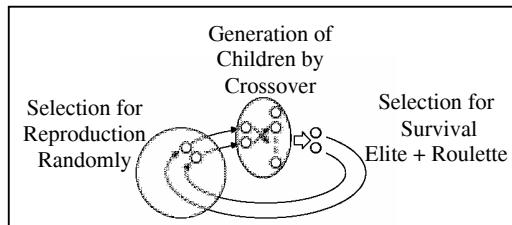

Fig. 2. Minimal Generation Gap (MGG) model (Satoh et al., 1996)

Here are the directions for the process:

- Take two individuals at random from the population for use as parents are called $R_{S \to D}^{1}$ and $R_{S \to D}^{2}$.
- Apply a crossover to the parents to produce offspring, where the crossover site is at one intersection or more, which they have in common. If there is no such common intersection(s), then choose an intersection of one route at random (s_1) and find the intersection nearest to it (s_2). The Modified Dijkstra algorithm can be used to find the shortest path based on the length criterion between these two intersections , so we have:

Table 2. A particular crossover parents and children

No.	Parents	Children		
1	$R_{S \to D}{}^1$	$R_{S \to s_1}{}^1 \to R_{s_1 \to s_2} \to R_{s_2 \to D}{}^2$		
2	$R_{S \to D}{}^2$	$R_{S \to s_2}{}^2 \to R_{s_2 \to s_1} \to R_{s_1 \to D}{}^1$		

- From the parents and their children, we select the best individual (elite route) and the one at random using the roulette wheel technique. By choosing them, the best individual is replaced to the nearest original parent.

In this model, original parents are two individuals, and replacing individuals are also two. Then we leave the elite individual for progress in solving a problem, and leave a random individual for maintaining diversity of population.

4.3.4 Viral Infection

4.3.4.1 Main Infection. Main infection substitutes the intersections contained by a virus for the intersections on the route, if the route has two intersections in common with the virus. In Figure 3, for example, if the route before infection is $R_{old}(k)=$ (…,109,110,111,112,113,114,…), then the route after the infection will be $R_{new}(k) =$ (…,109,110,106,107,113,114,…), where 110 and 113 are common intersections. The new route then replaced in old route.
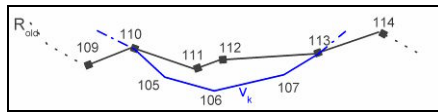

Fig. 3. An example of Main infection

4.3.4.2 Bridge Infection. Figure 4 shows an example of bridge infection. If the virus v_k crossed both route R_1 and R_2, two new routes $R_3 =$ (…,109,110,111,112,98,107, 108,…) and $R_4 =$ (…,103,104,105,106,107,98,112,113,114,…) are generated. The elite route is chosen from R_1, R_2, R_3, R_4 and then another route is also chosen from the rest of the routes by roulette wheel selection. These two routes are replaced with R_1 and R_2.
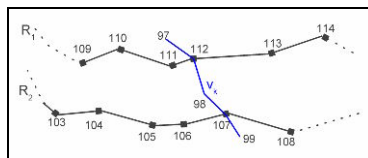

Fig. 4. An example of Bridge infection

4.3.4.3   Local/Site Infection. Here are the directions for the process:

- Let $N = \{N_1, N_2, ..., N_m\}$ be a set of intersections selected at regular intervals from the intersections of R_k. We call the element of N virtual nodes. Choose a sample node N_i at random.

- Let $v_{k'}$ be the virus nearest to N_i within a radius of r with 'a' and 'b' as its initial and final intersection. Let I_t within R_k be the intersection nearest to $v_{k'}$. Let N_{before} and N_{after} within N_i be the intersections before and after I_t respectively, and I_{t-1} and I_{t+1} within R_k are the nearest intersections to N_{before} and N_{after}. Generate three routes from $v_{k'}$ to I_{t-1}, I_t and I_{t+1} using the modified Dijkstra algorithm based on the length criterion algorithm (Figure 5).
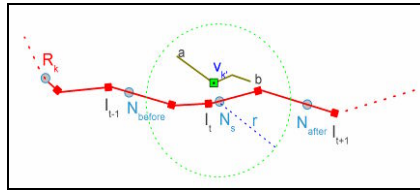

Fig. 5. Illustration of site infection

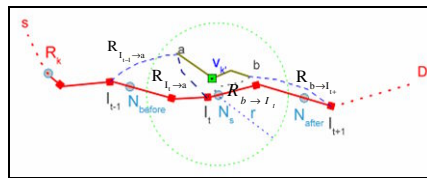- Select the elite route from among the following four routes(Figure 6):

Fig. 6. The routes achieved form site infection

1) R_k

2) $R_{s \to I_{t-1}} \to R_{I_{t-1} \to a} \to v_{k'} \to R_{b \to I_{t+1}} \to R_{I_{t+1} \to D}$

3) $R_{s \to I_t} \to R_{I_t \to a} \to v_{k'} \to R_{b \to I_{t+1}} \to R_{I_{t+1} \to D}$

4) $R_{s \to I_{t-1}} \to R_{I_{t-1} \to a} \to v_{k'} \to R_{b \to I_t} \to R_{I_t \to D}$

5   Experiments

The proposed method due to GA in GIS was implemented by ARCGIS utilization and customization. ArcGIS has a feature in architectural design, which enables it to be developed by COM programming in any visual environment.

To evaluate the performance of outlined method, we performed experiments using actual road maps of a part of the north-west of Tehran network at a scale of 1:2000. Table 3 shows the characteristics of the maps used in the following experiments, where the nodes correspond to the intersections. In addition, each result given bellow was performed on AMD Athlon(tm) XP 1600+ (1.40 GHz).

Table 3. Characteristics of the map used

	Number of main viruses	Number of site viruses	Number of links	Number of nodes
Map	160	7	5121	4389

For the purpose of novel approach capability proof, it has been discussed the achieving results in each step of approach improvement from Minimal Generation Gap (MGG) model to proposed novel approach in a case of three experiments. As in first experiment (Figure 1 without lines 10 and 11) we have done GA design and implementation by MGG utilization for quasi multi-criteria optimized route guidance by considering unspecified site satisfaction. In second experiment (Figure 1 without line 11), virus theory and viral infection improved the first experiment and finally in third experiment (Figure 1) by adding local/site infection we have come up to second experiment improvement.

Figure 7 shows the average fitness of thirty independent runs for different population size, when the crossover rate is 0.1. In this case, the population size is the main reason of evolution. As it is seen the figure 7 by increasing population size over one hundred, there is no critical change in average fitness. The obtained results of three experiments are shown in table 4. The obtained lower distance between z_i and z_i^* in experiment 3 shown in table 4 and higher average fitness over 30 independent runs shown in table 5 in comparison to two other experiments indicates the better capability of implemented approach in experiment 3 over experiment 1 and 2.
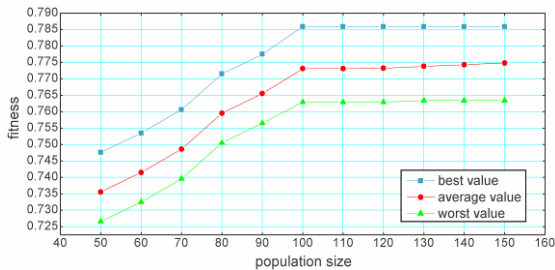

Fig. 7. The average fitness over thirty independent runs

Table 4. The obtained results of three experiments

Importance weights				Pop_ size	The No. of Runs	Ideal solution z^*_i: (L,T,C,D)	Best compromise solution z_i:(L,T,C,D)	Dist. z_i to z^*_i
w_1	w_2	w_3	w_4					
GA design and implementation by MGG utilization (Experiment #1)								
0.25	0.25	0.25	0.25	100	30	(6839.19,13.98,0,0.1363)	(7207.81,17.35,0.0122,0.3389)	0.127
0.2	0.4	0.2	0.2	100	30	(6839.19,13.98,0,0.1363)	(7408.78,16.98,0.1810,0.3221)	0.176
0.1	0.7	0.1	0.1	100	30	(6839.19,13.98,0,0.1363)	(7773.86,15.02,0.1060,0.3802)	0.101
0	1	0	0	100	30	(6839.19,13.98,0,0.1363)	(7445.13,14.91,0.0118,0.3513)	0.066
GA design and implementation by MGG and viral infection utilization (Experiment #2)								
0.25	0.25	0.25	0.25	100	30	(6839.19,13.98,0,0.1363)	(7258.11,17.15,0.0121,0.3330)	0.124
0.2	0.4	0.2	0.2	100	30	(6839.19,13.98,0,0.1363)	(7603.41,15.36,0.0116,0.3570)	0.108
0.1	0.7	0.1	0.1	100	30	(6839.19,13.98,0,0.1363)	(7444.85,14.81,0.0119,0.3418)	0.072
0	1	0	0	100	30	(6839.19,13.98,0,0.1363)	(7823.85,14.55,0.0113,0.3885)	0.041
GA design and implementation by MGG , viral infection and site infection utilization (Experiment #3)								
0.25	0.25	0.25	0.25	100	30	(6839.19,13.98,0,0.1363)	(7225.55,16.99,0.0122,0.3022)	0.064
0.2	0.4	0.2	0.2	100	30	(6839.19,13.98,0,0.1363)	(7505.46,14.21,0,0.3615)	0.071
0.1	0.7	0.1	0.1	100	30	(6839.19,13.98,0,0.1363)	(7343.44,14.15,0.0090,0.4065)	0.044
0	1	0	0	100	30	(6839.19,13.98,0,0.1363)	(7039.92,14.08,0.0304,0.3641)	0.007

w_1, w_2, w_3 , w_4:The importance of length, time, congestion and degree of difficulty respectively . **Pop_size:** population size. **(L,T,C,D):** (Length(meter), Time(minute), Congestion (percentage of route length, Degree of   difficulty (percentage of route length). **Dist z_i ti z_i^*:** the distance between z_i and z_i^* according to Eq. 9. **Note:** Ideal solution is obtained by four independent run of Dijkstra algorithm for each criterion.

Table 5. The best individual average fitness in each three experiments

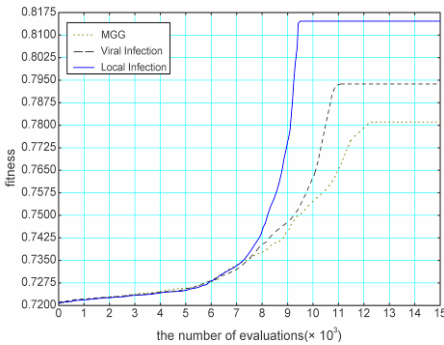| Importance weights				The No. of runs	Fitness Average of best solutions over 30 runs under Pop_size = 50 for Generic GA and Pop_size = 100 for the others		
w_1	w_2	w_3	w_4		MGG (Experiment #2)	Viral infection (Experiment #3)	Local infection (Experiment #4)
0.25	0.25	0.25	0.25	30	0.7813	0.7937	0.8147
0.2	0.4	0.2	0.2	30	0.6433	0.8215	0.8681
0.1	0.7	0.1	0.1	30	0.8243	0.8901	0.9136
0	1	0	0	30	0.9218	0.9519	0.9895

Fig. 8. The best individual average fitness when each criterion weight chosen by the driver is w_1= 0.2, w_2=0.25, w_3= 0.2, w_4=0.2
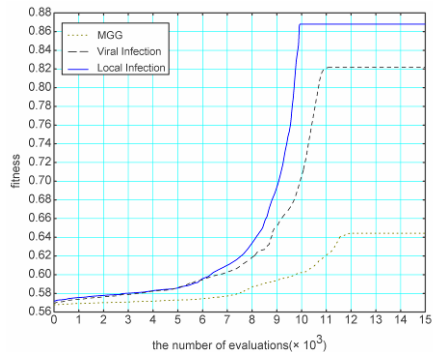
Fig. 9. The best individual average fitness when each criterion weight chosen by the driver is w_1= 0.2, w_2=0.4, w_3= 0.2, w_4=0.2
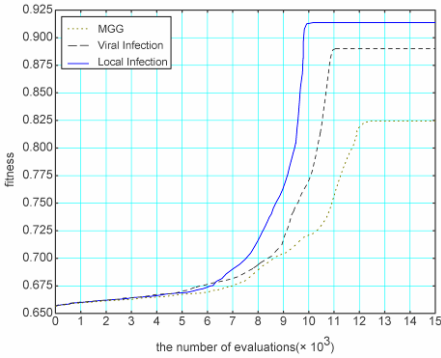
Fig. 10. The best individual average fitness when each criterion weight chosen by the driver is $w_1 = 0.1$, $w_2 = 0.7$, $w_3 = 0.1$, $w_4 = 0.1$
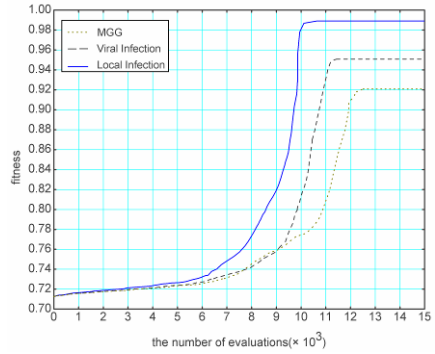

Fig. 11. The best individual average fitness when each criterion weight chosen by the driver is $w_1 = 0$, $w_2 = 1$, $w_3 = 0$, $w_4 = 0$
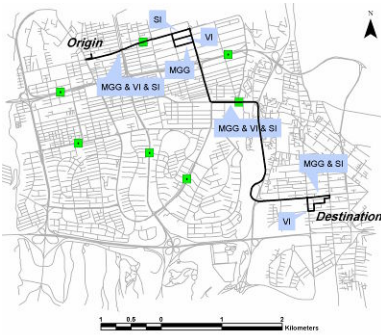

Fig. 12. The best compromise route when each criterion weight chosen by the driver is $w_1 = 0.25$, $w_2 = 0.25$, $w_3 = 0.25$, $w_4 = 0.25$.
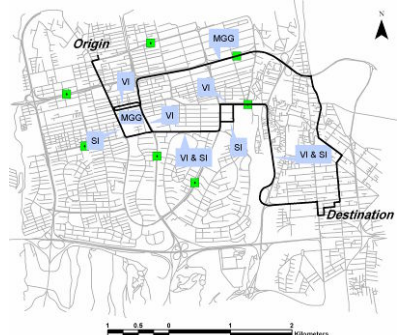
Fig. 13. The best compromise routes when each criterion weight chosen by the driver is $w_1 = 0.2$, $w_2 = 0.4$, $w_3 = 0.2$, $w_4 = 0.2$.
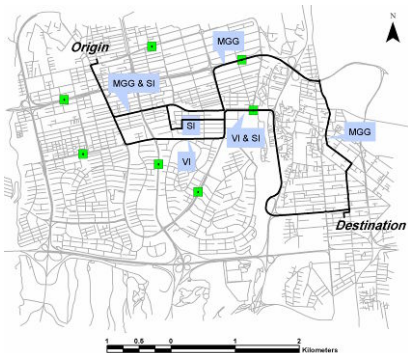

Fig. 14. The best compromise routes when each criterion weight chosen by the driver is $w_1 = 0.1$, $w_2 = 0.7$, $w_3 = 0.1$, $w_4 = 0.1$
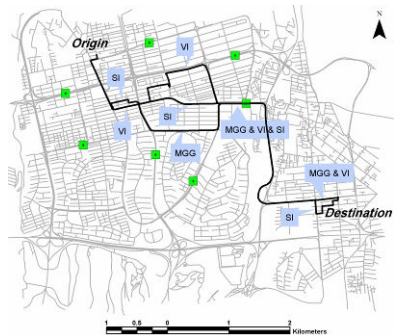

Fig. 15. The best compromise routes when each criterion weight chosen by the driver is $w_1 = 0$, $w_2 = 1$, $w_3 = 0$, $w_4 = 0$

Note: The MGG, VI and SI labels indicate the obtained results from three experiments respectively

In figures 8 to 11, it is represented the best individual average fitness changes over thirty independent runs with the population size equal to one hundred and also with determined weight by driver for each criterion in three experiments moreover the search rate in experiment 3 is higher than the others.

The determined routes labeled MGG, VI and SI, which indicate the obtained results from three experiments respectively are represented in Figures 12 to 15.

6   Conclusion

By proposing an innovative approach on the bases of route guidance navigation system principles, virus theory (viral infection and site infection) and by GIS and GA utilization, this paper is come up to rate of search improvement in urban multi-criteria optimized route guidance by considering unspecified site satisfaction on real network with multiple dependent criteria. For the purpose of novel approach capability proof, it has been discussed the achieving results in each step of approach improvement from Minimal Generation Gap (MGG) model to proposed innovative approach in a case of three experiments. As in first experiment, we have done GA design and implement-tation by MGG utilization for quasi multi-criteria optimized route guidance by considering unspecified site satisfaction. In second experiment, virus theory and viral infection improved the first experiment and finally in third experiment by adding site infection we have come up to second experiment improvement.

Passing through unspecified sites, accepting unlimited criteria, being "range (scale)-independent ranking" method, taking the "importance" of each criterion chosen by driver, utilizing priority knowledge with viral infection and site infection, converging GA to "best compromise" solution and proposing a quality metric for assessing the best compromise solution are the major characteristics of this innovative approach. Further efforts will be made on expanding the algorithm in dynamic route selection.

References

1. Anderson, N.G.: Evolutionary significance of virus infection. The Journal of Nature (1970) 1346-1347
2. Atallah M. J.: Algorithms and Theory of Computation Handbook. CRC press LCC, Washington, USA (1999)
3. Bennett, D., Xiao, N., Armstrong, M.: Using genetic algorithms to create multicriteria class intervals for choropleth maps. Annals of the Association of American Geographers., Vol.93, No.3. (2003) 595–623
4. Bently, P. J.: Generic Evolutionary Design of Solid Objects using a Genetic Algorithm. Ph.D. Thesis, University of Huddersfield, Heddersfield, UK (1996)
5. Chakhar, S., Martel, J.-M.: Enhancing geographical information systems capabilities with multi-criteria evaluation functions. Journal of Geographic Information and Decision Analysis., Vol.7, No.2. (2003) 47–71
6. Chemin, Y., Honda, K., Ines, A.: Genetic algorithm for assimilating remotely sensed evapotranspiration data using a soil-water-atmosphere-plant model. In: FOSS/GRASS User Conference. Bangkok, Thailand. (2004) 88-92
7. Fernandez, J., Gonzalez, J., Mandow, L., Perez-de-la-Cruz, J.: Mobile robot path planning: A multicriteria approach. Engineering applications of Artificial Intelligence (1999) 543–554

8. Gandibleux, X., Beugnies, F., Randriamasy, S.: Martins' algorithm revisited for multi-objective shortest path problems with a maxmin cost function. 4OR Quarterly Journal of the Belgian, French and Italian Operations Research Societies (2004) 1–16

9. Gen, M., Cheng, R.: Genetic Algorithmsd and Engineering Optimization. A Wiley-Interscience Publication, USA (2000)

10. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison Welsey Publishing Company (1989)

11. Hallam, C., Harrison, K., Ward, J.: A multiobjective optimal path algorithm. Digital Signal Processing., Vol.11, No.2 (2001) 133–143

12. Huang, B., Cheu, R., Liew, Y.: GIS and genetic algorithms for HAZMAT route planning with security considerations. Vol. 18 (2004) 1–19

13. Kanoh, H., Hasegawa, K., Matsumoto, M., Nishihara, S.: Solving Constraint Satisfaction Problems by a Genetic Algorithm Adopting Viral Infection. IEEE SMC'96 (1996) 626-631

14. Li, X, Yeh, A.: Integration of genetic algorithms and GIS for optimal location search. International Journal of Geographical Information Science, Vol. 19, No. 5 (2005) 581-601

15. Mandow, L. ,Perez de la Cruz, J.: A heuristic search algorithm with lexicographic goals. Engineering Applications of Artificial Intelligence (2001) 751–762

16. Martins, E. D. Q., dos Santos, J. L. E.: The labelling algorithm for ultiobjective shortest paths. Technical report, Department of Mathematics, University of Coimbra, Portugal (1999)

17. Martins, E., Pascoal, M. M. B., Santos, J. L. E. D.: Deviation algorithms for ranking shortest paths. International Journal of Foundations of Computer Science (IJFCS)., Vol.10, No.3 (1999) 247–262

18. Nakahara, H., Sagawa, T., Fuke, T.: Virus theory of evolution. Bulletin of Yamanashi Medical College, Vol. 3 (1986) 14-18

19. Nepal, K., Park, D.: Routing algorithms for transportation systems and service improvement projects in urban transportation networks. Tech. rep., Department of Civil Engineering, Tokyo Institute of Technology,2-12-1, Ookayama, Meguro-ku, Tokyo 152-8552, Japan (2003)

20. Pahlavani, P., Samadzadegan, F.: Fuzzy-assisted in a GIS-based Dynamic Urban Traffic Congestion Model. In: 25th UDMS Conference, AAlborg, Denmark (2006)

21. Pahlavani, P.: The design and implementation a GIS for optimal urban rout selection based on Genetic Algorithm. M.Sc. Thesis, University of Tehran, Tehran, Iran (2005)

22. Ran, B., Boyce, D.: Modeling Dynamic Transportation Networks. Second Revised Edition, Springer (1996)

23. Roy, A., Banerjee, N., Das, S. K.: An efficient multi-objective QoS routing algorithm for real-time wireless multicasting. Technical report, Center for Research in Wireless Mobility and Networking, Department of Computer Science, University of Texas at Arlington, TX 76019-0015 (2002)

24. Satoh, H., Yamamura M., Kobayashi, S.: Minimal generation gap model for GAs considering both exploration and exploitation. Proc. IIZUKA '96 (1996) 494-497

25. Skriver, A., Andersen, K. A label correcting approach for solving bicriterion shortest-path problems. Computers and Operations Research 27 (2000) 507–524

26. Spears W. M.: The role of mutation and recombination in evolutionary algorithms. Ph.D. Thesis, University of George Mason, Virginia, USA (1998)

27. Wilson, I., Ware, M. & Ware, A.: A genetic algorithm approach to cartographic map generalisation. Computers in Industry: Special Issue: Soft Computing in Industrial Applications., Vol.52, No.3 (2003) 291–304

28. Zitzler E.: Evolutinary Algortithms for Multiobjective Optimization Methods and Applications. Ph. D. Thesis, Swiss Federal Institute if Technology Zurich, Swiss (1999)

Ontological Analysis of Observations and Measurements

Florian Probst

Institute for Geoinformatics, University of Münster, Robert-Koch-Str. 26-28,
48149 Münster, Germany
`f.probst@uni-muenster.de`

Abstract. Geographic information is based on observations or measurements. The Open Geospatial Consortium (OGC) has developed an implementation specification for observations and measurements (O&M). It specifies precisely how to encode information. Yet, the O&M conceptual model does not specify precisely which real-world entities are denoted by the specified information objects. We provide formal semantics for the central O&M terms by aligning them to the foundational ontology DOLCE. The alignment to a foundational ontology restricts the possible interpretations of the central elements in the O&M model and establishes explicit relations between categories of real world entities and classes of information objects. These relations are essential for assessing semantic interoperability between geospatial information sources.

1 Introduction

Geographic information is based on observations or measurements. The Open Geospatial Consortium (OGC) has developed an implementation specification for observations and measurements (O&M). The purpose of this specification is to provide "a conceptual model and encoding for observations and measurements" [1, p.1].

This purpose indicates that two fundamentally different things are specified. First, a conceptual model specifies the relevant concepts in the domain of interest [2]. Second, an information encoding model specifies how to produce information items *representing* concepts and their instances specified in the conceptual model. The O&M specification provides an information encoding model in UML. It specifies precisely the internal structure of information objects. Yet, the conceptualization of the geospatial real-world entities is described only in natural language and reflected only implicitly in the UML diagrams.

Textual descriptions allow a wide scope of interpretation of what the specified information objects denote in geospatial reality. The current O&M conceptual model is not suitable for assessing and achieving semantic interoperability between information sources. It specifies the conceptualization of an information object which *represents* an observation; however, it makes not formally explicit the conceptualization of an observation as such. Observations are central to the creation of geographic information. Thus, it seems desirable for the GI community to have a formal and explicit method at hand to express the meaning of terms related to observation events, independent of the information encoding.

We follow Guizzardi [3] in defining ontology modeling as special type of conceptual modeling. Ontology modeling focuses on structuring the entities existing in the

domain of interest. Thus, one can distinguish an ontology-based conceptual model for domain entities and a conceptual model for information objects.

The work is driven by the need for a framework enabling information providers to specify precisely which real world entities are denoted by the information objects they provide.

The authors of the O&M specification request that "an ontology of observable phenomena must be available" [1 p.15] in order to improve discovery and retrieval of O&M conformant data sources. We present, as a first step towards this goal, an ontology of the core O&M terms which is aligned to the foundational ontology DOLCE [4]. The alignment is performed in two steps:

- Model internal analysis: The central elements of the O&M conceptual model are interpreted in the DOLCE context. Model internal inconsistencies are identified and resolved.
- Model external analysis: To support semantics-based information discovery, the concepts specified in a conceptual model should identify precisely the set of intended real world entities. Thus, overly general meanings of the central O&M terms are identified and remodeled in order to restrict the scope of interpretation.

The work presented continues the work of turning the O&M model into an ontology-based representation format[1] and is a step towards an ontology-based semantic reference system [5, 6] which enables information providers to semantically annotate their O&M-conformant information models in a consistent and reproducible manner.

Throughout the article, the following formatting is used. Categories and feature types start with an upper-case letter and are written in *italics*; concepts start with a lower-case letter and are written in *italics*. When we refer to a term or notion or to an individual entity no additional formatting is used. For example, the category (or feature type) *Observation* is conceptually based on the concept *observation* which provides semantics to the term observation.

The remainder of the paper is structured as follows. Section 2 gives an introduction to DOLCE and the process of ontology alignment. In section 3, we introduce the central O&M terms, analyze the definitions with respect to internal conceptual inconsistencies and suggest alignments to DOLCE categories. Section 4 combines the results of chapter 3 into a first cut of a DOLCE-aligned ontology for observations. Section 5 draws conclusions and points to future work.

2   Background

In this section we introduce the general process of aligning domain ontologies to a foundational ontology. This is followed by an introduction to the central categories of the foundational ontology DOLCE.

2.1   Ontology Alignment Process

"Foundational ontologies are axiomatic theories of domain-independent top-level notions such as object, attribute, event, parthood, dependence, and spatio-temporal

---

[1] See http://seres.uni-muenster.de/ for a discussion paper and OWL-based ontologies.

connection" [7, p.1]. Foundational ontologies provide a rigorous formal semantics for the top-level notions. They are intended to serve as conceptual foundation for domain ontologies, which in turn provide specialized views on a certain domain. The semantics of the vocabulary used in domain ontologies are specified using top-level concepts.

Approaches to ontology alignment are presented in [8-11]. The alignment process to a foundational ontology can be seen as committing to the foundational ontology's "view" on how entities are categorized. In other words, the domain ontology engineer commits to the conceptualization underlying the foundational ontology and further refines it.

Generally, existing (informal) domain ontologies (e.g. the O&M model) were not intended to be aligned to a foundational level at the time they were developed. Thus, a careful analysis of the domain concepts is needed before an alignment can be established. In Fig. 1, three possible alignment situations are depicted and explained in the following.
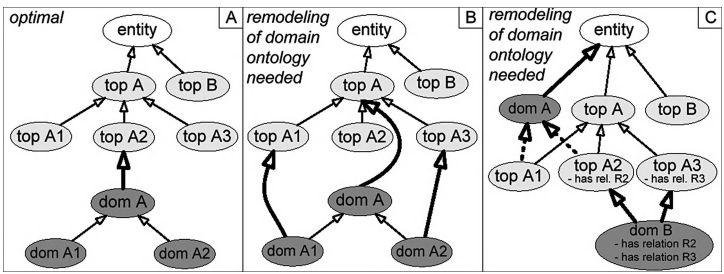

Fig. 1. Three general situations can be encountered when an existing domain ontology is intended to be aligned to a foundational ontology. Thick arrows indicate the *kind-of* relation between a domain concept (dom) and foundational concept (top). Remodeling of domain concepts is required in situations B and C in order to achieve a consistent alignment (situation A).

- A: The most general concepts of the domain ontology are aligned to the most specific concepts of the foundational ontology. This is the optimal alignment situation, and should be the result of an alignment process. It is unlikely that this situation is encountered without previous remodeling steps.
- B: Taxonomic relations of the domain ontology overlap with taxonomic relations of the foundational ontology. The domain ontology itself provides concepts which are more general than the foundational ontology's most specific concepts. Domain concepts might be removed to attain situation A.
- C: A domain ontology concept might act as superconcept of several foundational ontology concepts (see: dom A in Fig. 1C). Since this generalization is not considered in the foundational ontology it might indicate an overgeneralization in the domain ontology. Another suboptimal situation arises if a domain concept is specified to be the subconcept of two top-level concepts. This implies that the sets of characteristics (relations to other concepts) of two or more top-level concepts are all assigned to a single domain concept. In both cases a consistent alignment is impossible and thus a remodeling is required.

2.2   The Foundational Ontology DOLCE

Currently only a few full-fledged foundational ontologies are available. Among those are the General Formal Ontology (GFO) [12], the Object-Centered High-level Reference Ontology (OCHRE), [7]), OpenCyc [13], the Suggested Upper Merged Ontology (SUMO) [14] and the Basic Formal Ontology (BFO) [4, 15]. For this investigation, we have chosen the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [4] as foundational ontology[2]. The central categories of DOLCE are introduced in the following and depicted in Fig. 2. This introduction is based mainly on [4, 16]. The main purpose of DOLCE is the negotiation of meaning. It enables to sort the particulars (also called entities) encountered in the domain of interest into categories. Categories are understood as cognitive artifacts, which are more or less dependent on human perception, cultural imprints and social conventions [4]. We follow Sloman [17, p.192] stating that "[c]oncepts and categories are, to a large extent, flip sides of the same coin. Roughly speaking, a concept is the idea that characterizes a set, or category, of objects". In this sense, the categories in DOLCE reflect concepts and the concepts in turn provide the meaning to terms used in information systems. We are aware that the use of category in this sense has shortcomings as discussed in [18-20].

Endurants and Perdurants. Endurants and perdurants belong to the four top categories of DOLCE. Perdurants embrace entities generally classified as events, processes, phenomena, activities and states. Endurants can be divided into physical and non-physical endurants. Only physical endurants have physical and thus spatial qualities.

The difference between endurants and perdurants is related to their behavior in time. Endurants are wholly present at any time they exist. Their parts move with them in time. Perdurants, on the contrary, extend in time by accumulating different temporal parts. At any time that they are present, they are only partially present in the sense that some of their proper parts are not present anymore or are not yet present [16]. An
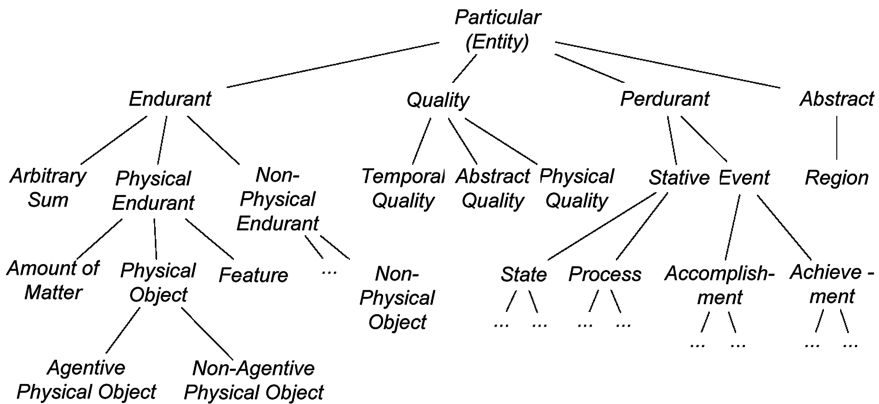

Fig. 2. The basic categories in DOLCE (reduced, for more details see [4])

---

[2]   Additionally some concepts are taken from the DOLCE extension DOLCE2.1-Lite-Plus (DLP) http://www.loa-cnr.it/DOLCE.html

endurant "lives" in time by participating in some perdurant(s). For example, a tree (endurant) participates in its lifespan (perdurant).

Qualities, Qualia and Quality Region. Qualities are seen as the basic entities we can perceive or measure, for example the volume of a certain lake, the color of a certain rose, or the length of a certain street [4]. The main characteristics of qualities are that they are observable and that they are inherent in other entities. Every (geospatial) entity comes with certain qualities, which exist as long as the entity exists. The (geospatial) entity itself can not be observed but only its qualities and the qualities of its parts. For example, we do not perceive an apple but its (indirect) qualities like weight, shape, color, taste or volume.

In O&M the notion property refers to what is called quality in DOLCE. However, DOLCE puts strong emphasis on the distinction between quality and property. A quality is a particular which is understood as individual entity. For the philosophical argument we refer to [4].

DOLCE distinguishes between physical, temporal and abstract qualities. Physical endurants can have only physical qualities. Non-physical endurants can have only abstract qualities and perdurants can have only temporal qualities. DOLCE defines a strict distinction between a quality, e.g., the depth of a specific lake, and its value (also called quale) which can be approximated with a depth measure, e.g. 10 m. To bring the above into a geospatial context: Geospatial entities are either conceptualized as perdurants or endurants. Endurants and perdurants are characterized through their qualities. Endurants and perdurants have unique qualities which are entities by themselves. Only endurants have spatial locations, whereas perdurants have indirect spatial locations via the endurants participating in them. Each quality of an entity has a quale which is represented as an atomic quality region. All regions at which qualities of a certain type can be located form together the quality space of that quality type. Quality regions, quality spaces and the values of qualities (qualia) are abstract entities.

3   Aligning the Central O&M Terms to DOLCE

This chapter introduces the notions feature, feature of interest, phenomena, event, procedure and instrument which are the notions employed by the O&M model to specify the central notion observation. Aligning these notions to DOLCE yields the following benefits:

− The central terms of the O&M model inherit the formal semantics provided by the foundational ontology. (DOLCE is fully axiomatized in the modal logic S5 plus the Barcan Formula [21].) When different GI information sources inherit the same top-level semantics, the benefit emerges that further refined terms become comparable.
− Internal ontological inconsistencies are identified when the elements of the O&M conceptual model are interpreted in the DOLCE context. In information encoding specifications such as the O&M, a model element (e.g. a feature type) may have multiple relations to other elements. This model element represents a set of real world entities. A model element is inconsistent, if the set of represented entities needs to be changed in order to make sense of all the relations that the model element entertains (see section 3.2 Feature of Interest and section 3.5 Procedure and

Instrument). If the conceptual model is mainly based on natural language, this inconsistency may not be obvious, since most natural language terms change the set of entities they denote dependent on the context in which they are used.

− Overly general definitions in the current O&M model are identified. A model may be internally consistent, yet the concepts are too general to support efficient semantics-based discovery of information sources.

On the left side of figures 4-7, the original O&M definitions are interpreted in terms of DOLCE. On the right side suggestions are given on how to arrive at an optimal alignment situation (see Fig. 1A).

The O&M categories of each figure are depicted in light grey. DOLCE concepts and DOLCE Lite Plus concepts are depicted in dark grey. Black open arrows represent taxonomic relations (kind-of). Black filled arrows represent non-taxonomic relations. Thick arrows indicate the alignment between domain and foundational ontology.

3.1 Feature

Feature is one of the core notions in all OGC specifications. It is remarkable that the definition given in most of the OGC standards' terminology section (a feature is an abstraction of real world phenomena) appears to be understood in two different senses.

− *Sense 1*: A feature can be any entity that exists in physical and social reality, for example the town you live in or your favorite national park. Thus, the process of abstracting real world entities into categories results in the definition of feature types, for example, the feature type town. Feature types are categories to which individual features belong.

− *Sense 2***:** Feature type and feature are understood as information objects. Such an information object can be used to *represent* either some category or some individual of a category.

The difference between those senses boils down that one can either understand feature as pure information object with the sole purpose of *representing* a real-world entity or understand a feature to *be* a (geospatial) real world entity.

Congruent with sense 1 is for example the section on conceptual modeling in ISO 19101 [22, p.11]. "Conceptual modeling is the process of creating an abstract description of some portion of the real world and/or a set of related concepts. As an example, a set of features such as watercourses, lakes, or islands might constitute a portion of the real world being modeled. […] The abstract description of these *real-world features* is called a *conceptual model*. Conceptual models may exist only in the minds of people who communicate them to each other verbally and often imprecisely".

Concepts are mental constructs according to which human agents form categories. We understand "abstract descriptions" in the quote above as concepts, if these descriptions exist in the mind of an information provider only. As soon as these "abstract descriptions" are made partially explicit via some representation language, we talk about models of concepts. Information system ontologies are concerned with providing such models of concepts. We understand concepts as providing the rules
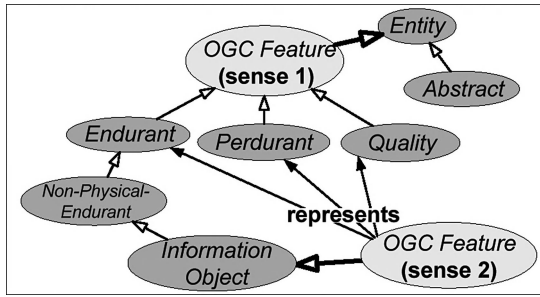
Fig. 3. The two possible interpretations of the notion feature are depicted in terms of DOLCE. The alignment situation for sense 1 corresponds to Fig. 1C. A feature in sense 2 is an information object that is used to represent perduring entities (e.g. processes) and enduring entities (e.g. physical objects). An information object itself is an enduring entity.

according to which we classify the entities in physical and social reality into categories. In this sense, the notion feature type is equivalent to the notion category.

Congruent with sense 2 is for example ISO 19109 [23, p.8]. It states that the feature is "[a] fundamental unit of geographic *information*".

In this definition, a feature is understood as information carrying unit or object. Thus, the feature's purpose is to model and to represent some entity. The definition restricts the notion of feature to denote exclusively information objects. According to this definition, a feature type can be used to *represent* an abstraction of real world phenomena (e.g. the category *Lake*) and an individual feature can be used to *represent* some entity (e.g. Lake Constance).

Semantic problems are unavoidable if the notion feature is used within the community to denote the real world entity as well as its representations. We suggest that in an OGC context, the meaning should be restricted to sense 2; a feature is an information object used to represent individuals of other categories.

3.2   Feature of Interest

According to the O&M specification, "the **feature of interest** is the object regarding which the observation is made, and whose value must be a feature instance" [1, p.12]."[A] phenomenon is associated with an identifiable object, which is the feature of interest of the observation. […] The key idea is that the observation result is an estimate of the value of some property of the feature of interest" [1, p.11].

Ontological Interpretation and Suggestions. In the O&M model, a *feature of interest* is the target of an observation. This implies that *feature* is understood in sense 1 as real-world entity (see Fig. 3). Understanding *feature of interest* in sense 2, implies that *only* properties[3] of information objects can be observed. For the discovery of information sources it is important to know whether the observation is performed on an individual real world entity or on an information object. For example, the width of a lake can be observed directly or it can be "observed" on a representation of a lake, e.g. in a GIS.

---

[3] In DOLCE terminology the notion quality is used instead of the notion property.
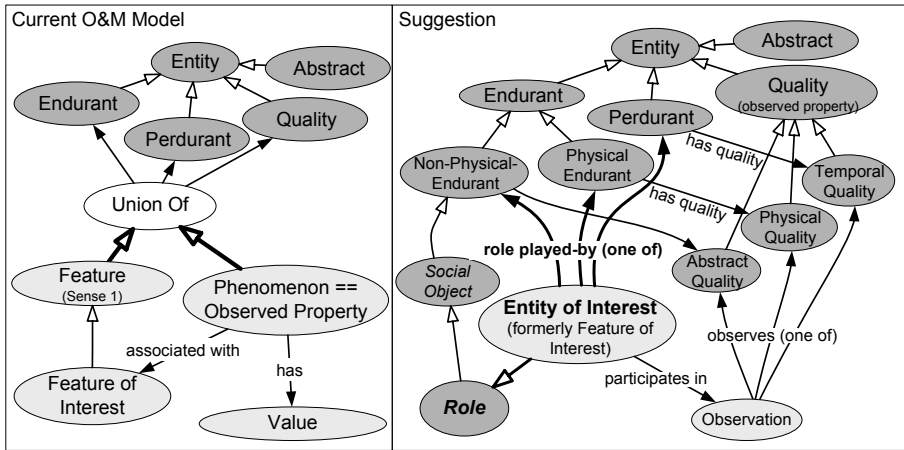
Fig. 4. Left: The model is consistent with the current definition of *Observed Property* only if *Feature of Interest* is understood in sense 1 (see Fig. 3). Right: In order to keep a consistent use of the notion feature, we suggest renaming the class *Feature of Interest* to *Entity of Interest*.

The O&M model is consistent only if *feature of interest* is understood in sense 1 (see Fig. 3 and Fig. 4). However, this conceptualization of *feature* leaves the model underspecified. The extension of the category *Feature* is equivalent to the extension of the category *Phenomenon* which may indicate that one is redundant. Further, the model does not specify which phenomenon can be associated with which feature of interest. For example, a subcategory of *Phenomenon* called S*peed* could be associated with a subcategory of *Feature of Interest* called *Lake*.

We suggest to emphasize the ontological distinction between a feature of interest and the entity whose properties are observed. We introduce the example of observing the depth of a lake to illustrate why this ontological distinction is useful. According to the O&M definition, the observed phenomenon (*depth)* is associated with an identifiable object (*lake).* The identifiable object (*lake*) is seen as an individual of the category *Feature of Interest*. According to [24, 25] an entity should entertain a IS-A (taxonomic) relation to a category only, if this relation holds as long as the entity exists. It is plausible that an entity can stop being the feature of interest and still exist. Therefore we suggest that *Feature of Interest* is *not* to be modeled as subcategory of *Feature* (in neither sense 1 nor 2).

DOLCE introduces the category *Role* being a kind of *Social Object* which in turn is a kind of *Non-Physical Endurant*. Any entity can entertain the non-taxonomic relation *plays-role* to individuals of the category *Role*. For example, a lake plays the role of a recreation area. Related to the O&M model, *feature of interest* is conceptualized as a *role* which any entity that has an observable property can play. In our example, a lake *plays* the role of a feature of interest only if it is the target of an observation. If it is not the target of an observation, the lake still is-a (individual of) *Lake* which is a kind-of *Physical Endurant* but the lake *is not* an individual of *Feature of Interest* anymore. The feature of interest stops to exist as soon as the observation event is finished. Modeling the semantics of the term lake as denoting an entity which stops to exist as soon

as it is not the feature of interest of a certain observation anymore is not helpful for a semantics-based search environment.

The O&M model has the purpose of specifying information encodings. Thus, the classes which use the stereotype <<feature type>> are understood in sense 2. However, in the case of the class *Feature of Interest*, the O&M model implicitly uses the notion feature in sense 1. In the context of standards for information encoding, we suggest to use the notion feature only in sense 2 in order to avoid inconsistencies explained in the next section.

3.3 Phenomenon

According to the O&M specification, "the observed property is usually a phenomenon, the estimate of whose value is the primary result of the observation" [1, p.14] "[T]he value of the observed property identifies or describes the phenomenon for which the observation result provides an estimate of its value " [1, p.12].

Ontological Interpretation and Suggestions. The O&M model intentionally gives a rather vague definition of what an observed property is (personal communication with O&M editors). Phenomenon, the second important term in the definition is also not precisely defined. We rely on the general OGC usage of the notion "real world phenomenon", assuming that the O&M notions phenomenon and observed property are intended to denote the same category. This would imply that any entity can be the observed property of another entity. Interpreting the O&M definition in terms of DOLCE implies that an entity belongs to the category Phenomenon if it also belongs to one of the categories Endurant, Perdurant or Quality. In the context of semantics-based discovery of information sources, the O&M definition of phenomenon does not allow to sufficiently specify a phenomenon.

The category *Observed Property* in the O&M model subsumes the category *Quality* in DOLCE. For the formal ontological distinction between property and quality we
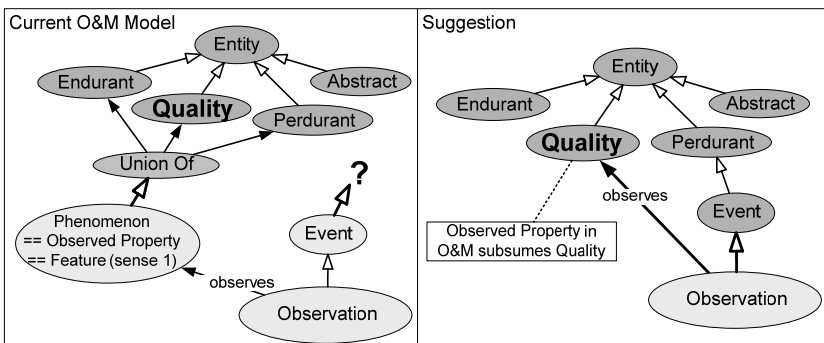

Fig. 5. Left: *Phenomenon* has the same extension as *Feature* in sense 1 (see Fig. 3). This leads to an underspecified model. The O&M category *Event* can not be aligned consistently to DOLCE. Right: According to DOLCE, *only* individual qualities are entities which can be observed.

refer to [4]. In DOLCE, only qualities are entities that can be observed. In this sense, a physical endurant, a lake for example, is not observable. In turn, the sum of all observed quality values allows the cognition of physical endurants. DOLCE provides a generic categorization of qualities, restricting which qualities an entity can have. We want to stress that it is important to enable the identification of the quality *without* referring to the information encoding with which the observation result is encoded.

3.4 Event

According to the O&M specification, an event "is a feature type characterized by a time whose value is a temporal object (TM_Object), a location whose value is either a spatial object described using coordinates (GM_Object) or a named place (EX_GeographicDescription). A number of persons or organizations (CI_ResponsibleParty) may be identified as responsible for the Event. "[1, p.11].

Ontological Interpretation and Suggestions. The textual definition of Event in the O&M model states that the category *Event* is a kind of feature type. Thus any individual event is a feature. This definition can be understood in two ways.

1. If *Feature* is understood in sense 2, then *Event* refers only to the specification of information objects representing real world events. This implies that the O&M model provides no definition of the underlying conceptualization of real world events. An event understood as information object has relations only to other information objects. This is reflected in the relations to *GM_Object*, *TM_Object* or *EX_GeographicDescription*, individuals of these categories are all information objects encoding time and location.
2. If *Feature* is understood in sense 1, then the semantics of the term event stay rather unspecific (see Fig. 3) since the direct supercategory of *Event* is *Feature* and almost all entities can be considered as features according to this definition.
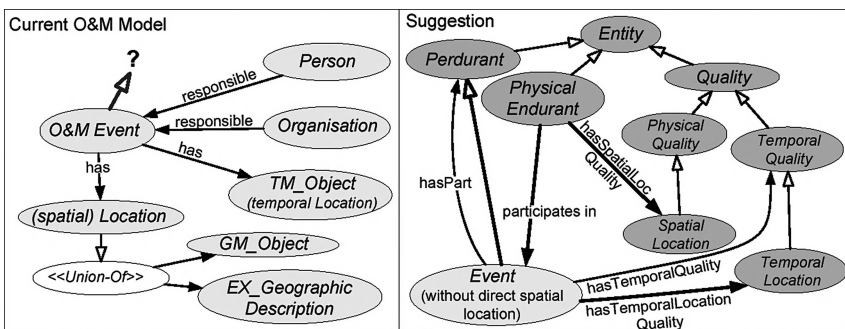

Fig. 6. Left: An event is defined to have both, a direct *spatial* and a *temporal location*. This prevents an alignment to DOLCE. Right: An event is a perduring entity, which has only temporal qualities. A physical object participates in an event and thus provides indirectly a spatial location for the event.

In the O&M model, any event has a relation to a time individual and a location individual. The conceptualization of this relation is not further specified. For the categories *Time* and *Location* no conceptualization is provided, they are defined in terms of information objects. Additionally any event can have a relation *responsible* which relates it to some person or some organization. The relation *responsible* is not further specified, too.

The current model of *Event* can not be aligned to DOLCE since an event has both, a direct spatial location and a direct temporal location (Fig. 6). This is not possible in DOLCE. In terms of DOLCE, an event is a *Perdurant;* it is an entity that happens in time (e.g. a soccer match, a departure, or an observation). An event entity is never fully present at a certain time. Take as example a particular soccer match event. The first chip pass is not present any longer when the first goal happens. In this sense, a soccer match is never completely present. Both, chip pass and goal are temporal parts of the event soccer match. Events are characterized by having only temporal qualities and by having at least one temporal location. An event can not have physical qualities. This leads to an important difference to the O&M *Event*. An event in the DOLCE sense can not have a spatial location, since a spatial location is a physical quality. Any perdurant can have physical qualities only indirectly via a physical object which participates in an event. For example a physical instrument participating in an observation event spatially locates the observation event. Events can have other perduring entities as parts.

Separating temporal and physical location has the advantage that one can ask for the spatial location of different physical entities which participate in an event. An observation event for example has two important but indirect spatial locations; the spatial location of the entity of interest, for example urban areas, and the spatial location of the physical instrument, for example a satellite, with which a quality of the entity of interest is observed. In the search for suitable data sources, it is useful to be able to specify in which of the two spatial location one is interested.

3.5   Procedure and Instrument

According to the O&M specification, a procedure is used to generate an observation [1, p. 16]. An "observation uses a procedure, which is often an instrument or sensor but may be a process chain, human observer, a computation or simulator"[1, p. 11]."The value of the procedure property is the description of a procedure" [1, p. 16].

Ontological Interpretation and Suggestions. The O&M textual section defines implicitly an anonymous category as direct super category of *Procedure* (see Fig. 7). This anonymous category is the union of the categories *Human Observer, Process Chain, Sensor, Computation, Simulator* and *Instrument.* In the UML diagrams, *Instrument* is specified to be the subcategory of *SimpleProcedure*. None of the other supercategories are specified in the UML diagrams of the O&M model. Circular taxonomic relations for *Instrument* result, when textual und UML-based definitions of the O&M model are viewed together (see Fig. 7). This is probably not intended.
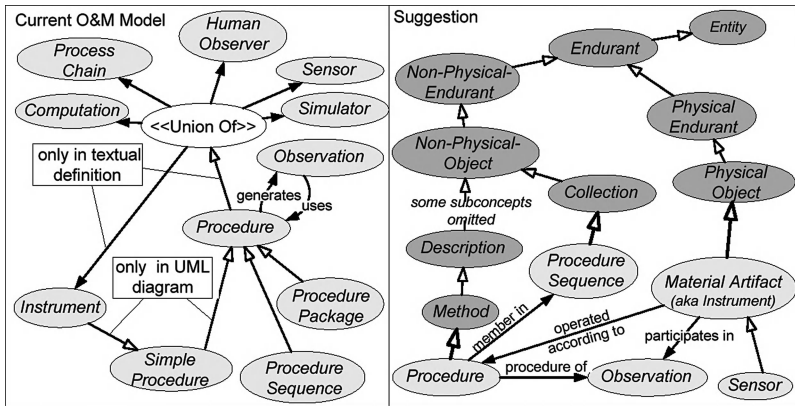
Fig. 7. Left: Textual definitions and UML-based definitions result in circular definitions. We suggest to define *Procedure* as a kind of *Method* according to which a physical instrument is involved in an observation.

According to [1, p. 16], the "value of the procedure property is the description of a procedure". According to this definition individuals of *Procedure* are descriptions. DOLCE provides a category *Description* which in turn has a subcategory *Method.* Descriptions and methods are non-physical, social objects. We consider *Method* as suitable supercategory for *Procedure*. A method is a description according to which a process or event can be performed. However, this would be inconsistent with the current O&M understanding that a procedure (actively) *generates* an observation. We suggest to replace the *generates*-relation by a newly introduced relation *procedure of,* which we specify as subrelation of the DOLCE relation *method of.*

If *Instrument* is understood as subcategory of *Physical Entity,* then the definition of *Procedure* as subcategory of *Method* causes an inconsistency with *Instrument*. Physical entities and methods are ontologically distinct categories and should not be confused. However, a physical instrument does operate according to a procedure or method. We suggest to remodel *Instrument* as *MaterialArtifact* which is a subcategory of *PhysicalObject* and as supercategory of *Sensor* and *Simulator* or any other newly introduced category accounting for measurement or observation devices. In the O&M model, *Instrument* has an attribute *SerialNumber*, indicating that instruments in O&M are understood as physical entities and not as procedures or methods.

Analogously to *Instrument*, the individuals of *HumanObserver* do act according to a method or a procedure but should not be modeled as supercategory of *Procedure*. According to DOLCE, *HumanObserver* can be seen as a kind of *Role* which is played by an individual of the category *Person*.

According to DOLCE, a simulation or a computation is a process which in turn is a perduring entity. Since observations, seen as perduring entities, can have other perduring entities as parts, we suggest to model simulation and computation processes to be temporal parts of observation events. Similar to observations, simulations and computations can be described by methods or procedures, yet they are no methods or procedures.

To summarize the above, simulations and computations are individuals of the category *Processes*, sensors and simulators are individuals of the category *Material Artifacts* and human observers can be seen as individuals of the category *Role* some person can play. Unifying the above categories into one anonymous super category of *Procedure* (see also Fig. 7) hinders a precise semantic annotation of observation encoding models drastically.

Another inconsistency in the O&M model is that currently the category *Station* is modeled as subcategory of *Feature of Interest*. This modeling decision is partly inconsistent with the textual descriptions in the O&M specification. On page 11 it is stated that "[t]he phenomenon is associated with an identifiable object, which is the feature of interest of the observation". For example, a thermometer observes the phenomenon air temperature. In this case, the surrounding air mass plays the feature of interest, not the station at which the thermometer is hosted. We suggest that *Station* never to be modeled as feature of interest. The *hosts* association which relates *Station* and *Procedure* is inconsistent with the new and partly with the old definition of *Procedure*. A station understood as physical entity which is not a physical instrument does not follow nor does it host a procedure or method. We suggest to model stations to *host* physical instruments, which in turn operate according to a procedure.

4   Basic Ontology of Observations

In the previous chapter, the concepts behind the O&M terms are ontologically analyzed. In the O&M context, these concepts serve the purpose to specify the most central concept of the O&M specification: *observation*. The results from the previous chapter are brought together in this chapter, resulting in a first cut on a DOLCE aligned ontology for observations (Fig. 8).

Ontological Interpretation and Suggestions. *Observation* is modeled as subcategory of *Accomplishment* which in turn is a subcategory of *Event*. As a perduring entity, an observation can have other perduring entities as parts. In particular, observations have processes as parts. The two central processes of an observation are:

1. The process of approximating the observed quality's value (called quale in DOLCE). This process turns the continuous quale of the observed quality into a discrete approximation entity which is of the same ontological nature as the quale.
2. The process of assigning a symbol (number, term, complex information item) to the approximation entity identified in the previous process. In order to communicate which approximation entity was identified in the previous step, symbols are needed.

A procedure describes an observation and thus provides the context needed to interpret the observation results. However, care should be taken when the procedure or the employed instrument is used to characterize the quality. For example, it seems odd to define the ontological nature of the quality temperature by stating that a temperature is a temperature because it is observed by a thermometer.

The categories Value, Quale and Abstract are not further investigated in this article and are subject to future work.
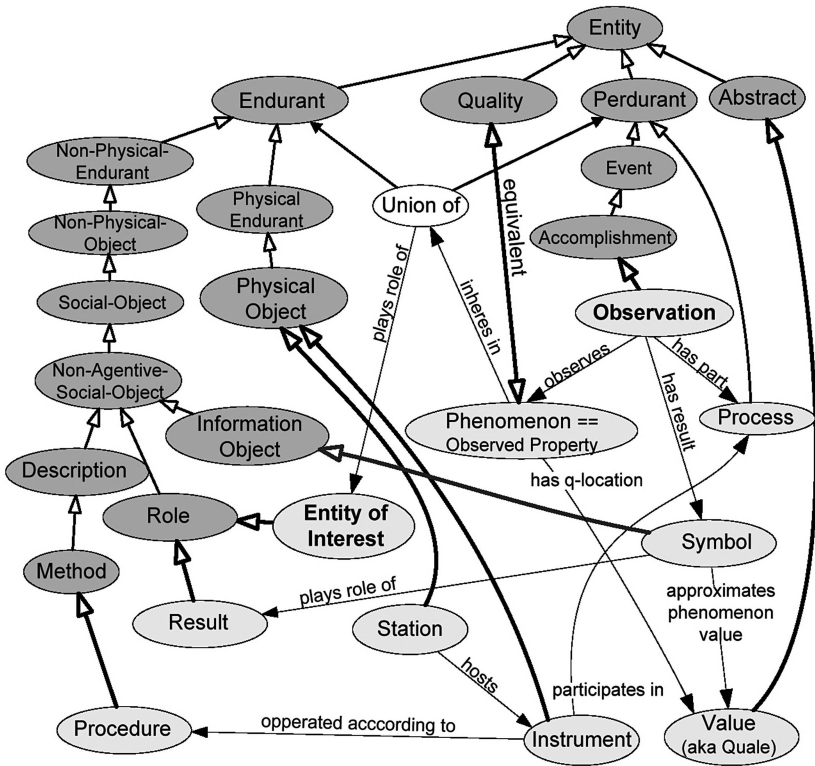
Fig. 8. Basic ontology for observations. See figures 3-7 for more details.

5   Result and Conclusion

Geospatial information sources can be discovered successfully if they are based on two explicitly specified types of conceptual models:

1. A conceptual model of the domain of interest for which representations are sup-plied. This includes specifications of the entities of interest, the observed qualities of these entities as well the procedures applied to achieve observation results which can be communicated.
2. A conceptual model of the information encoding in which the representations of the real world entities are supplied.

The O&M implementation specification provides only the second type explicitly. However, only if one knows which kinds of real world entities are represented, it is helpful to know how they are represented. In this paper we presented the formal and explicit basis of the missing first type of conceptual model. It is specified as an ontol-ogy. This ontology reflects as closely as possible the implicit O&M conceptual model and is at the same time ontologically consistent with the foundational ontology DOLCE.

The remodeling involved to

- restrict *Feature* to be a subcategory of DOLCE:*Information Object.*
- restrict *Feature of Interest* to be a subcategory of DOLCE:Role
- turn *Feature of Interest* into Entity of Interest.
- restrict *Observable Property* to be equivalent to DOLCE:*Quality*.

The ontological analysis of O&M showed that the notion of feature needs to be interpreted in two senses in order to keep the model consistent. This however leads to a model where the categories *Feature* and *Phenomenon* are equivalent on the ontological level leaving one notion redundant. In the current O&M model, the category *Procedure* is modeled with circular taxonomic relations. Remodeling it as subcategory of *Method* and adding non-taxonomic relations between the categories *Procedure* and the categories *Instrument*, *Sensor* and *Simulator* resolved this inconsistency.

The current O&M model relies on the human user to understand the meaning of the textual descriptions and the terms used in the UML descriptions. The presented ontology relates these terms to the rigorous formal semantics provided by the foundational ontology DOLCE.

Smith and Mark [18] claim that one benefit of developing and applying ontology in geospatial application is that "geographic information systems need to manipulate representations of geographic entities, and ontological study of the corresponding [categories], especially those at the basic level, will provide default characteristics for such systems". The results presented in this article are practical examples supporting this claim. The above mentioned benefit can be further specified: The risk of misinterpretation of O&M-conformant information sources is drastically restricted if they commit to the presented ontology or a specialization of it. The discovery of geospatial information sources is facilitated since reasoning about entities of interest and their potential observable qualities as well as the procedures employed to observe them becomes possible. A further benefit is that the occurrence of inconsistencies introduced unintentionally by the developer is decreased since automatic consistency checking becomes available.

Much work has been done to produce foundational ontologies like DOLCE as well as to produce domain specifications like the O&M model. A combination of these efforts by undergoing the alignment process is highly beneficial for current and future challenges in information discovery and retrieval.

6   Future Work

The attempt to align observation and measurement concepts to the foundational level has produced further questions regarding the modeling decisions taken in DOLCE. Further investigations are required on the concept *quality*. Especially for information sources dealing with observations and measurements, a precise specification of the observed qualities of the entities of interest is important. Questions regarding the combination of qualities forming compound or complex qualities need to be solved. Individuals of which quality categories can be combined and in which category of entity can the different qualities be inherent? In this context a detailed analysis on how to further combine the theory of conceptual spaces [26] with the quality spaces

provided in DOLCE is required. We envision a theory of semantic reference spaces and regions which allow to combine the ontologically important structures of quality spaces with spaces serving the solely purpose of approximating the generic value (quale) of a quality, thus allowing to specify the semantics of unit of measure, scale, reference datum and other conventionally agreed structures needed to perform observations and measurements. We consider this as the basis for methods enabling semantic translation between terms of different domain ontologies.

With respect to practical application, the task of annotating geospatial information sources needs to be facilitated. A framework is required guiding and assisting the information provider through the annotation process, by suggesting and explaining the formal structure of the ontology.

Acknowledgements

References

1. OGC: Observations and Measurements. Open GeoSpatial Consortium, OpenGIS Recommendation Paper OGC 05-087r1 (2005)
2. Guizzardi, G., Herre, H., Wagner, G.: On the General Ontological Foundations of Conceptual Modeling. In: Proc. 1st International Conference on Ontologies, Databases and Applications of Semantics (ODBASE 2002) (2002)
3. Guizzardi, G.: Ontological Foundations for Structural Conceptual Models. Enschede, Netherlands (2005)
4. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A. WonderWeb Deliverable D18, Ontology Library (final). Available: http://wonderweb.semanticweb.org/ delierables/documents/D18.pdf (2003)
5. Kuhn, W.: Semantic Reference Systems. International Journal of Geographical Information Science 17 (2003) 405-409
6. Kuhn, W.: Geospatial Semantics: Why, of What, and How? Journal on Data Semantics (2005) 1-24
7. Schneider, L.: Designing Foundational Ontology - The Object-Centered High-level Reference Ontology OCHRE as a Case Study. In: Proc. Conceptual Modeling - ER 2003, 22nd International Conference on Conceptual Modeling (2003) 91-104.
8. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A.: Sweetening WordNet with DOLCE. AI Magazine (2003)
9. Oberle, D., Mika, P., Gangemi, A., Sabou, M.: Foundations for service ontologies: Aligning OWL-S to DOLCE. In: Proc. World Wide Web Conference (WWW2004), Semantic Web Track (2004)
10. Noy, F. N., Musen, M. A.: SMART: Automated Support for Ontology Merging and Alignment. In: Proc. 12th Workshop on Knowledge Acquisition, Modelling and Management (KAW'99) (1999)

11. Borgo, S., Leitão, P.: The Role of Foundational Ontologies in Manufacturing Domain Applications. In: Proc. OTM Confederated International Conferences, ODBASE 2004 (2004) 670-688
12. Degen, W., B., H., H., H., Smith, B.: GOL: Towards an axiomatized upper level ontology. In: Proc. 2nd International Conference of Formal Ontologies and Information Systems (FOIS 01) (2001)
13. Lenat, D. B., Reed, S. L.: Mapping Ontologies into Cyc. In: Proc. AAAI workshop on Ontologies and the Semantic Web (2002)
14. Niles, I., Pease, A.: Toward a Standard Upper Ontology. In: Proc. 2nd International Conference of Formal Ontologies and Information Systems (FOIS 01) (2001) 2-9
15. Grenon, P., Smith, B.: SNAP and SPAN: Towards Dynamic Spatial Ontology. Spatial Cognition and Computation 4 (2004) 69-104
16. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening Ontologies with DOLCE. In: Proc. International Conference on Knowledge Engineering and Knowledge Management. AAAI (2002)
17. Sloman, S. A., Love, B. C., W.-K., A.: Feature centrality and conceptual coherence. Cognitive Science 22 (1998)
18. Smith, B., Mark, D.: Ontology and Geographic Kinds. In: Proc. 8th Int. Symposium on Spatial Data Handling, SDH'98 (1998) 308-320
19. Lakoff, G.: Women, Fire, and Dangerous Things: What Categories Reveal About the Mind. The University of Chicago Press, Chicago (1987)
20. Rosch, E.: Principles of Categorization. In: E. Rosch and B. Lloyd, (eds.): Cognition and Categorization. Lawrence Erlbaum Associates (1978) 27-48
21. Hughes, G. E., Cresswell, M. J.: A New Introduction to Modal Logic. Routledge, London (1996)
22. ISO/TC-211, OGC: 19101 Geographic information - Reference model. ISO TC 211, Draft International Standard (2002)
23. ISO/TC-211, OGC: Text for DIS 19109 Geogaphic information - Rules for application schema Vs. 2.0. Draft Version. ISO, (2001)
24. Guarino, N.: The Role of Identity Conditions in Ontology Design.
25. Welty, C., Guarino, N.: Supporting Ontological Analysis of Taxonomic Relationships. Data and Knowledge Engineering 39 (2001) 51 - 74
26. Gärdenfors, P.: Conceptual Spaces - The Geometry of Thought. Bradford Books, MIT Press, Cambridge, MA (2000)

Splitting the Linear Least Squares Problem for Precise Localization in Geosensor Networks

Frank Reichenbach[1], Alexander Born[2], Dirk Timmermann[1], and Ralf Bill[2]

[1] University of Rostock, Germany
Institute of Applied Microelectronics and Computer Engineering
{frank.reichenbach, dirk.timmermann}@uni-rostock.de
[2] University of Rostock, Germany
Institute for Geodesy and Geoinformatics
{alexander.born, ralf.bill}@uni-rostock.de

Abstract. Large amounts of cheap and easily deployable wireless sensors enable area-wide monitoring of both urban environments and inhospitable terrain. Due to the random deployment of these sensor nodes, one of the key issues is their position determination. Noisy distance measurements and the highly limited resources of every sensor node, due to tiny hardware and small battery capacity, demand the development of robust, energy aware, and precise localization algorithms.

We believe this can be achieved by appropriately distributing the complex localization task between all participating nodes. Therefore, we use a linearization tool to linearize the arising non-linear system of equations into a linear form that can be solved by a distributed least squares method. It is shown in this paper that we can save with this new approach more than 47% of computation cost whilst maintaining a low network traffic. Additionally, we describe memory optimizations to process the complex matrix operations with only a few kilobyte of memory on the sensor node.

1 Introduction

1.1 Sensor Networks

Wireless sensor networks (WSN) are composed of hundreds of tiny electronic devices, able to sense the environment, compute simple tasks and communicate with each other. Gathered information (e.g. temperature, humidity etc.) are transmitted in a multi hop fashion over direct neighbors to a data sink, where the data is interpreted [1]. With methods such as self configuration and self organization the network reacts to node failures.

Due to the desired node size of only a few cubic millimeters, the dimensions of the communication module and the battery are critical. Consequently, the scarcest resource within a network is the available energy [2]. Therefore, achieving a long lifetime of the sensor network requires low power hardware as well as optimized algorithms.
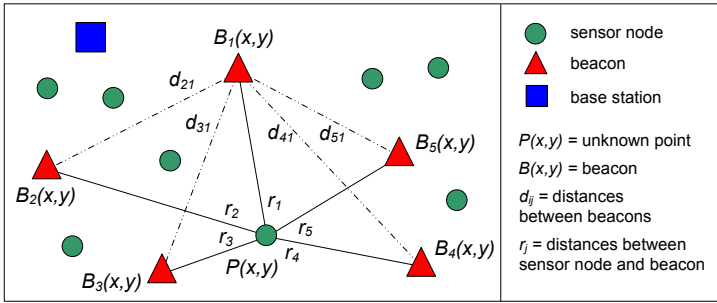
Fig. 1. Localization problem with one unknown point and five beacons (known points)

1.2   Problem Statement

After deploying the sensor network over an area of interest, initially the sensor nodes carry no position information. Sensor information are only useful if combined with their geographical position. Possible positioning technologies are the Global Positioning System (GPS), the Global System for Mobile Communication (GSM) or soon the European System Galileo [3],[4],[5]. However, these systems are unsuitable for miniaturized sensor nodes and could only be used for a small number of nodes, due to the size of the hardware, the high prices and the high energy requirements. Thus, it is a common technique to integrate an existing localization system on some more powerful nodes, further called beacons. Then, all remaining nodes estimate their own position with measurements such as distances to these beacons autonomously. Figure 1 shows an example constellation of a network with one unknown sensor node and five beacons.

For several reasons, a node's position is very important:

- Sensed data without a location where they were gathered are generally useless.
- Full covered sensor networks enable energy aware geographic routing.
- Self configuration and self organization are key mechanisms for robustness and can be easily realized with position information.
- In many applications the position itself is the information of interest.

A very comprehensive overview of geosensor networks can be found in [6].

In this paper we present a new approach to energy-saving determination of unknown coordinates with a higher precision compared to approximate positioning methods [7]. Using this method, the calculations are split between the resource-limited sensor nodes and the high-performance base station.

This paper is structured as follows: In Section 2 we give a basic overview of the methods for localization in wireless sensor networks. In Section 3 we describe the position estimation based on relationships to known points. In particular two methods to linearize the non-linear system of equations are discussed. Then, in Section 4, we study the robustness of our model to noisy distances and moreover, we examine the complexity of the least squares method. Next, we present in

Section 5 our new approach to split the least squares method with the aim to minimize the load on the sensor nodes. Furthermore, the new method is analyzed with respect to complexity, memory requirement, and communication effort. We finally conclude the paper with Section 6.

1.3   Motivation

As a scalable and decentralized system the wireless sensor network permits the automatized and area-wide observation of distributed, hardly accessible properties or phenomena in motion. In relevant fields of geoinformatics the main issue is to adapt geometric-topological localization algorithms to sensor networks. Thus, it is conceivable to develop hybrid methods with high precision and low computation effort. This is achievable by e.g. meshed triangles [8] or Radial-Sweep methods [9] that can be applied in 2d as well as in 3d. Wireless geosensor networks enable new possibilities for timely detection of wood fire, monitoring of slope slipping, and "Precision Farming". In these applications sensor data is gathered and analyzed using extensive interpolation algorithms. As an example, precision farming has to be mentioned. Sensors are arbitrary distributed over an agricultural crop field - e.g. dropped off by airplane. These sensors may measure parameters like humidity, pH-value, etc. over a specific period of time. The data can be collected at the base station at the farmers house. Afterwards or online in a spatial information system, surface models are generated from the point related data. For that an accurate position of the sensor nodes is needed.

2   Related Work: Localization Algorithms in Sensor Networks

The localization process in a WSN is based on different network topologies. We described in Section 1.2 that most of the localization algorithms assume beacons, initially knowing their own position. So, the first topology (shown in Figure 2a) consists of ad hoc connected sensor nodes and these beacons. In this
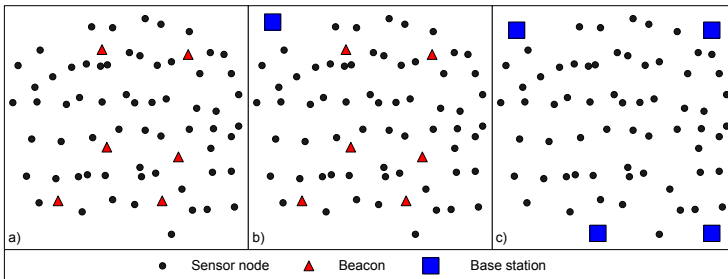

Fig. 2. Ad hoc connected sensor nodes: a) with beacons, b) with beacons and one fixed base station (**loosely coupled network**), c) without beacons and four base stations (**infrastructure case**)

case either the localization algorithm is processed fully distributed on all sensor nodes and/or additionally on beacons. This brings the advantage that every node is responsible for the computation of its own position with lowest communication traffic in the network.

Complementary, as shown in in Figure 2c, all sensor nodes transmit their data to the base stations (e.g. server, desktop-pc) in the network. Base stations relieve all sensor nodes from the complex computation tasks. However, this solution leads to the highest communication overhead and fails if the routes to the base stations are defective or the base stations are down. Beneficially, the base stations are part of an infrastructure that allows the determination of a geodetic date (origin, rotation and scale of a coordinate reference system). Thus, every sensor node can be defined in a world wide valid frame of reference system.

However, both topologies implies either overhead in communication or computation that qualifies a mixed topology in Figure 2b, where the incomplex tasks of the localization are processed on all sensor nodes and energy consuming tasks are delivered to the base station with respect to the communication overhead. This hybrid topology must be flexible enough to compensate all kinds of node failures to allow a fully distributed as well as a centralized computation. This yields in longest network lifetime.

Given these facts, all localization methods in WSNs can be classified into exact and approximate methods (see Figure 3), regarding the achievable precision and the complexity.

Approximated Localization. Many approximate approaches for the determination of sensor nodes exist in literature. These algorithms are resource-efficient but also result in higher positioning errors. Some approximate algorithms completely avoid the defective distance measurements to estimate a position.

For example Bulusu postulated the "Coarse Grained Localization" in [7]. In this algorithm in the first phase, all beacons send their position $B(x, y)$ to all sensor nodes within their transmission range. In the second phase, all sensor nodes calculate their own position by a centroid determination from all n positions of the beacons in transmission range. This algorithm requires only receiving few data on the sensor nodes and minimal computation overhead, but lead to higher localization errors. However, these errors can be acceptable in specific applications.

Tian et al. completely avoid distances in their approach [10]. First, triangles are combined with all beacon coordinates in the field. Then every unknown checks by a "Point in Triangulation-test" on which triangle surface it is situated. In this test only neighbor relations are used. A following intersection test with all filtered triangles shrinks the region where the unknown is probably located.

More examples of such geometric or proximity approaches are the hybrid methods [11], by using local coordinate systems [12], and the "Weighted Centroid Localization" [13].

Scene analysis with imagery is also a well researched method, but less important in sensor networks due to the size and the high energy consumption of the cameras.
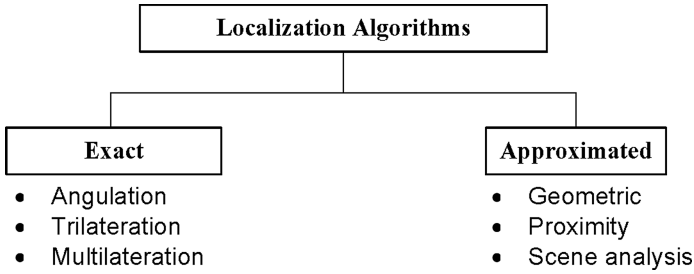
Fig. 3. Classification of common localization techniques in sensor networks

Exact Localization. In contrast to approximate algorithms, exact methods use the known beacon positions and the distances to the sensor nodes in order to calculate their coordinates through the solution of non-linear equations. Using a minimum of three beacons (in two dimensions), the coordinates of the sensor nodes may be determined using intersection (trilateration). The use of more than three beacons gives more information in the system and allows the refinement of the position and the detection and removal of outlying observations (multilateration). The least squares method (LSM) is used for the solution of the simultaneous equations. The LSM produces best linear unbiased estimators (BLUE) in a stochastic sense, however it is complex and resource-intensive and therefore rarely feasible on resource-limited sensor nodes.

Savvides et al. described methods to overcome these problems in [14]. For example, an iterative multilateration algorithm starts by estimating the position of the sensor node with the maximum number of beacons using atomic multilateration. When a sensor node estimates its location, it becomes a beacon. This process repeats until the positions of all the nodes that eventually can have three or more beacons are estimated. In a further collaborative multilateration algorithm it is possible to compute the position of sensor nodes with less than three beacons, which normally are not able to estimate its position.

Next, Kwon et al. presented a distributed solution using least squares whereby errors in acoustic measurements can be reduced [15]. Ahmed et al. published a new approach to combine the advantages of absolute and relative localization methods [16]. Moreover, Karalar et al. developed a low-energy system for positioning using least squares which can be integrated on individual sensor nodes [17]. A general overview of distributed positioning systems is given by Langendoen and Reijersin [18].

We demand exact localization methods that work on tiny sensor nodes with high limited energy resources. To achieve this, we transfer the complex calculations such as matrix multiplication or matrix inversion to the base station which can be e.g. a powerful desktop computer or a more energy-rich node in the network. Consequently, only simple calculations have to be executed on the sensor nodes. Additionally, we reduce the communication and memory requirements through optimizations of the proposed algorithm.

3   Background: Determination of a Position with Noisy Observations

3.1   Errors in Distance Measurement

All exact localization methods need distances (lateration) or angles (angulation) to known beacons, which are measured using different techniques. Due to the fact that measuring angles needs additional hardware on the sensor nodes (e.g. an array of antennas), the distance determination is the most common technique on sensor nodes. Dominating methods in geodesy such as laser or image correlation cannot be adapted to sensor networks, due to the limited energy resources. For this reason, only phase shifting of a signal, time of signal flight, and received signal strength measurement are feasible.

Figure 4 shows a characteristical received signal strength measurement over the distance for an 868 MHz Chipcon transceiver. The demand for high precision in measuring will not be achieved because:
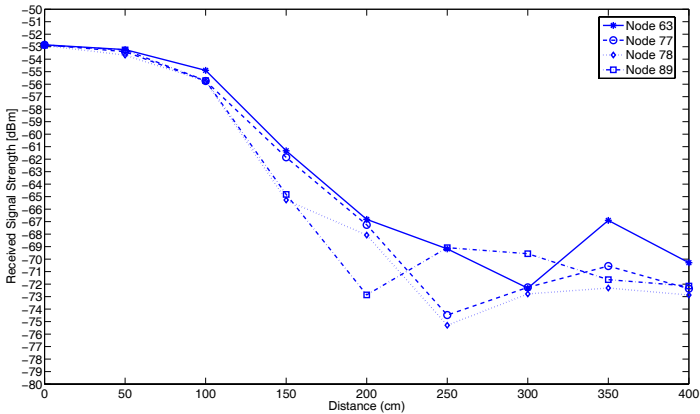
Fig. 4. Indoor received signal strength measurements in the institute's meeting room with our sensor node platform for four different modules

- Reflections of radio waves at obstacles cause multi-path propagation.
- Other electrical fields in the environment interfere with the transmitted radio waves.
- The resolution of the received signal strength indicator can be limited by the transceiver hardware.
- The required line of sight between nodes cannot be guaranteed.
- Node mobility influences the measurement.

Therefore, noisy distances must be considered when developing a robust localization algorithm.

3.2 Building a Model

Estimating the position of an unknown point $P(x, y)$ in two-dimensions requires at least three known points (see Figure 1). In this paper we will confine the explanation to two dimensions. Obviously an extension into the third dimension is possible by introducing the additional coordinate parameter z. This would not have any effect on the following calculation in principle. By using the formula of plane trigonometry the method of intersection will be most adequate, which uses the distances between the sensor nodes and the beacons for the calculation of the unknown coordinates. Due to inevitable measurement errors the observations are stochastic (expressed by σ^2). Therefore, an averaging or a statistically optimal estimate with simultaneous use of all observations by adjustment is feasible. Nevertheless the advantages of using the adjustment approach are:

- Unambiguous, statistically optimal coordinate determination
- Statistical tests to detect gross errors and outliers
- Quality evaluation of the results
  - Unambiguous accuracy prediction for the coordinates
  - Information about the reliability of the results

A detailed explanation of the adjustment is given [19].

Adjustment methods can be described as follows. A number of m observations is given, which may vary according to the character and accuracy: e.g. distance measurements, angles, local coordinates. With these values the $(m, 1)$-observation vector L is given. The u-unknown parameters X have to be determined, e.g. the coordinates for the unknown sensor nodes. These parameters are represented by the vector of the unknowns $\mathbf{X}$. The set of observations L have to be mapped onto the set of unknowns X: $L \rightarrow X$. The observations and the unknowns can be distinguished in:

$$\widetilde{L}, \widetilde{X} \text{ or } E(L), E(X) : \text{true, theoretical values or the expectation values}$$
$$L^0, X^0 : \text{approximated values, not stochastic}$$
$$\widehat{L}, \widehat{X} : \text{stochastic, estimated values}$$

Due to the fact that there is a very high number of sensor nodes in the WSN, this leads to an overdetermined system, e.g for 5000 nodes and 10% beacons results in a system of equations with $u = 2$ unknowns and $m = 500$ measurements. The cost for the solution is very high and will be described later. We will first explain two methods to set up this system of equations.

3.3 Linearizing the Euclidean Distances

In reality, the relationships between the observations r_i and the unknowns x are non-linear and cannot be described with the linear models of the adjustment. Regression functions will be used for non-linear parameters. In order to apply the estimations, the non-linear approaches must be linearized. In our case the system of equations is given by the Euclidean Distances:

$$(x - x_i)^2 + (y - y_i)^2 = r_i^2 \quad (i = 1, 2, \ldots, m). \tag{1}$$

This system of equations must be linearized with either Taylor series [20] or a linearization tool [21]. The two different methods will be described in the following.

3.4   Taylor Series

Linearizing a non-linear function with Taylor series is established as a standard technique. This method is precise and simple to implement in software. Taylor series assume that every value of a function f at a point X^0+x can be represented by a series expansion in a Taylor series. Usually, the approximation is precise enough when truncating after the first element. Taylor series work well if the function f_0 is known in X^0 and a sufficient uniform behavior of a curve is given or the improvement of X is small.

Function $f(X)$ holds at the approximated value X^0 the value $f(X^0)$. A better approximation for the searched value is achieved, if the tangent at the approximated value $[X^0, f(X^0)]$ is determined, which is equal to the first derivation. At the point $X^0 + x$ this method is repeated iteratively. If the approximated values X^0 are known, every non-linear function can be represented by a Taylor series.

$$f\left(X^0 + x\right) = f\left(X^0\right) + \left(\frac{\delta f}{\delta X}\right)_{X=X^0} \cdot x + \frac{1}{2}\left(\frac{\delta^2 f}{\delta X^2}\right)_{X=X^0} \cdot x^2 + O^3 \qquad (2)$$

Following we describe a second linearization technique.

3.5   Linearization Tool

Although the linearization tool is not as exact as the Taylor series, it requires no mathematical differentiation. It works without a start value, and it is suitable for a distributed implementation (discussed later in Section 5). Thus, we use the j'th equation of (1) as a linearization tool. By adding and subtracting x_j and y_j to all other equations this leads to:

$$(x - x_j + x_j - x_i)^2 + (y - y_j + y_j - y_i)^2 = r_i^2 \ (i = 1, 2, \ldots, j-1, j+1, \ldots, m). \qquad (3)$$

With the distance r_j (r_i) that is the distance between the unknown point and the j'th (i'th) beacon and the distance d_{ij} that is the distance between beacon B_i and B_j this leads, after resolving and simplifying, to:

$$(x - x_j)(x_i - x_j) + (y - y_j)(y_i - y_j) = \frac{1}{2}\left[r_j^2 - r_i^2 + d_{ij}^2\right] = b_{ij}. \qquad (4)$$

Because it is not important which equation we use as a linearization tool, $j = 1$ is sufficient. This is equal to choosing the first beacon and if $i = 2, 3, \ldots, m$ this leads to a linear system of equations with $m - 1$ equations and $u = 2$ unknowns.

$$
\begin{aligned}
(x - x_1)(x_2 - x_1) + (y - y_1)(y_2 - y_1) &= \tfrac{1}{2}\left[r_1^2 - r_2^2 + d_{21}^2\right] = b_{21} \\
(x - x_1)(x_3 - x_1) + (y - y_1)(y_3 - y_1) &= \tfrac{1}{2}\left[r_1^2 - r_3^2 + d_{31}^2\right] = b_{31} \\
&\ \ \vdots \\
(x - x_1)(x_m - x_1) + (y - y_1)(y_m - y_1) &= \tfrac{1}{2}\left[r_1^2 - r_m^2 + d_{m1}^2\right] = b_{m1}
\end{aligned}
\qquad (5)
$$

This system of equations can be written in matrix form as:

$$A\mathbf{x} = \mathbf{b} \tag{6}$$

with:

$$A = \begin{pmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \\ \vdots & \vdots \\ x_m - x_1 & y_m - y_1 \end{pmatrix}, \mathbf{x} = \begin{pmatrix} x - x_1 \\ y - y_1 \end{pmatrix}, \mathbf{b} = \begin{pmatrix} b_{21} \\ b_{31} \\ \vdots \\ b_{m1} \end{pmatrix}. \tag{7}$$

This is the basic linear form that now has to be solved using the linear least squares method.

3.6 Solving the Linear Least Square Problem

Due to the fact that overdetermined systems of equations with $m >> n$ have not exactly one solution for $A\mathbf{x} = \mathbf{b}$, we have to apply the L2-norm [19]. This is also called the Euclidean Norm, which minimizes the sum of the squares:

$$\begin{matrix} Minimize \\ x \in \Re^n \end{matrix} ||A\mathbf{x} - \mathbf{b}||^2. \tag{8}$$

To summarize, linear systems of equations can be solved iteratively using splitting techniques or directly either with the normal equations or by orthogonal factorization. Existing techniques are numerous but often the differences between them are small. For this reason, we focus our studies on solving the normal equations. For all others we recommend [22].

A trivial solution of the least squares problem is to reconvert after $\mathbf{x}$. In this case, the unique solution of $A\mathbf{x} \approx \mathbf{b}$ is given by:

$$||A\mathbf{x} - \mathbf{b}||^2 \rightarrow A^T A \mathbf{x} = A^T \mathbf{b}. \tag{9}$$

$$\mathbf{x} = \left(A^T A\right)^{-1} A^T \mathbf{b}. \tag{10}$$

Solving normal equations is a good choice if the linear system has many more equations than unknowns, i.e. $m >> n$, because after the multiplication $A^T A$ the result is only a quadratic $[n \times n]$-matrix, for one unknown point $n = 2$. This reduces the computation and makes it easy to implement it in software. Next, we will study the described methods in detail.

4 Analysis

4.1 Influence of Noisy Distances to the Linearization Method

We described in Section 3.1 that in reality distances are always influenced by measurement errors. Thus, we studied the influence of an increasing distance error to both linearization methods. To realize this, we replaced the exact distances with random values out of a Gaussian distribution $r_{stochastic} \sim \mathcal{N}(\mu_{exact}, \sigma^2)$.
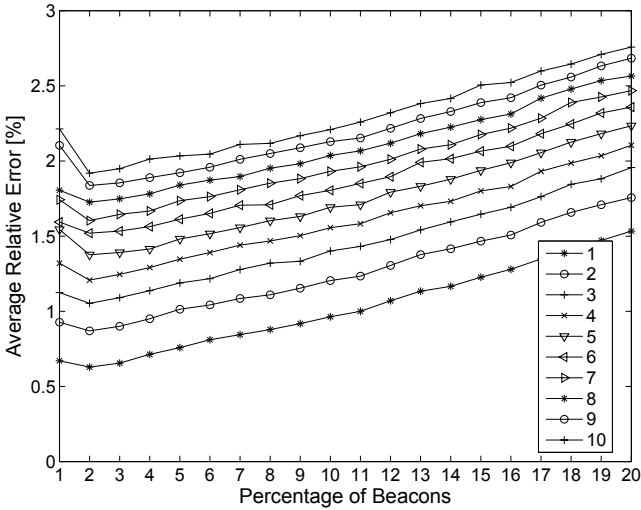
Fig. 5. Localization error over an increasing number of beacons with noisy distances. The non-linear system of equations was linearized with a linearization tool and solved with the linear least squares method. We estimated the positions of 500 sensor nodes in a field with the dimension 100×100.
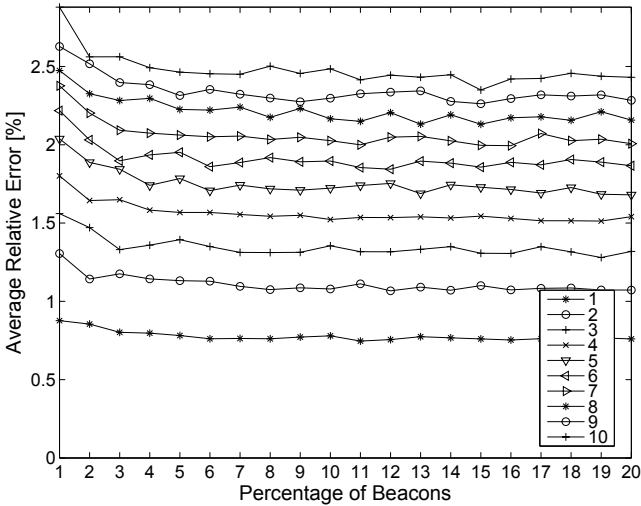
Fig. 6. Localization error with the same simulation settings like in the Figure above but with Taylor linearization

We determined the positions of 500 nodes with an increasing number of beacons to analyze the influence of the graduation. For this, we linearized the system of equations either with Taylor series or a linearization tool, solved the linear

system of equations with the linear least squares method and determined the error, i.e. the distance between the exact position and the calculated position. We repeated every series 50 times and averaged the results to avoid a strong influence of outliers. All simulations run with a test field of the size 100×100 where all nodes were placed in a uniform distribution.

The non-linear system of equations was linearized in Figure 5 with a linearization tool and in Figure 6 with Taylor series. The relative localization errors in both graphics are strongly influenced by an increasing variance. Here we can see the disadvantage of a linearization tool, because with many beacons the error increases. However, we are interested to equip as few nodes as possible with additional hardware. Thus, there exist a trade off between high precision and number of beacons. Nevertheless, we focus in this paper on the postulation of our algorithm. Summarized, Taylor series is more precise and the precision can be advanced with more observations. In contrast the distribution of Taylor series needs approximate values and the calculation requires more computation cost.

4.2   Complexity of the Normal Equations

In the following, we will analyze the complexity of the least squares method by solving the normal equations. Although the literature offers numerous specifications, later we will need the details of the calculation to reduce specific parts of it. In order to define the complexity mathematically, we count the number of floating point operations (flops), which is a commonly used method in literature. The required number of computation cycles strongly depends on the used hardware. Therefore, we count for every operation one flop whether it is an addition, subtraction, multiplication or division[1]. At this stage, we do not consider copying-operations in the memory, because this operation depends on the individual implementation of the algorithm.

As before, we will confine the explanation to two dimensions. Due to the linearization with a linearization tool the matrix A and the vector $\mathbf{b}$ have $(m-1)$-rows. For a clearer understanding we calculate with k-rows and substitute at the end: $m = k + 1$.

The linear system of equations:

$$\mathbf{x} = \left(A^T A\right)^{-1} A^T \mathbf{b} \tag{11}$$

has to be solved. We divide the calculation into the following complexities.

1. Multiplying the $[n \times k]$-matrix A^T with the $[k \times n]$-matrix A leads to $A^T A$ by $\frac{n(n+1)}{2}$ flops[2].
2. This $[n \times n]$-matrix $A^T A$ must be inverted[3] to $(A^T A)^{-1}$ with a complexity of n^3.

---

[1] It should be noted that in the arithmetic unit of a processor a division is a more complex operation than an addition. We will focus on theoretical analysis.

[2] Some operations can be saved by multiplying a transposed matrix with itself.

[3] The inversion of a matrix is very complex with n^3 flops (see [23]).

3. This $[n \times n]$-matrix $(A^T A)^{-1}$ must be multiplied with the $[n \times k]$-matrix A^T, which costs $2n^2 k - nk$ flops. This leads to the precalculated Matrix $A_p = (A^T A)^{-1} A^T$.
4. The matrix A_p must be multiplied with the k-vector $\mathbf{b}$. This step has a complexity of $2kn - n$ flops.
5. The calculation of $\mathbf{b}$ needs $5k + 1$ flops.

With $k = m - 1$ this leads to a total complexity of $15m - 5$ for the least squares method with m beacons and $n = 2$ unknowns. This means, with 100 beacons and the summation of x_1 and y_1 (see (7)) we need 1497 floating point operations. In the next section we will reduce this complexity.

5 Distributing the Least Squares Method

5.1 Topology and Nodes

For optimal self organization wireless sensor networks are structured hierarchically. As already introduced, there are many sensor nodes, some beacons, and a minimum of one base station to which the sensed data is transmitted. Beside resource limited sensor nodes, the beacons and especially the base station are more powerful with more battery capacity. The localization algorithm can be computed centrally on the base station only, shared between the base station and the sensor nodes or completely distributed, being calculated on the sensor nodes only. Since completely central computation results in high network traffic and completely distributed computation exhausts the sensor nodes battery, we decided to develop a hybrid computation of the algorithm. In this hybrid case, the algorithms computation steps are split in parts being distributed over the WSN. There are three methods to distribute the computation load for the localization algorithm:

1. Computation takes place at the base station only.
2. Computation is shared between base station and the sensor nodes.
3. Computation is done at the sensor nodes only.

5.2 Idea and Algorithm Description

Linearizing non-linear equations with a linearization tool has a significant advantage. All elements in matrix A are beacon positions $B_1(x, y)..B_m(x, y)$ only. Moreover, vector $\mathbf{b}$ consists of distances between the unknown sensor node and all beacons $r_1..r_m$ and distances $d_{21}..d_{m1}$ between the first beacon and all others. Consequently, we split the complex computation into two parts - a less complex and a very simple part.

So far, in sensor networks it is desired to execute the entire localization algorithm on every node; completely distributed. With this assumption, the precalculation of a least squares method would be exactly the same on every sensor
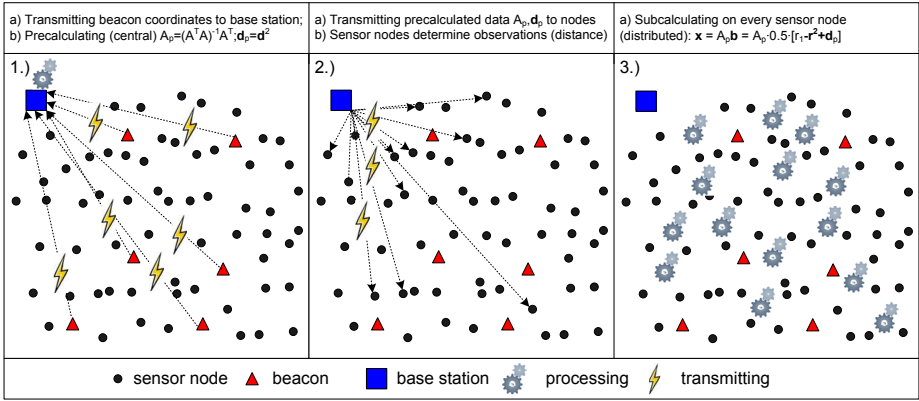
Fig. 7. Schematic of the distributed localization algorithm

node, because matrix A_p is the same for every sensor node in a static network. This wastes limited resources and produces high redundancy.

Now, with the distributed approach, the high-performance base station executes the complex precalculation and the resource-aware sensor nodes estimate their own position with the simple subcalculation. In this subcalculation all sensor nodes use the same precalculated information combined with their individual measured distances to every beacon. Before we describe the method in detail, we will postulate the entire algorithm. Figure 7 was created to visualize the following algorithm phases:

1. **Initialization Phase:**
   – All beacons send their position $B(x, y)$ to the base station.
2. **Complex Precalculation Phase (central):**
   – Base station builds matrix A and vector $\mathbf{d}$.
   – Starting the complex precalculation of A_p,$\mathbf{d}_p$.
3. **Communication Phase (distributed):**
   – Base station sends the precalculated matrix A_p and the vector of distances $\mathbf{d}_p$ to all sensor nodes.
4. **Simple Subcalculation Phase (distributed):**
   – Sensor nodes determine the distance to every beacon $r_1..r_m$.
   – Sensor nodes receive the precalculated matrix A_p and vector $\mathbf{d}_p$, built vector $\mathbf{b}$ and estimate their own position P_{est} autonomously.
5. **Communication Phase (distributed)**
   – Sensor nodes send the calculated position back together with the measured signals for quality investigation and outliers tests.

In the next section, we will analyze the performance of the new algorithm regarding complexity, communication and memory effort.

5.3   Performance of the Algorithm

Complexity: In Section 4.2 we already analyzed the complexity of the least squares method. At this point it is important to know what we can save on the sensor nodes without complex precalculations regarding only the remaining subcalculation.

We assume the constellation of the network in Figure 7 with sensor nodes, beacons, and a base station. On the basis of (11), the base station precalculates $A_p = (A^T A)^{-1} A^T$ and $\mathbf{d}_p = \mathbf{d}^2$. The matrix A_p and vector $\mathbf{d}_p$ are sent to all sensor nodes. Together with the distances $\mathbf{r}$ to all beacons, which every sensor node must determine itself, the subcalculation starts:

$$\mathbf{x} = A_p \frac{1}{2}(r_1^2 - \mathbf{r}^2 + \mathbf{d}_p) \qquad (12)$$

This computation requires $8m - 11$ flops, which leads with 100 beacons and the summation of x_1 and y_1 (see (7)) to 791 flops.

Communication Effort: As already described, communicating between sensor nodes is critical and must be minimized. Particularly, sending data over long distances stresses the energy capacity of sensor nodes. Communication between base station and beacons is less critical and must be preferred if possible. Therefore, we classify communication in two phases. In an uncritical phase, all beacons send their positions to the base station. This causes no energy loss on the sensor nodes. Additionally, in a critical phase the base station sends precalculated information to the sensor nodes that have, in theory, to receive only. Practically, transmitting/sending is never lossless, due to errors in the transmission channel and protocols that require acknowledge packets etc. Furthermore, the base station cannot reach every sensor node in one hop, which demands multi-hopping over some nodes.

Here, we focus on a theoretical examination of the algorithm that is, for the moment, independent of protocol definitions and media access operations. Hence, every sensor node must receive the precalculated matrix A_p and vector $\mathbf{d}_p$ with $[n \cdot (m - 1) + (m - 1)]$ elements. This results in receiving $(3m - 3)$ elements, which are 1188 bytes with 100 beacons and floating point representation of every element.[4]

In contrast, in the completely distributed case every beacon would send its coordinates directly to all sensor nodes. Thus, every sensor node has to receive $2(m)$ elements for x and y. In comparison to our algorithm this would result in 800 bytes, which is not much less.

Memory Considerations: The reduced calculations must be feasible on sensor nodes with a very small memory, mostly not more than a few kilobyte RAM. In our case, the memory consuming operation is always $A_p \cdot \frac{1}{2} \cdot (r_1^2 - \mathbf{r}^2 + \mathbf{d}_p)$. In the worst case A_p and $\mathbf{r}$ plus $\mathbf{d}_p$ must be stored temporarily in memory before the execution on the sensor node can start. In more detail, $[2 \cdot (m - 1)] + (m - 1) +$

---

[4] On common microcontrollers, that are presently integrated on sensor node platforms, every element is stored in floating point representation as a 4 byte number.
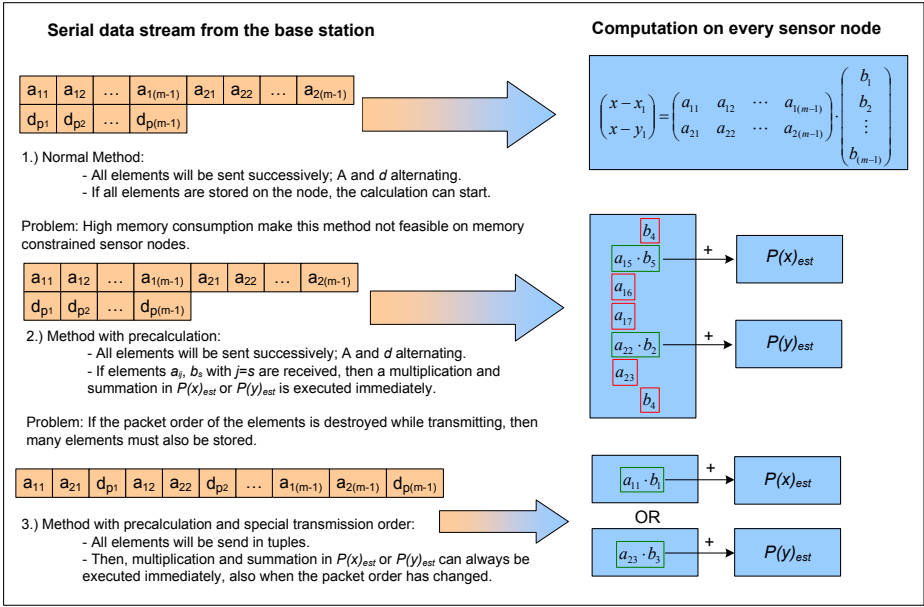
Fig. 8. Three different methods to execute the subcomputation, beginning from the normal method and with optimizations

$(m − 1) = (4m − 4)$ elements must be stored. In floating point representation with $m = 100$ beacons, already 0.796 kb must be allocated for localization only. Normally, the localization task is part of the middleware that has to execute many more tasks. Besides, temporary variables are needed which increase the memory consumption. Given these facts, we studied the critical operations in more detail.

Optimizations: In reality, input data for sensor nodes arrive in packets and will be disassembled into a serial data stream. Due to problems in the transmission channel (e.g. different paths or transmission errors) a sorted order of the incoming packets cannot be guaranteed. The data can arrive in an unsorted form and the calculation begins after receiving all data.

However, the reduced calculation has a further useful quality. Individual calculations of $A_p \cdot \mathbf{b}$ can be executed after the arrival of only some elements without collecting all data. Only one accumulator for the position $P_{est}(x)$ and one for the position $P_{est}(y)$ of the sensor node is needed.

With a_{ij} $(i = 1..2, j = 1..m − 1)$ and b_s $(s = 1..m − 1)$ the following assumptions can be made. If elements with $j = s$ are available, an immediate multiplication of $a_{ij} \cdot b_s$ and a subsequent accumulation in $P_{est}(x, y)$ is possible. The index i distinguishes into which accumulator it must be written; $P_{est}(x)$ at $i = 1$ or $P_{est}(y)$ at $i = 2$. Finally, the optimized reduced calculation requires a worst-case calculation time of $(m − 1)/2$ if the elements arrive in reverse order.

To avoid the case of unsorted data it is also possible to send the elements a_{ij}, d_s in appropriate tuples, e.g. $a_{11}, d_1; a_{12}, d_2; \ldots; a_{ij}, d_s$. In the best case, space in memory has then to be reserved for only a few temporary variables and two accumulators, which reduces memory consumption to a minimum. For a clearer understanding, we illustrate all three methods in Figure 8.

By applying the last optimization the algorithm is also memory aware and finally allows implementation on resource constrained sensor nodes.

6   Conclusion

We presented a distributed localization algorithm that allows precise localization on resource limited sensor nodes. Generally, precise localization is achieved by solving an overdetermined system of equations with the least squares method, which is formed by a multiple trilateration with many beacons and distances. Although the complexity of this solution is relatively high, we showed a method to split the linear least squares method into a complex part, precalculated on the high-performance base station, and a very simple subcalculation on every sensor node. Instead of linearizing the non-linear system of equations with the traditional Taylor series we used a linearization tool that helped to bring the system in the right matrix form. With this approach and based on a network containing 100 beacons we achieved 47.16% savings in computation on every sensor node in comparison to the complete calculation. Moreover, we achieved this with not more communication effort and very little memory consumption with our proposed optimizations. We are currently implementing the algorithm in the sensor network simulation framework J-Sim as well as on our sensor node platform to allow tests under more realistic condistions.

Acknowledgment

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: A survey. Computer Networks **38** (2002) 393–422
2. Min, R., Bhardwaj, M., Cho, S., Sinha, A., Shih, E., Wang, A., Chandrakasan, A.: Low-power wireless sensor networks. In: International Conference on VLSI Design. (2001)
3. Bill, R., Cap, C., Kohfahl, M., Mund, T.: Indoor and outdoor positioning in mobile environments a review and some investigations on wlan positioning. Geographic Information Sciences **10** (2004) 91–98
4. Gibson, J.: The mobile communications handbook. CRC Press (1996)
5. Alouini, M.: Global Positioning System: An Overview. California Institute of Technology (1996)

6. Stefanidis, A., Nittel, S.: GeoSensor Networks. CRC Press (2004)
7. Bulusu, N.: Gps-less low cost outdoor localization for very small devices. IEEE Personal Communications Magazine **7** (2000) 28–34
8. Delaunay, B.: Sur la sphere vide. In: Bulletin der Akademie der Wissenschaften der UdSSR. (1934) 793–800
9. Mirante, A., Weingarten, N.: The radial sweep algorithm for constructing triangulated irregular networks. In: IEEE Computer Graphics and Applications. (1982) 11–21
10. Tian, H., Chengdu, H., Brian, B.M., John, S.A., Tarek, A.: Range-free localization schemes for large scale sensor networks. In: 9th annual international conference on Mobile computing and networking. (2003)
11. Savarese, C., Rabaey, J., Langendoen, K.: Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In: USENIX Annual Tech. Conf. (Monterey, CA, June 2002). (2002) 317–327
12. Capkun, S., Hamdi, M., Hubaux, J.P.: Gps-free positioning in mobile ad hoc networks. Cluster Computing **5** (2002) 157–167
13. Blumenthal, J., Reichenbach, F., Timmermann, D.: Precise positioning with a low complexity algorithm in ad hoc wireless sensor networks. PIK - Praxis der Informationsverarbeitung und Kommunikation **28** (2005) 80–85
14. Savvides, A., Han, C.C., Srivastava, M.B.: Dynamic fine grained localization in ad-hoc networks of sensors. In: 7th ACM MobiCom (Rome, Italy,2001). (2001) 166–179
15. Kwon, Y., Mechitov, K., Sundresh, S., Kim, W., Agha, G.: Resilient localization for sensor networks in outdoor environments. In: 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05). (2005) 643–652
16. Ahmed, A.A., Shi, H., Shang, Y.: Sharp: A new approach to relative localization in wireless sensor networks. In: Second International Workshop on Wireless Ad Hoc Networking (WWAN) ICDCSW'05). (2005) 892–898
17. Karalar, T.C., Yamashita, S., Sheets, M., Rabaey, J.: An integrated, low power localization system for sensor networks. In: First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04). (2004) 24–30
18. Langendoen, K., Reijers, N.: Distributed localization in wireless sensor networks: A quantitative comparison. Computer Networks (Elsevier), special issue on Wireless Sensor Networks **43** (2003) 499–518
19. Wolf, P.R., Ghilani, C.D.: Adjustment Computations: Statistics and Least Squares in Surveying and GIS (Surveying & Boundary Control). Wiley-Interscience (1997)
20. Farebrother, R.: Fitting Linear Relationships. Springer (1998)
21. Murphy, W.S., Hereman, W.: Determination of a position in three dimensions using trilateration and approximate distances. (1999)
22. Lawson, C.L., Hanson, R.: Solving Least Squares Problems. Englewood Cliffs, NJ: Prentice-Hall (1974)
23. Golub, G.H., Loan, C.F.V.: Matrix Computations. The Johns Hopkins University Press (1996)

The Spatial Dimensions of Multi-Criteria Evaluation – Case Study of a Home Buyer's Spatial Decision Support System

Claus Rinner and Aaron Heppleston

Department of Geography, Ryerson University, 350 Victoria Street,
Toronto ON M5B 2K3, Canada
`crinner@ryerson.ca`
`rinner@geog.utoronto.ca`

Abstract. This paper explores the spatial aspects of GIS-based multi-criteria evaluation. We provide a systematic account of geographically defined decision criteria based on three classes of spatial relations: location, proximity, and direction. We also discuss whether the evaluation score of a decision alternative should be directly influenced by neighbouring scores and outline a methodology for distance-based adjustment of evaluation scores. A home buyer case study is employed to demonstrate how spatial criteria can be included in a spatial decision support system and to investigate the effect of geographically adjusting the evaluation scores of decision alternatives. The case study demonstrates how spatial criteria can be presented to decision-makers and their effects be observed in the decision outcome. Further, the spatial adjustment of evaluation scores using the performance of neighbouring properties smoothes the distribution of scores across the study area and allows decision-makers to consider a location's environment.

Keywords: Multi-Criteria Evaluation, Residential Real-Estate Choice, Spatial Decision Support Systems, Spatial Relations.

1 Introduction

When faced with important decisions, humans try to base their decision-making on a rational framework which often includes multiple decision criteria. For example, buying a home is the most important financial decision in life for many families. Home buyers will take more than just the listing price into account, as they will also consider criteria such as property size, number of bedrooms, appearance of the neighbourhood, and proximity to transportation, schools, and recreational areas. A rational approach to decision-making requires rules for the aggregation of these multiple criteria into an evaluation score for each decision alternative.

Multi-criteria evaluation (MCE) was first introduced to the Geographic Information Systems (GIS) discipline about 15 years ago (Janssen and Rietveld 1990, Carver 1991; Malczewski 1999, Thill 1999). Researchers have used the geographic location of decision alternatives for cartographic presentation of evaluation results. But no systematic account of geographically defined decision criteria has been given nor have MCE methods been suggested that explicitly account for spatial variation in

the method's parameters or in user preferences. In this paper, we focus on these spatial dimensions of MCE methods. We provide a framework for classifying decision criteria that are created based on spatial relations and suggest methods to include them as spatial criteria in GIS-based decision support. We also suggest including spatial aspects in the multi-criteria decision rules to address the question whether the evaluation score of a decision alternative should be influenced by neighbouring locations.

We will first review the literature on GIS-based MCE with respect to the treatment of spatial aspects in the MCE process (section 2). Classes of geographically defined decision criteria are proposed in section 3. Next, we suggest using post-hoc geographic weighting to modify the calculation of evaluation scores with inverse distance-based weights to adjust the suitability of locations to their neighbours' suitability values (section 4). The use of spatial decision criteria and a geographically adjusted decision rule is illustrated in a site selection case study with a multi-criteria decision support system for home buyers (section 5). The concluding section 6 discusses the results and outlines further research questions and potential applications of the proposed methods.

2   Spatial Aspects in GIS-Based Multi-Criteria Evaluation

Over the last 15 years, a number of multi-criteria evaluation methods have been implemented in GIS including Boolean overlay operations for conjunctive (AND) and disjunctive (OR) screening of feasible alternatives (Eastman 1997), weighted linear combination or simple additive weighting (Janssen and Rietveld 1990, Eastman 1997), ideal point methods (Carver 1991, Jankowski 1995, Pereira and Duckstein 1996), concordance analysis (Carver 1991, Joerin et al. 2001), the analytical hierarchy process (Banai 1993, Eastman 1997), and ordered weighted averaging (Jiang and Eastman 2000). In these studies, MCE methods are employed to combine criterion maps (raster model) or feature attributes (vector model) to create evaluation scores for decision alternatives. The spatial references of the criterion values (raster cells, vector features) allow for cartographic display of evaluation results. More recently, an exploratory approach to MCE has been suggested in which interactive maps are used to analyse decision outcomes using the principles of geographic visualization (Jankowski et al. 2001, Rinner and Malczewski 2002, Malczewski and Rinner 2005). No MCE decision rules that include spatial elements have been presented to date.

Specifically spatial aspects discussed in the multi-criteria decision-making literature include geographically distributed decision-makers and decision alternatives, decision objectives relating to geographic objects, and non-uniform weighting across space (van Herwijnen and Rietveld 1999). These authors explain that spatial elements relate to the nature of

   (1)   decision alternatives,
   (2)   objectives, and
   (3)   criterion weights.

(1) Alternatives refer to a choice between various locations. They can be explicitly spatial, implicitly spatial or non-spatial. A decision alternative is explicitly spatial

when it refers to a choice between different locations, implicitly spatial when it has spatial implications on the surrounding region, and it is non-spatial when it has no spatial dimensions. (2) Spatial objectives refer to situations where the objectives relate to geographic objects. Objectives also can be explicitly spatial, implicitly spatial or non-spatial. An objective is explicitly spatial when it relates to how a particular location scores in the evaluation process, implicitly spatial when the level of achievement for an objective is determined by spatial relations, and it is non-spatial when it has no spatial component. (3) Spatial weighting refers to situations where the spatial units are given non-uniform weighting. Feick and Hall (2004) examine the spatial dimensions of multi-criteria weight sensitivity. They test the effect of subjective weighting by decision-makers on the ranking of decision alternatives by mapping the weight sensitivity in order to detect localized variations of decision outcomes.

Decision criteria involved in a specific MCE problem often reflect some of the spatial properties of the decision alternatives, e.g. the slope of terrain in a suitability analysis, or the proximity to existing service facilities in a needs analysis. Rossini (1998) studied the search behaviour of home buyers in Adelaide, Australia and found that locational attributes played a major role in the purchase decision. Over 80 attributes were considered and in addition to price and view as common considerations for most buyers, locational attributes were the number one reason for a purchase. Not only did buyers want a house in a location with particular attributes, such as a beach or hill area, but they also wanted houses that were within a specific distance from their friends or family, city, work and various types of facilities. Zeng and Zhou (2001) develop a real-estate geographic information system (REGIS) to emphasize the significance of spatial factors in residential real-estate decision-making. Their evaluation criteria include attributes relating to the physical environment (slope and aspect, parks and natural reserves, rivers and beaches), amenity and transport (shops and shopping centres, schools and railway stations), pollution and noise (main road pollution and noise, railway noise, aeroplane noise), repayment time (loan, income and transport cost) and potential value increase (price trend). Milla et al. (2005) discuss the role of physical and environmental attributes in a hedonic price model of residential property values. Environmental attributes are defined by the proximity of each location to industrial, educational, and recreational facilities.

Malczewski (1999) proposes a framework for activities in multi-criteria analysis procedures, which includes: defining the decision problem, establishing evaluation criteria and constraints, determining decision alternatives and a decision matrix, applying a decision rule, performing sensitivity analysis, and making a recommendation. Table 1 identifies the spatial aspects of multi-criteria problems that can be differentiated along the activities of the MCE framework. As mentioned before, evaluation criteria and constraints are usually defined spatially in a way that depends on the geographic model. Certain criteria such as "distance to town centre" are explicitly spatial and are usually created using GIS functionality. This paper aims at classifying spatial decision criteria based on spatial relations used to create them. MCE decision rules such as weighted linear combination (or, simple additive weighting) are limited to algebraic combination of numeric attribute values. No MCE

methods have been developed that would include explicit spatial processing. In this paper, we also suggest one such method and discuss its possible uses.

Table 1. Spatial aspects of multi-criteria problems that can be differentiated along the activities of the MCE framework; *italic font* indicates aspects that are addressed in this paper

Activity in MCE framework	Spatial aspect in this activity
Evaluation criteria, constraints	Defined as criterion maps (raster model) or feature attributes (vector model)
	Use of spatial relations to create decision criteria
Decision alternatives	Defined as geographic features or cells
Decision matrix	Map-based exploration and spatial analysis of criterion performance
Decision rule	*Inclusion of spatial aspects in the calculation of multi-criteria decision rules*
Sensitivity analysis	Explore spatial patterns of sensitivity
Recommendation	Mapping of evaluation scores or ranks resulting from numerical calculations

3 Decision Criteria Based on Spatial Relations

The spatial relations between geographic objects are key elements of cartographic display and spatial analysis using GIS. Despite their importance, authors have been slow to develop a comprehensive theory of spatial relations, posing major problems for formal geographic modeling (Egenhofer and Mark 1995) and the construction of intelligent GIS (Egenhofer and Franzosa 1991, Robinson 2000). Creating definitions of spatial relations that satisfy real-world examples is difficult due to the many mathematical, cognitive, linguistic and psychological considerations. However, according to Egenhofer and Franzosa (1991) spatial relations can be grouped into three different categories: "topological relations which are invariant under topological transformations of the reference objects, metric relations in terms of distances and directions, and relations concerning the partial and total order of spatial objects as described by propositions such as *in front of, behind, above* and *below*". Egenhofer and Mark (1995) discuss topology, distance, and direction as elements of their "naïve geography".

We distinguish three classes of spatial relations applicable to multi-criteria decision-making: the *location* of decision alternatives (derived from topological relations), their *proximity* to desirable or undesirable facilities, and the *direction* relation between certain undesirable facilities and the decision alternatives. The following subsections explain these classes in more detail and explain how spatial relationships in each case can be transformed into standardized, numerical criterion values in order to apply regular MCE methods to them. Examples refer to the residential real-estate case study that follows.

3.1 Location as an Evaluation Criterion

Location refers to topological relations such as "is contained in". For example, decision alternatives such as houses for sale are conceptualized as points; point-in-polygon analysis would be used to determine the location of an alternative within a

Table 2. Summary of types of explicitly spatial evaluation criteria and types of geographic target objects, by which they are defined

	Relation to, and type of, target object	Example of target object
Location	Located in desirable area	Desirable school district
	Not located in an "avoid" area	Undesirable city neighbourhood
	Located along desirable line feature	River shoreline
Proximity	Located close to desirable public area	Park
	Located close to desirable public point	School
	Located away from undesirable area/point	Industrial area/facility
	Located close to desirable individual (user-specified) location	Place of work, friend's residence
Direction	Not located in an undesirable direction from a point/area	Direction of starting airplanes from an airport

desired or undesired city neighbourhood or school district. Location can be transformed into a Boolean value when standardizing evaluation criteria. A value of "true" represents locations within a desired, or outside an undesired, area, while "false" represents locations outside a desired, or inside an undesired, area.

3.2 Proximity as an Evaluation Criterion

Proximity to desirable facilities, e.g. parks for a home buyer, represents a benefit criterion, while proximity to undesirable facilities, e.g. industrial sites, represents a cost criterion. Proximity values have to be standardized through a common standardization function such as linear scale transformation, which stretches proximity values from 0 to 1. For a benefit criterion, 0 represents the furthest distance in the dataset, while 1 represents the shortest distance. Non-linear transformations are not discussed here, but can provide a useful alternative.

Proximity of decision alternatives to certain other locations can be further subdivided depending on the nature of the other locations. If proximity to a certain type of (public) locations, such as parks or industrial facilities, is considered, the proximity value will be measured as the distance to the closest such facility in the study area. If proximity to specific (individual) locations, such as the place of work or residence of friends, is considered, proximity will be measured as the distance to this specific location.

3.3 Direction as an Evaluation Criterion

Direction between two locations can play a role in decision-making when one location affects another location and this effect depends on direction. For example, houses near an airport may be affected differently by air traffic noise depending on typical start and landing directions. Houses near a slaughter house can be affected by odours depending on the predominant wind direction. Assuming direction as a cost criterion, it can be standardized by giving binary values of 0 to locations within a sector representing the cardinal direction from the origin, and values of 1 to all other locations.

The examples given in Table 2 show that many spatially explicit evaluation criteria can be modeled in different ways. For example, a home buyer could avoid a crime-ridden city neighbourhood by locating outside its boundaries, or by selecting locations with the greatest distance from that neighbourhood.

4 Post-Hoc Spatial Adjustment of Multi-Criteria Evaluation Scores

In this section we further investigate options for the inclusion of spatial dimensions in multi-criteria decision-making by suggesting the adjustment of evaluation scores with a spatially explicit factor. The fundamental observation that suggests adjusting evaluation scores in this way is the influence of nearby alternatives on the assessment of a decision alternative under consideration. In residential real-estate choice, the "potential" of a neighbourhood plays a role in decision-making but is difficult to quantify. However, differences in the aggregate evaluation of houses may indicate whether this potential for future improvement of the neighbourhood is present or not. In the case study described in the following section, the evaluation score for a house is adjusted using the inverse distance-weighted difference between its evaluation score and the scores of the neighbouring houses.

To achieve the distance-weighted adjustment of evaluation scores, we assume that decision alternatives A_i, $1 <= i <= m$ are characterized by decision criteria c_j, $1 <= j <= n$ through criterion values a_{ij}. Those values have been standardized using score-range transformation for benefit criteria resulting in $x_{ij} = (a_{ij} - a_{min,j}) / (a_{max,j} - a_{min,j})$, or for cost criteria resulting in $x_{ij} = (a_{max,j} - a_{ij}) / (a_{max,j} - a_{min,j})$. Criterion importance weights w_j are applied to the standardized values and aggregated resulting in evaluation scores $s_i = \sum_j w_j * x_{ij}$ for each alternative.

Rinner (2004) proposes a distance-based adjustment after completion of this regular MCE process. The evaluation scores are adjusted by distance-based weights of the form $v_{ik} = (1/1+d_{ik})^D$, $1 <= i,k <= m$. The distance weights v_{ik} range from 1 for i=k (the decision alternative itself) and other alternatives nearby, to close to 0 for alternatives that are located furthest away. When two alternatives i, k, are located close together, the weight represents relatively large mutual influence. If two alternatives are located far away from each other, the distance-based weight gets small, thus reducing the mutual influence. D is an exponential factor used to determine the type of decrease of influence with increasing distance, e.g. D = 2 for quadratic decrease. The geographically adjusted final evaluation scores are calculated by applying standardized distance weights to the original evaluation scores, resulting in $s_i' = (\sum_k v_{ik}*s_k) / (\sum_k v_{ik})$.

This method works in a similar way as spatial smoothing (Anselin 1992, Rushton 2003). However, it differs from smoothing in that it takes into account the score of the location at hand. A theoretical justification for modifying evaluation scores can be derived from Slocum et al. (2005, p. 90) who motivate cartographic pattern simplification with the random nature of measured attribute values.

5   Case Study: Residential Real-Estate Choice in Toronto

A residential real-estate case study has been employed to explore the use of spatial relations in decision criteria and demonstrate a method to geographically adjust the evaluation scores for decision alternatives. We created a home buyer's spatial decision support system (SDSS) using ArcGIS 9.0 and Visual Basic for Applications. A sample dataset from the Municipal Property Assessment Corporation contained attributes for 500 houses located in the west end of the City of Toronto, Ontario. For the purpose of this study, we assumed that the last sales price for each house could be substituted for the asking price.

The home buyer's SDSS provides the user with a toolbar to toggle various display layers and start the decision support wizard. The user can select from a list of twelve non-spatial decision criteria and a list of eleven spatial decision criteria.

Table 3. Non-spatial criteria provided in the home buyer's SDSS

Variable	Description	Unit
Asking Price	The selling price of the house	CAN $
Lot Size	The effective lot size of the house property	Square feet
Year Built	The construction year for the house	Year
Size of Living Space	The total area of the living space in the house	Square feet
Bathrooms	The total number of bathrooms in the house	Number
Bedrooms	Number of bedrooms in the house	Number
Basement Characteristics	The total area of the basement or the total area of the basement that is finished. The user can select a finished basement if it is preferred.	Square feet
Heating Type	The heating type for the house (electric, forced air, variety/gas radiator, hot water and other (e.g. – woodstove))	Nominal (EL, FA, GR, HW and OT)
Parking Space	The number of parking spaces for the house	Number
Air Conditioning	Whether or not the house has air conditioning	Boolean (Y or N)
Pool	Whether or not the house has a pool	Boolean (Y or N)
Fireplace	Whether or not the house has a fireplace	Boolean (Y or N)

The non-spatial criteria describe the physical characteristics of each of the houses (Table 3). They include attributes commonly found in most residential real-estate sales listings (e.g. Multiple Listing Service 2006; HomeLife Real Estate Services 2006). Although some of these criteria , such as lot size and size of living space, refer to geometric house characteristics, they can be considered as non-spatial criteria when houses are conceptualized as points. These attributes have been extracted from the original dataset and are included in the attribute table of the geo-coded house locations.

Spatial criteria include examples for the three types of spatial relations identified before: location, proximity, and direction (Table 4). They describe the neighbourhood environment of properties under consideration. The spatial criteria were generated with ArcGIS 9.0 tools and ancillary geospatial layers from DMTI Spatial Inc. and

Statistics Canada. GIS analysis and modeling tools provide the potential to generate many other spatial criteria.

Table 4. Spatial criteria provided in the home buyer's SDSS

Type of Spatial Relation	Criterion	Description	Unit
Location	Neighbourhood Waterfront	Houses located in specific neighbourhoods. Houses within a waterfront zone.	Boolean Boolean
Proximity	Distance to Public Transit	The straight line distance from a house to the nearest Toronto Transit Commission subway stop.	Metres
	Distance to Highway Exits	The straight line distance from a house to the nearest highway exit.	Metres
	Distance to Schools	The straight line distance from a house to the nearest school.	Metres
	Distance to Hospitals	The straight line distance between a house and the nearest hospital.	Metres
	Distance to Golf Courses	The straight line distance between a house and the nearest golf course.	Metres
	Distance to Parks	The straight line distance between a house and the nearest park.	Metres
	Distance to Downtown	The straight line distance between a house and the boundary of downtown Toronto.	Metres
	Distance from Industrial	The straight line distance between a house and the nearest industrial site.	Metres
Direction	Direction from Airport	Hypothetical flight paths of planes taking off from Pearson Intl. Airport.	Boolean

Two types of location criteria are included in the home buyer's SDSS: neighbourhoods and waterfront zone. Houses in the database are coded with an identifier that corresponds with the neighbourhood in which they are located. A waterfront zone has been defined using a buffer 750 metres from the shore of Lake Ontario. The buffer distance was arbitrarily selected so that an appropriate portion of the houses would be designated as waterfront locations for illustration purposes. A Boolean indicator is used to show which houses are on the waterfront. (The waterfront criterion could also be measured as a proximity criterion and houses could be evaluated based on their actual distance from the shore instead of whether or not they fall within a pre-defined waterfront zone.)

Eight proximity relations are used to provide examples of geographic features a home buyer might wish to be close to or far away from. For each proximity relation, the straight-line distance in metres is measured from a house to the nearest geographic object of a specific type. Distance is measured beforehand and recorded in a column in the property shapefile table to be accessed during run-time.

Two hypothetical flight paths were generated as examples of a direction relation, the third type of spatial criteria included in the home buyer's SDSS. The paths start from a point location representing Toronto's Pearson International Airport and cover
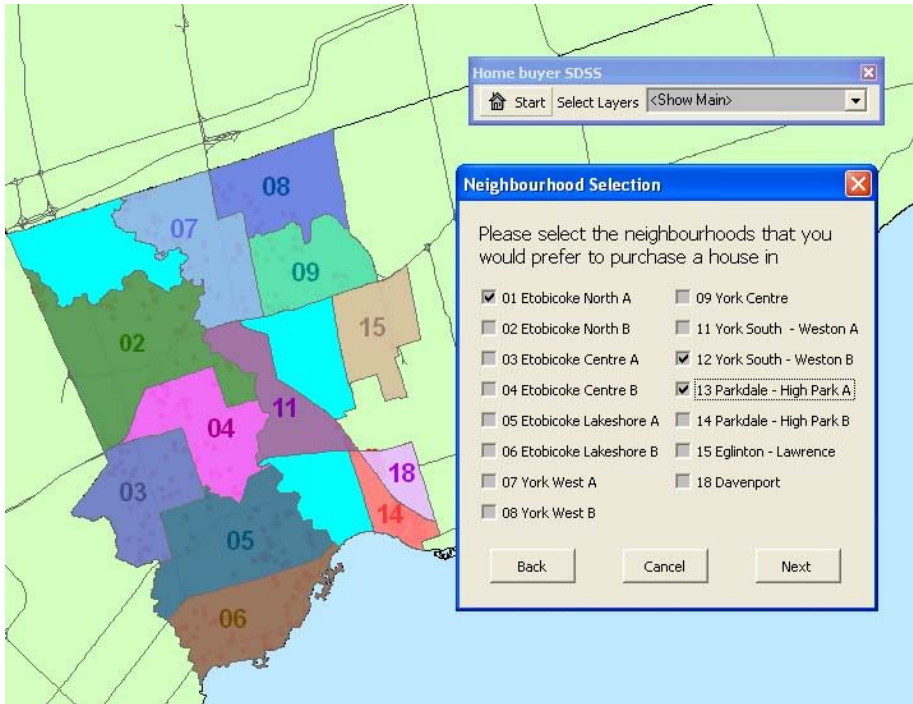
Fig. 1. Neighbourhood criterion selection in the residential real-estate case study

sectors to the northeast and southeast of the airport. Houses lying within each flight path polygon were coded with a Boolean indicator in the property shapefile table.

After the user has finished the selection of non-spatial and spatial criteria, a window opens for each of the spatial criteria. For the neighbourhood criterion, the user can indicate preferred neighbourhoods. They can do this by checking boxes that correspond with each of the neighbourhoods and their selections are displayed on the map (Figure 1). The SDSS then checks the property shapefile table to determine which houses are in the selected neighbourhoods. Houses that are in preferred neighbourhood locations receive a standardized value of 1 for this criterion, while those that are not, receive a value of 0. It is important to note that this criterion does not act as a constraint (or filter) but that it is just one factor in the overall evaluation procedure. If a waterfront location is selected as a criterion, the home buyer's SDSS displays the waterfront zone and highlights waterfront houses on the map (Figure 2). The Boolean values in the property shapefile table are accessed by the SDSS. Each house located within the waterfront boundary receives a standardized value of 1 while those that are not receive a 0.

For each selected proximity criterion, a window is provided for the user to designate whether they would like to maximize or minimize the distance between candidate houses and the target objects corresponding to the criterion. Once the user has indicated the type of criterion (benefit or cost criterion), the pre-calculated distance values are accessed from the property shapefile and the appropriate score range procedure is applied to standardize the values.
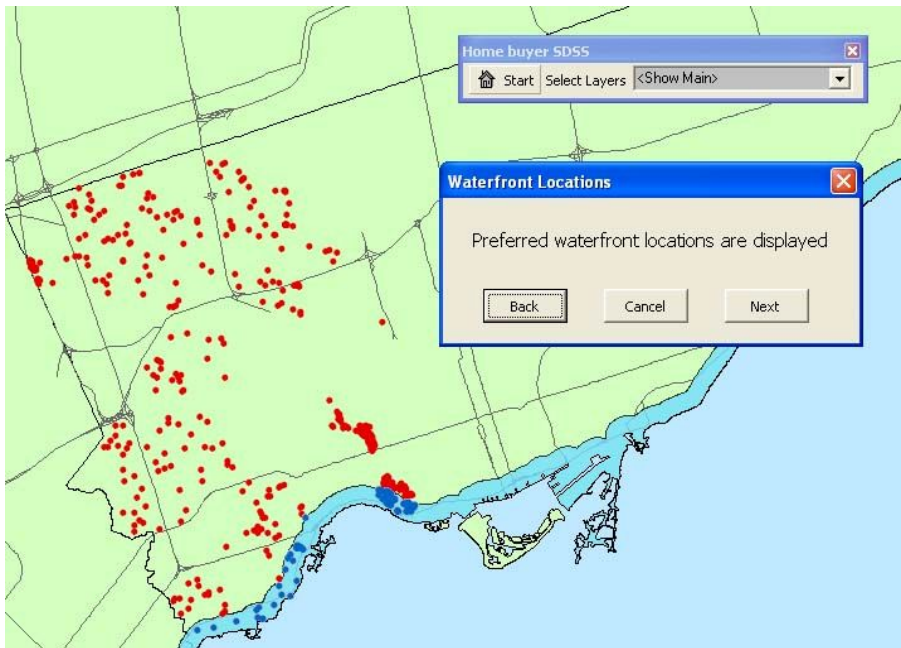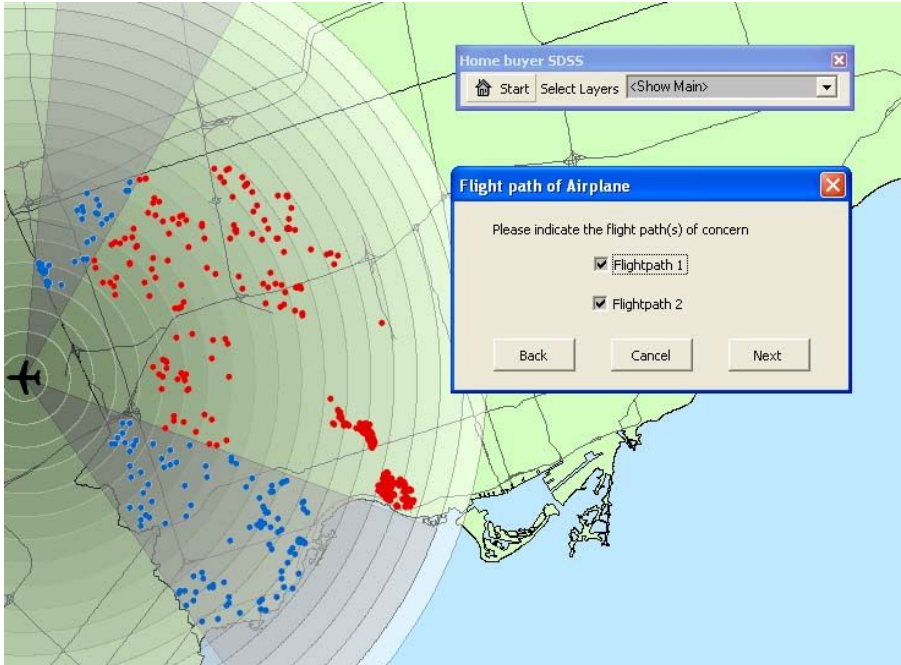
Fig. 2. Waterfront as a location criterion

Fig. 3. Example of a direction criterion

If the direction relation is included as a criterion, the hypothetical flight paths are displayed on the map and houses located within these sectors are highlighted (Figure 3). In the home buyer's SDSS the user can decide whether both flight paths are used or just one. For standardization, houses receive a 0 if they lie within a flight path and a 1 if they do not. This is modelled as a cost criterion because houses located within a flight path would be considered less attractive to buyers due to air plane noise.

After criterion selection and parameter settings, the user assigns preference weights to the criteria using sliders. The sliders are initialized to equal weights and a sum of 100. Sliders are dynamically linked so that a change in the value of one will automatically adjust the value of the others to keep their sum constant. Once the weighting is established, the home buyer can choose to calculate a standard evaluation score using the simple additive weighting (SAW) method, or the spatially adjusted method proposed in Section 4. When calculating the geographically adjusted evaluation score, the home buyer's SDSS accesses a table with pre-calculated distance weights. The table contains the straight-line distance in metres measured between each house and all other houses. First the SAW evaluation score is calculated for each house and then the method for spatial adjustment is applied.

The results for the highest scoring houses under the chosen evaluation method are provided in a display window (Figure 4). The house identifier, evaluation score, neighbourhood, and street address are listed in the upper list box. Figure 4 does not show the addresses from the sample dataset for privacy reasons. The user can select a house and it will be highlighted on the map with a blue house symbol, while a detailed description of its attributes will appear in the textbox below the list. The user also has the option of selecting the 'Highlight All Houses' button, which will display the top ten houses on the map.
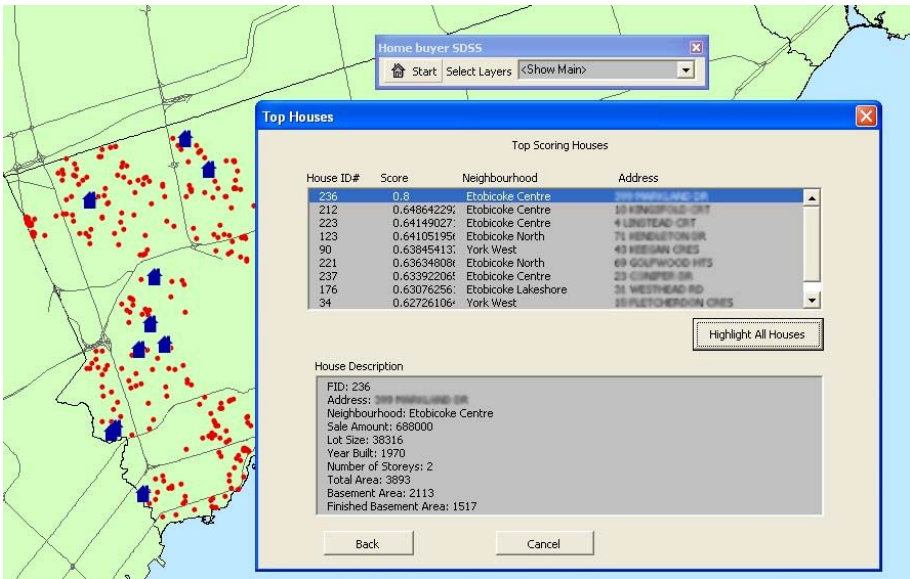

Fig. 4. Display of results in the home buyer's SDSS

The following three scenarios investigate the effectiveness of the implemented MCE methods and explore the influence of spatial relations and geographically weighted adjustment on decision outcomes. The first scenario uses only non-spatial criteria (Table 3) to calculate evaluation scores for each house. The second scenario uses non-spatial and spatial criteria (Tables 3, 4) to generate evaluation scores. It uses the same non-spatial criteria as the first scenario and keeps the weighting proportional to focus on the role spatial relations play in the ranking of decision alternatives through the addition of the spatial criteria. Finally, the third scenario uses spatial adjustment of the evaluation scores from scenarios 1 and 2.

Table 5. Criteria and weights for user scenarios

Criteria	Scenario 1 (non-spatial)	Scenario 2 (mixed)	Scenario 3 (mixed with spatial adjustment)
Asking price	14%	7%	7%
Size of living space	20%	10%	10%
No. of bedrooms	20%	10%	10%
No. of bathrooms	16%	8%	8%
Parking spaces	16%	8%	8%
Air conditioning	14%	7%	7%
Neighbourhood	-	18%	18%
Distance to parks	-	16%	16%
Direction of airport	-	16%	16%

In Scenario 1, houses are assessed by how well they satisfy the hypothetical buyer's non-spatial criteria selection and preference weighting. Houses that rank within the top ten have a low asking price and a high number of bedrooms, bathrooms and parking spaces, as well as air conditioning. The attributes of the houses are the only matter of concern and the home buyer's SDSS determines the best candidates in the study area.

The addition of spatial criteria in Scenario 2 eliminates many of the houses that performed well in the first scenario. This is because their location is not a good one according to the user input. Houses that perform well in Scenario 1 and poorly in Scenario 2 do so because: they are not located in a selected neighbourhood, lie under the flight path of airplanes, are not close to a park, or any combination thereof. Spatial criteria thus add a different layer of suitability measures that go beyond the physical characteristics of individual houses. Through standardization of all criteria, the SDSS balances between satisfying the non-spatial and spatial criteria. Several of the houses that scored well in Scenario 1 reappear in this scenario showing that the user preferences on non-spatial criteria are still being accounted for.

Geographically adjusting the evaluation scores in Scenario 3 affects the ranking of decision alternatives as well. The score for the highest ranking house drops when distance weighting is applied and the range between the scores decreases. This is a result of a spatial smoothing effect as the total variation between scores becomes smaller. Further, the relative ranking of the top ten houses shows significant change, with the ranking of some high scoring houses from Scenario 1 and 2 being altered in Scenario 3. Houses that experience an increase in their ranking do so because they are within close proximity to neighbours that also satisfy user criteria and preference

weighting well. Conversely, houses that fare poorly after their scores are geographically adjusted suffer from the influence of neighbouring houses that did not score well in the evaluation procedure for Scenario 2. In areas with a high density of houses the effect is often cumulative, with the change in one house being the result of all of the neighbours. In low density areas, the geographically weighted score for a house can be affected by only a few houses that have a strong influence due to close proximity, significant difference in score from the house in question, or both. The results demonstrate that geographic adjustment can be effectively used within MCE.

6   Conclusions and Outlook

In this paper, we study the influence of spatial factors at two stages in MCE: geographically explicit decision criteria and geographically adjusted evaluation scores. Building on previous work that used spatial decision criteria, we propose a classification of criteria based on the spatial relations they represent. In a case study, we give examples for all three classes based on location, proximity, and direction relations. Further, we suggest exploring spatial aspects of multi-criteria decision rules. As an example, we propose to adjust final evaluation scores based on neighbouring scores. A distance-based weighting of neighbouring scores was introduced and implemented in the case study of a home buyer's SDSS. Through this proof-of-concept we could demonstrate how spatial criteria can be presented to decision-makers and their effects be observed in the decision outcome. Further, the spatial adjustment of evaluation scores using those of neighbouring properties smoothes the distribution of scores across the study area and allows decision-makers to consider the performance of adjacent properties.

With the increasing popularity of map-based decision support tools explicit handling of spatial aspects of decision problems will likely become of larger interest to the Geographic Information Science community and to GIS and SDSS developers. With respect to spatially explicit decision criteria, more variants of the suggested classes could be explored. For example, decision-makers may want to avoid adjacency to desired facilities in keeping a minimum distance to schools or mall parking lots. Visibility should also be included in the criterion generating process as an additional spatial relation. In terms of distance-weighted adjustment, more research is needed into the interaction with possible spatial association among criterion values. Both proposed methods would benefit from taking into account non-linear scaling of distance-based criteria and adjustment factors. For many criteria, distance should be measured in street distance or travel time rather than straight-line distance. For spatial adjustment, distance could be replaced by another neighbourhood concept and the consequences be studied.

With reference to the case study implementation, the SDSS should be made flexible enough to take any map layer as a criterion. Further, we did not yet include individually defined reference locations for proximity criteria (e.g. work location, friend's residence) which are different for each decision-maker. These are likely to be among the most useful features of a home buyer's SDSS. In order to confirm this hypothesis and examine the utility of the spatially explicit decision support system, we plan to conduct expert interviews with real-estate agents and focus groups with actual home buyers.

Acknowledgements

References

Anselin, L.: Spatial Data Analysis With GIS: An Introduction to Application in the Social Sciences. National Center for Geographic Information and Analysis, Technical Report 92-10 (1992)

Banai, R.: Fuzziness in Geographic Information Systems: Contributions from the Analytic Hierarchy Process. Int J Geogr Inf Syst 7(4) (1993) 315–329

Carver, S.: Integrating Multi-criteria Evaluation with Geographical Information Systems. Int J Geogr Inf Syst 5(3) (1991) 321-339.

Densham, P.: Spatial Decision Support Systems. In: D. Maguire, M. Goodchild, D. Rhind (eds.): Geographical Information Systems: Principles and Applications. Longman, London (1991) 403-411

Eastman, J.R.: IDRISI for Windows, Version 2.0: Tutorial Exercises. Graduate School of Geography, Clark University, Worcester, MA (1997)

Egenhofer, M., Franzosa, R.: Point-Set Topological Relations. Int J Geogr Inf Syst 5(2) (1991) 161-174

Egenhofer, M., Mark, D.: Naive Geography. In: Frank A. and Kuhn W. (eds.): Spatial Information Theory – A Theoretical Basis for GIS (COSIT '95), Semmering, Austria (Lecture Notes in Computer Science, 988). Springer, Berlin (1995) 1-15

Feick, R., Hall, G.B.: A Method for Examining the Spatial Dimension of Multi-criteria Weight Sensitivity. Int J Geogr Inf Science 18(8) (2004) 815-840.

Feick, R.: Toward Localized Forms of GIS-MCDM Analyses. Abstract, Annual Meeting for the Association of American Geographers, Denver, CO (2005)

HomeLife Real Estate Services: Property Search: HomeLifeHigher Standards. Retrieved from: http://www.homelife.ca/buyers-findaproperty.html (28.2.2006)

Jankowski, P.: Integrating Geographical Information Systems and Multiple Criteria Decision Making Methods. Int J Geogr Inf Syst 9(3) (1995) 251-273

Jankowski P., Andrienko, N., Andrienko, G.: Map-Centered Exploratory Approach to Multiple Criteria Spatial Decision Making. Int J Geogr Inf Sci 15(2) (2001) 101-127

Janssen, R., Rietveld, P.: Multicriteria Analysis and Geographical Information Systems: An Application to Agricultural Land Use in the Netherlands. In: Scholten, H.J., Stillwell, J.C.H. (eds): Geographical Information Systems for Urban and Regional Planning. Kluwer, Dordrecht, (1990) 129-139

Jiang, H., Eastman, J.R.: Application of Fuzzy Measures in Multi-Criteria Evaluation in GIS. Int J Geogr Inf Syst 14(2) (2000) 173-184

Joerin, F., Thériault, M., Musy, A.: Using GIS and Outranking Multicriteria Analysis for Landuse Suitability Assessment. Int J Geogr Inf Sci 15(2) (2001) 153-174

Malczewski, J.: GIS and Multi-Criteria Decision Analysis. John Wiley & Sons, New York (1999)

Malczewski, J., Rinner, C.: Exploring Multicriteria Decision Strategies in GIS with Linguistic Quantifiers: A Case Study of Residential Quality Evaluation. J Geograph Syst 7(2) (2005) 249-268

Milla, K., Thomas, M.H., Ansine, W.: Evaluating the Effect of Proximity to Hog Farms on Residential Property Values: A GIS-Based Hedonic Price Model Approach. URISA Journal 17(1) (2005) 27-32

Multiple Listing Service Online: Property Search: Canadian Real Estate Association. Retrieved from: http://www.mls.ca/PropertySearch.aspx (28.2.2006)

Pereira, J.M.C., Duckstein, L.: A Multiple Criteria Decision-Making Approach to GIS-Based Land Suitability Evaluation. Int J Geogr Inf Syst 7(5) (1993) 407-424

Rinner, C., Malczewski, J.: Web-enabled Spatial Decision Analysis Using Ordered Weighted Averaging (OWA). J Geographical Systems 4(4) (2002) 385-403

Rinner, C.: Spatial Dimensions of Multi-Criteria Analysis. Abstract and presentation at the Annual Meeting of the Canadian Association of Geographers (CAG), 25-29 March 2004, Moncton, New Brunswick, Canada (2004)

Robinson, V.B.: Individual and Multipersonal Fuzzy Spatial Relations Acquired Using Human-Machine Interaction. Fuzzy Sets and Systems 113 (2000) 133-145

Rossini, P.: Assessing Buyer Search Behaviour for Residential House Purchasers in Adelaide. Fourth annual Pacific-rim Real Estate Society Conference, Perth, Australia (1998) 19-21

Rushton, G.: Public Health, GIS, and Spatial Analytic Tools. Annu. Rev. Public Health 24 (2003) 43-56

Slocum, T.A., McMaster, R.B., Kessler, F.C., and Howard, H.H.: Thematic Cartography and Geographic Visualization, 2nd edition, Prentice Hall, Upper Saddle River, NJ (2005)

Thill, J.-C. (ed.): Spatial Multicriteria Decision Making and Analysis: A Geographic Information Sciences Approach. Ashgate, New York (1999)

van Herwijnen, M., Rietveld, P.: Spatial Dimensions in Multicriteria Analysis. In: Thill, J.-C. (ed.): Spatial Multicriteria Decision Making and Analysis: A Geographic Information Sciences Approach. Ashgate, New York (1999) 77-99

Zeng, T., Zhou, Q.: Optimal Spatial Decision-Making Using a GIS: A Prototype of a Real Estate Geographical Information System (REGIS). Int J Geogr Inf Sci 15(4) (2001) 307-321

Graph-Based Navigation Strategies for Heterogeneous Spatial Data Sets

Andrea Rodríguez[1] and Francisco Godoy[2]

[1] Department of Computer Science, University of Concepción
Edmundo Larenas 215, 4070409 Concepción, Chile
`andrea@udec.cl`
[2] Center for Oceanographic Research in the Eastern South-Pacific
FONDAP-COPAS, University of Concepción
P.O. Box 160-C, Concepción, Chile
`fgodoyf@udec.cl`

Abstract. Querying heterogeneous spatial databases involves not only characterizing and comparing the information content of several databases, but also *navigating or accessing* the data sets with the query answer. This work proposes a formalism that relates the information content of data sets by three basic types of *correspondence relations*: *data equivalence*, *difference of data omission*, and *difference of data commission*. These correspondence relations define the *information space* over which a navigation process is carried out. Based on a complete or an incomplete information space, this work proposes strategies that optimize the retrieval process of information coming from different databases. The results of this study show the advantages of defining the information space to select and access databases. In particular, strategies that estimate the information contribution of data sets based on correspondence relations outperform a strategy that considers a random list or a list of data sets sorted by size.

1 Introduction

In presence of syntactic, schematic, and semantic differences [15], accessing distributed databases requires an important effort to compare user queries with data stored in diverse databases. Most research in this area has focused on establishing the relationship among databases; that is, similarity, difference, and equivalence between database schemas [8] [7] [12] [17] [9]. A subsequent, but not less important, issue in querying and accessing heterogeneous databases deals with the way users *navigate* among data sets that contain the query answer. Users are typically presented with only a ranked list of databases that totally or partially satisfy their requests and then, they have to decide which of these databases they will access.

This work considers the process of accessing different databases that contain desired data as a problem of information navigation. For example, consider a user who wants to retrieve data of utility networks in a given urban area. These

data may be obtained from different databases, each of them containing partial data of utility networks. The query to each database may result in different data sets (i.e., sets of tuples in relational databases), and users have to decide what data sets and in which sequence should be accessed. We claim that is not only important to know what databases contain the desired data, but to provide with strategies to organize the results of the query so that an efficient retrieval process is accomplished. In this context, efficiency refers to getting all or most information (i.e., new data) with the access to few databases.

The contribution of this work relates to the navigation in an information space composed of heterogeneous spatial data sets. This can be done after database schemas or metadata have been compared and databases that contain the desired type of data have been detected. This work proposes a formalism that allows one to relate spatial databases in terms of the equivalences and differences between their data sets.

We define an information space by characterizing the correspondences between data sets with three basic categories: *data equivalence*, *difference of data omission*, and *difference of data commission*. Using these categories in a graph-base representation, this work describes strategies that optimize the navigation of data sets with the query results. The objective of this optimization is to retrieve data that contribute to the answer, minimizing duplications while increasing the data retrieved. The work concentrates on spatial objects represented by regions. At an abstract level, the same approach may be useful in other domains of information.

The structure of the paper is as follows. Section 2 gives a brief review of related work. Section 3 describes the modeling of the information space with the correspondence relations between data sets and the graph-based representation. Section 4 presents the different strategies for information navigation, and Section 5 shows the implementation of the strategies. Final conclusions and future research directions are given in Section 6.

2   Related Work

In the area of information systems, the concept of information navigation has been associated with the visualization of retrieval results. In a retrieval process with heterogeneous data collections that totally and partially satisfied a user query, the problem becomes to select and browse the query results. The general idea for solving this problem is to use overviews of the information that may guide users in the retrieval process [2]. We consider the process of selecting and browsing documents as a process of information navigation, where two important approaches are:

– Category hierarchies. This approach assigns metadata to data sets based on their categories, which are then organized into a hierarchy. The problem with this strategy is that one could often need to look at a large collection of tags and, when the category has been found, search within the category. Examples of this type of approach are the computer classification system of the ACM

[1], which has developed a hierarchy of 1200 categories or the the popular search site Yahoo [19], which organized documents in many categories.
– Clustering techniques. There are several cluster-based solutions for helping users in searching and navigating data sets. They attempt to display overview information derived from the metadata or the extraction of common features in a collection. Then, clusters group data sets based on the similarity to one another. An early contribution is the paradigm Scatter/Gatter [3], where users are provided with a summary of the documents that have been clustered. More recent works have also considered dynamic clusters [20] and clustering and summary of query results for information navigation [14]. An example of such approaches is the web site VIVISIMO [16], which dynamically organizes the query results by topics.

A second perspective of information navigation relates to the process of accessing heterogeneous data sources. In this context, the navigation is carried out over a semantic structure, called information space, that connects different sources [13]. This work concerns the second perspective of a navigation process. It defines a structure upon which different strategies for selecting the path in the navigation process can be taken.

3   The Information Space

The information space describes the content relations between data sets. These relations are basic correspondence categories upon which a graph representation is defined to be used in the navigation process.

3.1   Types of Correspondences Between Data Sets

Three types of basic correspondence relations between data sets are: *equivalence*, *difference of commission*, and *difference of omission*. The distinction between *difference of commission* and *difference of omission* resembles the notions of *error of commission* and *error of omission*, respectively, associated with types of inaccuracy in an ontology of imperfection for information integration [18]. In such ontology, inaccuracy is the lack of correlation with the actual state of affair. Our notions of *difference of commission* and *difference of omission*, however, do not relate to errors. Commission and omission are here converse concepts related to the presence or absence of data, respectively. Since this work assumes no further information about the origin of data, all data sets are considered to equally contribute to the integration or retrieval of information. Thus, solving conflicting geometric representations (e.g., two different geometries for a same object) is a process carried out after all data are retrieved.
A definition of the correspondence relations follows.

– *Equivalence*. Considering the same thematic layer, two objects are said to be equivalent if they occupied the same space or have the same representative set of coordinates. We relax the definition of equivalence for not totally

equivalent objects by considering that two objects from different databases and a same thematic layer are represented by equivalent and different regions. The equivalent regions are the intersection of the geometric representations of objects, whereas the different regions are the non intersecting representations of objects. This difference between objects is further classified into *difference of commission* and *difference of omission*.

Let x, y, z be regions belonging to objects from different data sets, then the equivalence relation between x and y ($x \equiv y$) satisfies the following basic properties:

$$\forall \ x, y \ [x \equiv y \rightarrow y \equiv x] \ symmetry \tag{1}$$

$$\forall \ x, y, z \ [(x \equiv y \wedge y \equiv z) \rightarrow x \equiv z] \ transitivity \tag{2}$$

We define an *Equivalent Operator* between two data sets A and B ($E(A, B)$) to be the equivalent regions between the data sets. The total area of these equivalent regions will be expressed by $|E(A, B)|$.

– *Difference of commission.* The difference of omission between two data sets A and B ($C(A, B)$) corresponds to the regions that are in A and not in B.
– *Difference of omission.* As the converse relation of $C(A, B)$, the difference of omission between data sets A and B ($O(A, B)$) corresponds to the regions that are in B and not in A.

Figure 1 presents the basic cases of the three types of correspondence between data sets A and B when considering the same spatial window ω. In this case, features a and c are equivalent in both data sets, feature b is in data set A but not in B; that is, there is a difference of commission of feature b between data sets A and B, and a difference of omission of feature b between data sets B and A. Likewise, d is a feature in data set B and not in data set A.
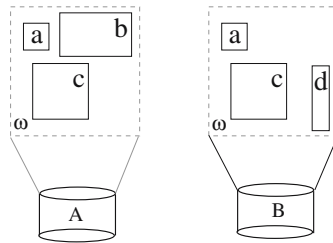
Fig. 1. Example of equivalences and differences between data sets

In the case of partial equivalence and matching of composite objects between two data sets A, B, the operators $|E(A, B)|$, $|C(A, B)|$ and $|O(A, B)|$ define the percentages of areas that are equivalent and different, without indicating how many objects are really equivalent or different. A further study could analyze when an object is equivalent or not, a problem that has been addressed in studies of data integration [18] [6] and consistency at multiple representations [5] [10] [4].

Let a and b be the total areas of objects in data sets A and B, respectively, the correspondence operators satisfy the following properties:

$$a = |E(A, A)| \tag{3}$$

$$a = |E(A, B)| + |C(A, B)| \tag{4}$$

$$a = |E(A, B)| + |O(B, A)| \tag{5}$$

$$b = (a - |O(B, A)|) + |O(A, B)| \tag{6}$$

3.2   Graph-Based Representation of the Information Space

The set of data sets with their correspondence relations can be seen as a directed and labeled graph, with nodes being data sets and edges being the correspondence categories between data sets. The graph can be simplified by using only one directed edge between nodes, since the inverse directed edge can be determined by the converse properties $|C(A, B)| = |O(B, A)|$.

For n data sets, a complete graph is of order $O(n^2)$. Under incomplete information, the composition of correspondence relations allows one to constrain the possible values of unknown information. Thus, a composition of correspondence relations between data sets A and B and between B and C indicates possible values for the correspondence relations between A and C.

Proposition 1. *Given three different data sets A, B, and C, with total areas of objects a, b, and c, respectively, and where known information are the results of the equivalent operator between A and B ($E(A, B)$) and between B and C ($E(B, C)$), the value of $|E(A, C)|$ has lower and upper bounds given by*

$$\forall A, B, C[max(0, |E(A, B)| + |E(B, C)| - b) \leq |E(A, C)| \leq min(a, c)] \tag{7}$$

Proof. A lower bound of equivalent areas between data sets A and C is determined by the minimum area of regions that are in common between regions in $E(A, B)$ and $E(B, C)$. These regions must be also in the equivalent regions between A and C by the transitivity property of equivalence (Equation 2). The minimum area can be obtained from the maximum area of regions that are different between the equivalent regions $E(A, B)$ and $E(B, C)$. In order to have a region in $E(A, B)$ that is not in $E(B, C)$, the equivalent region in $E(A, B)$ must be in $C(B, C)$. Thus, we cannot have more regions that are different between $E(A, B)$ and $C(B, C)$. Once $|C(B, C)|$ is less than $|E(A, B)|$, the lower bound of $|E(A, C)|$ is equal to $|E(A, B)| - |C(B, C)|$. Once $|C(B, C)|$ is larger than $|E(A, B)|$, the lower bound of $|E(A, C)|$ is equal to zero, since it assumes that all equivalent regions between A and B are also in the set of regions that are in B and not in C.

An upper bound of the area of equivalent regions between data sets A and C is given by the minimum area of the space used by objects in each data set, since one cannot have an area of equivalent regions larger than the total area of objects in the data sets.

Corollary 1. *Given that* $|C(A,C)| = a - |E(A,C)|$ *and* $|O(A,C)| = c - |E(A,C)|$, *then*

$$\forall A, B, C [a - min(a,c) \leq |C(A,C)| \leq a - max(0, |E(A,B)| - |C(B,C)|)] \quad (8)$$
$$\forall A, B, C [c - min(a,c) \leq |O(A,C)| \leq c - max(0, |E(A,B)| - |C(B,C)|)] \quad (9)$$

4   Information Navigation

The navigation strategies proposed in this paper aim at retrieving all data with a minimum effort or cost. Effort or cost in this case are considered to be the total amount of retrieved data, measured as the total area of objects retrieved. In such retrieval process, duplication may occur and the idea is to obtain all data, while accessing the minimum number of databases or accessing databases in a sequence that obtains most data as quick as possible. For all data we mean all different regions, since these regions are the real information contribution of data sets. If one of the data sets retrieved from a database contains all regions (objects) in a desired geographic window, the system should just access this database to obtain a complete answer. In an imperfect world, however, databases are incomplete. In such cases, to obtain a complete answer means to access more than one database, but hopefully, not all of them.

4.1   Navigation Strategies

There are different strategies retrieving all data from an information space. To illustrate the different alternatives, consider a simple case of four data sets (coming from different databases) (Figure 2), whose correspondence relations are codified by the triple $(|O()|, E()|, |C()|)$ as labels of the directed graph, and where the total area of objects in each data set is codified by the value in each corresponding node.

Common to all strategies is that they favor data sets with large objects' geometries. It is not the number of objects, but the objects that occupy more space what will guide the navigation strategy.

- *Size-based strategy.* A size-based strategy sorts the data sets based on the total area of objects in the sets, retrieving, for example, the data set in decreasing order of area. With this strategy, all data sets are retrieved. In the case of the example in Figure 2, the navigation sequence is given in Table 1.
- *Forward-based strategy.* A forward-based strategy takes the current position of the navigation (initially, the node of the data set with the largest area) and selects a next data set that has the largest area of commission with respect to the current data set, or inversely, a data set respect to which the current data set has the largest area of omission. Consider the example in Figure 2 and the Table 2 that presents the area of the difference of omission between two data sets. The forward-based strategy takes the largest data
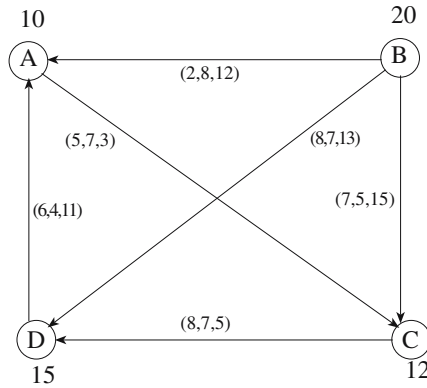
Fig. 2. Complete graph with four data sets

Table 1. An example of navigation with the size-based strategy

Data Set	Total area	Sequence order
A	10	4
B	20	1
C	12	3
D	15	2

Table 2. An example of navigation with the forward-based strategy

$O()$	A	B	C	D
A	0	**12**	5	11
B	2	0	7	**8**
C	3	**15**	0	8
D	6	**13**	5	0

set (in the example, data set B) and from that, it selects the *non previously selected* data set with respect to which the current data set has the largest difference of omission. This strategy avoids retrieving data sets with no new information. In this case, the sequence is given by B, D, A, and C.

– *History-based strategy:* This strategy uses the information about the relationship of already selected data sets to define which data set is the next in the navigation process. The idea behind this strategy is that all previous data sets, and not only the current data set, may also contain part of the data of a non-previously selected data set. So, new information in the navigation process may be less than the data commission of non-selected data sets with respect to the current data set. Like the forwarded-based strategy, history-based strategy avoids accessing data sets with no new information.

Estimating the area of expected new regions based on previously selected data sets assumes that all regions are equally likely to be shared among data

sets. Therefore, when considering any two data sets, the ratio between the area of equivalent objects and the total area in one of the data sets estimates the *likelihood* of a region in this data set of being in the set of equivalent objects.

To illustrate our derivation of the area of expected new regions in a navigation process, consider a case of a non previously selected data set A with area n and two already selected data sets B and C. The graph of the corresponding information space gives the regions that are equivalence between A and B and between A and C, $|E(A, B)|$ and $|E(A, C)|$, respectively. Likewise, the regions in A that are not in B and C, $|C(A, B)|$ and $|C(A, C)|$, respectively, are known. In this case, we distinguish the following cases when deriving the expected number of potential new regions when visiting A ($NEW(A)$):

1. When one of the previously selected data sets contains all regions stored in data set A (Figure 3), no new regions can be retrieved from visiting A:

$$|E(A, B)| = n \vee |E(A, C)| = n \rightarrow NEW(A) = 0$$
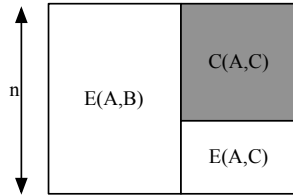

Fig. 3. History-based strategy: a case with no new regions

2. When one of the previously selected data sets contains none of the regions in A, and the other previously selected data set contains a partial set of regions in A (Figure 4), the expected number of new regions when visiting A is equal to the area of regions that *both* previously selected data sets did not contain.

$$|C(A, B)| \neq n \wedge |C(A, C)| = n \rightarrow NEW(A) = |C(A, B)| \vee$$
$$|C(A, B)| = n \wedge |C(A, C)| \neq n \rightarrow NEW(A) = |C(A, C)|$$

3. When $|E(A, B)| = |C(A, B)|$ and $|E(A, C)| = |C(AC)|$ (Figure 5), the expected area of new regions is equal to the half of $|C(A, B)|$ or $|C(A, C)|$, since we consider the same probability for all regions of being equivalent or different between two data sets. In the general case with two previously selected data sets, the estimated area of new regions is:

$$|C(A, B)| \neq n \wedge |C(A, C)| \neq n \rightarrow$$
$$NEW(A) = n \times \left(\frac{|C(A, B)|}{n} \right) \times \left(\frac{|C(A, C)|}{n} \right)$$
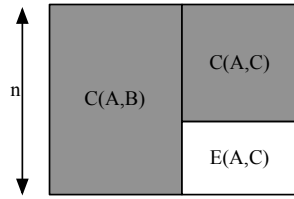$$= \frac{|C(A, B)| \times |C(A, C)|}{n}$$

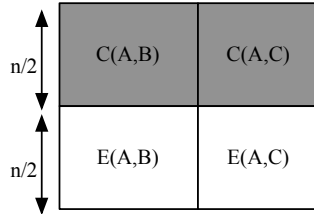**Fig. 4.** History-based strategy: a case with new regions

Fig. 5. History-based strategy: a case with half of new objects

The general expression of the expected new regions in a non-selected data set X of area n, when m data sets $\{Y_1, \ldots, Y_m\}$ have been previously visited is:

$$NEW(X) = \frac{\prod_{i \neq j} |C(Y_i, Y_j)|}{n^{m-1}} \tag{10}$$

In the example of Figure 2, this strategy takes the data set with the largest area and continues accessing the non-selected data sets in the following sequence: B, D, C and A.

4.2   Incomplete Information Space

Since the cost of constructing a complete information space may be very high, this section analyzes the strategies in presence of an incomplete representation of the information space. In this analysis, the total area of objects per data set is known, and the data equivalence, difference of data commission, and difference of data omission may be unknown. The focus of the analysis is on the equivalence, since knowing the data equivalence between data sets and the total area per data set allows us to quantify the differences between data sets.

The size-based strategy is not affected by the incomplete information space. The last two strategies, forward- and history-based strategies, however, need some derivation or approximation. Useful to this approximation are the lower and upper bounds defined by composition of correspondence relations described in Section 2.2. Such an approximation requires a connected information space (connected graph); that is, there exists at least one path between any two nodes in the graph representation. Consider for example the connected and incomplete information space in Figure 6.
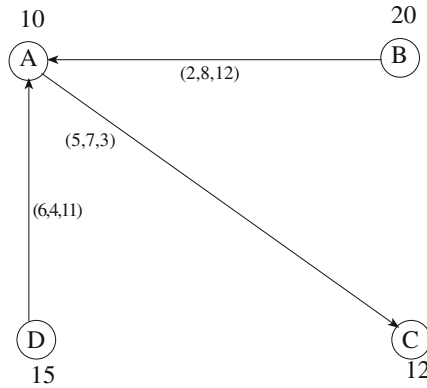
Fig. 6. Example 1: Incomplete and connected information space

In the information space of Figure 6, three edges of correspondence relations are missing: edge between B and C, edge between B and D, and edge between C and D. Based on the Equation 7, the lower and upper bounds of correspondence relations are:

$$5 \leq |E(B,C)| \leq 12 \; 2 \leq |E(B,D)| \leq 15 \; 1 \leq |E(C,D)| \leq 12$$
$$8 \leq |C(B,C)| \leq 15 \; 5 \leq |C(B,D)| \leq 18 \; 0 \leq |C(C,D)| \leq 11$$
$$0 \leq |O(B,C)| \leq 7 \;\; 0 \leq |O(B,D)| \leq 13 \; 3 \leq |O(C,D)| \leq 14$$

These bounds are used in the navigation strategies to approximate the value of non-previously retrieved regions (new regions) that will be obtained when accessing a non-retrieved data set. The *forward-based* strategy takes the difference of omission ($|O()|$) of the last retrieved data set and each of the non-retrived data sets. In an interval of possible values for $|O()|$, all these values are considered to be equally possible, so the strategy uses the media of the interval.

The *history-based* strategy requires the numbers $|E()|$ and $|C()|$ of non-retrieved data sets with respect to previously retrieved data sets. Like the *forward-based* strategy, the *history-based* strategy takes the media values of the derived intervals that bound $|E()|$ and $|C()|$. The resulting approximations are:

$$|E(B,C)| \backsim 8.5 \;\; |E(B,D)| \backsim 8.5 \;\; |E(C,D)| \backsim 6.5$$
$$|C(B,C)| \backsim 11.5 \; |C(B,D)| \backsim 11.5 \; |C(C,D)| \backsim 5.5$$
$$|O(B,C)| \backsim 3.5 \;\; |O(B,D)| \backsim 6.5 \;\; |O(C,D)| \backsim 8.5$$

When applying the approximations to the navigation strategies in the example given in Figure 6, the sequence in which data sets are accessed is the one given in Table 3. The results in this Table matches the results for the size-base and history-based strategies of the complete graph.

In the previous example, all non-existing edges (unknown correspondence relations) were derived by the composition of existing edges (known correspondence relations). In same cases, a derived edge must be needed to derive a second edge. Consider the example in Figure 7.

Table 3. Sequence of retrieved data sets for an incomplete information space

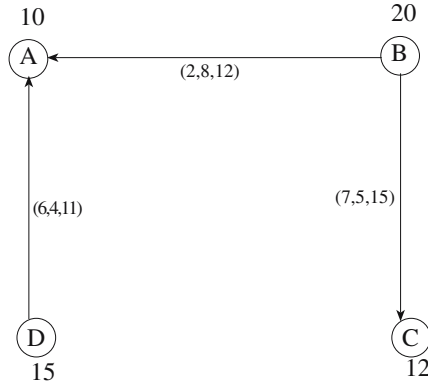| Order	Size-based	Forward-based	History-based
1	B	B	B
2	D	D	D
3	C	C	C
4	A	A	A

Fig. 7. Example 2: Incomplete and connected information space

In this example, to derive the correspondence relations between C and D requires having correspondence relations between A and C or between B and D, both of them being derived relations. In such case, we consider the process of completing the information in two steps: (1) a derivation from only known correspondence relations and (2) a derivation that combines known and unknown categories.

An algebraic analysis of using a derived correspondence relations to derive a new correspondence relations follows. Constants are the total areas per data set or the known correspondence relations. Consider the example in Figure 7, the constants a, b, c are the areas in data sets A, B, and C, respectively. The derivation of $|E(A, C)|$ follows the Equation 11.

$$max(0, |E(A, B)| - (b - |E(B, C)|) \leq |E(A, C)| \leq min(a, c) \qquad (11)$$

If $d \leq |E(A, B)| \leq e$, where d and e are previously derived lower and upper bounds, then

$$max(0, d - (b - |E(B, C)|) \leq |E(A, C)| \leq min(a, c)$$

includes all possible values of $|E(A, C)|$, since the effect of the upper bound of $|E(A, B)|$ is already considered within the interval between the lower bound of $|E(A, B)|$ and the upper bound of $|E(A, C)|$. Likewise, if $d \leq |E(B, C)| \leq e$, then

$$max(0, |E(A, B)| - (b - d) \leq |E(A, C)| \leq min(a, c)$$

includes all posssible values.

When deriving the correspondence relations, if there exist more than one composition path between data sets, it is the intersection of all intervals derived from composition paths of length 2 between data sets what defines the interval of a correspondence relation. Such intersection comes from the definition of path consistency of logical consistency in a graph [11].

In the intersection of these intervals, upper bounds of different composition paths between two data sets are the same, since these intervals are bounded by the minimum of total objects between data sets. For lower bound, the intersection is equivalent to the maximum of lower bounds. For example, the derivation of $|E(C,D)|$ in Figure 7 can be done by using $|E(C,B)|$ and $|E(B,D)|$ or by using $|E(C,A)|$ and $|E(A,D)|$. Thus, it is the intersection of both results what defines $|E(C,D)|$. The derived intervals of the example in Figure 7 are:

$$0 \leq |E(A,C)| \leq 10 \ 2 \leq |E(B,D)| \leq 15 \ 0 \leq |E(C,D)| \leq 12$$
$$0 \leq |C(A,C)| \leq 10 \ 5 \leq |C(B,D)| \leq 18 \ 0 \leq |C(C,D)| \leq 12$$
$$2 \leq |O(A,C)| \leq 12 \ 0 \leq |O(B,D)| \leq 13 \ 3 \leq |O(C,D)| \leq 15$$

With the previously estimated values for the areas of correspondence relations, the sequence in which data sets are accessed is as follows:

Table 4. Sequence of retrieved data sets for an incomplete information space with double derivation

Order	Size-based	Forward-based	History-based
1	B	B	B
2	D	C	C
3	C	D	D
4	A	A	A

The different configurations of incomplete graphs have an effect on the navigation strategies. Therefore, when handling incomplete graphs, the configurations with less needed derivations should be chosen (i.e., the direct derivation from known correspondence relations). An example is a star-like configuration, with the center being the data set with the largest area of objects.

5   Implementation of Strategies

To illustrate how the strategies for information navigation can be implemented, we use real data taken from the domain of a Forestry Cadastral System. We consider a thematic layer with 406 polygons of protected regions and we created sets of 10 different data sets. These data sets were created by randomly selected a percentage of the objects in the thematic layer. In the following illustrations we only used from 20% to 40% of the objects so that we could visually appreciate the differences between data sets. In addition to selecting different sets of objects, we introduce geometry changes over a percentage of the selected objects

(10%). These changes were translations in one or both dimensions equivalent to a random percentage of the size of the objects in the chosen directions. Thus, we expected to consider two cases when matching different data sets: incomplete information and partial overlapping. Figure 8 shows the complete thematic map and one data set derived with the process described above (20% of regions with a 10% of changes).
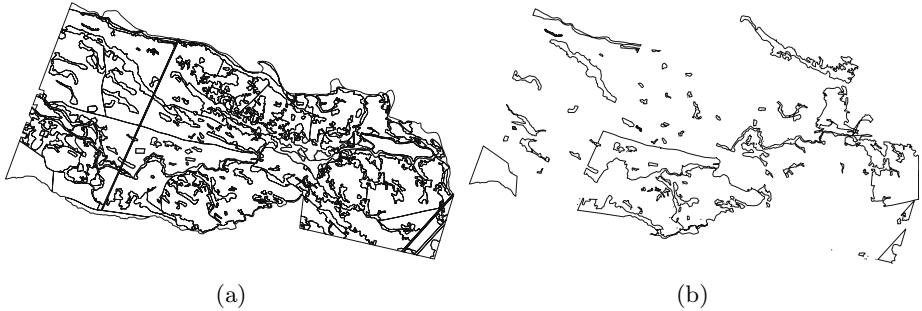

<table>
<tr><td>(a)</td><td>(b)</td></tr>
</table>

Fig. 8. Data: (a) complete thematic layer and (b) a derived data set

To implement the strategies, we approximated objects by their minimum boundary rectangles (MBRs). This approximation creates overlapping areas within a data set. We do so no only for computational simplicity of our implementation, but also because MBRs are the common approximation used by spatial databases for access methods and searching. Our final goal is to use these strategies on-line (i.e., with time constraints) and to relate our navigation strategies with searching in current databases.

We run the tree navigation strategies: size-based, forward-based and history-based strategies. The results were compared by determining the total regions (without duplication) that are actually obtained by accessing one after another data set. For example, consider only 3 data sets A, B, and C with an access sequence equal to B, C and A. The evaluation will say that for the first access we gain all data in B, for the second access, we gain all data in $C \cup B$, and for the last access, we gain all data in $A \cup B \cup C$. This can be represented in a graph in terms of percentages of new data (Figure 9).

The graph in Figure 9 indicates that the history-based strategy retrieves new regions quicker than the other two strategies. The history-based strategy best estimates the information content obtained in the retrieval process. With only accessing the tree first ranked data sets, this strategy obtains over 98% of information content in all data sets. The behavior of the size-based strategy is explained for the fact that two large data sets may have many duplicated regions so that we may not gain much information with accessing both of them. The forward-based strategy is limited to look to the next data set without considering information of the previous to the last data set that was selected.
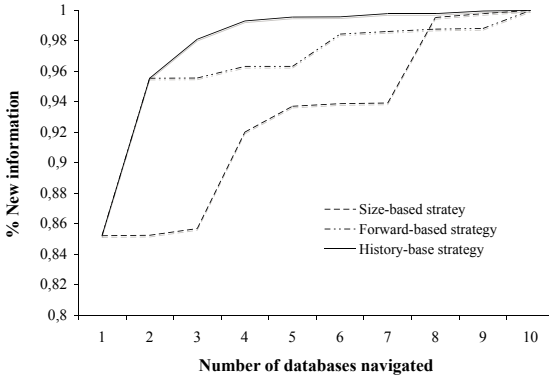
Fig. 9. Results in terms of percentages of new information

Figure 10 shows three of the ranked data sets: the first, the second and last ranked data set. It is possible to appreciate that the last ranked data set does not provide much new information.

(a)

(b)

(c)
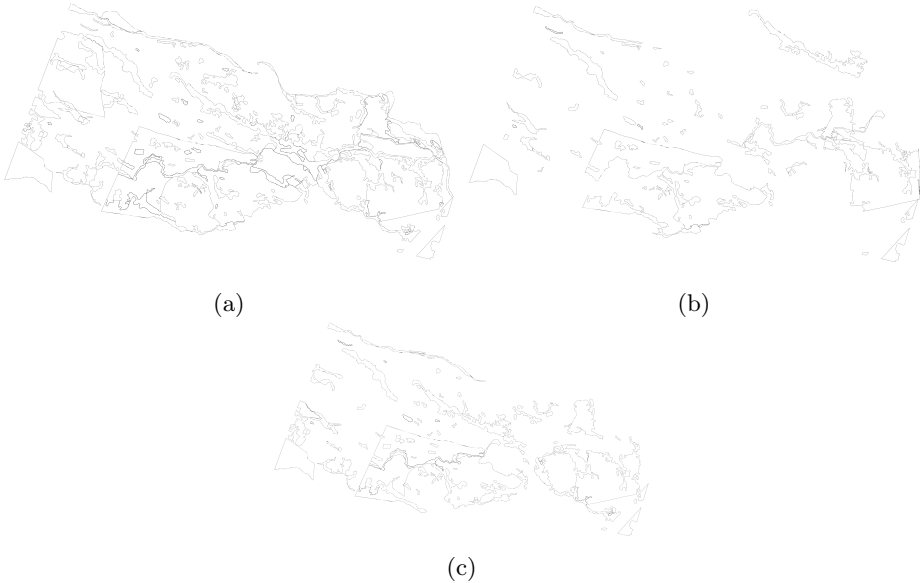
Fig. 10. Ranked data sets: (a) first, (b) second, and (c) last data set

The results shown here are examples of what we obtained with different sets of data. In all cases, the history-based strategy outperforms the forward-based and size-based strategies. Even more, in many cases, we needed only two o three data sets to obtain all information. Such cases consider data sets with larger

numbers of objects than the number of objects in the illustrated example and, therefore, data sets with larger numbers of duplications.

Finally, we considered an incomplete graph and we run the history-based strategy. The incomplete graph had a star-like configuration with one of the largest data sets being in the center of the star. The results in terms of the percentages of new information are shown in the graph of Figure 11. This graph shows no important differences between the results of the strategy with complete and incomplete graphs.
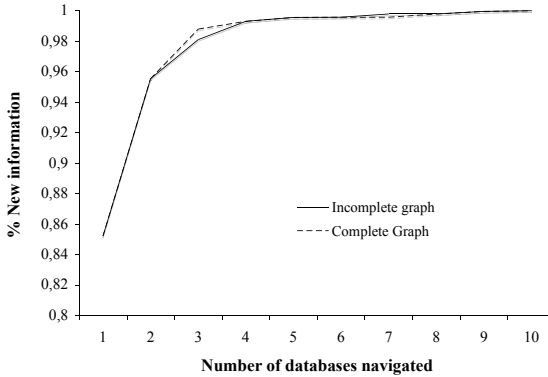

Fig. 11. Results for a complete and an incomplete graph

6   Conclusions

This work describes a formalism for relating the information content of data sets. Based on this characterization, it proposed a way to navigate or access these data sets to optimize the retrieval process. The experimental results show promising results for being applying in real cases of heterogeneous databases.

There are interesting topics for further investigation. We will make further evaluations of the strategies by comparing what the real impact of using MBR on the navigation strategies is. In this work we use the area of regions, but we expect to match objects and use the number of equivalent versus different objects in the navigation strategies. We would like to compare these two approaches: region versus objects. An additional evaluation will analyze the different strategies under incomplete information.

Using this formalism for query processing implies that one needs to characterize the information content of data sets associated with query windows in spatial databases. Since an on-line characterization seems inefficient and a characterization at the whole databases does not provide enough information of a particular spatial window, pre-defined aggregations at different levels of space partitions are needed. These aggregations may follow the organization given for spatial access methods and may use methods that approximate correspondence relations of overlapping regions.

Finally, we are planning to extend our work by assuming that additional information may exist about the origin of the data. In such situation, some databases may be considered more reliable than others, and some kind of database ranking may be included into the navigation strategies.

References

1. ACM. http://www.acm.org/class. 2006.
2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, 1999.
3. D. Cutting, D. Karger, and J. Oederson. Constant interaction-time scatter/gather browsing of very large document collection. In *16th Annual International ACM/SIGIR Conference*, pages 126–135, 1993.
4. M. Egenhofer, E. Clementini, and P. Di Felice. Evaluating inconsistency among multiple representations. In *Spatial Data Handling*, pages 901–920, Edinburg, Scotland, 1994.
5. M. Egenhofer and J. Sharma. Assessing the consistency of complete and incomplete topological information. *Geographical Systems*, 1(1):47–68, 1993.
6. R. Flowerdew. *Spatial Data Integration*, chapter Spatial Data Integration, pages 375–387. Longman Scientific & Technical, 1991.
7. F. Fonseca, M. Egenhofer, P. Agouris, and C. Camara. Using ontologies for integrated information systems. *Transactions in GIS*, 6(3):231–257, 2002.
8. A. Rodríguez and M. Egenhofer. Determining semantic similarity among entity classes from different ontologies. *IEEE Transactions on Knowledge and Data Engineering*, 15(2):442–456, 2003.
9. V. Kashyap and A. Sheth. Schematic and semantic similarities between batabase objects: A context-based approach. *The Very Large Database Journal*, 5(4):276–304, 1996.
10. B. Kuipers, J. Paredaens, and J. den Busshe. On topological equivalence of spatial databases. In F. Afrati and Ph. Kolaitis, editors, *6th International Conference on Database Theory ICDT97, LNCS 1186*, pages 432–446. Springer Verlag, 1997.
11. A. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.
12. E. Mena and A. Illarramendi. *Ontology-Based Query Processing for Global Information Systems*. Kluwer Academic Publishers, Norwell, MA, 2001.
13. S. Ram and S. G. Modeling and navigation of large information spaces: A semantic based approach. In *International Conference on System Science*. [http://computer.org/proceedings/hicss/0001/00016/00016020abs.htm], IEEE CS Press, 1999.
14. D. Roussinov and M. McQuaid. Information navigation by clustering and summary query results. In *International Conference on System Sciences*, page 3006. IEEE CS Press, 2000.

15. A. Sheth. *Interoperating Geographic Information Systems*, chapter Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics, pages 5–30. Kluwer Academic Plublishers, 1999.
16. VIVISMO. http://vivismo.com. 2006.
17. P. Weinstein and P. Birmingham. Comparing concepts in differentiated ontologies. In *12th Workshop on Knowledge Adquisition, Modeling, and Management*, Banff, Canada, 1999.
18. M. Worboys and E. Clementini. Integration of imperfect spatial information. *Journal of Visual Languages and Computing*, 12:61–80, 2001.
19. Yahoo! http://www.yahoo.com. 2006.
20. O. Zamir and O. Etzioni. Grouper: A dynamic cluster interface to web search results. In *WWW8*, 1999.

Correlation Analysis of Discrete Motions

Takeshi Shirabe

Institute for Geoinformation and Cartography
Technical University of Vienna
1040 Vienna, Austria
shirabe@geoinfo.tuwien.ac.at

Abstract. Points are dimensionless geometric elements often employed by geographic information systems to model a range of real-world objects such as people and vehicles. Many analytical tools have been developed for finding and validating certain patterns of static points such as clustering. In this paper, we present our preliminary efforts to develop statistical methods for analyzing patterns of moving points whose locations are tracked at equal intervals of time. Focus is placed on simple descriptive statistics—rather than more sophisticated inferential statistics—useful for detecting a tendency of two or more points to move in some coordinated fashion. A major implication of this paper is that statistical analysis complements spatial data query and modeling with respect to dynamics of sets of point-like objects in a way that potentially interrelated subsets are screened.

1 Introduction

In many applications of geographic information systems (GIS), points are used to model small discrete objects—relative to a given geographic scale—such as people and vehicles. While the kind of object represented by this geometric element depends on the specific context, there are some generic questions falling under the heading of "point pattern analysis" (see, e.g. [2, 4, 5]). Among the most basic are whether a given set of points is randomly distributed and, if not, how it differs from a random distribution. Many tools—from simple visualization to formal mathematical modeling—have been developed over the last decades.

With the recent advance of data collection technology, more data are now available on time-varying geographic phenomena. Some are concerned with how attributes of entities vary with time, others with how locations of entities change through time, and still others with both. To efficiently handle spatio-temporal data like these, GIS need new modeling schemes, for example, amendment vectors [13], triad framework [21], three-domain representation [25], identity-based model [1, 11], dynamic sets [22], and event-oriented approach [24]. See Güting and Schneider [10] for a recent overview.

Point data are not an exception. Like or not, even at this moment, millions of mobile agents are monitored by digital devices such as global positioning systems (GPS) and, more recently, geosensor networks [19]. There has been substantial research in search of suitable data models for such tracking data (e.g. [6, 12, 18]). Commercial GIS, too, have fairly well responded to this trend by improving their capability of

storing and managing a large volume of moving point data (e.g. ArcGIS Tracking Analyst from ESRI).

Once histories of moving points are successfully registered in GIS, it is natural to ask the systems to do more: query and analysis. There are cases where the user wants to examine the trajectory of each *individual*. Typical questions are how fast it moves, in which direction it moves, where it may or may not go, and, in a more involved instance, what activities it has access to within its space and time constraints [17]. In other cases, the user may be interested in the *collective* behavior of multiple points [7, 8, 14]—or "collective dynamic" termed by Galton [8]. An exemplar task is to characterize the way a group of people moves, grows, and fades as a whole. In addressing this, as Galton [8] suggests, it is implicitly assumed that the group acts in some coordinated manner. Otherwise it would be unreasonable to give a structured relation to a set of independent motions.

To see this, consider two independent "random walks" [20] plotted in Figure 1. The step lengths and directions of each walk are uniformly distributed between 0 and 1 and between 0 and 2π radians, respectively. Take a closer look at their first three steps. It appears that the two walkers first try to pull away from each other, and then turn to come close to each other. Have we found a notable coordination between them? Or is it just one of the things that must happen sometimes by chance? As we know that the two walks are independent, the latter is the case. In fact as we trace them further, we always find more such "surprises."
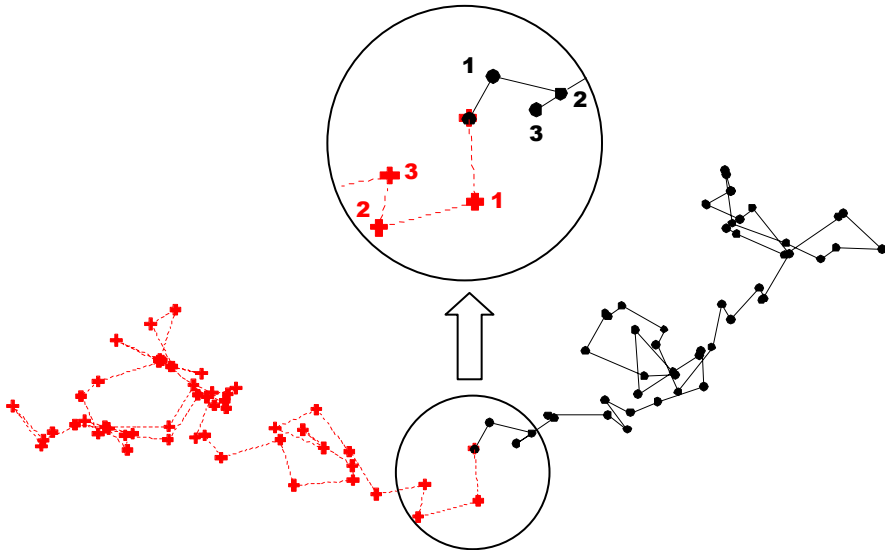

Fig. 1. Two random walks starting at the same location. The sequences of dots (connected by solid lines) and of crosses (connected by dashed lines) represent the changes of two walkers' locations. Their first three steps are extracted in the inset, where the number associated with each dot is the time at which the corresponding location is reached.

The purpose of this paper is to partially answer this "accident-or-not?" question. To do so, we present an approach to spotting statistically correlated motions in a given set of identifiable moving points, whose locations are recorded at equal intervals of time. The approach does not articulate specific patterns of motion such as those classified by Van de Weghe et al. [23]. Also it is not designed for inferential tasks, that is, interpolation of locations of points between sampled times or prediction of their locations at future times. These would require more careful analysis of relevant variables, which, for example, concern the intrinsic characteristics of points and of their environment. Nevertheless, we expect the present work to serve as a basis for formal statistical modeling of collective motion that may follow.

The rest of this paper is organized as follows. Section 2 introduces a statistical tool for measuring the strength of correlation between motions. Section 3 illustrates how it is applied to actual data. Section 4 sketches how it is extended to analyze other kinds of motion relation. Section 5 concludes the paper.

2   Description and Correlation of Motions

The phenomenon under consideration can be seen from at least two perspectives: time series of point sets and set of point time series (Figure 2). The former is more closely related to traditional point pattern analysis. That is, for those who are concerned with how a set of points is distributed at one moment, a primary interest is how that distribution changes over time. For instance, is it expanding or shrinking? Questions of this kind are answered by comparing point sets observed at different times in terms of such statistics as the nearest neighbor distance or the number of neighbors of each point[1]. In this case, it is not necessary to keep track of the identity of each point.

The latter, on the other hand, takes the whole trajectory of each point as one process. This makes it possible to compare one moving point to another and examine how they move in relation to each other. It is this view that we take in motion pattern analysis.

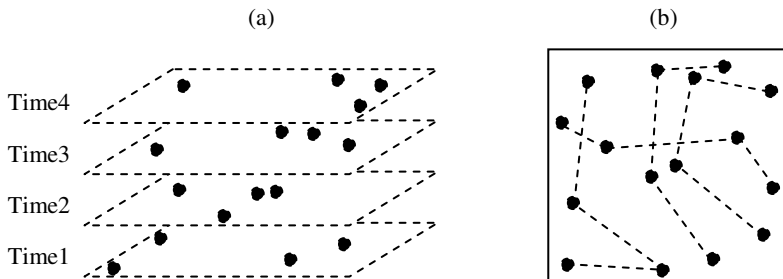**Fig. 2.** (a) time series of point sets and (b) set of point time series. In (a), each dashed layer encloses the locations of all points observed at one time. In (b), each dashed line traces the locations of one point observed at different times.

---

[1] It is well known that the mean square distance of random walks grows proportionally with time. Thus a certain degree of expansion is natural for moving points.

2.1 Motion as a Time Series of Vectors

Since every point is monitored at equal intervals of time, its motion can be discretized into a sequence of vectors, determined by successive pairs of observed locations. It is not certain how that sequence goes, so we assume that it does in a stochastic manner. Accordingly, a motion—which is associated with each identifiable point—is here regarded as a time series of observations on a set of random vector variables defined at equal intervals of time. These random variables do not necessarily follow a nice uniform distribution. For example, when you walk, your motion would peak in the forward direction and at a certain speed, and you would tend to walk faster in the forward direction than in the backward direction. In addition, your motion is likely to be influenced by others'. The resulting distribution of motion vectors would be more complicated than a uniform, or even a normal, distribution.

It may be an idea to decompose each motion vector into its direction and magnitude and evaluate them separately. This approach is indeed convenient as all variables become scalar. We have chosen not to do so, however, since handling a vector as a single mathematical object reduces potential bias[2]. To see this, try to describe how the point in Figure 3 tends to move while traveling from the tail of vector $\mathbf{x}_1$ to the head of vector $\mathbf{x}_6$. The mean vector $\mathbf{a}$ indicates the tendency of this motion more accurately than the vector $\mathbf{b}$ that is made of the mean direction and the mean magnitude.
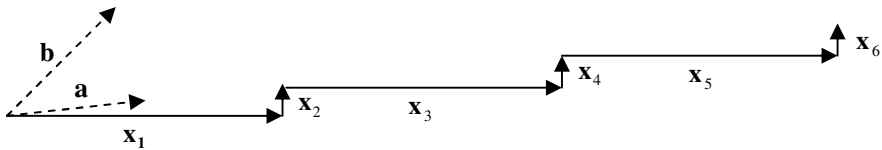
Fig. 3. Two kinds of vectors to show the tendency of a motion. The sequence of solid arrows labeled with $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, $\mathbf{x}_4$, $\mathbf{x}_5$, and $\mathbf{x}_6$ represents a given series of observed motion vectors. All the horizontal vectors are identical to (9,0) and all the vertical vectors (0,1). The dashed arrow labeled with $\mathbf{a}$ represents the mean vector, $(9/2, 1/2)$. The dashed arrow labeled with $\mathbf{b}$ represents the vector made of the mean length and the mean angle, which are 5 and $\pi/4$ radians, respectively.

In the remainder of this paper, we use the following notational conventions: (1) A vector is denoted by a **bold-face** letter, and a scalar by a non-bold-face letter. (2) A variable is denoted by an UPPER-CASE letter, and a realization of it by the same letter of lower case. (3) A subscript, i, associated with a vector indicates that the vector emanates from the location at time $i-1$ to that at time i; we call this vector simply a vector at time i, unless any confusion occurs. Accordingly, for example, $\mathbf{X}_i$ represents a random vector variable defined at time i, and $\mathbf{x}_i$ represents one particular observation on that variable.

---

[2] This is not the case if the speed of the point is constant. This assumption is often reasonable.

2.2   Correlation of Two Motions

A plausible signal of coordination between two motions is one's systematic response to the other. Simple examples are: when one moves forward, the other moves backward; when one makes a large step, the other makes a large step, too.

Correlation coefficient is a useful statistical tool to examine this kind of pattern. Suppose we make n serial observations on two motions, $\{\mathbf{x}_i : i = 1,...,n\}$ and $\{\mathbf{y}_i : i = 1,...,n\}$. Let $r_{\mathbf{xy}}$ denote their correlation coefficient. Recall that a motion is a vector series. Then $r_{\mathbf{xy}}$ is defined by:

$$r_{\mathbf{xy}} = \frac{\sum_{i=1}^{n}(\mathbf{x}_i - \overline{\mathbf{x}}) \cdot (\mathbf{y}_i - \overline{\mathbf{y}})}{\sqrt{\sum_{i=1}^{n}|\mathbf{x}_i - \overline{\mathbf{x}}|^2}\sqrt{\sum_{i=1}^{n}|\mathbf{y}_i - \overline{\mathbf{y}}|^2}} \tag{1}$$

where $\overline{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$ and $\overline{\mathbf{y}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{y}_i$. The numerator is the covariance, which, in the present context, indicates how the two motions deviate together from their respective means. Each term of the summation is the dot product of two deviation vectors $\mathbf{x}_i - \overline{\mathbf{x}}$ and $\mathbf{y}_i - \overline{\mathbf{y}}$. Geometrically, it is the product of their lengths multiplied by the cosine of the angle between them; intuitively, it measures the similarity between them. The square roots in the denominator are the standard deviations of the two motions, which make the correlation coefficient scale free.

The correlation coefficient ranges from –1 to 1 indicating that the closer to 1 the more positively correlated and the closer to -1 the more negatively correlated. Figure 4 illustrates examples: in (a), the two motions are positively correlated as they have similar deviation patterns; in (b), the two motions are negatively correlated as they have opposite deviation patterns. Be aware that the correlation coefficient depends not on the raw value of each observed motion vector, but on its deviation from the corresponding mean.

Fig. 4. (a) Positively correlated motions and (b) negatively correlated motions. Each sequence of solid arrows starting at a dot represents a motion, and the dashed arrow overlapping it represents the *direction* of its mean.

In general if two variables are statistically independent, their correlation coefficient is zero (or if the correlation is not zero, the two motions are not independent). The converse is not true, however, because the correlation coefficient measures only the strength of linear dependence. Thus the present method may overlook apparently co-ordinated motion patterns. Some are simply not the kind we can detect (see Section 4); others consist of sub-patterns that have equal degrees of correlation but with opposite signs. Figure 5 shows a contrived yet instructive example of the latter. The pair of motions in this example consists of two parts: one positively correlated and the other negatively correlated. They cancel each other and give rise to almost zero overall correlation. This problem could be resolved by separating the two components. In practice, however, it is not easy to find a discontinuity in given data unless some background information is provided (e.g. seasonality in animal behavior). Otherwise, automated breaking procedures would be valuable tools that have to be developed.
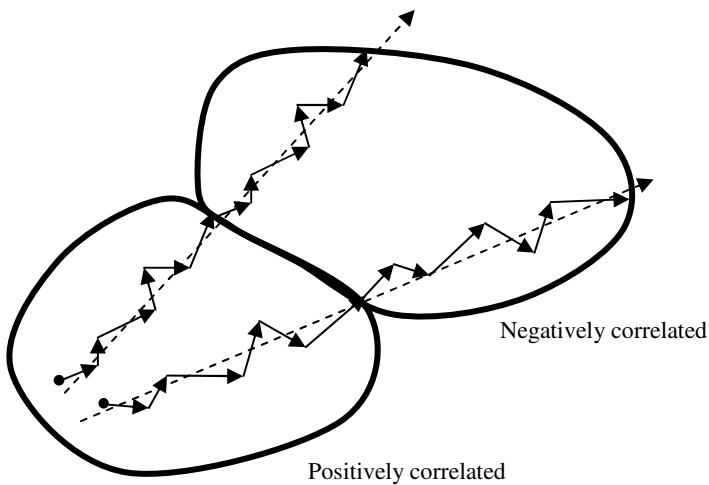
Fig. 5. Dependent but not correlated pair of motions. It consists of a positively correlated part (enclosed) and a negatively correlated part (enclosed).

2.3 Cross-Correlation of Two Motions

In correlated motions such as shown in Figure 4, two moving points (at least appear to) respond to each other instantaneously—i.e. *while* one goes this way, the other goes that way. Another, equally interesting kind of relation is found when one does move in response to the other but the response is not immediate. This may be regarded as an instance of "trendsetting" in Laube's terms [14-16].

The correlation coefficient in the form of Equation (1) is not effective for this kind of relation. Imagine a boy chasing a girl, for example. Although their motions certainly exhibit some relation, the girl can choose each step independently of the boy. Then the resulting correlation coefficient could be very low. Still each step of the boy should be strongly related to the *previous* step(s) of the girl.

An approach to such delayed relations is to measure the correlation coefficient of two motions in such a way that the vectors constituting one of them are shifted by a certain time lag. The result is called a "cross-correlation coefficient" [3]. Let $r_{xy}(k)$ denote the cross-correlation coefficient of two observed motions $\{\mathbf{x}_i : i = 1,...,n\}$ and $\{\mathbf{y}_i : i = 1,...,n\}$ at lag k. $r_{xy}(k)$ is formulated as:

$$r_{xy}(k) = \frac{\sum_{i=1}^{n-k} (\mathbf{x}_i - \overline{\mathbf{x}}) \cdot (\mathbf{y}_{i+k} - \overline{\mathbf{y}})}{\sqrt{\sum_{i=1}^{n-k} |\mathbf{x}_i - \overline{\mathbf{x}}|^2} \sqrt{\sum_{i=1}^{n-k} |\mathbf{y}_{i+k} - \overline{\mathbf{y}}|^2}} \tag{2}$$

where $\overline{\mathbf{x}} = \dfrac{1}{n-k} \sum_{i=1}^{n-k} \mathbf{x}_i$ and $\overline{\mathbf{y}} = \dfrac{1}{n-k} \sum_{i=1}^{n-k} \mathbf{y}_{i+k}$.

Like the correlation coefficient, this ranges from –1 to 1 indicating that the closer to 1 the more positively cross-correlated and the closer to -1 the more negatively cross-correlated.

At $k = 0$, the cross-correlation coefficient is identical to the correlation coefficient (cf. Equations (1) and (2)). At $k = 1$, it tells the strength of one motion's linear dependence on the immediate past of the other motion. As k increases, one can investigate further such delayed effects. It is important to note that unlike the correlation coefficient, the cross-correlation coefficient is not symmetric except for $k = 0$. This is intuitively understood by revisiting the boy-chasing-girl example. The girl, as the leader, is freer to determine her way than the boy, as the follower, does his. In an extreme case where the girl is not aware of being stalked by the boy, her cross-correlation against him may be very close to zero, although his cross-correlation against her is very high.

2.4 Effects of Autocorrelation

Recall that a motion is assumed to be a stochastic process and an observed motion is one possible realization of that process. Then, conceptually, if a mover is let go again and again, it could make many different motions. In this sense, the correlation coefficient discussed above is a sample statistic, which may differ from the unknown "true" value. In general, the variance of a sample statistic increases as the sample size grows, and this is true for the correlation coefficient of two motions. If either of the paired motions is autocorrelated—i.e. dependent on its own past, however, the variance of the sample correlation coefficient may inflate. This implies that Equations (1) and (2) could underestimate or overestimate the actual strength of correlation.

For more cautious analysis of motions, it is advised to inspect the autocorrelation within each motion involved. Similarly to cross-correlation, the degree of autocorrelation—called "autocorrelation coefficient" [3]—is defined at a different lag. The autocorrelation coefficient of motion $\{\mathbf{x}_i : i = 1,...,n\}$ at lag k, here denoted by $r_{xx}(k)$, is formulated as:

$$r_{\mathbf{xx}}(k) = \frac{\sum_{i=1}^{n-k}(\mathbf{x}_i - \overline{\mathbf{x}}) \cdot (\mathbf{x}_{i+k} - \overline{\mathbf{x}}')}{\sqrt{\sum_{i=1}^{n-k}\left|\mathbf{x}_i - \overline{\mathbf{x}}\right|^2} \sqrt{\sum_{i=1}^{n-k}\left|\mathbf{x}_{i+k} - \overline{\mathbf{x}}'\right|^2}} \tag{3}$$

where $\overline{\mathbf{x}} = \dfrac{1}{n-k}\sum_{i=1}^{n-k}\mathbf{x}_i$ and $\overline{\mathbf{x}}' = \dfrac{1}{n-k}\sum_{i=1}^{n-k}\mathbf{x}_{i+k}$.

Again, this statistic ranges from –1 to 1 indicating that the closer to 1 the more positively autocorrelated and the closer to -1 the more negatively autocorrelated.

To see possible interference of autocorrelation with correlation, let us return to the positively correlated motions illustrated in Figure 4 (a). A careful look at each motion reveals that any two consecutive vectors deviate from the mean in opposite directions, which indicates a high degree of autocorrelation—negative at odd-numbered lags and positive at even-numbered lags. It is not possible to tell if the observed pattern has arisen from the correlation between the two motions or the autocorrelation within each.

3 Example

To illustrate the utility of the statistics introduced in the preceding section, we present an example with real-life data. The data, provided by the Government of Yukon, Canada [9], contain the weekly (or more/less frequently recorded in some cases) locations of dozens of arctic deer, called porcupine caribou. They inhabit an area approximately 260,000 km^2 large, located between Kaktovik, Alaska, U.S. to Aklavik, NWT to Dawson City, Yukon, Canada. The animals have been tracked over nearly a decade by GPS receivers attached on their collars.

We have processed the original data to create a small subset for use of this study. It includes eight caribou's unbroken series of locations over 29 weeks, which define eight motions, each consisting of 28 vectors (Figure 6). In practice, we sometimes lose track of moving objects during the phase of data collection. The missing values then need to be replaced somehow, for example, with means of their preceding and following records. More sophisticated interpolation methods, however, may be required depending on the level of accuracy required by the application. In the current example, the dataset has been so deliberately chosen that there is no missing record. This means, however, that any statistical information derived may be biased. Thus this section is intended only for demonstration purpose.

The correlation coefficient of each pair of caribou motions is displayed in Table 1. Ten pairs—doubles of the same motion are not included—are found to have relatively high values (0.5 or larger). The corresponding cells are highlighted in the table. An interesting finding is that only one motion (associated with Caribou 1) is poorly correlated to every other. The rest have a fairly high degree of correlation to at least two other motions.
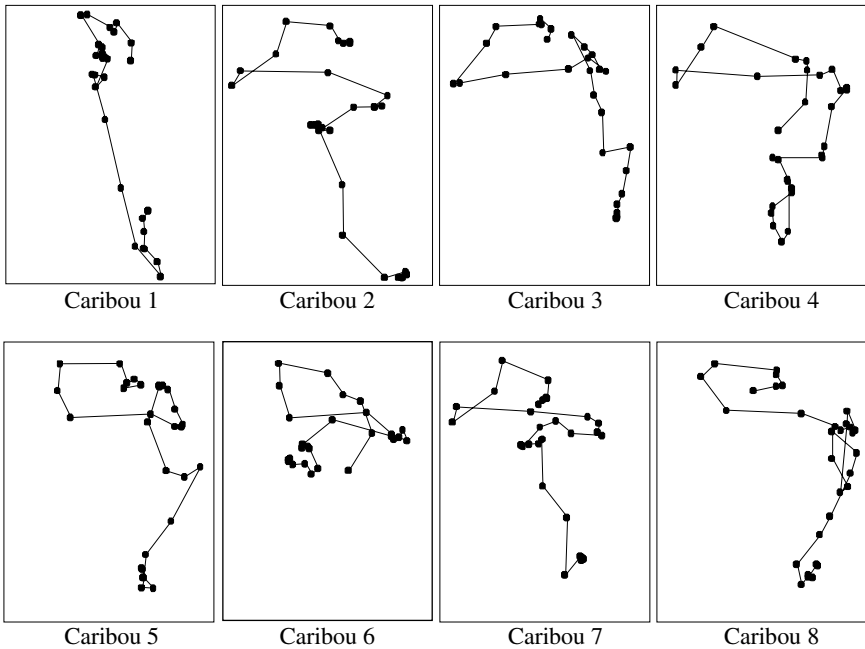
Fig. 6. Motions of eight caribou. In each map, a motion is represented by a sequence of line segments separated by dots that mark weekly locations of a caribou, starting at the upper end of the sequence. All maps share the same frame of reference, which encompasses approximately a 550km-by-430km area.

Table 1. Correlation Coefficients of Eight Caribou Motions

	Caribou1	Caribou2	Caribou3	Caribou4	Caribou5	Caribou6	Caribou7	Caribou8
Caribou1	1	0.431	0.257	0.384	0.229	0.306	0.041	0.431
Caribou2	0.431	1	0.601	0.648	0.357	0.437	0.67	0.309
Caribou3	0.257	0.601	1	0.643	0.489	0.401	0.654	0.269
Caribou4	0.384	0.648	0.643	1	0.356	0.502	0.667	0.499
Caribou5	0.229	0.357	0.489	0.356	1	0.602	0.37	0.584
Caribou6	0.306	0.437	0.401	0.502	0.602	1	0.36	0.636
Caribou7	0.041	0.67	0.654	0.667	0.37	0.36	1	0.362
Caribou8	0.431	0.309	0.269	0.499	0.584	0.636	0.362	1

Note that the value of each cell indicates the correlation coefficient between the caribou motions indicated by the corresponding row and column. Note also that the table is diagonally symmetric by definition.

Cross-correlation coefficients have also been computed for each pair of caribou motions. Table 2 shows those at lags 1, 2, and 3. Those at greater lags are omitted because of their increasing inaccuracy. That is, as the lag is increased by one, the sample size decreases by one—at lag 26, the sample size is only two. It has been found that

four pairs of caribou motions have a relatively high degree of cross-correlation at lag 1 (see the highlighted values in Table 2). Caribou 5 is the most involved player, as its motion is cross-correlated to those of Caribou 3, 4, and 7 when the latter are delayed by a week. Together with some of the findings in Table 1—i.e. the motions of Caribou 3, 4, and 7 are fairly correlated to each other but not to Caribou 5's, it may be hypothesized that Caribou 5 was setting a trend for Caribou 3, 4, and 7. The statistical analysis alone, however, does not prove the presence of a causal relation.

Table 2. Cross-correlation Coefficients of Eight Caribou Motions

Lag 1

	Caribou1	Caribou2	Caribou3	Caribou4	Caribou5	Caribou6	Caribou7	Caribou8
Caribou1	0.529	0.249	0.189	0.072	0.195	0.216	-0.051	0.073
Caribou2	0.356	0.354	0.165	0.186	0.351	0.462	0.15	0.407
Caribou3	0.166	0.411	0.368	0.414	0.536	0.455	0.39	0.41
Caribou4	0.289	0.434	0.259	0.326	0.507	0.474	0.231	0.464
Caribou5	0.236	0.06	-0.04	-0.016	0.346	0.234	-0.052	0.193
Caribou6	0.163	0.017	-0.026	0.033	0.245	0.359	-0.021	0.275
Caribou7	0.162	0.454	0.259	0.314	0.529	0.482	0.325	0.518
Caribou8	0.284	0.126	-0.102	0.097	0.331	0.278	-0.073	0.327

Lag 2

	Caribou1	Caribou2	Caribou3	Caribou4	Caribou5	Caribou6	Caribou7	Caribou8
Caribou1	0.075	-0.06	0.131	-0.13	0.112	0.046	-0.155	-0.254
Caribou2	0.324	-0.042	0.076	-0.044	0.33	0.373	-0.196	0.25
Caribou3	0.197	0.172	0.147	0.106	0.442	0.343	-0.004	0.442
Caribou4	0.172	0.069	-0.088	0.022	0.251	0.398	-0.101	0.294
Caribou5	0.272	-0.006	0.051	-0.044	-0.043	0.159	-0.166	0.108
Caribou6	-0.047	-0.089	-0.024	-0.199	0.102	0.101	-0.162	-0.05
Caribou7	0.41	0.231	0.01	0.173	0.359	0.393	0.003	0.406
Caribou8	0.107	0.018	-0.064	-0.146	0.025	0.036	-0.161	-0.16

Lag 3

	Caribou1	Caribou2	Caribou3	Caribou4	Caribou5	Caribou6	Caribou7	Caribou8
Caribou1	-0.113	-0.164	0.292	-0.032	0.174	0.058	-0.124	-0.284
Caribou2	0.128	-0.183	0.006	-0.053	0.152	0.067	-0.174	-0.164
Caribou3	0.1	-0.168	-0.125	-0.033	0.041	0.138	-0.258	0.032
Caribou4	0.16	-0.138	0.016	-0.01	0.007	0.234	-0.154	-0.016
Caribou5	0.156	-0.175	-0.158	-0.129	-0.208	-0.078	-0.34	-0.188
Caribou6	-0.126	-0.26	-0.284	-0.312	-0.233	-0.144	-0.244	-0.353
Caribou7	0.438	0.05	-0.022	-0.073	0.116	0.222	-0.217	0.089
Caribou8	0.002	-0.125	-0.025	-0.243	-0.147	-0.003	-0.235	-0.334

Note that the value of each cell indicates the cross-correlation coefficient between the caribou motions indicated by the corresponding row and column, the former of which is delayed by the specified lag.

Table 2 also shows the autocorrelation coefficient of each caribou motion along the diagonal. All have nonzero autocorrelation coefficients at lag 1, but they are not very high and fade quickly at greater lags. Still, we should keep in mind that the correlation and cross-correlation coefficients obtained might not be most accurate.

4   Extension

We have so far been concerned only with linear relation between motions. This section briefly describes a possible extension of the present approach so as to evaluate other kinds of relation.

Assume that we have two motions that are second-order stationary processes $\{\mathbf{X}_i : i = 1,...,n\}$ and $\{\mathbf{Y}_i : i = 1,...,n\}$. If they have a significantly high correlation, it is, in theory, said that one of them (partially) explains the other. More formally, the two series have the following relation.

$$\mathbf{Y}_i - \mathbf{\mu_Y} = \beta(\mathbf{X}_i - \mathbf{\mu_X}) + \varepsilon_t \qquad (4)$$

where $\mathbf{\mu_X} = E(\mathbf{X}_i)$, $\mathbf{\mu_Y} = E(\mathbf{Y}_i)$, ε_i is a residual term at time i, and β is a constant scalar. The scalar β is essentially the same as the correlation coefficient of the two motions. When one is zero, the other is zero. Roughly speaking, Equation (4) can be interpreted such that $\mathbf{Y}_i - \mathbf{\mu_Y}$ is the result of a scaling transformation of $\mathbf{X}_i - \mathbf{\mu_X}$.

If we set $\beta = \beta\mathbf{I}$ in Equation (4) where $\mathbf{I}$ is the 2-by-2 identity matrix, the equation can be seen as a special case of a more general matrix transformation expressed as:

$$\mathbf{Y}_i - \mathbf{\mu_Y} = \mathbf{B}(\mathbf{X}_i - \mathbf{\mu_Y}) + \varepsilon_i \qquad (5)$$

where $\mathbf{B}$ is a 2-by-2 matrix. This includes basic transformations including scaling, rotation, reflection, and their combinations. From observed motions, it is possible to test if the matrix $\mathbf{B}$ contains at least one element significantly different from zero. This is an extension of the correlation analysis we have discussed.

It is tempting to specify β or $\mathbf{B}$ from a given dataset. The resulting model may help improve understanding the system of the two motions, but is unlikely to be useful for forecasting one's future from the other. This is partly because there are still too many important factors missing, such as autocorrelation, trend, and cycle that may be associated with each motion. Furthermore, there may exist some kind of feedback between the two motions—i.e. motion $\mathbf{X}_i$ affects motion $\mathbf{Y}_i$, which, in turn, affects motion $\mathbf{X}_i$. Such effects considerably jeopardize the validity of Equation (5).

5   Conclusion

We have presented a simple descriptive-statistical approach to detection of potentially coordinated motions. A motion—which is attributed to each identifiable point—is defined as a discrete time series of vector observations made at equal intervals of time. We have shown that the correlation coefficient is a valuable tool to find linearly correlated motions, and that the cross-correlation coefficient is an alternative measure

when one motion is correlated to another in a lagged manner. An advantage of use of these statistics is that they take into account all observations associated with each point so that they are not too sensitive to coincidental abnormalities that occasionally appear. At the same time, they reduce the chances of overlooking significant relations that are not obvious from a microscopic perspective. The flip side of this, however, is that the present method does not seek local, short-term motion patterns. Detection of these would need other tools such as Laube's.

We have also reported some drawbacks. First, the accuracy of correlation and cross-correlation coefficients can be disturbed by the presence of autocorrelation within each motion. Second, the present approach is not most effective when coordinated motion patterns arise from nonlinear relations or from two (or more) linear relations of opposite signs. These should be addressed by future research.

Acknowledgements

References

1. Al-Taha, K. and Barrera, R. Identities through time. In: M. Ehlers and D. Steiner (Eds.) Proceedings of International Workshop on Requirements for Integrated Geographic Information Systems, New Orleans, LA, Environmental Research Institute of Michigan (ERIM), 1-12 (1994)
2. Bailey, T. C. and Gatrell, A. C. Interactive spatial data analysis, Longman Scientific & Technical, Essex, England (1995)
3. Chatfield C. The Analysis of Time Series An Introduction. Chapman and Hall (2004)
4. Diggle, P. J. The statistical Analysis of Spatial point patterns, Academic Press, London, England (1983).
5. Boots, B. N. and Getis, A. Point Pattern Analysis, Sage Publications, Newbury Park, CA (1987)
6. Erwig, M., Güting, R. H., Schneider, M., and Vazirgiannis, M. Spatio-temporal data types: an approach to modeling and querying moving objects in databases. Geoinformatica Vo. 3(3), 269-296 (1999).
7. Frank, A. U. Socio-economic units: their life and motion. In: A. U. Frank, J. Raper, and J. P. Cheylan (Eds.) Life and Motion of Socio-Economic Units, London: Taylor & Francis, 21-34 (2001)
8. Galton, A. Dynamic Collectives and Their Collective Dynamics. In: A. G. Cohn and D. M. Mark (Eds.) Spatial Information Theory: Proceedings of International Conference COSIT 2005, Lecture Notes in Computer Science, Vol. 3693, Springer, Berlin-Heidelberg, 300-315 (2005)
9. Government of Yukon and Porcupine Caribou Technical Committee, Canada. Porcupine caribou herd satellite collar project, http://www.taiga.net/satellite/
10. Güting, R.H. and Schneider, M. Moving Objects Databases. Morgan Kaufmann Publishers, 2005.

11. Hornsby, K. and Egenhofer M. J. Identity-based change: a foundation for spatio-temporal knowledge representation. International Journal of Geographical Information Science, Vol. 14(3), 207-224 (2000)

12. Hornsby, K. and Egenhofer M. J. Modeling moving objects over multiple granularities. Annals of Mathematics and Artificial Intelligence Vol. 36 (1-2), 177-194 (2002)

13. Langran, G. Time in Geographic Information Systems, London: Taylor & Francis (1992)

14. Laube, P. and Imfeld, S. Analyzing Relative Motion within Groups of Trackable Moving Point Objects, Proceedings of the Second International Conference on Geographic Information Science, Lecture Notes In Computer Science, Vol. 2478, Springer, Berlin-Heidelberg, 132-144 (2002)

15. Laube, P., Kreveld, M. van, and Imfeld, S. Finding REMO—Detecting Relative Motion Patterns in Geospatial Lifelines. In: P. F. Fisher (Ed.) Developments in Spatial Data Handling, Proceedings of the 11th International Symposium on Spatial Data Handling, Leicester, UK, 23rd-25th August 2004. Springer, Berlin-Heidelberg, 201-214 (2004)

16. Laube, P., Imfeld, S., and Weibel R. Discovering relative motion patterns in groups of moving point objects. International Journal of Geographical Information Science Vol. 19, 639-668 (2005)

17. Miller, H. J. Modeling accessibility using space-time prism concepts within geographical information systems, International Journal of Geographical Information Systems, Vol. 5, 287–301 (1991)

18. Moreira, J., Ribeiro, C. and Saglio, J. M. Representing and Manipulation of Moving Points: An extended data model for location estimation. Cartography and Geographic Information Science, Vol. 26(2), 109-123 (1999)

19. Nittel, S., Duckham, M., Kulik, L. Information dissemination in mobile ad-hoc geosensor networks. In: Egenhofer, M. J., Freksa, C. and Miller, H. J. (Eds.) Lecture Notes in Computer Science, Vol. 3234, Springer, Berlin-Heidelberg, 206-222 (2004)

20. Pearson, K. The problem of the random walk. Nature, Vol. 72 (1905) 294

21. Peuquet, D. It's about time: a conceptual framework for the representation of temporal dynamics in GIS. Annals of the American Association of Geographers, Volume 84 (3), 441-461 (1994)

22. J. Stell, G. Granularity in change over time. In: Duckham, M., Goodchild, M., and Worboys, M (Eds.) Foundations of Geographic Information Science, Taylor and Francis, 95-115 (2003)

23. Van de Weghe, N., Kuijpers, B., Bogaert, P., and De Maeyer, P. A Qualitative Trajectory Calculus and the Composition of its Relations. In: M. A. Rodriguez, I. F. Cruz, M. J. Egenhofer, S. Levashkin (Eds.) Proceedings of Geospatial Semantics: First International Conference, Geos 2005, Lecture Notes in Computer Science, Vol. 3799, Springer, Berlin-Heidelberg, 60-76 (2005)

24. Worboys, M. F. Event-oriented approaches to geographic phenomena. International Journal of Geographical Information Science 19(1): 1-28 (2005)

25. Yuan, M. Use of three-domain representation to enhance GIS support for complex spatio-temporal queries. Transactions in GIS, Vol. 3 (2), 137-160 (1999)

Representing Topological Relationships for Moving Objects

Erlend Tøssebro[1] and Mads Nygård[2]

[1] Department of Electronics and Computer Science, University of Stavanger,
NO-4036 Stavanger, Norway
`erlend.tossebro@uis.no`
[2] Department of Computer and Information Science, Norwegian University
of Science and Technology, NO-7491 Norway
`mads@idi.ntnu.no`

Abstract. Several representations have been created to store topological information in normal spatial databases. However, not that much work has been done to store such relationships for spatiotemporal data. This paper extends the representation of moving objects from [7] so that it can also store and enforce some of the topological relationships between the objects. This is done in a fashion similar to the Node-Arc-Area model for normal spatial databases. One use of such a representation is storing a changing spatial partition.

1 Introduction

For purely spatial databases, there are several ways to represent topology. One such way is the Node-Arc-Area representation [13]. Each line segment stores a link to the two regions that it borders, each point stores a link to each line segment that begins or ends in that point and each face (connected region) stores a link to at least one of its border curves. This ensures that if the border of one region is updated, the borders of all other regions are automatically updated as necessary to maintain known topological relationships.

An important use for topological information is storing a spatial partition, since without topological information it is difficult to control whether a given set of regions forms a partition or not. [5] presents an abstract model for spatiotemporal partitions called the "honeycomb model". This is called an abstract model in this paper because it is based on infinite point sets. A discrete model, by contrast, is based on constructs that one could realistically store in a database such as straight line segments. Both raster and vector models are discrete models. The "honeycomb" model represents time as an extra dimension and represents a spatiotemporal partition as a three-dimensional partition with the limitation that at any time instant the partition should be a legal two-dimensional partition.

However, no discrete model for spatiotemporal partitions is known to the authors. Nor is a discrete model of spatiotemporal information that takes topological relationships into account.

1.1 Examples of Moving Partitions

Here is a collection of examples of where the ability to store a spatiotemporal partition might be useful. The first was also mentioned in [5].

Example 1: Subdivision of the world into countries: The countries of the world make up a partition because they cover all the land and do not overlap each other. The borders of countries may also change (such as when east and west Germany merged or when Yugoslavia split apart), but only in discrete steps.

Example 2: Land cover: The type of vegetation that covers different areas changes continuously over time. It would be theoretically possible to monitor these changes very often, but the mapping agencies do not have the manpower for this. So instead snapshots are created that may be decades apart. To visualize the changes in land cover it is better to produce an interpolation that yields a best guess as to how the borders moved than just do discrete jumps. Land cover within a given area might be updated simultaneously. However, land cover regions neighbouring this area may be updated at entirely different times.

Example 3: Soil type classification: All land regions have a soil type and the classes are usually distinct, thus making this into a partition. Soil type may also change continuously in time. Usually this change is very slow, measured over millennia or more, but in some cases changes may happen far more rapidly, such as when forest cover is removed and erosion becomes much higher than it was.

1.2 Examples of Topology Not Involving a Partition

Here are some examples of situations in which storing explicit topology might be useful even though they do not involve partitions:

Example 4: Visualization of how the landscape has looked in the past. Landscapes change. Rivers alter their courses and lakes grow smaller or larger over time. Glaciers grow and shrink. In this case one not only needs to create temporal version of the objects, but one also needs to "glue" them together to avoid noticeable inconsistencies. For instance, if a river flows out of a lake, it should end at the lake rather than just next to it or slightly inside it.

Example 5: In an ordinary map database, one might have a glacier that ends over a lake. Both the glacier and the lake may grow or shrink over time, but they often border each other.

Example 6: When a major oil spill occurs, one might want to find out if any animals for which its position is known were inside the spill area at any time.

2   Related Work

Representations for topological relationships have been studied extensively for purely spatial databases as well as for spatiotemporal databases with step wise discrete changes. This section describes those earlier works that this paper directly builds on.

[10] describes a temporal topology that he calls "tropology". This tropology describes the possible chains of events that may occur for a single object. It does not,

however, deal with multiple connected objects and does not assume any particular storage model.

A system for reasoning about changes in the topological relationships between moving objects is described in [3].

[7] describes a discrete model for independent spatiotemporal objects. This model is based on time slices. A time slice is a period of time in which the object moves according to a simple function. Points move linearly. The end points of line segments can move like other points except that the line segment is not allowed to rotate. Area objects are represented by their border lines. A rotating line segment is represented as two line segments that shrink to points in one or the other end of the time slice.

[12] describes a method for generating the representation of faces (area objects that consist of only one connected component) from [7] using snapshots of the faces.

For pure spatial data several vector representations have been created that represent topology explicitly. The Node-Arc-Area (NAA) representation that is presented in for instance [13] is one of them. In the NAA representation, lines store the area objects that are to the left and right of the line as well as their start and end points. The border of an area object is thus defined in the line objects. Thus if the border of an area is updated, the borders of its neighbours are also automatically updated.

[11] describes how to implement topological relationships on complex regions using plane-sweep algorithms on a realm. That paper says nothing on how to make that realm accurate from the outset.

[6] formally describes how to handle topological predicates in spatiotemporal database systems. Their basic method is to *lift* spatial predicates to the spatiotemporal case. Lifting a predicate converts a spatial predicate into a spatiotemporal one with a temporally varying output. For any time instant the output of the lifted predicate is the same as for the spatial predicate with the same objects at that time instant.

They then define quantification on these such that one might ask whether the predicate is true at some point in the time interval or over the whole time interval. They then use the universally quantified operations to define temporal aggregations, for example *Enters*. A point *p Enters* a region *R* if *p* starts outside the region, then meets it and then is inside it. They further show that such lifting and defining temporal predicates gives a far more expressive language than using the basic Egenhofer relations from [4] on 3D objects.

[2] defines spatiotemporal objects by an initial snapshot and a transformation function. Some closure properties of this model are then analysed. For instance, for a linear transformation function, the model is closed under union, intersection and difference for rectangles, but only under union for arbitrary polygons.

[1] and [8] define discrete models for spatiotemporal data based on constraints. A convex region is defined as a set of linear constraints (such as: $x + 2y \leq 5$). A non-convex region is defined as a union of convex regions. The advantage of such a system is that it can be easily extended to arbitrary dimensions and can use ordinary relational algebra operations to express many spatial operations that need special operators in abstract data type approaches.

One disadvantage of this approach is that it does not take topology into consideration. A region is stored as a set of linear constraints, but their model has no way of indicating explicitly that a given region shares a set of line segments (equivalent to linear constraints) with another region. This means that if one updates one of the regions, there will be an inconsistency unless the relationship is somehow stored

explicitly. One can only treat topological relationships indirectly and this makes inconsistencies much more likely.

Another problem is that it is very difficult to find the border of a region from the region representation. Changing the constraints from $x + 2y \leq 5$ to $x + 2y = 5$ is not enough as a conjunction of such constraints is, in general, unsatisfiable. Rather the individual constraints must be converted into line segments, each defined by three constraints (one for the infinite line and two for the end points). The border line is the disjunction of these line segment constraints.

3   Possible Models

In this section three models for storing topological information for moving objects are described and analysed. The first subsection discusses which relationships to store and says something about what needs to be stored for points, lines and regions. The subsequent three subsections describe three different models that might be used to store the shared boundaries given in the first subsection.

3.1   Which Relationships to Store

The relationships that need to be stored explicitly are those which involve the borders of regions or lines as the borders are infinitely thin and even tiny errors may change the result of the corresponding predicate. The following definitions of the relationships from [4] will be used:

Table 1. Topological Relationships

Operation	Meaning
Disjoint	The two objects do not share either border or interior
Meet	The two objects share borders but not interiors
Overlap	The two objects overlap. This means that the interiors overlap and the borders cross each other
Overlap with disjoint border	The interiors of the two objects oevrlap but the borders do not cross
Cover	One object is inside the other but shares a part of its border
CoveredBy	The reverse of Covers
Inside	One object is entirely inside the other
Contain	The reverse of Inside
Equal	The two objects have equal shapes

For a pair of regions the relationships **meet**, **covers**, **coveredBy** and **equal** involve the border and therefore must be handled explicitly. The other predicates (**overlaps**, **overlaps with disjoint border**, **inside**, **contains** and **disjoint**) deal only with the interiors of the regions and can therefore be computed using the geometry of the objects.
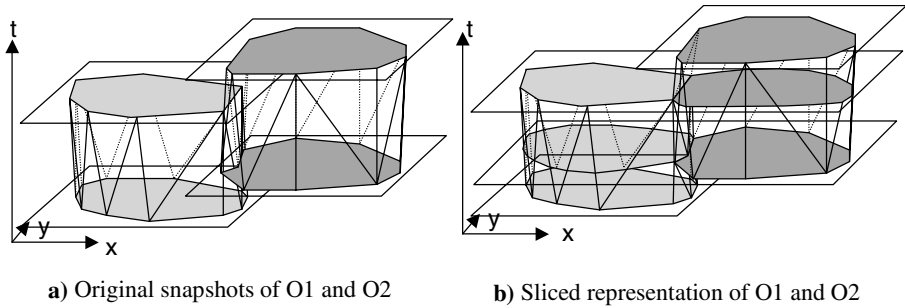
a) Original snapshots of O1 and O2   **b)** Sliced representation of O1 and O2

Fig. 1. Time slices with meeting relationship

For a point and a region, **meet** needs to be stored explicitly while **disjoint** and **inside** can be computed from the geometry. **Meet**[1] can be computed from the geometry if the point moves from outside the region to inside. However, the point may also stay on the border of the region for some time, or it may graze the region. In these last cases the relationship must be stored explicitly or the database might not recognize it.

For a pair of lines both overlap in the end points and interiors must be stored explicitly as in both cases even tiny errors may cause a different answer.

For a point and a line, one must store explicitly any period of time in which the point lies on the line. Crossings can be computed from the geometry.

The only way to identify shared borders reliably is to store the shared part in a shared location like it is done in the Node-Arc-Area representation. For spatio-temporal data, it therefore becomes important to store shared boundaries between spatiotemporal objects.

Line crossings are points and should therefore be stored as a shared moving point. Point crossings may either be a shared moving point (if the point remains on the line or region border over time) or a single spatiotemporal point (with a single time location rather than moving) if the crossing occurs at a particular time instant.

3.2   Time Slices

To maintain the spatial topology in a time slice model, all neighbouring objects must have the same time slices. Inside a time slice the shape of the object is linearly interpolated. Therefore, for each time there is a new snapshot for one of the neighbouring objects, the interpolation of all must change if the borders are to remain equal. Changes in topology should only be allowed between time slices.

The main problem with this approach is that it results in a lot of time slices that are unnecessary for each region but necessary for the whole. One cannot assume that all the regions are updated at the same time (If they were, the time slice approach would work fine). This problem in illustrated in Figure 1.

A partial solution to this problem is as follows: Whenever one object has a snapshot and therefore begins a new time slice, make a new time slice for all neighbouring

---

[1] A point meets a region when it is on the border of the region.

objects as well, but not objects that are further away. This means that not all objects must have all time slices, but it does mean that each object must have one time slice for each time one of its neighbours is updated as well as for when the object itself is updated. If an object borders four other objects, it will have 5 times as many time slices as if it were isolated.

In the land cover type this problem might be reduced because land cover is usually updated in large areas rather in individual regions. All land cover regions within a given satellite image is probably updated at the same time. Thus this problem only occurs for those land cover regions that lie on the border between different satellite images.

A time slice model in which all objects have the same time slices may use a basic node-arc-area model in each time slice as topology only changes in the instants between time slices.

One problem with a pure time slice approach is handling temporal topological relationships like *enters* and *crosses*. As the time slices of the two objects are probably different, they cannot be used directly as a basis for the topological relationships.

3.3 3D Model

One way of avoiding the problems of time slice models is to use general 3D data types to store moving objects. However, this would require a new model for moving objects as well as a new algorithm for creating moving regions from snapshots.

One drawback of 3D types when compared to a sliced representation is that you lose the direct correspondence between the non-temporal and moving object types. However, one needs only fairly simple operations to extract snapshots from the 3D data types.

Time slices have one other advantage: When you query about a time instant or short time interval, the database only needs to fetch those time slices that are relevant for the query, which may be only a small fraction of the total. For a pure 3D model, on the other hand, the entire geometry of the object needs to be fetched.

3.4 Hybrid Model

This last advantage of time slices may be retained if one defines a hybrid model that looks as follows:

At certain instants in time (typically when there is a snapshot available) the shape of the object is stored. The intervals in between these time instants are time slices in which the shape of the object is inferred. However, rather than using just a linear function to infer the shape, the shape may be represented by any 3D surface that can be represented by a set of 3D triangles and that yields a legal 2D object at all time instants in the time slice. The sliced representation described in [7] would be a special case of this representation.

4   Defining the Hybrid Model

In this section, the hybrid data model is defined in more detail. In Section 4.1, a set of 3D data types is defined. These are then used as building blocks for the hybrid model. The hybrid data types are defined in Section 4.2.

4.1   Building Blocks of the Hybrid Data Types

In this section, data types for 3D points, lines and triangles are defined. Additionally, types for time intervals and temporal line segments are defined.

3D Point: *A 3D point is defined by its coordinates:*
$$3DPoint \equiv \{(x, y, t) | (x \in \Re \wedge y \in \Re \wedge t \in \Re)\}$$

3D Line Segment: Each line segment has two end points:
$$3DLineSegment \equiv \{(s, e) | (s \in 3DPoint \wedge e \in 3DPoint)\}$$

3D Triangle: To make the algorithms for computing intersections easier, surfaces should be piecewise straight. The only way to ensure a piecewise straight surface is to create it as a set of triangles. Therefore, the 3D surface element of this model is a triangle:
$$3DTriangle \equiv \{(p_1, p_2, p_3) | (p_1 \in 3DPoint \wedge$$
$$p_2 \in 3DPoint \wedge p_3 \in 3DPoint)\}$$

Time Interval: A time interval is a connected set of numbers with a given start and end:
$$Interval \equiv \{(s, e) | (s \in \Re \wedge e \in \Re \wedge s < e)\}$$

Temporal Line Segment: A temporal line segment is a 3D line segment where the end points have ascending times:
$$TempLineSeg \equiv \{s | (s \in 3DLineSegment \wedge s.s.t < s.e.t)\}$$

4.2   Definitions of Spatiotemporal Data Types That Store and Enforce Meeting Relationships

In the hybrid approach outlined in Section 3.4, moving objects are represented by time slices. However, the interpolation between the snapshots is more general than the one given in [7]. For most of the types, all that needs to change is still the definition of the **unit**, or time slice. The overall type can in most cases remain unchanged from the type from [7].

In these definitions, the units are given a semantic meaning of their own. A unit is assumed to be the period between two updates of the object. Thus for every time a moving curve is updated, a new curve unit is added. The same applies to points and regions. Examples of the point and line types are given in Figure 2 and Figure 3. Examples of the face types are given in Figure 4.

4.3   Moving Point

The **moving point** may be modelled as a set of polylines that for each time instant gives only a single point.
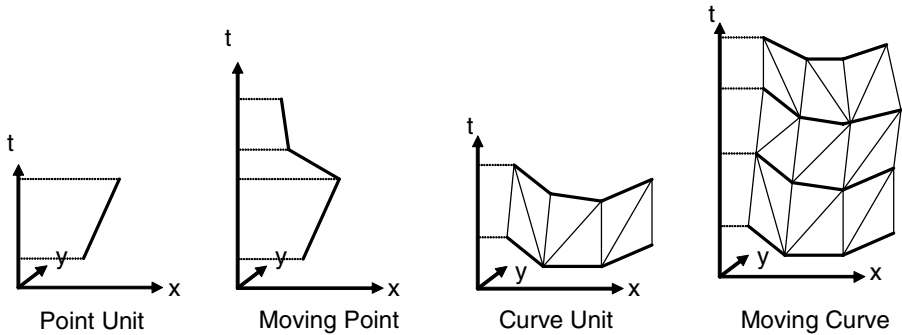
Fig. 2. Moving Point and Curve

The moving point may serve two functions: It may be an *independent* database object in its own right (such as a car, building or marked animal), or it may represent the *meeting point* of several moving curves or regions. These two functions have different storage needs. In this definition, both these types are combined into the single moving point, but it might be argued that they should be different types. They are defined as the same type here to keep the type system small and therefore manageable, and because these two roles are not mutually exclusive.

The following three topological relationships must be dealt with for moving points:

On line: If the point remains on a given moving curve (including the border of a region) over time, this must be stored explicitly as only minor inaccuracies can make the point be outside the line. In most cases the point will be on only one line at a time and for the other case one can conclude that the lines have the same location. The point therefore only needs to store a link to a single line. Since a point is not necessarily on the line during its entire lifetime, this relationship should be stored with the point units rather than in the main point object.

End point of curve: A moving point may serve as a *meeting point* for several moving curves or regions. The most efficient way to store this relationship is to store it in the curves as a curve can have only two end points but a point may be the end point of an arbitrary number of curves. Discovering which curves end in a given point can be done by querying a spatial index with the position of the point. This is guaranteed to return all the curves that end in the point. Any other curves returned can be filtered out by checking their end points.

Meet: Two moving points may have the same position at a particular time instant or in a particular time interval. This relationship may be stored by having the two moving point objects share point units when they are at the same position. Meeting at a time instant can be stored by letting them share a degenerate point unit that is valid only at that time instant.

Temporal meet: Moving points may be connected in time. If for instance one moving point splits into several, there are moving points that meet in time. At the end time of one point it is at the same place as another point object begins.

The **Point Unit** is therefore defined as follows:

$$UPoint \equiv \{(s, c) | s \in TempLineSeg \wedge c \in MCurve\}$$

In this definition, s defines the movement of the point and c represents the *on line* relationship. An *MCurve* is a moving curve and is defined later.

The **Moving Point** is defined as follows:

$$MPoint \equiv \{(U, M) | U \subset UPoint \wedge M \subset MPoint \wedge$$
$$\forall (a, b \in U):(a \neq b) \rightarrow \neg temp Overlap(a.s, b.s) \wedge$$
$$\forall (p \in M):tempMeet(this, p)\}$$

Where *tempMeet(p1, p2)* is true iff the valid time of *p1* meets the valid time of *p2* and the two points are at the same location at that time. *tempOverlap(s1, s2)* is true iff the valid times of *s1* and *s2* overlap.

In this definition, U is the set of point units that make up this moving point and M is the set of points that *temporally meet* this point.

4.4   Moving Line

The **moving line** type from [9] is defined as a set of moving curves. According to [7], any set of moving line segments makes a valid moving line according to this definition. However, it would be very difficult to deal with topological information with such a construct. It would have no end points, and the border curve of a region is the ideal place to store a pointer to a neighbouring region.

As for a moving point, a moving line may serve two purposes. It may be an *independent* database object in its own right, or it may mark the *border* between two particular regions.

A simple and straightforward definition of a moving curve would be this: A moving curve is a 3D surface consisting of planar facets whose intersection with any plane parallel to the x-y plane would be a valid curve (continuous set of line segments). A moving line could then be defined as a set of moving curves. The moving curve may additionally have to store the following topological information:
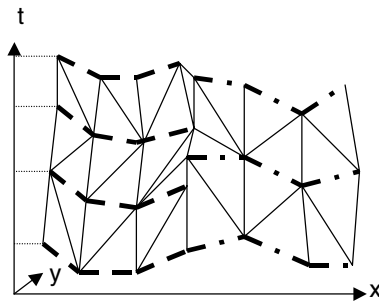

Fig. 3. Moving Line

Bordering regions: The curve may be a part of the border of up to two regions. This is stored in the main curve object. This makes a curve that serves as a *border* represent the border between two particular regions.

End points: Any curve has two end points in space. However, these points only needs to be stored explicitly if they serve as *meeting points* for several curves.

Points on line: If one wants to query which cars are on a particular road, one may want to store which cars are on the road at any given moment. However, this information may take up a lot of storage space and can also be discovered through a spatial search of the points combined with the **on line** relationship for points. It is therefore not necessary to store directly.

Meet: Two *moving line* objects may share *moving curve* objects. If this relationship changes over time, it is handled the same way as two regions that stop bordering one another.

Temporal meet: If two area objects that used to border each other no longer do, then the border curve should split into two new curves, one for each area object. These new curve objects should store the fact that they are continuations of an old curve.

The **curve unit** can be defined as follows:

$$UCurve \equiv \{(vt, T) | vt \in Interval \land T \subset 3DTriangle \land \\ \forall (t \in vt):(AtInstant(t, T) \in Curve)\}$$

The *AtInstant* function creates the intersection between a flat plane at a given time and a given set of 3D objects. The *Curve* type represents a non-temporal curve.

The **moving curve** is a set of curve units:

$$MCurve \equiv \{(C, e1, e2, f1, f2, TM) | C \subset UCurve \land \\ e1 \in MPoint \land e2 \in MPoint \land \\ f1 \in MFace \land f2 \in MFace \land TM \subset MCurve \land \\ \forall (a \in C)\forall (b \in C):(Overlap(a.i, b.i) \rightarrow (a = b)) \land \\ endPoint(e1) \land endPoint(e2) \land \\ borderFace(f1) \land borderFace(f2) \land \\ \neg Overlap(f1, f2) \land \\ \forall (mc \in TM):TempMeet(this, mc)\}$$

In this definition, *endPoint*(*p*) indicates that the point is an end point of this curve, and *borderFace*(*f*) indicates that the face is bordered by the curve. The *MFace* type represents a moving face and is defined later.

The **moving line** is a set of moving curves. It does not require that those curves have the same time slices. The reason for this is given in the next section. In the example in Figure 3 the different dash patterns indicate different moving curves.

$$MLine \equiv \{C | C \subset MCurve\}$$

4.5   Moving Region

The **moving region** type from [9] is defined as a set of non-overlapping faces. A face is a connected area object that may have any number of holes.

The main topological relationship between faces that should be handled explicitly is **bordering**, that is, which regions border this region. This relationship can be deduced from the *bordering regions* relationship for moving lines.
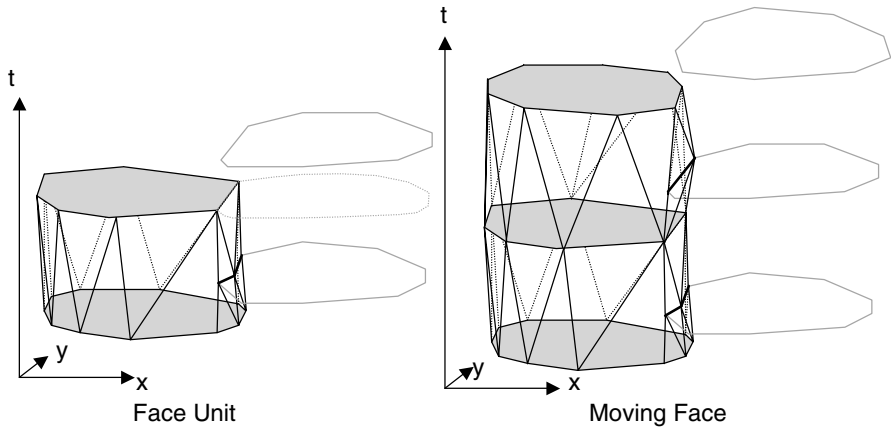
Fig. 4. Moving Face

A **moving cycle** is a set of moving curves that for any time instant in the time period that the cycle object is valid forms a ring. In the models from [9], a moving cycle is considered to consist of only one curve. However, when one wants to store topology it is better to think of a cycle consisting of a set of curves. Each curve in the set represents the boundary between two particular regions. The moving cycle is therefore defined based on the *moving line* type.

$$MCycle \equiv \{(l, vt) | l \in MLine \wedge vt \in Interval \wedge$$
$$\forall(t \in vt):AtInstant(t, l) \in Cycle\}$$

The curve units in each moving curve represent the boundary between two particular region units. Since the regions bordering a particular cycle may be updated at different times and therefore have snapshots at different times, the curve units in each curve in an line must be allowed to have different time slices.

A moving face consists of one moving cycle representing the outer boundary of the face and N moving cycles defining the holes. Discrete changes in the moving face are assumed to happen in the instants between time slices. All the cycles in a given face unit should be valid in the time that the face unit is valid because changes in topology such as the appearance of new holes should only occur between time slices. The **face unit** is therefore defined as follows:

$$UFace \equiv \{(oc, HC, vt) |$$
$$oc \in MCycle \wedge HC \subset MCycle \wedge vt \in Interval \wedge$$
$$\forall(hc \in HC):(Inside(hc, oc) \wedge (hc.\text{vt} \supseteq vt)) \wedge$$
$$(oc.\text{vt} \supseteq vt)\}$$

In this definition, *Inside*(*a* ,*b*) is true iff *a* is inside *b*.

Notice that in this definition the face units may be different from the units of the moving curves. This is because a curve has a new unit whenever one of the faces that it borders is updated. Thus if curve *a* borders faces *b* and *c*, *a* has a number of units equal to the total number of units of *b* and *c*.

A **moving face** is defined as a non-overlapping set of face units:

$$MFace \equiv \{UF | UF \subset UFace \; \wedge$$
$$\forall(a \in UF)\forall(b \in UF):(Overlap(a.\text{vt}, b.\text{vt}) \rightarrow (a = b))\}$$

A **moving region** is a set of moving faces. These are not required to have the same time slices. If a region consists of several faces one is not guaranteed that snapshots of all the faces from the same time exist.

$$MRegion \equiv \{F | F \subset MFace \; \wedge$$
$$\forall(a, b \in F):(a \neq b \rightarrow Disjoint(a, b))\}$$

5   Constructing the Hybrid Model

This section presents a method for constructing the hybrid model of moving regions from a series of snapshots of the individual regions. This method assumes that the regions are updated periodically but not necessarily simultaneously. The topological relationships between the regions are stored in an adjacency graph. All topological relationships should be maintained unless this graph is explicitly changed.

Modelling a partition and modelling a network are two sides of the same coin as the borders between the regions in a partition forms a network. A model for one also works for the other. This also applies to partial partitions (ones that cover only a part of the space of interest).

5.1   Constructing a Moving Partition

When creating the component objects of a moving partition, it is easier to interpolate the border curves than the regions themselves. One approach based on regions would be to interpolate each region separately and then ensure that their borders meet. Writing a procedure that could do this and ensure consistency in places where more than two regions meet would be quite complex. Therefore, the algorithm presented here for creating a moving partition is based on interpolating the border curves rather than the regions. The algorithm is also based on an adjacency graph[2] supplied by the cartographer. This adjacency graph must be explicitly updated when the topology changes.

The algorithm assumes that there is a pre-existing partition that one wants to update. When creating a partition for the first time, one runs a similar algorithm for each cycle in the adjacency graph of the initial partition as well as for each edge that does not belong to a cycle.

When modifying the regions so that they fit together in the partition, the system always stores the original versions as well as the modified ones. The original versions are used for interpolations whenever a new version of a region is inserted. This is dome to ensure that the interpolated versions of the regions stay as close to the original as possible, especially if one region is updated several times while another is not.

---

[2] An undirected graph with one node for each face and an edge between each pair of faces that border each other.

Algorithm. *UpdatePartitionInterpolation*(*nf*, *FS*, *ag*)

Input: A new face snapshot *nf*, the set of faces in the partition *FS*, and an adjacency graph for the faces *ag*.

Output: The face set with a new snapshot added

Method:

Let *of* be the previous snapshot of *nf*

Let *fn* be the node in *ag* that represents *of*

Let *mf* be a copy of *nf*

For each cycle in *ag* that contains *fn* **do**

> Compute a meeting point for all the faces in the cycle using the original rather than the modified snapshots. This is done by adding a buffer of the same size for all the faces (if there is a gap) or subtracting an area of the same size for all the faces (if they all overlap).

End for

For each edge in *ag* that ends in *fn* **do**

> Construct a meeting line between the two faces by adding a buffer to the other face and removing overlap until they meet. For each side where there is a cycle ensure that the lines end in the meeting points.

> Update *mf* by replacing the original line with the new meeting line (See Section 5.2)

End for

Interpolate the lines of the face *mf* from their version in *of*

Add the interpolation and *mf* to *FS*

Add *nf* to *FS* as the original face (for use in later runs of this algorithm)

return *FS*

End *UpdatePartitionInterpolation*

The adjacency graph is supplied by the cartographer who knows which regions are supposed to border each other. Updates to the topology is reflected in updates of the adjacency graph. The following updates are possible:

- Removing an edge
  - At the edge of the partition: This region no longer borders that region. The interpolation system should ensure that from the point in time in which the edge was removed there is a small gap between these two regions. (Regions in the adjacency graph should not overlap. A region may be added to the graph as an isolated node to indicate that it should not overlap any of the other regions).
  - In the middle of the partition: These two regions no longer border each other. Reduce the meeting line to a point at the time in which the edge was removed. This point may then expand into a line between a new pair of regions. At the instant when the line is removed neither it nor any newly inserted lines exist thus forming a possibly temporary cycle.
- Removing a node
  - As nodes correspond to faces, this means that a face no longer exists. Insert a single point representing the face as a new version and interpolate neighbouring faces to this point. This point is the new meeting point of the eventual new cycle created by removing the node. All the edges from that node are also removed.

Adding an edge or node is just like removing one except that the process is reversed in time.

Whenever the adjacency graph is updated, all affected faces must have a new time slice, where the state at the beginning of this new time slice is the state according to *UpdatePartitionInterpolation* after the change in the graph.

5.2   Constructing a Moving Network

A network may consist either of nodes (points) with curves between them (like graphs except that the shape of the curves may be important) or routes and intersections. The difference between these two is that the routes-and-intersections model can be equivalent to a non-planar graph as routes may cross each other without intersecting (one road goes in a tunnel below another with no means of switching from one to the other).

If the entire network is updated at the same time instants, representing a changing network is trivial. A network unit is simply a collection of moving curves which end in the same moving points (nodes). Changes in topology (removal of edges or nodes, merging of nodes and edges) happen only between the time slices (at the end of one and beginning of the next) in this model.

When the curves are updated individually, they are interpolated as normal. Additionally, a new version of the end points are stored. The end points are interpolated in time between all the end points of all the curves that meet there. Thus the meeting point of four curves would be updated whenever any of the four curves is updated. This means that the final line segment of each curve changes more often than the other lines, and the interpolation of these line segments cannot use the normal algorithm from [12]. One alternative method would be this:

- Create the projection of the final line segments and the changing end point on a plane that is parallell to the time axis and has an angle to the x-y axis that is the average of the angle of the final line in the two snapshots.
- Create the Delaunay triangulation of the projected lines.
- Use this as the triangulation of the final line segments. Going back to a full 3D representation is easy since the triangulation does not insert any new points and the 3D coordinates of the points used are known.

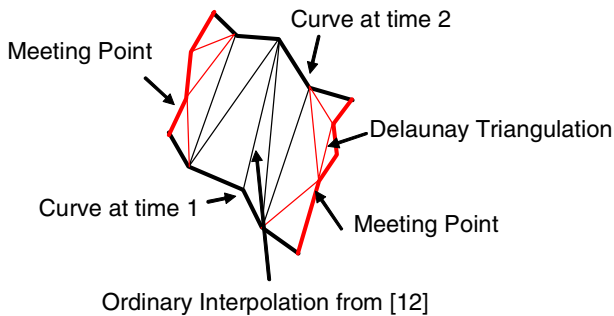One example of such a curve and the proposed interpolation algorithm is shown in Figure 5.

Fig. 5. Interpolating a line in a network

For networks consisting of routes and intersections, a modified version of this method can be used. Each route is equivalent to a connected set of edges and each intersection is equivalent to a meeting point. Places where routes cross but there is no intersection are not represented as points. This is a simple way of distinguishing such crossings from regular intersections.

6 Handling Current Time

The database described so far handles historical spatiotemporal data quite well. However, when asked about the current state, it can only return the last state of the various polygons, and this state can be inconsistent for those polygons which have not been updated for some time. The most straightforward method is to assume that the objects are static after the last update. This method works well if the objects do not move too much. The method basically goes as follows:

- Take the most recent snapshot in the area of interest.
- Insert a new time slice of each object that extends from the last snapshot of the object to the current time. The object is considered to be static in this time slice.
- In this new time slice, all the moving points and curves are static and do not change from their most recent state. This includes the curves that form the boundaries of moving regions.

This is a fairly simple method that works well in many cases. However, it does not take the movement of the objects into account and may therefore produce highly inaccurate results in cases in which the objects move fast.

One alternative is to try to extrapolate them based on past movement. To do this for a general line or region is quite complex. A naïve approach would be to extrapolate based on the movement of the individual points that make up the line or region border. This produces artifacts like shown in Figure 6.
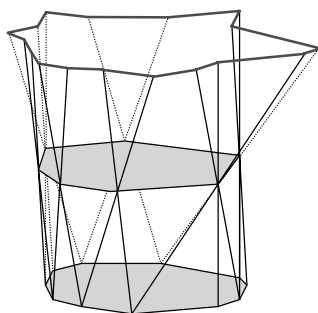

Fig. 6. Extrapolating using a Triangle Representation

7 Implementing the Model

In this model many objects should have references to each other. For instance, the region object stores its boundary as a set of curve objects. This ensures consistency,

but it also creates a problem. To fetch the geometry of a region, one must access all the curve objects that make up its boundary, and all the point objects that make up the meeting points for those curves. Unless these are stored together, this may slow down the system. Border curves cannot be stored together with both of the regions that they border as these regions may be stored in entirely different places on the disk.

An alternative is to store several copies of these objects. For the curve, one would store one copy for each of the two regions that have it as part of their boundaries. This would make retrieval more efficient, but would introduce the possibility of inconsistencies. To solve this one would need a database system that could handle integrity rules of the type "Objects A and B should always be equal". Such a system could then enforce this by always updating both curves when one is updated. This would increase the cost of updating as well as storage cost, but would reduce query costs.

For many of the relationships, such as a point serving as an end point, the relationship is stored in only one object, typically the object for which the relationship has the lowest cardinality. This is common design practise in relational databases and helps to ensure consistency. However, one may choose to store the relationships both ways instead. This will increase storage cost as well as update cost to avoid inconsistencies, but may reduce query cost.

[6] defines two relationships as examples of temporal topological relationships:

- **Enters:** The object starts outside the region and moves inside it.
- **Crosses:** The object enters the region and later leaves it.

The time slices in the hybrid model may be used to reduce the amount of data that needs to be fetched to do these computations. If each time slice has its own spatio-temporal bounding box, an index may be used to find those time slices of each object in which they may overlap. Then only those time slices need to be used to compute the temporal topology as for all other time slices the objects are known to be **disjoint**.

8   Summary

This paper has presented an extension to the sliced representation from [7] that is capable of representing explicit topology. This is necessary for several important operations, including checking whether two regions border each other. The new representation is slightly more complex than the original representation, but many of the same algorithms are applicable to both.

References

1. J. Chomicki and P. Z. Revesz: Constraint-Based Interoperability of Spatiotemporal Data-bases. In *Proc. 5th Int. Symp. on Large Spatial Databases*, pp. 142-161, 1997.
2. J. Chomicki and P. Z. Revesz: A Geometric Framework for Specifying Spatiotemporal Objects. In *Proc. 6th Int. Workshop on Temporal Representation and Reasoning (TIME'99),* pp. 41-46, 1999\

3. M. J. Egenhofer and K. K. Al-Taha: Reasoning about Gradual Changes of Topological Relationships. In A. Frank, I. Campari, and U. Formentini (eds.): *Theory and Methods of Spatio-Temporal Reasoning in Geographic Space*, vol. 639 LNCS, Springer-Verlag, pp. 196-219. 1992

4. M. J. Egenhofer and R. D. Franzosa: Point-Set Topological Spatial Relations. In *Int. Journal of Geographical Information Systems*, 5(2), pp. 161-174, 1991

5. M. Erwig and M. Schneider: The Honeycomb Model for Spatio-Temporal Partitions. In *Proc. Int. Workshop on Spatio-Temporal Database Management*, pp. 39-59, 1999

6. M. Erwig and M. Schneider: Spatio-Temporal Predicates. In *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 14(4), pp. 881-901, 2002

7. L. Forlizzi, R. H. Güting, E. Nardelli, M. Schneider: A Data Model and Data Structures for Moving Objects Databases. In *Proc. ACM SIGMOD Int. Conf. on Management of Data* (Dallas, Texas),pp. 319-330, 2000

8. S. Grumbach, M. Koubarakis, P. Rigaux, M. Scholl, S. Skiadopoulos: Spatio-Temporal Models and Languages: An Approach Based on Constraints. In *Spatio-Temporal Databases - the CHOROCHRONOS Approach*, LNCS 2520, Springer-Verlag, pp. 177-201, 2003

9. R. H. Güting, M. F. Böhlen, M. Erwig, C. S. Jensen, N. A. Lorentzos, M. Schneider, M. Vazirgiannis: A Foundation for Representing and Querying Moving Objects. In ACM Transactions on Database Systems 25(1), 2000.

10. A. Renolen: Concepts and Methods for Modelling Temporal and Spatiotemporal Information. *Dr. Ing Thesis,* Norwegian University of Science and Technology (NTNU), 1999

11. M. Schneider: Implementing Topological Predicates for Complex Regions. In *Proc. 10th Int. Symp. on Spatial Data Handling (SDH),* pp. 313-328, 2002

12. E. Tøssebro and R. H. Güting: Creating Representations for Continuously Moving Regions from Observations. In *Proc. 7th Int. Symp. on Spatial and Temporal Databases*, pages 321-344, 2001

13. M. Worboys: *GIS: A Computing Perspective.* Taylor & Francis, 1995

UMN-MapServer: A High-Performance, Interoperable, and Open Source Web Mapping and Geo-Spatial Analysis System

Ranga Raju Vatsavai[1,4], Shashi Shekhar[1], Thomas E. Burk[2], and Stephen Lime[3]

[1] Department of Computer Science and Engineering, University of Minnesota
EE/CS 4-192, 200 Union Street. SE., Minneapolis, MN 55455
{vatsavai, shekhar}@cs.umn.edu

[2] Remote Sensing Laboratory, Department of Forest Resources, University of Minnesota.
115, Green Hall, 1530 N. Cleveland Ave, St. Paul 55108
teb@mallit.fr.umn.edu

[3] Minnesota Department of Natural Resources, St. Paul, MN 55155
steve.lime@dnr.state.mn.us

[4] IBM Research Labs, Block 1, Indian Institute of Technology, Hauz Khas, N. Delhi. 110 016

Abstract. Recent advances in Internet technologies, coupled with wide adoption of the web services paradigm and interoperability standards, makes the World Wide Web a popular vehicle for geo-spatial information distribution and online geo-processing. Web GIS is rapidly evolving and adapting to advances in Internet technologies. Web GISes are predominantly designed under a "thin-client / fat-server" paradigm. This approach has several disadvantages. For example, as the number of users increases, the load on the server increases and system performance decreases. Recently the focus has been shifted towards client-side Web GISes, which are heavy-duty, stand-alone systems. We take an opposing approach and present a load balancing client/server Web-based spatial analysis system, UMN-MapServer, and evaluate its performance in a regional natural resource mapping and analysis (NRAMS) application which utilizes biweekly AVHRR imagery and several other raster and vector geo-spatial datasets. We also evaluate alternative approaches and assess the pros and cons of our design and implementation. UMN-MapServer also implements several open standards, such as, WMS, WCS, GML and WFS. In this paper, we also describe in detail the WMS, WCS, and GML extensions from the interoperability point of view, and discuss issues related to adoption of such standards.

Keywords: WebGIS, WMS, WMT, GML, WFS, Load-balancing Architecture.

1 Introduction

There has long been a need for Web GIS, and our efforts to develop an efficient means of delivering spatial data products on the Web dates back to the ForNet [8] project. The tools developed under this project, namely ImageServer and MapServer [13], became quite popular in the GIS community. Although, these initial prototypes were limited in functionality, they demonstrated an important application the user community, that spatial data can be queried and visualized over the Web. The success of the ForNet project

led to another collaborative research project, called TerraSIP, [23] between the University of Minnesota, NASA, and a consortium of land management organizations. One of the main objectives of TerraSIP is to investigate ways of efficiently implementing a general purpose, Internet-based system for the analysis of geo-spatial data. This investigation led us to the development of the Web-based Browsing and Spatial Analysis System (WeBSAS), which is an extended version of UMN-MapServer. Advancements in the Internet development environments have helped us to develop an innovative load-balancing client server architecture and incorporate for the first time some online geo-spatial analysis capabilities into the WebSAS.

In this paper, we describe and validate this load-balancing high-performance architecture and as well describe extensions based on three different OGC specifications. We use the terms MapServer, WeBSAS and UMN-MapServer interchangeably. Preliminary work on MapServer can be found in [26,21].

1.1 Contributions of This Paper

Our research on WebGISes has resulted in several innovative and practical solutions in the form of enhanced UMN-MapServer. The contributions range from a novel "load-balancing architecture" (which facilitates on-line geo-processing) to efficient integration of open standards and open software components into the MapServer. Specifically, the contributions fall in four areas.

Innovative Architecture: As part of this research, we have identified the core components of WebGISes that affect the overall performance of a WebGIS application. We have developed an innovative architecture for WebGIS and implemented it as a load-balancing client/server system in the UMN-MapServer. This system was prototyped as the core of a demanding Web-based natural resource analysis and mapping application (NRAMS). This system gave better performance than either client-centric or server-centric WebGISes.

Online Geo-processing: Geo-processing is often compute intensive; as a result, typical WebGISes are limited to simple query processing and on-line visualization of geo-spatial datasets. However, in this paper we show that with innovative algorithms, several compute intensive tasks can be implemented in the WebGIS framework.

Interoperability: The main hindrance for building true interoperable distributed geographic information systems is the lack of any standard exchange mechanism between the diverse GISes connected over the web. Recent efforts by the OpenGIS Consortium (OGC) have resulted in several specifications to alleviate these problems. Web Map Service (WMS), Web Coverage Service (WCS) and Geographic Markup Language (GML) are such standards for developing interoperable Web based Geographic Information Systems (Web-GIS). GML is an XML (eXtensible Markup Language) encoding for the transport and storage of geographic information, including both geometry and properties of geographic features. GML has great promise as a medium of geographic data exchange between disparate WebGISes and building client-centric systems. However, query processing over GML documents incurs huge performance penalty. We evaluated two well-known parsing approaches – simple API for XML (SAX) and the document object model (DOM) for single and multiple passes over GML documents. Our study shows that SAX performs better than DOM for single passes; thus for simple

applications like visualization and subsetting, the SAX model is superior. However, for intensive applications involving queries requiring multiple passes over documents or integration of multiple documents in a distributed environment, DOM based parsing offers a better solution. On the other hand WCS specification is geared toward delivering GIS raw data over the Web in a standard way. We have successfully implemented WCS specification in the MapServer and demonstrated its utility in browsing, subsetting, and downloading raw satellite imagery over the web.

Applications: UMN-MapServer has been heavily used and tested in a plethora of WebGIS applications over a decade. So, apart from evaluating performance and developing cost models, we extensively tested UMN-MapServer in real-world applications. In this paper, we describe the NRAMS application, which highlights some of the innovative and high-performance features of our system.

The remainder of this paper is organized as follows. Section 2 surveys current WebGISes and evaluates the critical components. Section 3 describes UMN-MapServer, its design and implementation. Section 4 describes online geo-spatial processing (OLGP) capabilities and Section 5 deals with the interoperability features of the MapServer. The Natural Resource Analysis and Mapping System (NRAMS) application is described in Section 6. Finally, in section 7 we provide conclusions and future directions.

2   Related Work

The Internet and the Web are changing the ways in which spatial data are accessed and disseminated. The Web has had a great impact on many fields, and GIS is no exception. First we will review existing Web GISes from the technology perspective. The main components of any Web GIS are the client, the server, and the network. Initial developments concentrated on map visualization [15] and were quickly followed by query capabilities [25], [13]. The earliest Web GISes appeared in 1995, with the first noticeable contribution being GRASSLinks [20], based on the public domain GIS and image processing system GRASS. The initial response of the geo-spatial technology industry to the growing popularity of the Internet was to develop Common Gateway Interface (CGI) wrappers to their standalone GIS software. This resulted in thin-client/fat-server systems. In this design, the server is overburdened with both data access and geo-spatial analysis tasks. The Web browser provides access to these servers. Ignoring performance issues, such systems are the best choice since the user does not need to install any additional resources on his local machine. But this solution becomes impractical, as the server cannot handle a large volume of online users.

In recognition of this limitation and with advancements in development environments, the focus of development gradually shifted towards client-side GIS. Client-side GIS is now practical due to various enabling technologies like Java applets, ActiveX Controls, and extendible Web clients. Several researchers have explored client-side GIS architectures [1] [19]. This approach is very promising and appealing due to progress in Internet programming environments, namely Java. Java provides an architectureneutral platform, which is essential given the heterogeneous hardware and software environment of the Internet. Java applets are a kind of mini application designed to be run in any modern Web browser or applet viewer. Applets are written only once and

reside on a server. They are downloaded along with the data upon a user request and are executed on the user machine connected to the Internet with a Java-enabled Web browser. However, client-side GISes also share some of the limitations of Server-side GISes. Firstly they are heavy-duty and stand-alone, secondly they do not exploit the fact that geo-spatial computation can be distributed between client and server. We developed an innovative architecture that exploits the nature of a given geo-processing task (e.g., compute intensive or communication intensive) to determine the best location (server or client) to perform the task. Our research into Load-balancing WebGIS architectures, GML query processing, and WCS extensions are among the very early works in this area.

3   Architecture of UMN-MapServer

The architecture of MapServer is based on a "balanced client/server" paradigm, as opposed to client-centric or server-centric approaches. We will explain the balanced nature of MapServer in the following sections. MapServer follows a standard 3-tier architecture as shown in Figure 1. The three layers are the client, the server, and the geo-spatial database access system (GSDAS). The server layer is further subdivided into the CGI, geo-spatial analysis and communication subsystems. For detailed description see [26].

3.1   Client/Server Interaction - A Load-Balancing Approach

The distinction between the "thin-client/fat-server" and "thick-client/thin-server" paradigms lies in whether the geo-spatial analysis functionality is supported on the server or the client respectively. One of the major reasons for the initial "thin-client/fat-server" systems is the fact that HTTP/1.1 is a stateless and connectionless protocol. By connectionless we mean that the client and server are connected only for the duration of the transaction. Stateless refers to the fact that the server forgets about the client once the transaction is over. Another reason for the development of these systems is the fact that the Web client does not understand anything about geographic elements. As a result, in such systems all of the data access and processing is performed on the server with the results being converted into an image (e.g. JPEG or GIF) and passed to the Web client.

However, recent advancements in development environments have made it possible to extend the Web client in the form of plug-ins and applets. Plug-ins are software programs that extend the capabilities of the browser in a specific way. For example, audio plug-ins allow users to play music files and video plug-ins allow the user to watch movies online. Recently, several commercial vendors began offering GIS plug-ins: MAPublisher [14] from Adobe, MapGuide [12] from Autodesk, and Vdraft [28] from SoftSource. Applets are simple application programs downloaded from the server, along with the data, which run on a client machine and act on the data to produce a desired output. Even though both of these mechanisms extend the capability of the client to understand, process, and render geographic elements locally, they possess subtle differences.
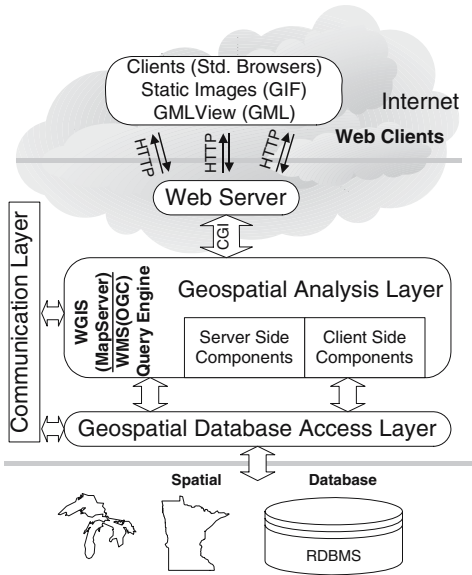
Fig. 1. WMS Compliant MapServer Architecture

Applets are machine and operating system independent programs, whereas plug-ins are specifically designed for particular platforms, operating systems, and browsers. Plug-ins must first be installed on the client machine, whereas applets are downloaded along with the data only when a request is made. Plug-ins have the advantage that they can access local data, whereas applets have restricted access to resources on both the client and the server for security reasons. Both of these approaches enable the design of "thin-server/thick-client" systems, facilitating geo-spatial analysis on the client. Under this design, a thin-server can be implemented using some simple CGI scripts which parse the URL to reformulate the client request into a database query or a function call. These calls can then be submitted to a standard GIS or a specifically designed GIS server.

Alternatively, one could design a customized GIS server which internally implements HTTP and parses the URL to service the client request (e.g. ArcIMS). These client/server configurations are two extreme approaches to realizing Web GIS. Both place a heavy burden on either the client or the server. Careful study, however, reveals that certain operations will give better overall performance if they are applied on the server rather than the client, and vice versa. The main focus of our present research is to analyze and identify these operations and determine ways of implementing them in a well-balanced client/server Web-based spatial analysis system which gives better response and throughput.

For the purposes of this analysis, we define the following terms. When a request (GET/POST) from the client is submitted to the Web server, the following general operations occur. The Web server parses the request and reformulates it as a query suitable for the application server (e.g. GIS or RDBMS). The application server then fetches the data (e.g. a subset of an image or a vector layer), applies the operation (e.g. a geo-spatial analysis or summary of the records fetched from an RDBMS), and transmits the results to the client, which then renders the results in a client window. We assume that any geo-spatial task can be applied on either the client or the application server. Now we will evaluate each of these tasks.

Performance is the most important feature in modern client/server systems [3] and is critical for two reasons: the bandwidth bottleneck of the Internet and the large volume of data to be transmitted in GIS and multimedia applications. Performance of Web GISes can be improved by 1) increasing the speed of Internet connections and 2) developing more efficient software. Network performance has improved dramatically over last five or six years. GIS researchers have focused on the development of efficient

algorithms to improve system performance. For example, [4] presented a progressive vector transmission technique, [32] presented a layer decomposition technique for efficient spatial data transmission, and [6] presented a method for progressive transmission and rendering for 3D-GIS. Progressive transmission and rendering improves performance in the "thin-client/fat-server" model, but implementation on thick-clients is difficult. Similarly, simplification, line generalization, and polygon amalgamation are meaningful only after analyses are performed. Our approach is to minimize communication between client and server by implementing the geo-spatial analysis operation on the server if the resultant output size is less than the input size. To load balance the client/server system and to reduce the communication cost, we apply the following general rules, based on observations of NRAMS system performance and analysis of associated MapServer logs:

"Apply geo-spatial analysis function (f_g) on server, (i) if the computational cost (t_f) is less than communication cost (t_c), (ii) if the resultant data (d_o) is less than input data (d_i), otherwise apply the f_g on client."

Based on the status of a run-time parameter defined in the configuration file (named $< projectname >$.map), MapServer determines where the given geo-spatial analysis task should be launched. If f_g is computed on the server, then the result is sent back to the client. If f_g is determined to be executed at the client, then the data and as well as the code (applet) will be pushed-down to the client.

4   Online Geo-Spatial Processing (OLGP) System

MapServer supports online geo-spatial analysis capability thanks to its load-balancing architecture. For purposes of illustration we give two examples taken from the NRAMS application. The first example is temporal profile generation from satellite images. In order to generate the mean temporal profile for a sample window of 50 rows x 50 columns from 100 multi-temporal AVHRR NDVI images, we would need to transmit 250000 bytes over the network if this operation were to be implemented on the client. Input to the profile is simply the mean value from the window for each of the 100 time periods. The cost of this operation (250000 additions and 100 divisions) is negligible for modern computers. If this operation were performed on the server, then the resulting dataset to be transmitted, only 100 bytes in size, would be significantly less (i.e. 2500 times smaller) than the original dataset. Similarly, a change detection operation (i.e. subtracting the values of one image from another), if performed on the server, would reduce communication by one half. Certain operations, such as filtering, will give the same amount of data output as input, so performance would generally be improved by implementing them on the client. Other operations may result in even more data output than input (e.g. buffer analysis) and hence need to be implemented on the client. MapServer has been extended after careful analysis of many such practical examples. Its analytical capabilities are distributed between server and client based on the objective criterion presented above.

Apart from load-balancing architecture to improve the overall system performance, we need to employ smart algorithms and data structures to further improve the performance of WebGISes. The second example from the NRAMS application illustrates

the importance of data organization in improving the performance. Please refer to our earlier work [27] where we discussed several other approaches to improve the performance.

Subsetting (Range Query): Subsetting is the process of extracting a small portion or region of interest (ROI) from the original source. In general the ROI will be much smaller than the original image. From Web log analyses, we have observed that most user requests result in extracting a small subset of data, typically less than 20% of the original image or vector layer. A good review of several multi-dimensional dimensional access methods can be found in [10]. Our application server, MapServer, uses the well known quadtree indexing scheme.

Most of the raster data utilized in Web applications are single-channel images, generally thematic layers. Multi-spectral (e.g. Landsat, SPOT, and IRS) and multi-temporal (e.g. AVHRR) images require special organization of data for efficient access. Our prototype application, NRAMS, utilizes multi-temporal data, namely approximately 100 biweekly composite AVHRR Normalized Difference Vegetation Index (NDVI) images. Computation of estimated Net Primary Productivity (NPP) for different land cover types from these data generally requires accessing only a small subset of these images, which makes data organization a critical consideration. Data access patterns for single and multiple channel GeoTIFF images vs. generic binary BIP, band interleaved by line (BIL), and band sequential (BSQ) file formats were analyzed. Tests showed that the BIP format gave the best results for multi-channel access.

5   Interoperability

Satellite remote sensing data are a common base for the development of geographic information system (GIS) implementations. The GIS community is large and diverse, and is a relatively untapped resource when considering the usual target audiences of earth sciences enterprise (ESE) data development efforts. Until the mid 1990s GIS users commonly obtained their data by ordering media and generated outputs as paper products. With the advent of the Web, that quickly changed. Many GIS applications require timely data, a need that Web-based delivery can meet. Further, the availability of Web Mapping tools allowed GIS developers to electronically deliver their products more broadly and in a more dynamic and useful manner.

The data formats supported by GIS software do not overlap those commonly utilized in science applications of remote sensing data. This has been a roadblock to the broad usage of ESE data by the GIS user community. The majority of Web Mapping tools are products of commercial companies and many have licensing issues that hinder their usage by all but the largest GIS developer concerns. Subject-area specific GISes, for example those targeting terrestrial resources, commonly share base datasets. For example, Landsat imagery is a common component of many GISes concerning land resources. An ability to share common datasets in a standard way would have obvious advantages. In this section we discuss various OGC standards related to WebGIS interoperability and our experience with extending MapServer to incorporate these standards.

Recent efforts by the OpenGIS consortium have resulted in three standards that address interoperability, i.e. Web Mapping Sever (WMS) [18] (also called Web Mapping

Testbed (WMT)), and Geographic Markup Language (GML) [16] and the Web Coverage Service (WCS) [17]. WMS specification standardizes the way in which maps are requested by clients and the way that servers describe their data holdings. These maps are generally rendered in a standard format like Graphics Interchange Format (GIF), Portable Network Graphics (PNG), etc. However, the output generated by WMS may not be easily consumed by other systems. On the other hand GML is an XML encoding for the transport and storage of geographic information, including both the spatial and non-spatial properties of geographic features.

GML specification is more suitable for vector data exchange between WebGISes. More recent efforts to produce a parallel specification for delivering GIS raw data over the Web in a standard way has resulted in the Web Coverage Service (WCS) specification. Owing to its obvious importance in the earth sciences enterprise (ESE) data sharing over the Web, the University of Minnesota, along with UND and JPL partners have developed the WCS extension of the MapServer, with a SEEDs grant received from the NASA. We now describe these three standards and how they were incorporated into the MapServer.

5.1   GML and MapServer

XML, the *eXtensible Markup Language*, is fast becoming the standard of data exchange between web applications. HTML (Hypertext Markup Language) and XML are both subsets of Standard Generalized Markup Language. However there is a subtle difference between HTML and XML: HTML tags describe how the data items should be displayed, while XML tags describe the data itself. This difference is very important, because the self-describing nature of XML allows the programs to interpret the data in XML documents. Recent research has been focused on modeling the data stored in XML documents, e.g. semi structured data models [5], [22], standardizing the query language for XML [7], [30], optimization and compression. Interpretation of data stored in XML documents is possible due to the existence of Document Type Descriptors or XML Schema. GML will have a great impact on the ability of organizations to share geographic information with one another and to enable linked geographic datasets. GML also provides several challenges in terms of query processing, indexing etc. Previous studies [11] have considered wrapper based XML solutions for web mapping. We not only extended MapServer to generate geographic data in GML format, which allows building interoperable systems, we also studied the computational issues related to spatial query processing on GML documents. The initial release of GML conforms to OGC's "Simple Features" specification. GML provides support for the geometry elements corresponding to the Point, Linestring, LinearRing, Polygon, MultiPoint, MultiLineString, MultiPolygon, and GeometryCollection. It also provides a coordinate element for encoding coordinates and a Box element for defining spatial extents.

Parsing: Parsing is the process of assigning structure to sentences with a given grammar. There are two basic models for XML document parsing, namely SAX and DOM. A general comparative analysis of DOM and SAX, which differ significantly in origin, scope, and programming style can be found in [29]. The models are not in direct competition with each other; each has strengths and weaknesses and can even be combined

in certain applications. The most important difference between SAX and DOM is that SAX presents the document as a serialized "event stream" (a sequence of calls to a handler function as each chunk of XML syntax is recognized) rather than a 'tree' as in DOM. A major disadvantage of this approach is that SAX does not support random-access manipulation of the document, and it is the responsibility of the application to store the data if its needed after parsing is completed. But this disadvantage turns into an advantage where the user can discard information that will not be needed. Therefore the use of SAX can result in reduced memory overhead compared to DOM, which requires retaining the complete document as a tree in memory. It is possible to combine SAX and DOM within a single system. We used Xerces Java Parser 1.2.0 which supports the XML 1.0 recommendation and contains advanced parser functionality, such as XML Schema, DOM Level 2 version 1.0, and SAX Version 2. We have chosen the Xerces parser mainly because of its easy availability, functionality, and, more importantly, the close integration with the Apache web server. GML maps generated form the MapServer were validated using this parser.

GML View: We use DOM representation for rendering the map from the GML document. DOM representation is convenient for rendering maps, as it is easy and efficient to traverse the tree and extract the required information from each node. Therefore, we choose DOM in lieu of SAX for map rendering of a GML document. The parsed map can be either rendered directly or can be converted into static images. Advantages and disadvantages of these two approaches can be found in [19]. Conversion to a standard image does not require any client support, but client side querying and processing are limited. On the other hand GMLView takes GML data directly from the MapServer and renders the map in a client browser using applets. This approach is flexible and allows client side processing like query and integration of maps from different map servers.

Spatial Computations on GML Data (Client). Queries operate on a single document or on fixed collections of documents. The selected information can span more than one document. Different queries, such as point, range, and join can be performed on GML documents. Queries can be broadly divided into two categories: single scan query (point query, range query) and multi scan query (join query). In a single scan query, a record (tuple) in the table (relation) being queried has to be accessed at most once. Thus in the worst-case scenario, each record in the table will be accessed and processed to verify whether it meets the query criterion. For example, "List the names of all hotels which are within 15 miles of the Minneapolis downtown," is a single scan query. The result of this query will be all hotels, which intersect a circle of radius 15 miles centered on the courthouse (assuming courthouse is the center of Minneapolis downtown). In rendering the map from the GML document, the coordinates have to be extracted from the GML document. This operation is a special case of a range query (where range is defined in terms of a geometry object).

A join query is basically an example of a multi-scan. In the context of spatial databases, when the joining attributes are spatial in nature, the query is referred to as a spatial join query. For a join operation, a nested loop strategy is applied which requires multiple scans over the GML document. Because a record in one table can be associated with more than one record in a second table, records may have to be accessed more than once to complete the join. For detailed formulation of cost models, please refer to [21].

From these models, we observed that if *n, i.e.,* the number of nodes in the GML file is large, the SAX model is better for searching a single element than the DOM model; the DOM search model starts a search operation only after it has parsed the whole document. However, for multiple passes, i.e., when x is large, the DOM model is better since it needs to parse the document only once, and all x searches can be performed on the same tree structure of the document, which resides in memory.

5.2 WMS (WMT) Compliant MapServer

The original MapServer [13] was designed before OGC specifications were published. We extended MapServer to support various OGC standards including WMT standard. The goal of the MapServer/WMT integration effort was to develop native WMT compatibility into the MapServer in such a way that would allow users of the CGI application and those writing custom applications using MapScript (a scripting interface to the MapServer) to take advantage of it. WMT capabilities requests require the presentation of service and metadata. MapServer was designed originally as a map and spatial query engine so generic metadata storage was added to MapServer configuration files (see the sample fragment) to fully support WMT requirements.

MapServer applications are built using a single configuration file called a "map file". Map files are used to control virtually all aspects of an application such as what layers are to be drawn, how they are rendered, and how they are queried. This file becomes the means for a MapServer developer to add WMT compatibility to an application. The WMT functions work off the map file to develop content matching the users WMT request. Any standard browser (or client system) can ask a WMS compliant server for one or more of the following services: map images (GetMap), service-level metadata (GetCapabilities), and optionally, information about particular features (GetFeatureInfo).

These requests are submitted to the sever in the form of Uniform Resource Locators (URLs). URLs are formed using a set of standard parameters [18](e.g., width, height, bbox, srs, etc.), based on user interaction with the client system. For example, a request to an AVHRR hurricane image may translate into the following URL for MapServer: http://terrasip.gis.umn.edu/mapserver.cgi?VERSION=1.1.0&   REQUEST=GetMap& SRS=   EPSG%3A4326&   BBOX=-97.105,24.913,78.794,36.358&   WIDTH=560& HEIGHT= 350&LAYERS=AVHRR-09-27&STYLES=default&FORMAT = image/gif.

WMT request handling is managed through a simple broker function that examines the parameters passed into a MapServer application. This function then determines which, if any, type of WMT request needs to be processed. If no WMT compliant request is made, then control is handed back to the calling function for normal processing. In this way WMT functionality can easily be added to any MapServer application even though that application may also be a normal MapServer application. Individual WMT request handling (e.g. getMap, getCapabilities or getFeatureInfo) is done using helper functions written in C using the MapServer API. The resulting spatial data is converted into GIF format (as requested, see FORMAT tag in the above URL) and is returned to the client. This representation has the advantage that the maps can be viewed in a standard browser, but client side query processing cannot be achieved.

```
MAP
  NAME 'example'
  SIZE 300 300
  IMAGECOLOR 255 255 255
  IMAGETYPE PNG
  EXTENT 4274 52280 4518 52540
  PROJECTION
    +init=epsg:26915
  END
  WEB
    METADATA
      wms_title 'Example WMT App'
      wms_onlineResource 'http://...'
      wms_contactPerson 'John D'
      wms_contactOrganization 'IBM.'
      ...
    END
  END
  LAYER
    NAME 'water'
    DATA '/usr/shpfiles/water'
    CLASS
      COLOR 0 0 255
    END
  END
END
```

[MAP1a].   Sample   mapfile
$< projectname.map >$

5.3 WCS Compliant MapServer

Even before the Web Coverage Service (WCS) specification was published, the MapServer design allowed building of applications that provide web access to raw geo-spatial data products. MapServer has the capability to access seamlessly various data products (of different formats) residing on any server, and generate downloadable data products based on the user defined region of interest (on the fly - for reasonable sized products). However, the generated product is limited to generic binary format (with BIL, BIP, and BSQ interleaving), with few well known header formats, which allows the usersto grab the downloaded data product into well known commercial and public domain Remote Sensing and GIS software. Although the data format issues can be addressed with specialized libraries like GDAL [9], the main problem still remains with client-side software systems. First of all there is no standard way to request for data products, and secondly there is no standard way to describe the associated ancillary information (metadata, e.g., projection system). These limitations have hindered MapServer's potential as a tool to access raw data holdings on distributed (Web) servers. This is exactly where the WCS standard comes into play. In a nutshell, MapServer has many components and desirable features that are needed for building a WCS server. Let us now have a brief look at the recently published WCS standard.

A WCS enabled server provides access to potentially detailed and rich sets of geospatial information. As opposed to the WMS server, which filters and generates data products in the form of static maps, the WCS server actually generates the data products that meet user defined criteria in specific (well-known) formats that can be consumed by the client-side software (e.g., ARC/INFO, Imagine, Matlab, IDL). The WCS specification provides the following three operations: GetCapabilities, GetCoverage,and DescribeCoverage.

The *GetCapabilities* operation returns an XML document describing the service and brief descriptions of the data collections from which clients may request coverages. Clients generally run the GetCapabilities operation and cache its result for use throughout a session, or reuse it for multiple sessions. If GetCapabilities cannot return descriptions of its available data, that information must be available from a separate source, such as an image catalog.

The *DescribeCoverage* operation lets clients request a full description of one or more coverages served by a particular WCS server. The server responds with an XML document that fully describes the identified coverages.

The *GetCoverage* operation of a Web Coverage Service is normally run after GetCapabilities and DescribeCoverage replies have shown what requests are allowed and what data are available. The GetCoverage operation returns a coverage (that is, values or properties of a set of geographic locations), bundled in a well-known coverage format. Its syntax and semantics bear some resemblance to the WMS GetMap and WFS GetFeature requests, but several extensions support the retrieval of coverages rather than static maps or discrete features.

As noted previously, MapServer already has the bells and whistles to build a WCS server. We now briefly describe the key components we developed to make it a fully complient WCS server. A WCS request consists of one of the three allowed operations; GetCapabilities, GetCoverage,and DescribeCoverage. The get GetCapabilites and the DescribeCoverage services require generating a good amount of metadata that describes data holdings on the server as well as individual coverages. Metadata is harvested through two distinct sources, one from the data source itself, either through the header files or through the associated ancillary files (e.g., world file). Secondly, new vocabulary (key-words) has been added to the MapServer configuration file (.map). Let us now look at these extensions in more detail.

WCS-CGI Module: This module processes the client request with the help of a configuration file with an extension of .map. The key-value pair encoded URL is scanned first by this module. If the request is a valid WCS request, then the MapServer responds appropriately; otherwise it generates an error message. The configuration file is parsed, and in memory data structures are built which then become key resources for various other modules.

Metadata Harvesting Module: This module is responsible for servicing the GetCapabilites and the DescribeCoverage requests. The output of this module is an XML document that conforms to the WCS 1.0 XML schemas defined in the WCS standard.

From a WCS application development point of view, users have to know more about two key resources, namely the new .map file definitions and spatio-temporal tile indexing.

New MapServer Configuration File Definitions: Two important key words defined inside the layer object are TILEINDEX, and METADATA. The following are some of the properties that MapServer utilizes when defined in a METADATA block within a layer object.

```
...
  wcs_formats "GEOTIFF_INT16"
  wcs_nativeformat "raw binary"
```

```
  wcs_timeposition "2002-001,2002-033,
   2002-049,2002-065,...,2002-193"
...
```

Another important set of properties deals with range definitions. MapServer supports three kinds of range sets in addition to temporal ranges. They are *Image band range sets, Pixel range sets* (e.g., elevation values), and *arbitrary image tile attribute range sets* (e.g. cloud cover percentage). The pixel range sets are used to support range set queries proposed in the WCS. The range sets are used to describe the properties assigned to each location in the domain, for example, a given location may have various recordings like; current land use, mean daily rain fall and temperature, spectral reflectance values from a satellite image, elevation from a digital elevation model etc. As this function is closely tied to the raw values associated with image pixels, we are working on directly supporting this function in the GDAL.

There are two layer metadata properties necessary to set up a range set(s) for a layer: (i) `wcs_rangeset_axes` – a comma delimited list of the range set name(s) available for this layer. In many cases this may just be a single value like 'bands'. (ii) `wcs_<name>_<tag>` – a collection of range set specific metadata. The following main tags are available (more granular control is available but is not documented here): `wcs_<name>_description,wcs_<name>_name wcs_<name>_label`, `wcs_ <name>_values` (a comma delimited list of single values that comprise the range set (e.g. 1,2,3,4,5,6)) `wcs_<name>_interval` (a comma delimited (changing to '/' to be consistent with request interval notation) list of the form min,max,resolution, only a single interval is allowed at the moment), `wcs_<name>_rangeitem` (this tells MapServer what type of a range this is). Two special values '_bands' and '_pixels' will trigger specific GDAL functionality, while any other value is interpreted as an actual tileindex column name upon which to filter. See TAB1b for an example layer object definition. Please note that the `wcs_bands_name` is the value that is supposed to be the parameter name in the GetCoverage? request, so use need to use something appropriate. Now let us look at the spatio-temporal tile index.

Spatio-Temporal Tile Indexes: MapServer has long supported a method of breaking a dataset into smaller, more manageable pieces or tiles. In this case, a shapefile is used to store the boundary of each tile, and an attribute holds the location of the actual data. Within a MapServer configuration file (.map) the layer keywords TILEINDEX and TILEITEM are used to activate tiling.

Consider the example where an organization wants to serve hundreds or even thousands of MODIS scenes. Five images cover the spatial extent and each group of five varies by date of acquisition. This turns out to be a fairly common scenario for organizations interested in WCS, one that the existing tiling support does not adequately address. In previous versions of MapServer a developer would have to create one tile index and one layer definition for each group of five images. This could result in configuration files that are prohibitively long and difficult to manage.

In order to more efficiently support the WCS specification, we implemented a new tiling scheme within MapServer, one that supports spatial sub-setting, but also ad hoc sub-setting based on any attributes found within the tile index. In many cases a temporal attribute could be used, but sub-setting is not limited to that case. The new scheme

introduces the concept of tile index layers, that is, a separate layer definition is used to describe the tile index dataset. With this we get all the benefits of any MapServer layer, but most importantly we can apply MapServer filters to the data. Filters can be defined at runtime using MapServer CGI, MapScript or via the WCS server interface. The syntax for the layer using the index remains unchanged except that the value for TILEINDEX refers to the index layer, by name, instead of an external shapefile.

So, looking at the example above again we can reduce our MapServer configuration to two layer definitions, one for the tile index and one for the imagery itself. Extracting a single date's worth of imagery is now a matter of setting the appropriate filter within the tile index layer (see MAP1c).

```
MAP
  NAME 'WCS Application'
  ...
  LAYER
    ...
    METADATA
      ...
      wcs_rangeset_axes 'bands'
      wcs_bands_name 'infra-red'
      wcs_bands_description 'Description of
        each TM band here...'
      wcs_bands_label 'Band Number'
      wcs_bands_values '1,2,3,4,5,6,7'
      wcs_bands_rangeitem '_bands'
      ...
  END
  DUMP TRUE !Enable this layer for WCS access
  TILEINDEX "modis_idx" !Tileindex layer name
END
```

[MAP1b]. WCS Example

```
MAP
  ...
  LAYER
    NAME "fpar"
    ...
  END
  ...
  DUMP TRUE
  TILEINDEX "fpar_idx"
END
!Define tileindex layer
LAYER
  NAME "fpar_idx"
  TYPE TILEINDEX
  DATA "mod13"
  FILTERITEM "imgdate"
  FILTER "%time%"
END
...
END
```

[MAP1c]. Tile Index Example

Developing these tile indexes is more difficult than basic indexes simply because there are no ready-made tools to do so. Fortunately we can leverage existing tool available within MapServer or supporting libraries such as GDAL by post processing their output. First, the basic spatial index needs to be built. The GDAL utility *gdaltindex* already does this. Simply point gdaltindex at the directory containing the collection of MODIS images and it will build a shapefile index suitable for use with MapServer. The next step would be to add the temporal information. Briefly, we 1) open the index .dbf file for reading, 2) create a new column to hold the image acquisition date, 3) for each image – a) extract the image acquisition date, b) insert it into the new column, and finally, 4) close the index .dbf file. This general approach could be used for many cases. Scripting languages such as Perl, PHP or Python work well since they all have readily available modules for manipulating .dbf files. A worst case would involve hand editing

the resulting .dbf file using a desktop tool such as Mircosoft Access or ESRI Arcview. We are now ready to develop and deploy WebGIS applications using the MapServer.

6   Applications

Natural Resource Analysis and Mapping System (NRAMS) [26]: The success of any Web-based GIS application largely depends on its efficiency and ease of use. Building a Web GIS application consists of two main phases: the preparation of the data and the overall design of the user interface to the application.
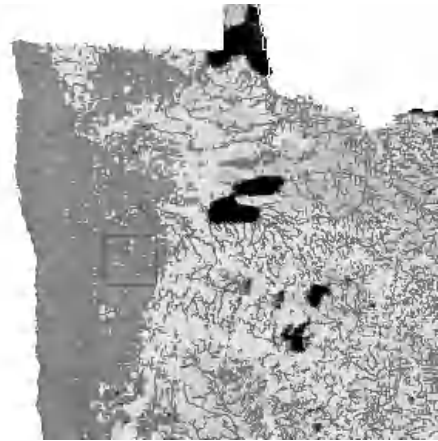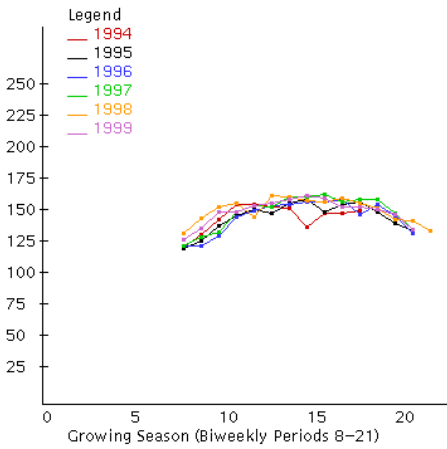

Fig. 2. Temporal Profiles for the ROI (see Fig. 3)

Fig. 3. ROI for temporal profile generation

Natural resource analysis and modeling is a complex task. Several studies based on remote sensing and GIS techniques have resulted in the development of useful products. However, they have seldom been widely applied due to the lack of an efficient and easy-to-use delivery mechanism. It has long been recognized that public participation is critical for managing our natural resources effectively. It is also important that we better manage our information and knowledge resources. In order to provide timely access to these data products, we need an efficient mechanism for information delivery. The Internet, specifically the Web, provides a natural choice. NRAMS, our prototype Web-based GIS application, demonstrates the capabilities of providing both online access and analysis to a large collection of multi-temporal AVHRR NDVI imagery along with related spatial and non-spatial datasets for use by land managers and the general public. Furthermore, it serves as a testbed for MapServer, the Web GIS described in section 3. In this section we describe NRAMS, not only in the context of data preparation, but also in relation to the data browsing and analysis capabilities provided to the end user. We utilized both MVC NDVI imagery and Fourier-adjusted imagery in the NRAMS application, since interpretation of temporal profiles generated from MVC NDVI imagery

is well understood, while the corresponding interpretation for the Fourier-adjusted profiles is still under development. Fourier-adjusted NDVI image composites have the additional advantage that the first three harmonics are sufficient for our analysis. This effectively reduces the number of required bands from 26 biweekly NDVI bands to five Fourier coefficient bands.

NDVI Profiles and $\Sigma NDVI$: A temporal profile is a graphical plot of sequential NDVI observations against time. These profiles quantify the remotely sensed vegetation's seasonality and dynamics. These profiles can be described with simple parameters, like the amplitude, mean, and standard deviation. We can understand the onset and peak of greenness and the length of growing season from analyzing these profiles. Using MapServer, a temporal profile can be generated for a sample location, a window (i.e. an arbitrary rectangular region), or a polygon (e.g. using a county vector layer). For both window and polygon regions, profiles can be generated for mean, median, or mode values. The plots are rendered in either a single window or multiple windows (for comparison) using Java applets. To improve performance, temporal profiles for each county could be pre-realized and stored in an RDBMS, since the county boundaries are fixed. However, this cannot be applied to ROI-based queries since the user can draw a window of arbitrary size and shape at any location. Sample profiles generated are shown in Figure 2 for the area located in northern Minnesota (Figure 3).

Application Development and the GUI: Application development with MapServer is fairly simple and consists of two parts: the front-end, which defines the graphical user interface to the application, and the application configuration file. An application is configured using several objects, like Map Object, Label Object, Layer Object, Feature Object, and Web Object. Each object controls a certain aspect of the application. More detailed descriptions of these objects can be found in the MapServer [13] documentation. Efforts are underway to describe the configuration and data layer definitions using XML. Geo-spatial analysis can be customized by using a separate process configuration file, which defines the relationship between a CGI variable and the analysis function to be applied and where to do the processing (i.e. on the client or the server). The front-end of the application consists of the graphical user interface, described by standard HTML tags and style sheets. JavaScript is used for preprocessing user requests (e.g. options selected or ROI selection), creating new windows and changing the output targets, etc.

The NRAMS application is quite dynamic, and new NDVI images and other geo-spatial data sets are added constantly as they become available. MapServer seamlessly incorporates new data sets into the visualization, analysis and query processes. NRAMS is only one of many WebGIS applications that can be built using MapServer. Plethora of well documented applications can be found at the MapServer's Gallery page [13]. These applications developed by the user community demonstrates various advanced features and versatility of the MapServer.

7   Discussion and Conclusions

We have evaluated both client-centric and server-centric Web GIS architectures and proposed a new load-balancing client/server architecture. MapServer was extended using this new architecture and has been prototyped through a demanding Web GIS application called NRAMS. The new application is much faster than older implementations

based on the server-centric approach. The new approach has several advantages, as enumerated in the paper, but it requires additional coding since some functions need to be implemented on both client and server. Adaptation of GML in WebGIS environment comes with a computational overhead. Our study concludes that SAX performs better than DOM for single pass; thus for simple applications like visualization, subsetting, the SAX model is superior. However, for intensive applications involving queries requiring multiple passes over documents or integration of multiple documents in a distributed environment, DOM based parsing offers a better solution. Further research is needed to index GML documents, and represent topology.

Our Web log analysis shows that there is a clear trend for users to generally access adjacent areas in subsequent operations. Therefore, pre-fetching these data will greatly improve the overall performance of the system. However, implementation is a problem with the present HTTP/1.1. The proposed HTTP-NG [31] offers persistent connections, thereby opening new optimization opportunities for Web GISes, such as pre-fetching and cacheing. Another issue that needs to be addressed is persistent access to database systems, as there is considerable overhead in opening a new connection with each user request. The PERL DBI module offers a quick solution, but it needs additional wrappers as most of the Web GISes are implemented in C/C++. A faster solution would be a socket-based daemon which maintains a persistent connection to a database server and processes Web GIS requests. Our core MapServer was developed in "C" language. Given the performance improvements, it makes sense to use the Java platform along with useful APIs (e.g. Advanced Imaging, 3D, Servlet, JSF, AJAX, JSTL, WSDP) to build newer WebGISes. Additionally, persistent connection between client and server can be achieved through Servlet API. Similarly, adaptation of web services and service oriented architecture [2,24] will facilitate building more sophisticated, scalable, and interoperable WebGISes.

We conclude that in order to develop efficient WebGISes, it is necessary first to understand Web technologies and Internet development environments and various design choices. Efficient application development requires understanding and intelligent usage of WebGIS, and the identification of the datasets that should be pre-realized. Developers must consider the many alternative ways of implementing solutions to a given problem, identify and use shortcuts where possible, understand user access patterns, and, most important, fine-tune the system accordingly.

Acknowledgments

References

1. David J. Abel, Kerry Taylor, Ross Ackland, and Stuart Hungerford. An exploration of gis architectures for internet environments. *Computers, Environment and Urban Systems*, 22(1):7–23, 1998.
2. Douglas K. Barry. *Web Services and Service-Oriented Architectures: The Savvy Manager's Guide*. Morgan Kaufmann, 2003.
3. Hal Berghel. The client's side of the world wide web. *Communications of the ACM*, 39(1):30–40, 1996.
4. Michela Bertolotto and Max J. Egenhofer. Progressive vector transmission. *ACMGIS*, 1999.
5. Peter Buneman. Semistructured data. In *SIGMOD/PODS*, 1997.
6. Volker C. and Sascha F. Integrating levels of detail in a web-based gis. *ACMGIS'*, 1998.
7. A. Deutsch, M. Fernandez, D. Florescu, A. Levy, and D. Suciu. Xml-ql: A query language for xml. 1998, http://www.w3.org/TR/NOTE-xml-ql/.
8. ForNet. Internet delivery of spatial and related data. http://www.gis.umn.edu/.
9. W. Frank. Gdal - geospatial data abstraction library. http://www.remotesensing.org/gdal/.
10. Volker G. and Oliver G. Multidimensional access methods. *ACM Computing Surveys*, 30, 1998.
11. A. Shaheen J. Zhang, M. Javed and Le Gruenwald. Prototype for wrapping and visualizing geo-referenced data in a distributed environment using xml technology. In *ACMGIS*, 2000.
12. MapGuide. Davecentral: Mapguide plug-in. http://www3.autodesk.com/.
13. MapServer. Mapsever home page. http://mapserver.gis.umn.edu/.
14. MAPublisher. Mapublisher by avenza software. http://www.adobe.com/store/plugins/.
15. MapViewer. Xerox parc map viewer. http://mapweb.parc.xerox.com/map/.
16. OGIS. Geography markup language (gml). http://www.opengis.net/gml/.
17. OGIS. Web coverage service. http://www.opengis.net/wcs.html.
18. OGIS. Web map server interfaces implementation specification. http://www.opengis.org/techno/specs/00-028.pdf.
19. Zhong-Ren Peng. An assessment of the development of internet gis. In *Proceedings of the ESRI User Conference*, 1997.
20. REGIS. Grasslinks 3.1. http://www.regis.berkeley.edu/grasslinks/.
21. S. Shekhar, Ranga R. Vatsavai, Namita Sahay, Thomas E. Burk, and Stephen Lime. Wms and gml based interoperable web mapping system. In *ACMGIS*, 2001.
22. D. Suciu. An overview of semistructured data. *Sigact News*, 29(4):28–38, 1998.
23. TerraSIP. Terrasip introduction. http://terrasip.gis.umn.edu/.
24. S. Tu, M. Flanagin, Y. Wu, M. Abdelguerfi, E. Normand, V. Mahadevan, J. J. Ratcliff, and K. Shaw. Design strategies to improve performance of gis web services. In *ITCC (2)*, 2004.
25. USCB. Tiger mapping service. http://tiger.census.gov/.
26. Ranga R. Vatsavai, T. E. Burk, B. T. Wilson, and S. Shekhar. A web-based browsing and spatial analysis system for regional natural resource analysis and mapping. In *ACMGIS*, 2000.
27. Ranga R. Vatsavai, Thomas E. Burk, S. Shekhar, and M. H. Hansen. An efficient query strategy for integrated remote sensing and inventory (spatial) databases. In *SSDBM*, pages 115–123, 2001.
28. Vdraft. Vdraft intenet tools. http://www.vdraft.com/vtools.html.
29. W3C. Architecture domain, document object model faq. http://www.w3.org/DOM/faq#SAXandDOM.
30. W3C. Conference on standardization of xml query languages. 1998, http://www.w3.org/TR/NOTE-xml-ql/.
31. W3C. Http-ng overview. http://www.w3.org/Protocols/HTTP-NG.
32. Zu-Kaun Wei and at. el. Efficient spatial data transmission in web-based gis. *WIDM'99*, 1999.

Author Index