Lecture Notes in Artificial Intelligence 1003

Subseries of Lecture Notes in Computer Science
Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science
Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Springer
*Berlin*
*Heidelberg*
*New York*
*Barcelona*
*Budapest*
*Hong Kong*
*London*
*Milan*
*Paris*
*Santa Clara*
*Singapore*
*Tokyo*

P. Pandurang Nayak

# Automated Modeling of Physical Systems

Springer

Series Editors

Jaime G. Carbonell
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3891, USA

Jörg Siekmann
University of Saarland
German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany

Author

P. Pandurang Nayak
NASA Ames Research Center, Recom Technologies
Moffett Field, CA 94035, USA

Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

**Nayak, P. Pandurang:**
Automated modelling of physical systems / P. Pandurang Nayak. - Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ; Hong Kong ; London ; Milan ; Paris ; Santa Clara ; Singapore ; Tokyo : Springer, 1995
(Lecture notes in computer science ; 1003)
Zugl.: Diss.
ISBN 3-540-60641-6
NE: GT

CR Subject Classification (1991): I.2, I.6, F.1, J.2, C.5.4

ISBN 3-540-60641-6 Springer-Verlag Berlin Heidelberg New York



Typesetting: Camera ready by author
SPIN 10487123 06/3142 - 5 4 3 2 1 0 Printed on acid-free paper

*To Mala and my parents*

# Foreword

It is a privilege for me to write the foreword to this important, powerful, and elegant piece of Artificial Intelligence research. In both substance and method, it is the quintessence of what today's AI research should be like.

The prelude to doing an important work is to seek out an important problem. Nayak's problem area is important for two reasons.

- For most tasks whose solution requires intelligence, the AI will have to be able to reason about the world of physical objects, physical systems, and engineered artifacts. The path from the earliest AI programs (e.g. the Logic Theorist and the Geometry Theorem Proving Program) to the expert systems of the 1980s and 1990s has not included much work on physics and engineering problem solving. The exception of course is the thinly populated field of qualitative physics research, of which Nayak's work is an example.
- In expert systems ("where the rubber meets the road" for most of applied AI), the models of domains are very extensive and detailed. Indeed, it is the hallmark of expert systems that their knowledge bases are relatively large and their reasoning methods simple (usually just Aristotelian and/or Bayesian logic). A narrow pinnacle of knowledge (literally a pinnacle of excellence) allows the expert system to perform at high levels of competence for a narrow class of problems. But from pinnacles it is possible to fall precipitously, for example when the problem presented lies outside of the narrow band of knowledge. What is needed is a soft landing on a much more gentle slope of generalized knowledge. Such knowledge contains the generalized models and equation systems that we learn in courses on physics and engineering. Of course, this "generalized" knowledge is also somewhat domain-specific, but its breadth is much greater than the specialists' knowledge used in expert systems. The increased scope and power of the "broad" knowledge does not come for free. To use it, the reasoning processes can no longer be simple – more powerful methods are needed. To develop the necessary representation and reasoning methods is the goal of qualitative physics research, and the contribution of Nayak's research.

Here are some specific points to keep in mind as you read this monograph.

- Carefully chosen problem representations are crucial to effective problem-solving. Amarel, as early as the 1960s, discussed this issue extensively in his landmark paper "On representations of problems of reasoning about actions" [Amarel, 1968], where he provides various representations for the Missionaries and Cannibals problem. The choice of problem representations is equally crucial in engineering problem-solving, where one has to carefully balance the veracity of a model against its complexity. An overly detailed model can be too complex, while simple models may miss important problem features. Hence, in modeling physical systems, it is important to identify just the relevant physical phenomena, and to identify just the appropriate level of detail to model each phenomenon. Nayak's research elegantly treats and combines both the theoretical and practical sides of this problem.
- On the theoretical side, the research develops a precise formalization of the problem of selecting adequate models of physical systems. It casts the problem as a search problem, and provides clear definitions for the search space and the goal criterion. Such a precise formalization is essential for applying computational tools to the problem. It then uses the formalization to analyze the difficulty of finding adequate models, and for understanding the different reasons for its intractability. Finally, it uses the analysis of the reasons for intractability to identify special cases of the problem that can be solved very efficiently. These special cases stem from the identification and use of a special class of approximations called causal approximations. Importantly, the research shows that causal approximations are very common in modeling the physical world, making the special cases broadly applicable and useful.
- On the practical side, Nayak's work analyzes the representation and reasoning issues that arise in building practical systems for constructing adequate device models. It develops a class-level description for device models that facilitates knowledge base construction and supports focused generation of device models. It develops a novel order of magnitude reasoning algorithm for reasoning about a device's behavior, and incorporates it into the efficient model selection algorithm developed above. The resulting algorithm has been implemented in Common Lisp.
- Nayak tests and validates his methods and algorithms by running his implementation on a large knowledge-base of device models. The range of devices is not only interesting but significant, and the testing is a necessary activity to ground the research in reality and to add credibility to Nayak's approach.

In recommending Nayak's work to the Association for Computing Machinery for its dissertation award, I remember using a word that I rarely use: "brilliant." In reviewing the work again as a prelude to writing this foreword,

I am once again excited by the work and am reminded of why I used the strong B word. Proceed with haste to the first page so that you can find out for yourself. You will be richly rewarded.

*October 1995*

*Edward A. Feigenbaum*
*Professor of Computer Science, Stanford University*
*and Chief Scientist, United States Air Force*

# Preface

Effective reasoning about complex physical systems requires the use of models that are adequate for the task. Constructing such adequate models is often difficult. In this dissertation, we address this difficulty by developing efficient techniques for automatically selecting adequate models of physical systems. We focus on the important task of generating parsimonious causal explanations for phenomena of interest. Formally, we propose answers to the following: (a) what is a model and what is the space of possible models; (b) what is an adequate model; and (c) how do we find adequate models.

We define a model as a set of *model fragments,* where a model fragment is a set of independent equations that partially describes some physical phenomenon. The space of possible models is defined implicitly by the set of applicable model fragments: different subsets of this set correspond to different models. An adequate model is defined as a simplest model that can explain the phenomenon of interest, and that satisfies any domain-independent and domain-dependent constraints on the structure and behavior of the physical system.

We show that, in general, finding an adequate model is intractable (NP-hard). We address this intractability, by introducing a set of restrictions, and use these restrictions to develop an efficient algorithm for finding adequate models. The most important restriction is that all the approximation relations between model fragments are required to be *causal approximations*. In practice this is not a serious restriction because most commonly used approximations are causal approximations.

We also develop a novel *order of magnitude* reasoning technique, which strikes a balance between purely qualitative and purely quantitative methods. The order of magnitude of a parameter is defined on a logarithmic scale, and a set of rules propagate orders of magnitudes through equations. A novel feature of these rules is that they effectively handle non-linear simultaneous equations, using linear programming in conjunction with backtracking.

The techniques described in this dissertation have been implemented and have been tested on a variety of electromechanical devices. These tests provide empirical evidence for the theoretical claims of the dissertation.

**Acknowledgements** In preparing this dissertation I have been greatly influenced by a number of different people, and have benefited enormously from

my discussions with them. I would like to thank my advisor, Ed Feigenbaum, for his guidance, encouragement, and support. He gave me the freedom to pursue my research interests, while always reminding me of the importance of real problems. Leo Joskowicz participated actively in my research and helped me clarify the ideas. Sanjaya Addanki, Richard Fikes, Rich Keller, and Jean-Claude Latombe provided valuable feedback on various aspects of this research. Surajit Chaudhuri and Alon Levy shared the ups and downs of graduate student life with me and helped to make research genuinely fun. Members of the How Things Work project, Bob Englemore, Tom Gruber, Yumi Iwasaki, and Jim Rice, provided a challenging and educational environment for research.

Grace Smith, Michelle Perrie, and Margaret Timothy provided superb administrative support. IBM supported me financially during the course of this research through an IBM Graduate Technical Fellowship. Additional support for this research was provided by the Defense Advanced Research Projects Agency under NASA Grant NAG 2-581 (under ARPA order number 6822), by NASA under NASA Grant NCC 2-537, and by IBM under agreement number 14780042.

On a more personal note, my friends from IIT and Stanford made my years at Stanford enjoyable and memorable. My family in the US helped alleviate any home-sickness I might have felt. My parents and my sister supported me with their love and encouragement throughout my life. My parents taught me that in any endeavor, it is the effort that is important, not the result. It is because of them that I am what I am. And last, but by no means least, Mala's constant friendship, support, and love has made this research possible. This dissertation is as much a result of her efforts, as it is a result of mine.

*P. Pandurang Nayak*
*October 1995*

# Table of Contents


**1. Introduction** ..... 1
- 1.1 Models and tasks ..... 1
- 1.2 Problem statement ..... 4
- 1.3 Proposed solution: An overview ..... 4
  - 1.3.1 What is a model and what is the space of possible models? ..... 4
  - 1.3.2 What is an adequate model? ..... 5
  - 1.3.3 How do we find adequate models? ..... 8
- 1.4 Contributions ..... 10
- 1.5 Readers guide ..... 11

**2. Models and model fragments** ..... 13
- 2.1 Models of the behavior of physical systems ..... 13
  - 2.1.1 Parameters ..... 13
  - 2.1.2 Equations ..... 14
- 2.2 Multiple models ..... 15
- 2.3 Model fragments ..... 16
  - 2.3.1 What is a model fragment? ..... 16
  - 2.3.2 Advantages of model fragments ..... 17
  - 2.3.3 Composing model fragments ..... 18
  - 2.3.4 Relations between model fragments ..... 19
- 2.4 Space of possible models ..... 22
  - 2.4.1 Device structure ..... 22
  - 2.4.2 Structural abstractions ..... 23
  - 2.4.3 Possible models of a component ..... 24
- 2.5 Model fragment classes ..... 24
  - 2.5.1 What are component and model fragment classes? ..... 25
  - 2.5.2 Typographic conventions ..... 25
  - 2.5.3 Defining component and model fragment classes ..... 26
  - 2.5.4 Difference between component and model fragment classes ..... 31
- 2.6 Summary ..... 31

**3. Adequate models** ..... 33
3.1 Tasks and models ..... 33
3.2 Causal explanations ..... 34
3.2.1 Importance of causal explanations ..... 34
3.2.2 Types of causal explanations ..... 34
3.3 Causal ordering ..... 35
3.3.1 Loops in the causal ordering ..... 36
3.3.2 Equations ..... 36
3.3.3 Computing the causal ordering ..... 37
3.4 Consistency and completeness of models ..... 46
3.4.1 Model consistency ..... 46
3.4.2 Model completeness ..... 47
3.5 Representing the phenomenon of interest ..... 48
3.6 Constraints from the structural context ..... 50
3.6.1 Structural context ..... 50
3.6.2 Constraints ..... 52
3.7 Constraints from the behavioral context ..... 54
3.7.1 Behavioral context ..... 54
3.7.2 Constraints ..... 54
3.7.3 Thresholds in behavioral constraints ..... 56
3.8 Simplicity of models ..... 57
3.9 Summary ..... 58

**4. Complexity of model selection** ..... 61
4.1 Formalizing the problem ..... 61
4.1.1 Formalizing the input ..... 61
4.1.2 Problem statement ..... 67
4.2 Complexity analysis ..... 68
4.2.1 Problem size ..... 69
4.2.2 Preliminaries ..... 69
4.2.3 The SELECT MODEL FRAGMENTS problem ..... 70
4.2.4 The SELECT ASSUMPTION CLASSES problem ..... 75
4.2.5 The SATISFY CONSTRAINTS problem ..... 76
4.2.6 The intractability of finding causal models ..... 80
4.3 Other complexity results ..... 81
4.3.1 Fixed causal orientations ..... 81
4.3.2 Finding coherent models ..... 82
4.4 Summary ..... 85

**5. Causal approximations** ..... 87
5.1 Upward failure property ..... 88
5.1.1 Efficient model selection algorithm ..... 89
5.1.2 Discussion ..... 93
5.2 Preliminary restrictions ..... 93
5.3 Causal Approximations ..... 95

5.3.1 Definitions ... 95
5.3.2 Causal approximations and the upward failure property 97
5.3.3 Causal approximations are common ... 101
5.4 Selecting assumption classes ... 101
5.4.1 Causal approximations are not enough ... 102
5.4.2 Parameter ownership ... 103
5.4.3 Ownership constraints ... 103
5.4.4 Monotonicity of causal relations ... 104
5.4.5 Discussion ... 106
5.5 Individually approximating model fragments ... 106
5.6 Expressivity of constraints ... 108
5.7 Efficiently simplifying a coherent model ... 109
5.7.1 Simplifying a model by approximating ... 110
5.7.2 Simplifying a model by dropping model fragments ... 113
5.8 Summary ... 117

**6. Differential equations** ... 119
6.1 Canonical form ... 120
6.2 Approximating differential equations ... 121
6.3 Exogenizing differential equations ... 123
6.4 Equilibrating differential equations ... 123
6.5 Monotonicity of causal relations ... 125
6.6 Efficiently equilibrating differential equations ... 135
6.6.1 Equilibrating differential equations can be hard ... 136
6.6.2 Self-regulating parameter ... 137
6.6.3 Discussion ... 142
6.7 Summary ... 143

**7. Order of magnitude reasoning** ... 145
7.1 Motivating example ... 146
7.2 Order of magnitude reasoning in NAPIER ... 147
7.2.1 Inference rules in NAPIER ... 148
7.2.2 Set of simultaneous equations ... 149
7.2.3 Backtracking algorithm ... 150
7.2.4 Example ... 151
7.3 Order of magnitude reasoning is intractable ... 153
7.4 Approximation algorithms in NAPIER ... 157
7.4.1 Ordering the equations ... 157
7.4.2 Loss of accuracy ... 158
7.5 Error estimation ... 160
7.5.1 Estimating the error of heuristic rules ... 160
7.5.2 Alternate order of magnitude rules ... 162
7.5.3 Discussion ... 164
7.6 Related work ... 165
7.7 Summary ... 166

**8. Model selection program and results** ..... 169
8.1 Finding an initial causal model ..... 170
8.1.1 Component interaction heuristic ..... 170
8.1.2 Finding an initial causal model ..... 173
8.2 Simplifying the model ..... 181
8.3 Implementation and results ..... 181
8.3.1 Overview of the knowledge base ..... 181
8.3.2 Overview of the devices ..... 182
8.3.3 Results ..... 185
8.4 Summary ..... 187

**9. Related work** ..... 189
9.1 Automated modeling of physical systems ..... 189
9.1.1 Compositional modeling ..... 189
9.1.2 Graphs of models ..... 191
9.1.3 Fitting approximations ..... 192
9.1.4 Critical abstractions ..... 192
9.1.5 Model-based diagnosis with multiple models ..... 193
9.1.6 Reasoning about model accuracy ..... 193
9.1.7 Microscopic ontologies ..... 194
9.2 Logical approaches ..... 195

**10. Conclusions** ..... 197
10.1 Summary and contributions ..... 197
10.2 Future work ..... 198

**A. Examples of causal approximations** ..... 201

**B. Example devices** ..... 207
B.1 Bimetallic strip temperature gauge ..... 207
B.2 Bimetallic strip thermostat ..... 208
B.3 Flexible link temperature gauge ..... 209
B.4 Galvanometer temperature gauge ..... 210
B.5 Electric bell ..... 211
B.6 Magnetic sizing device ..... 212
B.7 Carbon pile regulator ..... 213
B.8 Electromagnetic relay thermostat ..... 214
B.9 Tachometer ..... 215
B.10 Car distributor system ..... 216

**C. Composable operators** ..... 217
C.1 Influences ..... 217
C.2 Kirchhoff's laws ..... 218
C.3 Other methods ..... 220

**D. Matchings in bipartite graphs** ..... 223

**References** ..... 227

# List of Figures


1.1 A temperature gauge ..... 2
1.2 The possible models of a wire. ..... 3
1.3 A possible model of the temperature gauge ..... 5
1.4 Causal ordering of the parameters ..... 6
1.5 Algorithm for finding a minimal causal model. ..... 9

2.1 A temperature gauge ..... 15
2.2 A set of equations describing the temperature gauge ..... 16
2.3 Model fragment describing a wire as a resistor. ..... 17
2.4 Model fragment describing a wire as an ideal conductor. ..... 17
2.5 Model fragment describing the temperature dependence of the wire's length. ..... 17
2.6 Model fragments describing electrical conduction in a wire. ..... 20
2.7 Approximation relation between the electrical conduction model fragments. ..... 21
2.8 Model fragments describing a wire's resistance. ..... 22
2.9 Structural description of the temperature gauge. ..... 23
2.10 The `Resistor` model fragment class. ..... 26
2.11 The `Two-terminal-electrical-component` model fragment class. 28

3.1 A set of equations with two onto causal mappings ..... 39
3.2 Graph representing a set of equations ..... 42
3.3 Causal ordering algorithm ..... 43
3.4 A possible model of the temperature gauge ..... 44
3.5 Bipartite graph representing the equations in Figure 3.4. ..... 45
3.6 The direct causal dependencies generated by the causal mapping corresponding to the complete matching shown in Figure 3.5. .... 45
3.7 Algorithm for checking whether a model can explain the expected behavior. ..... 49

4.1 Mapping of parameters to equations ..... 74

5.1 Function *find-minimal-causal-model* ..... 90
5.2 Model fragments describing electrical conduction in `wire-1` ..... 96
5.3 Model fragments describing a wire's resistance. ..... 96

5.4 Causal orderings generated from $M_1$ and $M_2$ ..... 102
5.5 Causal ordering generated from $M_1'$ ..... 102
5.6 Model fragments in the `Electrical-conductor(wire-1)` assumption class ..... 103
5.7 The function *simp-by-approximating* ..... 111
5.8 Simplifying a model by dropping model fragments ..... 116

6.1 Partial causal mappings $H$ and $H'$ ..... 125
6.2 A motivating example ..... 126
6.3 Example of the graph $G'$ ..... 130
6.4 The resulting onto causal mapping $H''$ ..... 133
6.5 Assumption classes and model fragments ..... 136
6.6 Model fragment describing the velocity of a falling raindrop ..... 139
6.7 Model fragment describing the raindrop's terminal velocity ..... 139
6.8 Assumption classes, and their model fragments, describing the temperature of objects 1 and 2, and the heat path between them.. 143
6.9 Causal ordering before and after equilibration ..... 143

7.1 Ionization reactions that occur on dissolving $AH$ in water ..... 146
7.2 Rules for order of magnitude reasoning ..... 148
7.3 A backtrack tree ..... 150
7.4 Alternate rules for order of magnitude reasoning ..... 163

8.1 Algorithm for finding an initial causal model ..... 172
8.2 Components and their initial models ..... 173
8.3 Component models after the expected behavior parameters have been included ..... 176
8.4 Component models after the heuristic coherence constraints have been repeatedly satisfied ..... 178
8.5 The initial causal model ..... 180
8.6 The two stages of simplifying the initial causal model ..... 182

B.1 Bimetallic strip temperature gauge ..... 207
B.2 Bimetallic strip thermostat ..... 208
B.3 Flexible link temperature gauge ..... 209
B.4 Galvanometer temperature gauge ..... 210
B.5 Electric bell ..... 211
B.6 Magnetic sizing device ..... 212
B.7 Carbon pile regulator ..... 213
B.8 Electromagnetic relay thermostat ..... 214
B.9 Tachometer ..... 215
B.10 Car distributor system ..... 216

D.1 Matchings in a bipartite graph ..... 224
D.2 Algorithm for finding a maximum matching ..... 225

# List of Tables

1.1 Examples of causal approximations ..... 9

2.1 Composable operators ..... 19

7.1 NAPIER's run times on an Explorer II, with and without causal ordering. ..... 157
7.2 Maximum value of $\Delta$ for each example. ..... 159
7.3 Properties of the causal ordering graph ..... 160

8.1 Number of components and operating regions used in each device. 183
8.2 Summary of experimental results ..... 185
8.3 Number of model fragments in the most accurate model, the initial causal model, and the minimal causal model constructed by the program. ..... 186
8.4 Number of model fragments that were dropped and approximated in simplifying the initial causal model. ..... 187

# 1. Introduction

One of the earliest important ideas in Artificial Intelligence is that effective problem solving requires the use of adequate models of the domain [Amarel, 1968]. Adequate models incorporate abstractions and approximations that are well suited to the problem solving task. In most Artificial Intelligence research, models are hand-crafted by a user. The user must decide what domain phenomena are relevant, and must select appropriate abstractions and approximations that adequately describe these phenomena. In most real-world domains, constructing such models is a difficult, error-prone, and time-consuming task. Automating the construction of adequate models overcomes these drawbacks and provides future intelligent programs with a useful modeling tool. In this thesis we investigate the problem of selecting adequate models in the domain of physical systems.

## 1.1 Models and tasks

Consider, for example, the schematic of a bimetallic strip temperature gauge, from [Macaulay, 1988], shown in Figure 1.1. This temperature gauge consists of a battery, a wire, a bimetallic strip, a pointer, and a thermistor. A thermistor is a semi-conductor device; a small increase in its temperature causes a large decrease in its resistance. A bimetallic strip has two strips made of different metals welded together. Temperature changes cause the two strips to expand by different amounts, causing the bimetallic strip to bend.

Consider, now, the task of explaining how the temperature gauge works, i.e., how the temperature of the thermistor determines the position of the pointer along the scale. A trained engineer is able to look at this schematic for a few moments, and provide the following explanation: the thermistor senses the water temperature. The thermistor's temperature determines the thermistor's resistance, which determines the current flowing in the circuit. This determines the amount of heat dissipated in the wire, which determines the temperature of the bimetallic strip. The temperature of the bimetallic strip determines its deflection, which determines the position of the pointer along the scale.

A crucial part of how the engineer constructs the above explanation is his or her ability to pick out *just* the relevant phenomena that are needed to

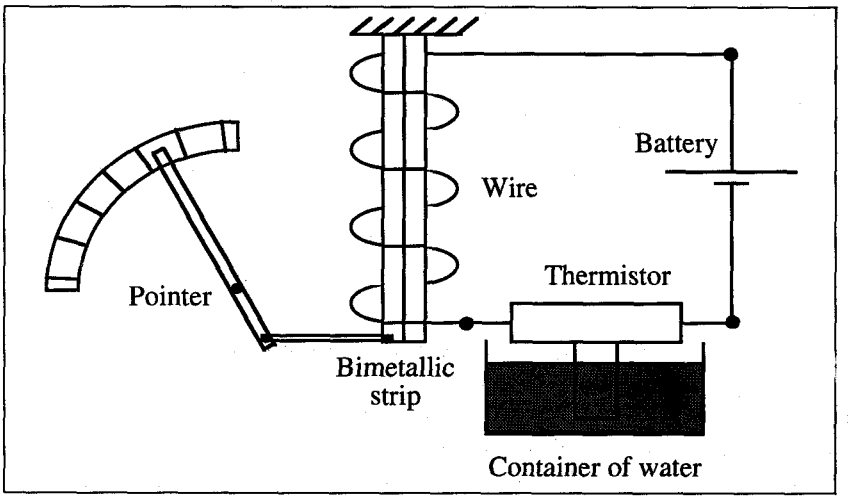



**Fig. 1.1.** A temperature gauge

be modeled. In particular, the engineer decided that the important thing to model about the wire is that it generates heat as current flows through it. The explanation is not cluttered by references to irrelevant phenomena, such as the electromagnetic field generated by the current flow in the wire.

Now, consider a slightly different task: the task of explaining how the atmospheric temperature affects the working of the temperature gauge, i.e., how the temperature of the atmosphere affects the position of the pointer along the scale. The engineer's explanation would be as follows: the temperature of the atmosphere determines the temperature of the bimetallic strip, which determines the deflection of the bimetallic strip. The amount of the deflection determines the position of the pointer along the scale.

In constructing the above explanation, the engineer completely disregarded the electrical properties of the wire, the battery, and the thermistor. Modeling these phenomena is not relevant to explaining how the temperature of the atmosphere affects the position of the pointer along the scale.

In addition to being able to decide which phenomena must be modeled, the engineer is also able to identify just the right models for each relevant phenomena. For example, in modeling electrical conduction in the wire, the engineer had to choose between modeling it as an ideal conductor, a constant resistance resistor, or a resistor whose resistance depends on its temperature. The engineer chose the constant resistance resistor model because (a) no heat is dissipated by an ideal conductor, and hence modeling the wire's resistance is crucial to understanding how the temperature gauge works; and (b) modeling the dependence of the wire's resistance on its temperature is unnecessary—assuming that the resistance is constant is adequate for explaining the temperature gauge's functioning.

Of course, what is meant by "just the right model" for each relevant phenomena is task dependent. For example, consider the following analysis task: predict the position of the pointer along the scale for a particular thermistor temperature. If a high fidelity prediction is required, i.e., if the pointer's position must be predicted with high accuracy, then the engineer would model the dependence of the wire's resistance on its temperature. On the other hand, if a lower fidelity prediction is acceptable, the engineer would once again use the simpler, constant resistance model for the wire, thereby simplifying the prediction process.

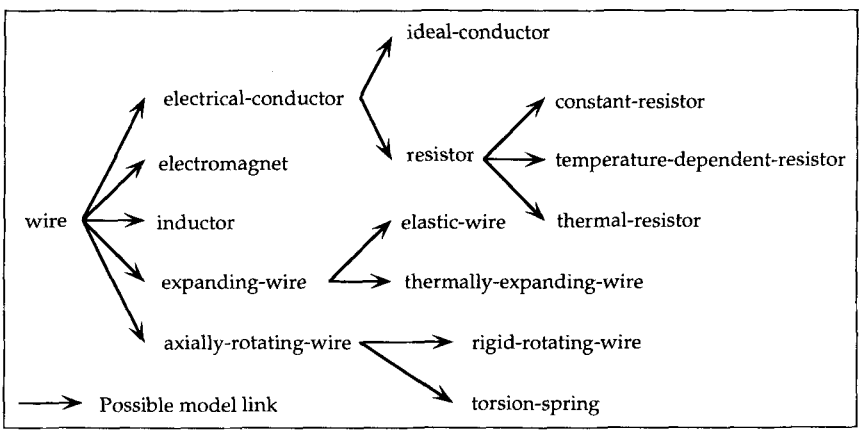



**Fig. 1.2.** The possible models of a wire.

What makes the above modeling decisions particularly intriguing is that there is usually a *very* large space of possible models to choose from. Figure 1.2 shows part of the space of possible models of a wire. We can choose to model its electrical, electromagnetic, or thermal properties, or we can choose to model its expansion or rotation. If we choose to model it as an electrical conductor, we must choose between modeling it as an ideal conductor, or as a resistor, in which case we must choose between modeling the resistance as a constant, or as dependent on the temperature. In addition, we can choose to model the heat generated in the wire due to current flow.

All the parts of the temperature gauge have a similarly large set of possible models. Hence, the set of possible models of the temperature gauge, constructed by selecting an appropriate subset of models for each of its parts, is combinatorially large. And yet, an engineer, after only a little thought, is able to select an adequate model that is specifically tailored for each task.

## 1.2 Problem statement

This thesis is about automating the engineer's ability to select adequate models for specific tasks. We cast the problem of selecting adequate models as a search problem. To do this, we must answer the following three questions:

- *What is a model, and what is the space of possible models?* (What is the search space?)
- *What is an adequate model?* (What is the goal criterion?)
- *How do we search the space of possible models for adequate models?* (What is the search strategy?)

The thesis proposes answers to each of the above questions in the domain of physical systems and for the task of generating *parsimonious causal explanations*.

## 1.3 Proposed solution: An overview

In this section we give a brief overview of our answers to the questions raised in the previous section. The rest of the thesis develops these ideas in detail.

### 1.3.1 What is a model and what is the space of possible models?

In this thesis, we will be concerned with models of the behavior of physical systems. Such models are best expressed as a set of algebraic and/or differential equations, that describe various phenomena of interest. Hence, a model is a set of equations. However, rather than viewing a model as just a set of equations, we will view it as a set of *model fragments*. A model fragment is a set of equations that partially describe a single phenomenon, usually a single mechanism. For example,

$$\{V_w = iR_w\}$$

is a model fragment describing electrical conduction in the wire. Note that it is a partial description of electrical conduction, since it does not include any description of the variation of the resistance $R_w$. Figure 1.3 shows the model fragments, and associated equations, in a possible model of the temperature gauge shown in Figure 1.1. Model fragments provide an appropriate level of description: (a) they are much easier to create than complete models; (b) unlike complete models, they are significantly more reusable; and (c) not all meaningful physical phenomena can be represented by a single equation.

The space of possible models is defined implicitly by the set of model fragments that can be composed to form models. The set of model fragments that can be so composed is defined by the *structure* of the physical system. The structure of the physical system is a description of the parts of the

| | |
|---|---|
| `Linkage(bms,ptr)`: | $\theta_p = k_1 x_b$ |
| `Thermal-bms(bms)`: | $x_b = k_2 T_b$ |
| `Heat-flow(bms,atm)`: | $f_{ba} = k_3(T_b - T_a)$ |
| `Heat-flow(wire,bms)`: | $f_{wb} = k_4(T_w - T_b)$ |
| `Constant-temperature(atm)`: | $exogenous(T_a)$ |
| `Thermal-equilibrium(bms)`: | $f_{ba} = f_{wb}$ |
| `Thermal-equilibrium(wire)`: | $f_{wb} = f_w$ |
| `Resistor(wire)`: | $V_w = i_w R_w$ |
| `Constant-resistance(wire)` | $exogenous(R_w)$ |
| `Thermal-resistance(wire)`: | $f_w = V_w i_w$ |
| `Electrical-thermistor(thermistor)`: | $V_t = i_t R_t; \quad R_t = k_5 e^{k_6/T_t}$ |
| `Constant-voltage-source(battery)`: | $exogenous(V_v)$ |
| `Kirchhoff's laws`: | $V_v = V_w + V_t; \quad i_v = i_t; \quad i_t = i_w$ |
| `Input`: | $exogenous(T_t)$ |

| | |
|---|---|
| $\theta_p$: Pointer angle | $x_b$: Bms deflection |
| $R_w$: Wire resistance | $R_t$: Thermistor resistance |
| $i_t$: Thermistor current | $V_t$: Thermistor voltage |
| $i_w$: Wire current | $V_w$: Wire voltage |
| $i_v$: Battery current | $V_v$: Battery voltage |
| $T_b$: Bms temperature | $T_a$: Atm temperature |
| $T_w$: Wire temperature | $T_t$: Thermistor temperature |
| $f_{ba}$: Heat flow (bms to atm) | $f_{wb}$: Heat flow (wire to bms) |
| $f_w$: Heat generated in wire | $k_j$: Exogenous constants |

**Fig. 1.3.** A possible model of the temperature gauge

system, and how they are put together. The parts that can be used to describe a system's structure are drawn from a component library. Each component in this library is associated with a set of model fragments, describing different aspects of the component's behavior. For example, the set of model fragments associated with a wire would include those shown in Figure 1.2. A model of the physical system is a subset of the model fragments associated with each of the components in the system.

Hence, our answer to the first question is:

- A model is a set of model fragments.
  The space of possible models of the physical system are defined by the structure of the system and a component library.

### 1.3.2 What is an adequate model?

We define the adequacy of a model using three criteria: (a) the task; (b) domain dependent constraints; and (c) simplicity.

**The task.** The adequacy of a model can only be determined with respect to a task. In this thesis, we will be concentrating on the task of providing *causal explanations* for a phenomenon of interest. A causal explanation is an explanation in terms of the underlying causal mechanisms of the domain.

For example, the explanations in the previous section were causal explanations. We have chosen this task because of its importance in reasoning about physical systems. Weld and de Kleer [Weld and de Kleer, 1990, page 612] summarize its importance as follows:

> ... humans expect to be provided explanations in causal terms. ... Part of the motivation for developing a theory of causality is as a vehicle for a system to explain its conclusions.
> Many qualitative physics researchers have adopted the far stronger position that causality is fundamental and plays a central role in reasoning about physical systems. ... Causal explanations are important to engineers because they are an explicit representation of how a device achieves its behavior. This explanation itself forms the basis for subsequent reasoning. In design tasks it is important to reason backward from effects to causes to identify what changes to make to a device to better achieve its specifications. In diagnosis tasks, it is important to reason backward to pinpoint what could have caused the symptoms. The causal explanation can guide subsequent quantitative analysis ...

Hence, given a phenomenon of interest, the fundamental criterion for the adequacy of a model is whether or not it is able to provide a causal explanation of the phenomenon. To check whether or not a model can provide an explanation for a phenomenon, we generate the *causal ordering* [de Kleer and Brown, 1984; Forbus, 1984; Williams, 1984; Iwasaki and Simon, 1986b; Iwasaki, 1988] of the parameters of the model using the equations of the model. The causal ordering of the parameters is a dependency ordering of the parameters that reflects an engineers notion of causal dependence between the parameters. The causal ordering is used to check whether or not a model can provide an explanation for a phenomenon.

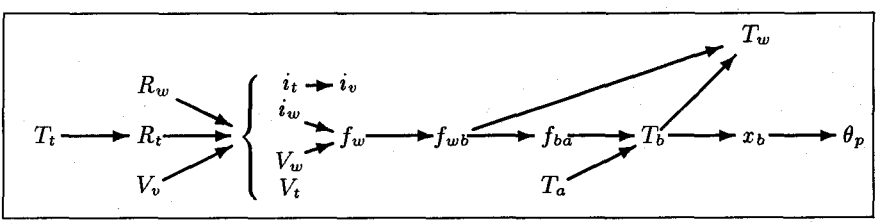



**Fig. 1.4.** Causal ordering of the parameters

For example, suppose we want to explain how the temperature gauge in Figure 1.1 works, i.e., to explain how the temperature of the thermistor ($T_t$) causally determines the angular position of the pointer ($\theta_p$). Figure 1.4 shows the causal ordering generated from the model in Figure 1.3. Since $\theta_p$ is causally dependent on $T_t$ in this causal ordering, the model in Figure 1.3 is

adequate for the task of explaining how the temperature gauge in Figure 1.1 works.

**Domain dependent constraints.** In addition to requiring that an adequate model be able to explain the phenomenon of interest, an engineer may want it to satisfy a set of domain dependent constraints. Such constraints can stem from the structure and the behavior of the physical system. For example, the following constraint:

```
(implies
   (and (Electromagnet ?object)
        (Wire ?object)
        (coiled-around ?object ?core)
        (magnetic-material ?core))
   (Magnet ?core))
```

requires that if the electromagnetic field generated by a wire is modeled and the wire is coiled around a core made of a magnetic material, then the core must be modeled as a magnet. The justification for this domain dependent constraint is that the core amplifies the magnetic field by three or four orders of magnitude, converting the core into a powerful magnet. Hence, under these circumstances, an engineer would not consider the model to be adequate unless the core were modeled as a magnet. More generally, an adequate model must satisfy all such domain dependent constraints.

**Simplicity.** Not all explanations of a phenomenon are *parsimonious*. A parsimonious causal explanation is a causal explanation with a minimum of irrelevant detail. Irrelevant detail is introduced into explanations because either (a) irrelevant phenomena are modeled; or (b) needlessly complex models of relevant phenomena are used. For example, we could introduce irrelevant detail into an explanation of how the temperature gauge in Figure 1.1 works by modeling the electromagnetic field generated by the wire. We could also introduce irrelevant detail into this explanation by modeling the temperature dependence of the wire's resistance, since approximating the wire's resistance by assuming that it is constant is adequate for explaining how the temperature gauge works.

To minimize the amount of irrelevant detail in an explanation, the model generating the explanation must be as *simple* as possible. The notion of model simplicity that supports the generation of parsimonious causal explanations is based on a primitive *approximation* relation between model fragments. The intuition underlying our definition of model simplicity is that modeling fewer phenomena more approximately leads to simpler models. An adequate model is required to be as simple as possible according to this ordering.

Hence, our answer to the second question is:

- An adequate model
  - is able to provide causal explanations for the phenomenon of interest;

- satisfies any domain dependent constraints; and
- is as simple as possible.

Let us say that a model is a *causal model*, with respect to a phenomenon of interest, if and only if it is able to explain the phenomenon and if the domain dependent constraints are satisfied. Hence, an adequate model is a minimal causal model, i.e., a causal model such that no simpler model is a causal model.

### 1.3.3 How do we find adequate models?

Given the structure of the physical system and a component library, there is an exponentially large space of possible models of the physical system. We will show later that the problem of finding an adequate model in this space of possible models is intractable (NP-hard). Intuitively, this means that, to find an adequate model, we can do little better than check each model in the exponentially large space of possible models. Even for small systems, this space is extremely large, so any brute force approach is out of the question. However, this seems to contradict the observation that expert engineers are able to provide parsimonious causal explanations for phenomena after only a little bit of thought. This means that the world provides additional structure, which can be exploited to develop an efficient, polynomial time model selection algorithm.

**Upward failure property.** One property that is likely to be satisfied in modeling the physical world is the *upward failure property*. The upward failure property states that if a model is not a causal model, then no simpler model is a causal model. Intuitively, this seems like a reasonable property. After all, if a model is unable to explain the phenomenon of interest, then there is little reason to believe that a simpler model is able to provide an explanation. If the upward failure property is satisfied, then, given an initial causal model, the algorithm shown in Figure 1.5 can be used to efficiently find an adequate model, i.e., a minimal causal model. In this algorithm, $M$ is the initial causal model, with an immediate simplification of $M$ being produced by either replacing a model fragment in $M$ by an immediate approximation, or by dropping a model fragment. The algorithm works by continually replacing $M$ by an immediate simplification that is a causal model, until all the immediate simplifications of $M$ are not causal models. The upward failure property then tells us that $M$ is a minimal causal model.

**Causal approximations.** The upward failure property is useful because it leads to a polynomial time algorithm for finding an adequate model. However, checking whether or not the space of possible models satisfies the upward failure property is, in general, difficult. This is because the upward failure property is a global property. To address this shortcoming we have identified a set of local properties, which can be easily checked as we build up the component library, that entail the upward failure property. In particular, we have

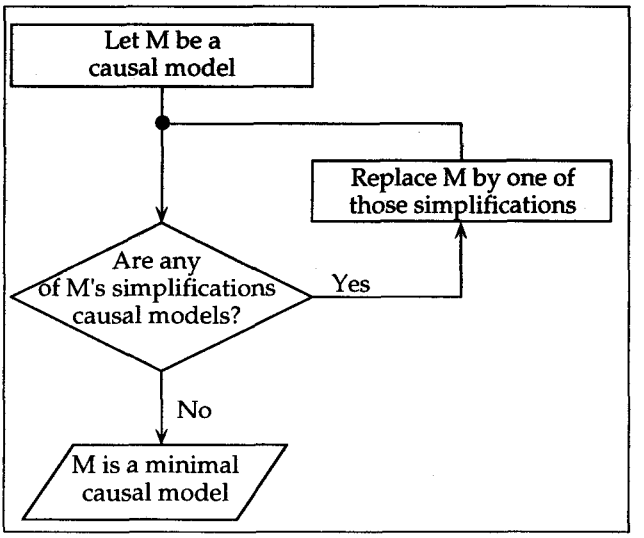



**Fig. 1.5.** Algorithm for finding a minimal causal model.

identified an important class of approximations called *causal approximations*. When all the approximations are causal approximations, replacing a model fragment by a more accurate model fragment results in a superset of causal relations between parameters. This forms the basis for proving the upward failure property, and the use of the algorithm shown in Figure 1.5. Causal approximations are particularly useful because they are common in modeling the physical world. For example, Table 1.1 shows a number of commonly used approximations, all of which are causal approximations. These approximations are described in greater detail in Appendix A. The exact definition of a causal approximation is found in Chapter 5.

**Table 1.1.** Examples of causal approximations

| | |
|---|---|
| Inertialess objects | Rigid bodies |
| Inviscid flow | Elastic collisions |
| Frictionless motion | Ideal gas law |
| Zero or constant gravity | Ideal heat engines |
| Non-relativistic mass and motion | No thermal expansion |
| Ideal thermal insulators and conductors | Constant thermal conductance |
| Ideal electrical insulators and conductors | Constant resistance and resistivity |

**Finding an initial causal model.** The algorithm in Figure 1.5 requires us to find an initial causal model $M$, from which to start the simplification. A natural choice for this model is the most accurate model describing the physical system. However, starting with the most accurate model is often undesirable. Hence, we introduce a heuristic method, based on the *compo-*

*nent interaction heuristic*, that allows us to find a initial causal model. For example, one component interaction heuristic is the following:

```
(implies
   (and (terminals ?object ?term1)
   (voltage-terminal ?term1)
   (connected-to ?term1 ?term2))
(voltage-terminal ?term2))
```

which says that if any terminal of a component is modeled as a voltage terminal, then all terminals connected to that voltage terminal must also be modeled as voltage terminals. This allows the components corresponding to the connected terminals to interact by sharing voltages at those terminals. Note that the above constraint does not require all connected terminals to be modeled as voltage terminals; it only says that if a terminal is a voltage terminal, then terminals connected to it must also be voltage terminals. We use such heuristic constraints to build up a causal model, and then use the algorithm in Figure 1.5 to find a minimal causal model.

In summary, the answer to the third question is as follows:

- When all the approximations are causal approximations, an adequate model can be found efficiently by first identifying an initial causal model, and then simplifying it.

## 1.4 Contributions

The thesis makes the following important contributions:

- It introduces a novel criterion for defining model adequacy: the criterion that a model must be able to provide a parsimonious causal explanation for a phenomenon of interest.
- It presents a clear formalization of the model selection problem, making the problem amenable to theoretical analysis.
- It uses the above formalization to analyze the complexity of finding adequate models, and shows that the problem is intractable. This analysis yields three different sources of intractability, which can be summarized as follows: (a) deciding *what* phenomena to model; (b) deciding *how* to model selected phenomena; and (c) having to satisfy all the domain dependent constraints.
- It introduces a new class of approximations, called causal approximations, which are commonly found in modeling the physical world. Causal approximations are important because they lead to the development of an efficient algorithm for finding adequate models.

- It introduces a novel order of magnitude reasoning method which is used to generate the behavior of a physical system. The method strikes a balance between purely quantitative and purely qualitative reasoning, and is based on defining the order of magnitude of a quantity on a logarithmic scale. This makes the method applicable even in the presence of non-linear simultaneous equations.
- It introduces the component interaction heuristic that is useful in finding causal models.
- It describes an implemented representation methodology for representing the space of possible models of a physical system.
- It describes the results of testing our implementation of the model selection algorithm on a variety of electromechanical devices.

## 1.5 Readers guide

The rest of the thesis presents the details of our solution to the model selection problem. Chapter 2 is a detailed answer to the first of our three questions. It describes models and model fragments, and shows how they are represented. Chapter 3 is a detailed answer to the second of our three questions. It describes our criteria for the adequacy of a model. These two chapters are central to understanding this thesis.

Chapter 4 presents a formalization of the model selection problem, and uses this formalization to analyze the complexity of finding adequate models. It shows that the general problem of finding adequate models in NP-hard, and identifies three different sources of intractability. Section 4.1, which presents the formalization, is necessary for understanding Chapters 5 and 6. However, readers not interested in the details of the proofs of intractability can skip the rest of the chapter.

Chapter 5 contains some of the main results of this thesis. It introduces the upward failure property, and uses the upward failure property to develop an efficient algorithm for finding an adequate model. It then introduces a number of local properties of a knowledge base that ensure that the global upward failure property is satisfied. In particular, this chapter introduces the class of causal approximations, and discusses their role in ensuring that the upward failure property is satisfied. Chapter 6 generalizes the results of Chapter 5 to models involving differential equations.

Chapter 7 presents the novel order of magnitude reasoning method that we use to generate the behavior of the physical system. This behavior is used to evaluate some of the domain dependent constraints introduced in chapter 3. This chapter is self-contained, and can be read independently of the rest of the thesis.

Finally, Chapter 8 presents the component interaction heuristic and the implemented program for model selection. It also reports on our experimental

results. Related work is discussed in Chapter 9, and conclusions and future work are discussed in Chapter 10.

We conclude this introductory chapter with a brief note on papers that describe various aspects of this dissertation. The main results of Chapters 4 and 5 are presented in [Nayak, 1994]. A shorter exposition of the same ideas can be found in [Nayak, 1992a]. An overview of Chapters 2, 3, and 8 is presented in [Nayak and Joskowicz, 1995], with a shorter description in [Nayak *et al.*, 1992]. Finally, much of Chapter 7 is reproduced in [Nayak, 1992b].

# 2. Models and model fragments

In this chapter we describe the types of *models* that we consider in this thesis. Fundamentally, we will be considering models of the behavior of physical systems, that are best represented as sets of equations. Section 2.1 discusses the different types of equations that can be used in models of physical systems, and Section 2.2 discusses the need for multiple models of a single system. The next two sections introduce model fragments, and show how model fragments can be used to represent the space of possible models of a physical system. The final section of this chapter discusses the actual representational mechanisms that we use to implement these ideas. In particular, we introduce a class level description of components and model fragments, and show how these classes are organized.

## 2.1 Models of the behavior of physical systems

In this thesis we will be concerned with models of the behavior of physical systems, typically of engineered devices. (In the rest of the thesis we will use "device" as a synonym for "physical system.") Models of device behavior are best represented as a set of *equations* that relate a set of *parameters*.

### 2.1.1 Parameters

A parameter is a numerical attribute representing a physical property of the device, e.g., temperature of an object, voltage drop across an electrical conductor, magnetic field in a region. Parameters are usually functions of both time and space, e.g., the temperature of an object can vary with time and with location within the object.

It is common to disregard the dependence of parameter values on space and/or time. A *lumped parameter* model disregards the dependence of parameter values on spatial location. Such models make the assumption that the variation of parameter values over a specific region of space is negligible, with the primary variation being as a function of time. For example, we may choose to model the temperature of an object as a lumped parameter, i.e., assume that the temperature is uniform throughout the object, though the temperature may still vary with time.

An *equilibrium* model disregards the dependence of parameter values on time. Such models are useful for modeling the asymptotic behavior of devices, i.e., device behavior after a sufficiently long time has elapsed, so that any transient behavior has died out. For example, consider a wall separating a heated room from the cold air outside. An equilibrium model can be used to model the eventual temperature profile in the wall.

### 2.1.2 Equations

Equations are relations between parameters. Different types of equations are used to represent different types of models. The most general types of equations are *partial differential equations*. Partial differential equations can model the variation of parameter values over both time and space. For example, the well known Navier-Stokes equation [Welty *et al.*, 1984] is a partial differential equation that describes fluid flow as a function of both time and space.

*Ordinary differential equations* can model the variation of parameter values only as a function of a single independent variable, such as time. Hence, ordinary differential equations are used to represent lumped parameter device models. For example, Hooke's law [Halliday and Resnick, 1978] is an ordinary differential equation that describes the behavior of a simple harmonic oscillator like a spring-block system.

*Algebraic equations* do not contain any partial or total derivatives. Hence, they can be used to represent equilibrium, lumped parameter device models. For example, Ohm's law [Halliday and Resnick, 1978] is an algebraic equation describing the relationship between current flow through a resistor and the voltage drop across the resistor.

Another widely used type of equation is the *qualitative equation* [Bobrow, 1984; Kuipers, 1986]. Qualitative equations do not relate the exact numerical values of parameters. Instead, they represent functional dependencies and monotonicity relations between parameters. For example, if we do not know the exact functional form of the relation between the resistance of a wire and its temperature, we could use a qualitative equation to express the fact that the resistance functionally depends on the temperature, and that increasing the temperature results in an increase in resistance.

In this thesis, we will only consider lumped parameter models. However, we will consider both time-varying models, as well as equilibrium models. Hence, we have the following:

- A device model is a set of algebraic, qualitative, and/or ordinary differential equations, relating a set of parameters.

Figure 2.1 reproduces the temperature gauge introduced in the previous chapter. Figure 2.2 shows a set of equations that describe this temperature gauge. This set of equations represents an equilibrium model of the temperature gauge, since no differential equations are used. The equation

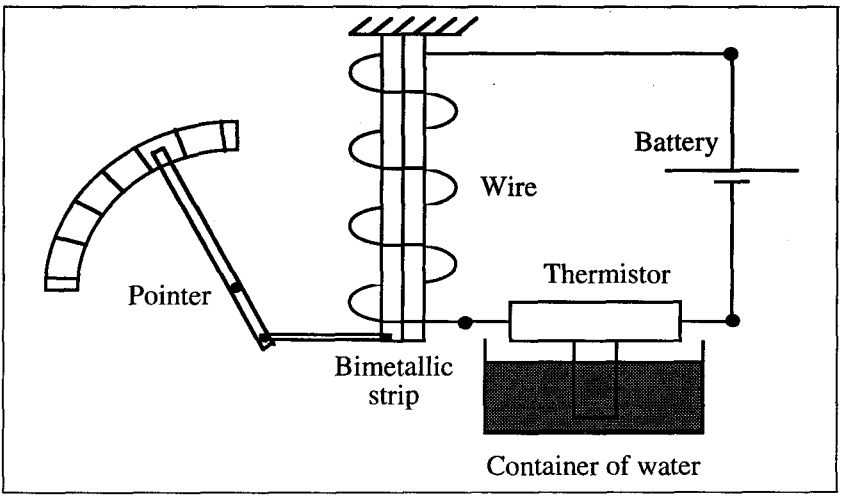



**Fig. 2.1.** A temperature gauge

*exogenous*$(Q)$ represents the fact that the value of $Q$ is determined exogenously; it can be viewed as a shorthand for the equation $Q = c$, for some constant $c$. The equation $M\text{-}(Q_1, Q_2)$ is a qualitative equation representing the functional dependence of $Q_1$ on $Q_2$, and the fact that if $Q_2$ increases then $Q_1$ decreases [Kuipers, 1986].

## 2.2 Multiple models

Any device can be modeled in many different ways, i.e., it can be described by different sets of equations. Different device models differ because they give different answers to the following two fundamental questions:

- *What must be modeled?* Different models can differ because they choose to model different physical phenomena. For example, the physical phenomena modeled in Figure 2.2 include the heat generated due to current flow in the wire, but not the electromagnetic field generated by the same current flow. Models can also differ because they choose different granularities, i.e., they choose a different set of objects to model. For example, the model in Figure 2.2 chose a granularity that includes the bimetallic strip as a single object. A different model might have chosen a different granularity, such as one that separately modeled the two strips of the bimetallic strip.
- *How must the chosen things be modeled?* Even though models may choose to model the same phenomena at the same level of granularity, they may differ based on the specific models they choose. For example, the model in Figure 2.2 models electrical conduction in the wire as a constant resistance resistor. However, other models could have chosen to use different models of

$$\theta_p = k_1 x_b$$
$$x_b = k_2 T_b$$
$$f_{ba} = k_3(T_b - T_a)$$
$$f_{wb} = k_4(T_w - T_b)$$
$$exogenous(T_a)$$
$$f_{ba} = f_{wb}$$
$$f_{wb} = f_w$$
$$V_w = i_w R_w$$
$$exogenous(R_w)$$
$$f_w = V_w i_w$$
$$V_t = i_t R_t$$
$$M\text{–}(R_t, T_t)$$
$$exogenous(V_v)$$
$$V_v = V_w + V_t$$
$$i_v = i_t$$
$$i_t = i_w$$
$$exogenous(T_t)$$

| | |
|---|---|
| $\theta_p$: Pointer angle | $x_b$: Bms deflection |
| $R_w$: Wire resistance | $R_t$: Thermistor resistance |
| $i_t$: Thermistor current | $V_t$: Thermistor voltage |
| $i_w$: Wire current | $V_w$: Wire voltage |
| $i_v$: Battery current | $V_v$: Battery voltage |
| $T_b$: Bms temperature | $T_a$: Atm temperature |
| $T_w$: Wire temperature | $T_t$: Thermistor temperature |
| $f_{ba}$: Heat flow (bms to atm) | $f_{wb}$: Heat flow (wire to bms) |
| $f_w$: Heat generated in wire | $k_j$: Exogenous constants |

**Fig. 2.2.** A set of equations describing the temperature gauge

electrical conduction, e.g., by modeling the the wire as an ideal conductor, or as a resistor whose resistance depends on its temperature. Similarly, the model in Figure 2.2 uses an equilibrium model for the temperature of the wire, while other models might choose to use a differential equation model to model the transient behavior of the wire's temperature.

## 2.3 Model fragments

Since a device can be modeled in a variety of different ways, it is important that we be able to represent the space of possible models of the device. We represent this space using *model fragments*.

### 2.3.1 What is a model fragment?

A model fragment is a set of independent equations that partially describe some physical phenomena at some level of granularity. Different model fragments can describe different phenomena, or can be different descriptions of

the same phenomena. For example, Figure 2.3 shows a model fragment that describes electrical conduction in a wire by modeling the wire as a resistor. Figure 2.4 shows a different model fragment that describes the same phenomena for the wire by modeling the wire as an ideal conductor. Finally, Figure 2.5 shows a model fragment that describes the temperature dependence of the wire's length, a completely different phenomena.

$$\{V_w = i_w R_w\}$$

**Fig. 2.3.** Model fragment describing a wire as a resistor.

$$\{V_w = 0\}$$

**Fig. 2.4.** Model fragment describing a wire as an ideal conductor.

$$\{l_w = l_{w0}(1 + \alpha_w(T_w - T_{w0}))\}$$

**Fig. 2.5.** Model fragment describing the temperature dependence of the wire's length.

In general, model fragments are only *partial* descriptions of phenomena. For example, the model fragment in Figure 2.3 only specifies the relation between the voltage ($V_w$) and the current ($i_w$); it does not say anything about the variation of the resistance of the wire. Additional model fragments describing the resistor's resistance are necessary to complete this description.

Model fragments can be viewed as either component model instances [de Kleer and Brown, 1984; Williams, 1984], or process instances [Forbus, 1984]. Component model instances and process instances usually have *applicability* conditions (e.g., operating conditions [de Kleer and Brown, 1984; Williams, 1984] or quantity conditions [Forbus, 1984]), that determine when the equations can be used. There are well developed techniques for handling such applicability conditions [Forbus, 1990; Crawford *et al.*, 1990; Iwasaki and Low, 1991]. Hence, in this thesis, rather than explicitly modeling and reasoning about these applicability conditions, we assume that the only model fragments under consideration are the ones whose applicability conditions are satisfied.

### 2.3.2 Advantages of model fragments

A device model is constructed by composing a set of model fragments, i.e., rather than viewing a model just as a set of equations, it is much more useful to think of it as a set of model fragments. Hence, we have the following alternative definition of a model:

- A model is a set of model fragments that describe some set of phenomena at some level of detail.

This viewpoint has a number of advantages. First, because model fragments are partial descriptions of a single phenomena, they usually consist of a small number of equations. Hence, constructing a library of model fragments is relatively easy. On the other hand, device models usually consist of a large number of equations, sometimes as many as hundreds of equations, because they are complete descriptions of a number of phenomena. Hence, constructing a model is much more difficult than constructing a model fragment.

Second, a set of model fragments is an implicit representation of a very large set of models. This is because any subset of this set of model fragments can be composed to form a model.[1] Hence, a set of model fragments is an implicit representation of an exponentially large set of models. Alternate representations of this large space of models, by explicitly representing each model, are unrealistic. To put it another way, explicitly representing the space of possible models restricts us to representing a much smaller set of models.

Third, model fragments are reusable, not just in different models of the same device, but in different models of different devices. For example, the model fragments shown in Figure 2.3–2.5 can be reused, not only in a number of different models of the temperature gauge shown in Figure 2.1, but also in models of other devices that use wires. This means that the effort of constructing a library of model fragments can be amortized over their use in a variety of different models.

### 2.3.3 Composing model fragments

The equations of a device model are created by composing the equations of the model fragments used to construct the model. In most cases, the composition is a straightforward union of the equations in the model fragments. However, because model fragments are partial descriptions of phenomena, there is a need to have special types of expressions that provide only partial information about equations. Such partial descriptions have associated with them a set of composition rules that are used to combine different partial descriptions to create a complete equation in the model.

Consider, for example, a bathtub partially filled with water. Suppose that a tap has been turned on to fill up the bathtub. Simultaneously, suppose that the drain plug in the bathtub has been opened to try and empty the bathtub. The net effect of these two water flows (i.e., from the tap into the bathtub, and out of the bathtub through the drain) can be described by the equation:

$$\frac{dV_{tub}}{dt} = f_{tap} - f_{drain}$$

where $V_{tub}$ is the volume of water in the tub, $f_{tap}$ is the rate at which water enters the tub through the tap, and $f_{drain}$ is the rate at which water leaves

[1] As we shall see later, not every subset of model fragments can be viewed as a model, but the basic observation still holds.

the tub through the drain. For maximum flexibility, it is useful to describe the two water flows using separate model fragments. What might the equations of these model fragments be? Intuitively, the model fragment describing the tap water flow must say that the water flowing through the tap tends to increase the volume of water in the bathtub. Similarly, the model fragment describing the drain water flow must say that the water flowing out of the drain tends to decrease the volume of water in the bathtub.

We can express this using the $I+$ and $I-$ operators introduced by Forbus [Forbus, 1984]. $I+(q_1, q_2)$ says that $q_2$ is a positive influence on $q_1$, while $I-(q_1, q_2)$ says that $q_2$ is a negative influence on $q_1$. Given a set of influences on a parameter $q$, we use the closed world assumption that these are the only influences on $q$ to construct an equation. For example, the model fragment describing the tap water flow would have the equation $I+(V_{tub}, f_{tap})$, and the model fragment describing the drain water flow would have the equation $I-(V_{tub}, f_{drain})$. Combining these two model fragments, and assuming that these are the only influences on $V_{tub}$, we get the equation

$$\frac{dV_{tub}}{dt} = f_{tap} - f_{drain}$$

**Table 2.1.** Composable operators

| | |
|---|---|
| $I+$ | Positive influence on derivatives |
| $I-$ | Negative influence on derivatives |
| *sum-term* | Term in a sum |
| *sum-to-zero* | Quantities that add up to zero<br>(Used for Kirchhoff's current law) |
| *same-value* | Quantities are equal<br>(Used for Kirchhoff's voltage law) |
| *same-reference* | Quantities have a common reference (potential)<br>(Used for Kirchhoff's voltage law) |
| *same-circuit* | Flows belong to the same circuit<br>(Used for Kirchhoff's current law) |

The use of composable operators like $I+$ and $I-$ are crucial to our use of model fragments as partial descriptions of phenomena. Table 2.1 shows the composable operators that we use. Brief descriptions have been included in this table, and more detailed descriptions are provided in Appendix C.

### 2.3.4 Relations between model fragments

We now turn to a discussion of some important relations between model fragments: *contradictory* and *approximation*. We also introduce *assumption classes*, and the *required assumption classes* of a model fragment.

**The *contradictory* relation.** As mentioned earlier, different model fragments can be descriptions of different phenomena, or can be different descriptions of the same phenomena. When model fragments describe the same phenomena, they often make contradictory assumptions about the domain. For example Figure 2.6 shows three different model fragments describing electrical conduction in a wire, which make contradictory assumptions. In particular, the ideal conductor model fragment assumes that the resistance of the conductor is zero, the ideal insulator model fragment assumes that the resistance of the conductor is infinite, while the resistor model fragment assumes that the resistance of the conductor is non-zero and finite.

| | |
|---|---|
| `Ideal-conductor(wire-1):` | $V_w = 0$ |
| `Ideal-insulator(wire-1):` | $i_w = 0$ |
| `Resistor(wire-1):` | $V_w = i_w R_w$ |

**Fig. 2.6.** Model fragments describing electrical conduction in a wire.

We represent the fact that model fragments make contradictory assumptions about the domain using the *contradictory* relation. If $m_1$ and $m_2$ are model fragments, then $contradictory(m_1, m_2)$ says that $m_1$ and $m_2$ make contradictory assumptions about the domain. It is important to note that the *contradictory* relation is a primitive, domain-dependent relation which cannot, in general, be derived from the equations of the model fragments. For example, there is nothing intrinsically contradictory about the equations of the ideal conductor model fragment and the ideal insulator model fragment, i.e., it is certainly possible that both the current through a conductor and the voltage drop across the conductor is zero. The contradiction between them is a domain fact. However, we assume that the *contradictory* relation is irreflexive (so that model fragments cannot contradict themselves), and symmetric (so that model fragments can only be mutually contradictory):

$$\neg contradictory(m_1, m_1) \tag{2.1}$$

$$contradictory(m_1, m_2) \Rightarrow contradictory(m_2, m_1) \tag{2.2}$$

**The *approximation* relation.** As discussed above, when two model fragments describe the same phenomenon, they often make contradictory assumptions about the domain. In addition to specifying that model fragments contradict each other, an engineer may be able to specify that one model fragment is a more *approximate* description of the phenomenon than the other. This means that the predictions made by the more accurate model fragment are "closer to reality" than the predictions made by the more approximate model fragment. We represent such knowledge using the *approximation* relation between model fragments. In particular, $approximation(m_1, m_2)$ says that the model fragment $m_2$ is a more approximate description of some phenomena than the model fragment $m_1$. For example, Figure 2.7 shows some

of the approximation relations between the model fragments shown in Figure 2.6.

*approximation*(Resistor(wire-1), Ideal-conductor(wire-1))
*approximation*(Resistor(wire-1), Ideal-insulator(wire-1))

**Fig. 2.7.** Approximation relation between the electrical conduction model fragments.

Once again, it is important to note that the *approximation* relation is a primitive, domain-dependent relation, and this relation cannot, in general, be derived from the equations of the model fragments. For example, there is nothing about the equations of the ideal conductor model fragment that tells us that it is necessarily a more approximate description of electrical conduction than the resistor model fragment; this just happens to be a domain fact discovered by scientists and engineers. However, we require that the *approximation* relation be irreflexive, anti-symmetric, and transitive (so that model fragments are not approximations of themselves, and *approximation* forms a partial ordering on the relative accuracy of the model fragments describing a phenomena):

$$\neg approximation(m_1, m_1) \tag{2.3}$$

$$approximation(m_1, m_2) \Rightarrow \neg approximation(m_2, m_1) \tag{2.4}$$

$$\begin{aligned} &approximation(m_1, m_2) \wedge approximation(m_2, m_3) \\ &\quad \Rightarrow approximation(m_1, m_3) \end{aligned} \tag{2.5}$$

Furthermore, since approximations make different, and hence contradictory, predictions about the same phenomenon, we require that all approximations are also mutually contradictory:

$$approximation(m_1, m_2) \Rightarrow contradictory(m_1, m_2) \tag{2.6}$$

**Assumption classes.** An *assumption class* is a set of model fragments that make different, contradictory assumptions about the domain. This means that an assumption class is a set of mutually contradictory model fragments, i.e., if $m_1$ and $m_2$ are model fragments, and $A$ is an assumption class, we have:

$$(m_1, m_2 \in A) \wedge m_1 \neq m_2 \Rightarrow contradictory(m_1, m_2) \tag{2.7}$$

One can see that the model fragments in Figure 2.6 form an assumption class describing electrical conduction in the wire. Figure 2.8 shows two model fragments forming an assumption class describing the resistance of a wire.

Recall that model fragments are partial descriptions of phenomena. Additional model fragments are required to complete this description. We represent the set of model fragments that can be used to complete a description by associating with each model fragment a set of *required assumption*

| | |
|---|---|
| `Constant-resistance(wire-1)` | $exogenous(R_w)$ |
| `Temperature-dependent-resistance(wire-1)` | $R_w = R_{w0}(1 + \alpha_w(T_w - T_{w0}))$ |
| | $exogenous(R_{w0})$ |
| | $exogenous(\alpha_w)$ |
| | $exogenous(T_{w0})$ |

**Fig. 2.8.** Model fragments describing a wire's resistance.

*classes*. Let $A$ be an assumption class required by model fragment $m$ (written $requires(m, A)$). This means that to complete the description of the phenomena described by $m$, we must include a model fragment from the assumption class $A$. For example, to complete the description of electrical conduction described by the resistor model fragment, we require a description of the resistance, i.e., the `Resistor(wire-1)` model fragment requires a model fragment from the assumption class shown in Figure 2.8.

## 2.4 Space of possible models

In the previous section, we have argued that a set of applicable model fragments form a compact representation of a very large space of possible models. In this section we discuss the following issue: given a device description, how do we decide which model fragments are applicable. Our answer to this issue can be summarized as follows:

- The set of applicable model fragments is the union of the model fragments associated with the *components* of the device.

We now discuss this in detail.

### 2.4.1 Device structure

The *structure* of a device is a description of the device which specifies the *components*, or parts, of the device, physical properties of these components, and how these components are put together to form the device.

The components that can be used to describe the structure of a device are drawn from a library of component types. For example, to define the structure of the temperature gauge shown in Figure 2.1, the component library must contain component types like `thermistor`, `wire`, `battery`, `bimetallic-strip`, and `pointer`.

Components are put together to form a device description with the use of *structural relations*. These relations are drawn from a library of structural relations, and include relations such as `connected-to` (indicating that two component terminals are connected), `coiled-around` (indicating that a wire is coiled around a component), and `meshed` (indicating that a pair of gears mesh with each other). Figure 2.9 shows a structural description of the temperature gauge in Figure 2.1.

```
(defdevice bimetallic-strip-temperature-gauge
           ((?V Battery)
            (?T Thermistor)
            (?W Wire)
            (?B Bimetallic-strip)
            (?P Pointer)
            (?L Linkage)
            (?ATM Atmosphere)
            (?A1 Axis)
            (?A2 Axis))
   (connected-to (battery-terminal-one ?V)
                 (wire-terminal-one ?W))
   (connected-to (battery-terminal-two ?V)
                 (thermistor-terminal-one ?T))
   (connected-to (thermistor-terminal-two ?T)
                 (wire-terminal-two ?W))
   (connected-to (bms-terminal-two ?B)
                 (linkage-terminal-one ?L))
   (connected-to (pointer-terminal-two ?P)
                 (linkage-terminal-one ?L))
   (coiled-around ?W ?B)
   (immersed-in ?B ?ATM)
   (immersed-in ?P ?ATM)
   (immersed-in ?V ?ATM)
   (immersed-in ?L ?ATM)
   (fixed-object (bms-terminal-one ?B))
   (can-rotate ?P ?A2)
   (bms-deformation-axis ?B ?A1))
```

**Fig. 2.9.** Structural description of the temperature gauge.

### 2.4.2 Structural abstractions

The structure of a device specifies the basic set of components in the device. This basic set of components can be augmented by recognizing *structural abstractions*. Structural abstractions are components that represent a set of other components in specific structural configurations. For example, components of the `Coil-structure` component type represent objects corresponding to a wire coiled around another object.

The component library contains rules that can be used to recognize instances of a structural abstraction in the structural description of a device. For example, the following rule is used to recognize `Coil-structures`:

```
(implies
   (and (Wire ?object)
        (coiled-around ?object ?core))
   (exists
      ?struc Coil-structure
      (and (coil-structure-wire ?struc ?object)
           (coil-structure-core ?struc ?core))))
```

Therefore, the set of all components of a device consist of the union of the set of basic components specified in the structural description, and the set of all structural abstractions that can be recognized using the rules in the component library. For example, applying the above rule to the device structure shown in Figure 2.9, we see that an instance of the `Coil-structure` is recognized, corresponding to the wire, `?W`, being coiled around the bimetallic strip, `?B`.

### 2.4.3 Possible models of a component

The space of model fragments that can be used to construct a device model is defined by associating with each component of the device, whether a basic component or a structural abstraction, a set of model fragments that can be used to describe that component. For example, we could associate with a wire, `wire-1`, the following model fragment describing electrical conduction in the wire:

$$\{V_w = i_w R_w\}$$

As discussed earlier, model fragments can be viewed either as "component models" [de Kleer and Brown, 1984; Williams, 1984] or "process models" [Forbus, 1984]. Hence, a model fragment associated with a component is a partial description of some physical phenomena, including some physical process, occurring in that component. It is worth noting that model fragments associated with structural abstractions can be used to represent physical processes that take place over more than one basic component. For example, if `cs1` is a structural abstraction representing the wire coiled around the bimetallic strip, then we could associate with it the following model fragment, describing heat flow from the wire to the bimetallic strip:

$$\{f_{cs1} = \gamma_{cs1}(T_{w1} - T_{b1})\}$$

where $f_{cs1}$ is the heat flow, $\gamma_{cs1}$ is the thermal conductance, $T_{w1}$ is the temperature of the wire, and $T_{b1}$ is the temperature of the bimetallic strip.

In summary, the space of possible models of a device is represented implicitly by the set of applicable model fragments that can be composed to form models of the device. The set of applicable model fragments is the union of the model fragments associated with each of the components of the device. In the next section we discuss our representation of the space of model fragments that can be used to describe a component.

## 2.5 Model fragment classes

Thus far we have been talking about components and model fragments as instance level descriptions, i.e., a component is a specific component used in a

specific device, and a model fragment is the specific set of equations describing some physical phenomenon in a specific component. However, building a library of components and model fragments requires that we provide class level descriptions, i.e., descriptions of classes of components and model fragments that can be instantiated to create structural descriptions and models for a variety of devices. To this end, we have devised an implemented language for specifying class level descriptions of components and model fragments. We now describe this language, and show how we represent the information described above.

### 2.5.1 What are component and model fragment classes?

Component and model fragment classes are just classes, where a class is viewed as a set of instances. Component classes are class level descriptions of components: components are just instances of the corresponding component classes. Model fragment classes are class level descriptions of phenomena. A component is modeled by a particular model fragment class by making the component an instance of the class.

Following [Hayes, 1979], classes can be viewed as unary predicates that are true of their instances. Functions and higher arity predicates are implemented as slots on instances. For example, if `s` is a binary predicate, and `u` and `v` are instances, then the literal `s(u,v)` is represented by placing the instance `v` on the `s` slot of the instance `u`.

Component and model fragment classes inherit various properties to their instances. The most important property that a model fragment class inherits to its instances is the equations describing the phenomena. These inherited equations form the model fragment describing the physical phenomena for that instance.

### 2.5.2 Typographic conventions

A few notes on typographic conventions.

- Names of components, model fragments, component classes, and model fragment classes will be typeset in `typewriter` font.
- Class names begin with an uppercase letter, while slot names and instance names begin with a lower case letter.
- If `M` is a model fragment class, and `c` is a component, then `M(c)` denotes the model fragment resulting from modeling `c` as an instance of `M`. `M(c)` will also be used to represent the ground literal expressing the fact that `c` is an instance of `M`. It will always be clear from the context whether `M(c)` represents a model fragment or a literal.
- Instances of component classes will often have names formed by concatenating the name of the component class with a number.

- Variables names will start with the "?" character. The variable "`?object`" used in class definitions is bound to the class instance under consideration.

To illustrate some of the above conventions, let `Wire` be a component class representing the set of all wires, and let `Resistor` be a model fragment class representing the set of all resistors. Let `wire-1` be an instance of `Wire`. To model `wire-1` as a resistor, we would make it an instance of `Resistor`, with the corresponding model fragment being `Resistor(wire-1)`. Note that, since `wire-1` is now an instance of both `Wire` and `Resistor`, the literals `Wire(wire-1)` and `Resistor(wire-1)` are both true.

### 2.5.3 Defining component and model fragment classes

Component and model fragment classes are defined using the `defmodel` macro. Figure 2.10 shows the definition of the `Resistor` model fragment class. We now discuss various parts of this definition.

```
(defmodel Resistor (Electrical-conductor)
  ((attributes
     (resistance
       :range Resistance-parameter
       :documentation "The resistor's resistance"))
   (equations
     (= (voltage-difference ?object)
        (* (resistance ?object)
           (current (electrical-terminal-one ?object)))))
   (assumption-class electrical-conductor-class)
   (approximations Ideal-conductor)
   (required-assumption-classes resistance-class)
   (possible-models Constant-resistance
                    Temperature-dependent-resistance)))
```

**Fig. 2.10.** The `Resistor` model fragment class.

**Generalization hierarchy.** Component and model fragment classes are organized into a generalization hierarchy, representing the "subset-of" relation between classes. The use of a generalization hierarchy, in conjunction with inheritance, is a very powerful tool for building knowledge bases because it facilitates reuse and knowledge base maintenance: (a) knowledge represented with a class can be used, not just by direct instances of the class, but also by instances of many different classes that are subclasses (specializations) of the class; and (b) since knowledge needs to be represented only with the most general class to which the knowledge is applicable, knowledge base maintenance is facilitated since most changes tend to be localized.

The second argument to the `defmodel` macro specifies the list of classes that are immediate generalizations of the defined class. Hence, the `Electri-`

cal-conductor class is an immediate generalization of the Resistor class. Logically this is equivalent to the following axiom:

Resistor(?object) ⇒ Electrical-conductor(?object)

From the point of view of model fragments used in a model, this means that any model that includes the model fragment Resistor(?object), also includes the model fragment Electrical-conductor(?object).

**Parameters and other attributes.** Parameters are represented as instances of a subclass of the Parameter class. For example, parameters representing voltages are instances of the Voltage-parameter class, while parameters representing resistances are instances of the Resistance-parameter class. Both Voltage-parameter and Resistance-parameter are subclasses of Parameter.

Recall that parameters represent numerical attributes of a device, in particular, of components. The relationship between a component and a parameter that represents a particular attribute of the component is represented by unary functions, called *parameter functions*. For example, voltage-difference is a parameter function that returns the instance of Voltage-parameter which represents the voltage difference across a component being modeled as an Electrical-conductor.

The attributes clause in the definitions of model fragment classes defines the parameter functions that can be used on components being modeled by that model fragment class. The definition of the parameter function includes a :range specification, which is the class of the parameter returned by the function. For example, the Resistor model fragment class defines the resistance parameter function, which returns an instance of Resistance-parameter representing the resistance of components being modeled as Resistors.

The attributes clause is also used to define functions that return other attributes of components. For example, two important attributes of an electrical conductor are the two *terminals* of the conductor. (Conceptually, terminals are parts of the component that allow the component to interact with other components by sharing parameters [de Kleer and Brown, 1984].) Figure 2.11 shows the definition of the Two-terminal-electrical-component model fragment class. The attributes clause in this definition defines the functions electrical-terminal-one and electrical-terminal-two, which return the two Electrical-terminals of the electrical component.

The attributes that a component inherits from a model fragment class are often related to attributes that it inherits from a component class. For example, in modeling a wire as an electrical conductor between its two ends, the two terminals of the electrical conductor correspond to the two ends of the wire. We enforce such relationships using a set of rules, which are similar to *articulation axioms* in [Hobbs, 1985].

**Equations.** The equations that a model fragment class inherits to its instances are defined using the equations clause. These equations are defined

```
(defmodel Two-terminal-electrical-component (Model-fragment)
  ((attributes
     (electrical-terminal-one
       :range Electrical-terminal
       :documentation "One end of the electrical component")
     (electrical-terminal-two
       :range Electrical-terminal
       :documentation "The other end of the electrical component"))
   (equations
     (= (current (electrical-terminal-one ?object))
        (current (electrical-terminal-two ?object)))
     (same-circuit (current (electrical-terminal-one ?object))
                   (current (electrical-terminal-two ?object)))
     (same-reference (voltage (electrical-terminal-one ?object))
                     (voltage (electrical-terminal-two ?object))))))
```

**Fig. 2.11.** The `Two-terminal-electrical-component` model fragment class.

using *equation schemas*. Equation schemas are exactly like equations, except that parameters are replaced by terms like `(resistance ?object)`. To instantiate such equation schemas for specific instances of the model fragment class, the variable "`?object`" is bound to the instance, and the terms are replaced by the parameter resulting from evaluating the term. For example, if `resistance(wire-1) = resistance-parameter-1`, then evaluating the term `(resistance ?object)` for the instance `wire-1` results in `resistance-parameter-1` Hence, if `wire-1` is modeled as a `Resistor`, then `wire-1` inherits the equation:

```
(= voltage-parameter-1 (* resistance-parameter-1
                          current-parameter-1))
```

**Assumption classes.** The `assumption-class` clause in a model fragment class specifies the assumption class of the model fragments which are instances of the model fragment class. More precisely, let `c` be a component and let `M1` and `M2` be model fragment classes. The model fragments `M1(c)` and `M2(c)` are in the same assumption class if and only if the `assumption-class` clause in both `M1` and `M2` specify the same assumption class. Let both `M1` and `M2` specify `A` in their `assumption-class` clause. We let the expression `A(c)` denote the assumption class of the model fragments `M1(c)` and `M2(c)`.[2] Furthermore, we will sometimes say "the assumption class of `M1` is `A`," meaning that for any component `c`, the model fragment `M1(c)` is in assumption class `A(c)`.

[2] This is a slight abuse of notation. While it is similar to our convention that `M1(c)` denotes the model fragment resulting from modeling the component `c` as an instance of the `M1` model fragment class, it certainly does not mean that `A` is a unary predicate so that `A(c)` is a literal meaning that `c` is an instance of `A`. To prevent any confusion, we will always refer to `A(c)` as "the assumption class `A(c)`."

For example, we can see that the `assumption-class` clause in `Resistor`'s definition specifies `electrical-conductor-class`. Suppose that the `assumption-class` clause in `Ideal-conductor`'s definition also specifies `electrical-conductor-class`. This means that for a component such as `wire-1`, the model fragments `Resistor(wire-1)` and `Ideal-conductor(wire-1)` are in the assumption class `electrical-conductor-class(wire-1)`.

**Approximations.** The `approximations` clause in a model fragment class specifies the model fragments that are approximations of instances of that class. More precisely, let `c` be a component and let `M1` and `M2` be model fragment classes. The model fragment `M2(c)` is an approximation of the model fragment `M1(c)` if and only if the `approximations` clause in `M1` specifies `M2`. For example, we can see that the `approximations` clause of the `Resistor` model fragment class specifies `Ideal-conductor`. This means that for a component such as `wire-1`, the model fragment `Ideal-conductor(wire-1)` is an approximation of the model fragment `Resistor(wire-1)`. To relate this to terminology introduced in a previous section, we have:

*approximation*(`Resistor(wire-1)`, `Ideal-conductor(wire-1)`)

Similarly, the `contradictory` clause in a model fragment class specifies the model fragments that contradict the instances of that class. Figure 2.10 does not show a `contradictory` clause because the contradiction between `Resistor` and `Ideal-conductor` can be inferred from the `approximations` clause.

**Required assumption classes.** The assumption classes required to complete the description of instances of a model fragment class are specified in the `required-assumption-classes` clause of the model fragment class. More precisely, suppose that `c` is a component and `M` is a model fragment class. Suppose that `M` specifies `A` as a `required-assumption-class`. This means that, to complete the description specified by the model fragment `M(c)`, we must include a model fragment from the assumption class `A(c)`. For example, the `required-assumption-classes` clause of the `Resistor` model fragment class specifies `resistance-class`. This means that for a component such as `wire-1`, the description specified by the model fragment `Resistor(wire-1)` must be completed by including a model fragment from the assumption class `resistance-class(wire-1)`, i.e., by including either `Constant-resistance(wire-1)` or `Temperature-dependent-resistance(wire-1)`. To relate it to terminology introduced earlier, we have:

*requires*(`Resistor(wire-1)`, `resistance-class(wire-1)`)

**Possible models.** Recall that the space of device models was defined by the set of model fragments that can be used to describe the device. This set of model fragments was the union of the model fragments that can be used to describe the components of the device. This means that we need a representation of the set of model fragments that can be used to model a

component. A straightforward way to represent this set of model fragments is to associate with each component class the set of model fragment classes which can be used to model instances of that component class.

For example, some of the model fragment classes we could associate with the `Wire` component class would include `Ideal-conductor`, `Constant-resistance`, and `Temperature-dependent-resistance`. This would represent the fact that any instance of `Wire` can be modeled as an instance of the associated model fragment classes.

While the above approach is, in principle, correct, a much better approach is to use a *possible models hierarchy*. The basic intuition underlying this approach is the observation that model fragment classes like `Ideal-conductor`, `Constant-resistance`, and `Temperature-dependent-resistance` are all models of electrical conduction. Hence, it would be much better if we only had to represent the fact that an instance of `Wire` can be modeled as an `Electrical-conductor`, with additional electrical conductor models being associated with `Electrical-conductor`. Similarly, rather than associating all the electrical conductor models with `Electrical-conductor`, we would associate only `Ideal-conductor` and `Resistor` with it, and associate `Constant-resistance` and `Temperature-dependent-resistance` with `Resistor`. In essence, we build a hierarchy of possible models.

The advantage of the possible models hierarchy are very similar to the advantages of a generalization hierarchy. First, it leads to compact representations. For example, one only needs to specify that instances of `Wire` can be modeled as `Electrical-conductor`, with additional ways of modeling instances of `Wire` being inferred from the hierarchy. Second, knowledge base maintenance is simplified. For example, if we want to add an additional model fragment class describing yet another electrical conductor model, e.g., the dependence of the resistance on length, then this change need only be made to the possible models hierarchy below `Resistor`; definitions of component classes, like `Wire`, need to be modified.

The possible models of a model fragment class are defined in the `possible-models` clause of the `defmodel` macro. For example, we can see that the model fragment classes that can be used to model instances of `Resistor` include `Constant-resistance` and `Temperature-dependent-resistance`.

Note that the generalization hierarchy and the possible models hierarchy often overlap. For example, `Resistor` is both a specialization and a possible model of `Electrical-conductor`. However, the two hierarchies are not the same. For example, the `Thermal-thermistor` model fragment class, which models the dependence of a thermistor's resistance on its temperature, is a specialization of the `Thermal-object` model fragment class. However, it is evident that not all components being modeled as `Thermal-objects` can be modeled as `Thermal-thermistors`, only thermistors can be modeled as `Thermal-thermistors`. Hence, `Thermal-thermistor` is a specialization of `Thermal-object`, but not a possible model of it.

### 2.5.4 Difference between component and model fragment classes

Thus far we have been talking about component classes and model fragment classes as separate types of classes. But what exactly is the difference? The answer is that, fundamentally, there is no difference! Both model fragment classes and component classes are partial descriptions. For example, while the `Resistor` model fragment class is a partial description of electrical conduction, the `Wire` model fragment class is a partial description of what it means for an object to be a wire.

The only difference between component classes and model fragment classes is their position in the possible models hierarchy. Component classes are the classes that are at the top of the possible models hierarchy, i.e., component classes are not models of any other class. Therefore, component classes can be viewed as primitive descriptions. The decision to model an object as an instance of a component class is, therefore, the responsibility of the human user providing the input (the structural description), and is outside the scope of the model selection program.

An interesting consequence of the above observation is that a human user may choose to define the structure of the device in terms of model fragment classes, rather than just component classes. For example, the user may use an instance of the `Electrical-conductor` model fragment class as part of a device. The ability to specify structural descriptions using model fragment classes provides the user with a valuable abstraction tool. This is useful, for example, during design, where the designer may know that there is an electrical conductor at some place in the device, without knowing what specific component implements this electrical conductor.

## 2.6 Summary

In this chapter we defined a model to be a set of model fragments, where a model fragment is a set of algebraic, qualitative, and/or ordinary differential equations, describing some phenomena at some level of detail. Viewing a model as a set of model fragments is useful because model fragments are easier to construct and more reusable than complete models. In addition, the set of applicable model fragments is an implicit description of an exponentially large space of possible models. The set of applicable model fragments is defined by the device structure and a component library. The component library specifies the model fragments that can be used to model each component of the device.

We introduced two important relations between model fragments: *contradictory* and *approximation*. Model fragments related by the *contradictory* relation make contradictory assumptions about the domain. In addition to being mutually contradictory, model fragments can differ in the relative accuracy with which they model phenomena. The relative accuracy of model

fragments is represented using the *approximation* relation. In addition to these two relations, we also introduced assumption classes, which are sets of mutually *contradictory* model fragments that describe the same phenomena.

Finally, we concluded this chapter with a discussion of the actual representational mechanisms we use to implement the above ideas. In particular, we introduced a class level representation of components and model fragments and showed how these classes are organized. In this representation, a model fragment is the result of making a component an instance of the corresponding model fragment class. Component and model fragment classes are organized into two hierarchies: a generalization hierarchy and a possible models hierarchy. These hierarchies lead to more compact representations, and facilitate knowledge base maintenance.

# 3. Adequate models

In this chapter we discuss the adequacy of device models. The adequacy of a device model is fundamentally determined by the task that needs to be solved. We will define the adequacy of a model with respect to the task of generating causal explanations for a phenomenon of interest. We also show that additional constraints on model adequacy can stem from the structure and the behavior of the device. Finally, we define model simplicity based on the intuition that modeling fewer phenomena, more approximately, leads to simpler models. An adequate model is required to be as simple as possible.

## 3.1 Tasks and models

The adequacy of a model is closely tied to the task for which the model is to be used. Simulations carried out during the final stages of the detailed design of a device require the use of high fidelity models that incorporate accurate, quantitative descriptions of all significant phenomena. For example, a high fidelity model of the temperature gauge shown in Figure 1.1 would include a quantitative, nonlinear equation describing the dependence of the thermistor's resistance on its temperature.

On the other hand, models that support analysis during the initial, conceptual design of a device can be much coarser. For example, during the conceptual design of the temperature gauge, it is sufficient to use a qualitative model [Bobrow, 1984] of the thermistor, which states that the thermistor's resistance is inversely proportional to its temperature.

Similarly, Hamscher [Hamscher, 1988, page 11] argues that:

> For complex devices the model of the target device should be constructed with the goal of troubleshooting explicitly in mind.

He then presents a set of representation and modeling principles that assist the efficient diagnosis of complex digital circuits [Hamscher, 1988; Hamscher, 1991]. These principles are an informal specification of the adequacy of a model with respect to the task of diagnosis.

In this thesis, we define the adequacy of a model *with respect to the task of generating causal explanations for phenomena of interest*. In the next section we discuss the importance of this task, both as a vehicle for communication,

as well as an important subtask for other tasks such as analysis, diagnosis, and design.

## 3.2 Causal explanations

Causation and causal reasoning are ubiquitous in human reasoning. People are always asking *why* something happened, expecting some sort of a causal explanation in reply. However, while the notion of causation seems intuitively clear to everyone, providing a good definition for it has not been easy. Philosophers have argued about the true nature of causation for a long time (e.g., see [Mackie, 1974]). In this thesis we choose not to get mired in this debate. Instead, we take the view, common in Artificial Intelligence [Bobrow, 1984; Iwasaki and Simon, 1986b; Patil *et al.*, 1981; Pople, 1982; Rieger and Grinberg, 1977; Shoham, 1985; Wallis and Shortliffe, 1982; Weiss *et al.*, 1978], that causal explanations are explanations of phenomena based on a set of underlying mechanisms, that are *assumed* to provide a description of how (the relevant aspect of) the world really works, i.e., these mechanisms are assumed to be causal mechanisms. (See [Nayak, 1989] for an overview of this literature.)

### 3.2.1 Importance of causal explanations

Causal explanations play an important role in automated reasoning systems as a vehicle for the system to communicate with its human user. Such explanations can be used for instructional purposes, as in various Intelligent Computer Aided Instruction systems [Brown *et al.*, 1982; Forbus and Stevens, 1981; Weld, 1983], or as a method for explaining the system's line of reasoning to a human user [Patil *et al.*, 1981; Weiss *et al.*, 1978; Wallis and Shortliffe, 1982].

In addition to their role in communication, causal explanations play a central role in focusing other forms of reasoning [Weld and de Kleer, 1990]. Causal explanations are used in diagnosis to focus the reasoning only on those elements that could have caused a particular symptom [Davis, 1984]. Causal explanations focus design and redesign by focusing the reasoning on just those mechanisms that can produce the desired behavior [Williams, 1989; Williams, 1990]. Causal explanations can also guide quantitative analysis by providing an overall structure for solving the problem at hand [de Kleer, 1977].

### 3.2.2 Types of causal explanations

Causal explanations are generated by stringing together causal relations of the form "x causes y." Different types of causal explanations are generated

depending on the particular vocabulary used for modeling these causal relations, i.e., the types of "x" and "y" and the meaning of the "causes" relation. In many medical diagnosis systems (e.g., CASNET [Weiss *et al.*, 1978], CADUCEUS/INTERNIST [Pople, 1982], ABEL [Patil *et al.*, 1981]) the causal relation relates different possible *states* of a patient, while the causal relation itself represents the likelihood of observing the effect given the cause. A similar approach is used in Bayesian networks, where the causal relation represents conditional probabilities between random variables [Pearl, 1988]. Reiger and Grinberg, in their work on understanding physical mechanisms [Rieger and Grinberg, 1977], identify 10 different types of causal relations that relate *events* like actions, tendencies, states, and statechanges. Shoham's logical account of causation [Shoham, 1988] relates *temporal propositions*, with the causal relation being an INUS condition [Mackie, 1974], i.e., the cause is an Insufficient but Necessary condition of an Unnecessary but Sufficient condition for the effect.

In this thesis we adopt the representation of the causal relation widely used in the literature on qualitative reasoning about physical systems [Weld and de Kleer, 1990]. In this representation, the causal relation relates *parameters* used to model the physical system, and the causal relation itself represents a dependence of the value of the "effect parameter" on the "cause parameter." We discuss this in detail in the next section.

## 3.3 Causal ordering

The causal relation between the parameters, introduced above, is a transitive relation that induces an ordering on the parameters called a *causal ordering*. The dependency of the "effect parameter" on the "cause parameter" in such a causal ordering takes one of two forms: *functional dependency* and *integration*.

The functional dependency of a parameter $p_1$ on a parameter $p_2$ corresponds to a causal mechanism that "instantaneously" determines the value of $p_1$ as a function of the value of $p_2$ (and, possibly, some other parameters). We have quoted the word "instantaneously" to emphasize that what counts as "instantaneously" is a modeling decision related to the time scale of interest [Iwasaki, 1988; Kuipers, 1987]. For example, at a time scale of minutes, a thermistor's resistance is functionally dependent on its temperature; a change in the temperature can be viewed as instantly causing a change in the resistance. However, at a much smaller time scale one can actually observe a delay in the change in resistance due to the change in temperature. Causal relations as functional dependencies have been studied in [de Kleer and Brown, 1984; Williams, 1984; Iwasaki and Simon, 1986b] and in [Forbus, 1984], where they are called *indirect influences*.

The other type of causal relations between parameters is the integration relation between a parameter and its derivative. In contrast to functional

dependencies that act instantaneously, the integration relation acts over a period of time. For example, the total amount of charge stored in a capacitor depends on the net flow of current into the capacitor over a period of time; the amount of stored charge is calculated by integrating the current flow over that period of time. Causal relations as integration have been studied in [Iwasaki, 1988] and in [Forbus, 1984], where they are called *direct influences*.

### 3.3.1 Loops in the causal ordering

As mentioned above, the causal relation between parameters is transitive. However, we do not insist that the causal relation be anti-symmetric, i.e., a parameter $p_1$ can simultaneously causally depend on, and can causally determine, a parameter $p_2$. Such loops in the causal ordering are manifestations of feedback in the behavior of the physical system. The proper handling of such feedback, and the resulting loops in the causal ordering, is the focus of much debate and ongoing research [Bobrow, 1984; Iwasaki and Simon, 1986b; de Kleer and Brown, 1986; Iwasaki and Simon, 1986a; Rose and Kramer, 1991]. In this thesis we adopt the (somewhat neutral) viewpoint, advanced in [Iwasaki and Simon, 1986b], of merely viewing such feedback as a set of interdependent parameters.

### 3.3.2 Equations

The causal ordering of a set of parameters used to model a physical system is derived from a set of algebraic, qualitative, and/or differential equations describing the physical system. Equations, as such, can be viewed as acausal representations of domain mechanisms. For example, the equation $V = iR$ (Ohm's law) is an acausal representation of a mechanism for electrical conduction. It merely states that the voltage across an electrical conductor, $V$, is proportional to the current through the conductor, $i$, with the resistance of the conductor, $R$, being the proportionality constant. However, it makes no causal claims like "the voltage depends on the current."

To have a causal import, equations must be *causally oriented*. A causally oriented equation represents the fact that one of the parameters of the equation is *directly causally dependent* on the other parameters of the equation. The dependent parameter is said to be *causally determined* by the equation. For example, the acausal equation $V = iR$ can be causally oriented so that it causally determines $V$, making $V$ causally dependent on $i$ and $R$.

The causal orientation of an equation can be fixed *a priori* [Forbus, 1984], or it can be inferred from the equations comprising a model of the system [de Kleer and Brown, 1984; Williams, 1984; Iwasaki and Simon, 1986b; Iwasaki, 1988]. Fixing the causal orientation of each equation *a priori* is overly restrictive, since different causal orientations are often possible. However, not all causal orientations fit our intuitions about causality. For example,

the equation $V = iR$ can be causally oriented in one of two ways: either $V$ can be causally dependent on $i$ and $R$, or $i$ can be causally dependent on $V$ and $R$. However, the third possibility, $R$ being causally dependent on $V$ and $i$, makes no sense because, in an ordinary electrical conductor, there is no way that changing $V$ and/or $i$ can cause a change in $R$.

The set of allowed causal orientations of an equation, $e$, can be represented by the set, $P_c(e)$, of parameters that can be causally determined by $e$. As a typographical aid, parameters that can be causally determined by an equation will be typeset in boldface, e.g., $\boldsymbol{V} = \boldsymbol{i}R$ says that this equation can causally determine $V$ and $i$ but not $R$. We extend the function $P_c$ to a set $E$ of equations in the natural way:

$$P_c(E) = \bigcup_{e \in E} P_c(e) \tag{3.1}$$

Similarly, we extend $P_c$ to a model $M$ as follows (recall that a model fragment $m \in M$ is just a set of equations):

$$P_c(M) = \bigcup_{m \in M} P_c(m) \tag{3.2}$$

In addition, let $P(e)$ be the set of all parameters in equation $e$. Extend $P$ to a set $E$ of parameters, and to a model $M$, as follows:

$$P(E) = \bigcup_{e \in E} P(e) \tag{3.3}$$

$$P(M) = \bigcup_{m \in M} P(m) \tag{3.4}$$

### 3.3.3 Computing the causal ordering

As mentioned earlier, the causal ordering of a set of parameters is derived from the set of equations representing a model of the system under consideration. Iwasaki and Simon provide an algorithm for computing the causal ordering [Iwasaki and Simon, 1986b; Iwasaki, 1988]. However, that algorithm is a worst-case exponential time algorithm. In this section we describe an efficient algorithm for computing the causal ordering based on the work of Serrano and Gossard [Serrano and Gossard, 1987].

Serrano and Gossard make the key observation that, given a set of equations, the causal ordering of the parameters can be generated by (a) causally orienting each equation such that each parameter is causally determined by exactly one equation; and (b) taking the transitive closure of the direct causal dependency links entailed by the causal orientations.[1]

[1] Serrano and Gossard do not actually talk about causal ordering or causal orientations. They are interested in efficiently evaluating a set of constraints. However, the parameter dependencies that they generate are identical to the causal ordering, and their algorithm can be viewed as causally orienting each equation. Hence, we attribute the above observation to them.

**Causal mappings.** We formalize Serrano and Gossard's observation by first defining a causal mapping:

**Definition 3.1 (Causal mapping).** *Let $E$ be a set of equations. A function $F : E \rightarrow P(E)$ is said to be a causal mapping if and only if (a) $F$ is 1-1; and (b) for each $e \in E$, $F(e) \in P_c(e)$. $F$ is an onto causal mapping if for each parameter $p \in P(E)$, there is an equation $e \in E$, such that $F(e) = p$.*

Hence, a causal mapping causally orients each equation such that each parameter is causally determined by at most one equation, while an onto causal mapping causally determines every parameter. A causal mapping is said to be *partial* if it is not defined on every equation.[2]

Note that the co-domain of $F$ in the above definition is $P(E)$ and not $P_c(E)$, even though condition (b) guarantees that the range of $F$ is a subset of $P_c(E)$. We have chosen $P(E)$ as the co-domain of $F$ to ensure that when $F$ is onto then each parameter in $P(E)$ is causally determined by an equation in $E$.

**Properties of causal mappings.** Let $E$ be a set of equations and let $F : E \rightarrow P(E)$ be a (possibly partial) causal mapping. The direct causal dependencies entailed by $F$ is denoted by $C_F$, and is defined as follows:

$$C_F = \{(p_1, p_2) : (\exists e \in E)\ F(e) = p_2 \ \wedge\ p_1 \in P(e)\} \tag{3.5}$$

In other words, $(p_1, p_2) \in C_F$ if and only if $p_2$ directly causally depends on $p_1$ in the causal orientations defined by $F$. Denote the transitive closure of $C_F$ by $tc(C_F)$. The following lemma states that the transitive closure of different onto causal mappings of $E$ are identical. (We will soon discuss conditions under which onto causal mappings exist.)

**Lemma 3.3.1.** *Let $E$ be a set of independent equations, and let $F_1 : E \rightarrow P(E)$ and $F_2 : E \rightarrow P(E)$ be onto causal mappings. Then $tc(C_{F_1}) = tc(C_{F_2})$.*

*Proof.* To show that $tc(C_{F_1}) = tc(C_{F_2})$ we need to show that $tc(C_{F_1}) \subseteq tc(C_{F_2})$ and $tc(C_{F_2}) \subseteq tc(C_{F_1})$. We prove the first containment, with the second containment following by a symmetric argument. To show that $tc(C_{F_1}) \subseteq tc(C_{F_2})$, it suffices to show that $C_{F_1} \subseteq tc(C_{F_2})$, since $tc(tc(C_{F_2})) = tc(C_{F_2})$.

Let $(q, p) \in C_{F_1}$, and let $e \in E$ such that $F_1(e) = p$, and hence $q \in P(e)$. We show that $(q, p) \in tc(C_{F_2})$. There are two cases:

1. If $F_2(e) = p$, then $(q, p) \in C_{F_2}$, and hence $(q, p) \in tc(C_{F_2})$.
2. If $F_2(e) \neq p$, construct the sequence $p_0, p_1, \ldots, p_m$ such that (a) $p_0 = p$; (b) $p_i = F_2(F_1^{-1}(p_{i-1}))$, for $1 \leq i \leq m$; (c) $p_m$ is the first repetition in the sequence, i.e., $p_i \neq p_j, 0 \leq i, j \leq (m-1), i \neq j$, and $p_m = p_i$, for some $i, 0 \leq i \leq (m-1)$. Such a sequence must exist because $F_1$ and

[2] Hence, causal mappings as defined in Definition 3.1 are more precisely named *total* causal mappings. However, for simplicity, we shall assume that all causal mappings are total, unless we explicitly mention them to be partial.

$F_2$ are onto causal mappings, and because there are a finite number of parameters. In addition, observe that $m \geq 2$, since if $m = 1$, it follows that $p_0 = F_2(F_1^{-1}(p_0))$, which leads to a contradiction as follows:

$$\begin{aligned} p &= p_0 \\ &= F_2(F_1^{-1}(p_0)) \\ &= F_2(F_1^{-1}(p)) \\ &= F_2(e) \end{aligned}$$

which contradicts the assumption that $F_2(e) \neq p$.
We now show that $p_m = p_0$. Suppose not, so that $p_m = p_i$ for some $i$, $1 \leq i \leq (m-1)$. Hence, it follows that $p_{m-1} = F_1(F_2^{-1}(p_m)) = F_1(F_2^{-1}(p_i)) = p_{i-1}$, which contradicts condition (c) above. Hence, $p_m = p_0$.
Next, let $e_i = F_1^{-1}(p_{i-1})$, for $1 \leq i \leq m$. Hence, $p_{i-1} \in P(e_i)$ and $p_i = F_2(e_i)$. Hence, it follows that $(p_{i-1}, p_i) \in C_{F_2}$. Hence, by transitivity, it follows that $(p_1, p_m) \in tc(C_{F_2})$, and since $p_m = p_0 = p$, it follows that $(p_1, p) \in tc(C_{F_2})$.
Now there are two cases: (a) if $p_1 = q$, then it follows that $(q, p) \in tc(C_{F_2})$; or (b) if $p_1 \neq q$, then since $p_1 = F_2(e)$ and $q \in P(e)$, it follows that $(q, p_1) \in C_{F_2}$, and hence by transitivity $(q, p) \in tc(C_{F_2})$. In either case, $(q, p) \in tc(C_{F_2})$, and we are done.

Intuitively, the above proof shows that if $F_1$ and $F_2$ differ on the parameter to which an equation $e$ is mapped, then the parameters $F_1(e)$ and $F_2(e)$ are causally dependent on each other. For example, consider the set of equations, and two different onto causal mappings $F_1$ and $F_2$, shown in Figure 3.1.

| Equations | $F_1$ | $F_2$ |
|---|---|---|
| $exogenous(\boldsymbol{x})$ | $x$ | $x$ |
| $\boldsymbol{x} = \boldsymbol{y}$ | $y$ | $y$ |
| $\boldsymbol{u} + \boldsymbol{v} = \boldsymbol{y}$ | $u$ | $v$ |
| $\boldsymbol{u} - \boldsymbol{v} = 0$ | $v$ | $u$ |

**Fig. 3.1.** A set of equations with two onto causal mappings

Note that $F_1$ and $F_2$ agree on the parameters assigned to the first two equations, while they differ on the parameters assigned to the last two equations. However, under $F_1$, $u$ causally depends on $v$ from the mapping of the third equation while $v$ causally depends on $u$ from the mapping of the fourth equation, i.e., $u$ and $v$ are interdependent. Similarly, under $F_2$, $u$ causally depends on $v$ from the mapping of the fourth equation while $v$ causally depends on $u$ from the mapping of the third equation; once again, $u$ and $v$ are interdependent.

**Causal ordering definition.** Using causal mappings and the above lemma, we now define the causal ordering of a set of parameters generated from a set of equations. Before we do this, we first introduce the *integration completion* of a set of equations. Recall that the integration relation between a parameter and its derivative constitutes a causal dependency from the derivative to the parameter. We represent this relation with the *int* equation: $int(\boldsymbol{p}_1, p_2)$ says that $p_2$ is the derivative of $p_1$. Note that $int(\boldsymbol{p}_1, p_2)$ can be causally oriented in only one way, to causally determine $p_1$ by integrating the value of $p_2$ over time. Given a set $E$ of equations the integration completion of $E$ makes explicit all such integration links among the parameters of $E$:

**Definition 3.2 (Integration completion).** *Let $E$ be a set of equations. The integration completion of $E$, denoted $ic(E)$, is defined as follows:*

$$ic(E) = E \cup \{int(\boldsymbol{q}, dq/dt) : dq/dt \in P(E)\}$$

i.e., whenever $P(E)$ contains a derivative, the integration completion of $E$ contains an *int* equation expressing the integration relation. Note that if $E$ contains no differential equations, then $E = ic(E)$.

We now define the causal ordering generated from a set of equations as the transitive closure of direct causal dependencies generated by *any* onto causal mapping of the integration completion of the set of equations:

**Definition 3.3 (Causal order).** *Let $E$ be a set of independent equations, and let $F : ic(E) \rightarrow P(E)$ be an onto causal mapping. The causal order of the parameters of $E$, denoted $C(E)$, is the transitive closure of $C_F$:*

$$C(E) = tc(C_F)$$

The causal ordering is well defined because Lemma 3.3.1 assures us that the transitive closures of all onto causal mappings of a set of equations are identical. This allows us to define the causal ordering of a set of equations as the transitive closure of *any* onto causal mapping. The use of $ic(E)$, instead of $E$, in the above definition ensures that causal dependencies due to integration links are included in the causal ordering.

Next, we investigate conditions under which the causal ordering exists, i.e., conditions under which an onto causal mapping exists.

**Existence of onto causal mappings.** We start by defining what it means for a set of equations to be *complete*, *overconstrained*, and *incomplete*. Informally, a set of equations is (a) complete if it has as many equations as parameters, and no subset of equations has fewer parameters than equations; (b) overconstrained if some subset of equations has more equations than parameters; and (c) incomplete if some subset of equations, that has no parameters in common with its complement, has more parameters than equations. More precisely, we have the following definitions:

**Definition 3.4.** *Let $E$ be a set of independent equations.*[3]

- *$E$ is said to be complete if and only if (a) $|ic(E)| = |P_c(ic(E))| = |P(E)|$; and (b) for every $S \subseteq ic(E), |S| \leq |P_c(S)|$.*
- *$E$ is said to be overconstrained if and only if there exists $S \subseteq ic(E)$ such that $|S| > |P_c(S)|$.*
- *$E$ is said to be incomplete if and only if there exists $S \subseteq ic(E)$ such that either (a) $P_c(S) \cap P_c(ic(E) \setminus S) = \emptyset$ and $|S| < |P_c(S)|$; or (b) $P(S) \cap P(ic(E) \setminus S) = \emptyset$ and $|S| < |P(S)|$.*

We now show that an onto causal mapping exists if and only if the set of equations is complete. This means that when a set of equations is complete, all the parameters in the equations can be causally determined.

**Lemma 3.3.2.** *Let $E$ be a set of independent equations. Then there exists an onto causal mapping $F : ic(E) \rightarrow P(E)$ if and only if $E$ is complete.*

*Proof.* To prove this lemma, we start by defining a bipartite graph representing the set of equations $E$.

**Definition 3.5.** *Let $E$ be a set of independent equations. Let $G = (X, Y, R)$ be a bipartite graph such that $X \cup Y$ is the set of nodes, $R$ is the set of edges, and each edge connects a node in $X$ to a node in $Y$. $G$ is said to represent $E$ if and only if*[4]

1. *$X = ic(E)$, i.e., there is a node in $X$ for each equation, including the integration equations;*
2. *$Y = P(E)$, i.e., there is a node in $Y$ for each parameter; and*
3. *$(x, y) \in R$ if and only if $x \in X (= ic(E))$, $y \in Y (= P(E))$, and $y \in P_c(x)$, i.e., an equation is connected to a parameter if and only if the equation can causally determine the parameter.*

For example, the bipartite graph representing the equations shown in Figure 3.1 is shown in Figure 3.2.

A *matching* in a bipartite graph is a set of edges such that no two edges in the matching share a common node. A matching is said to be *complete* if and only if each node in the graph is covered by an edge in the matching, i.e., each node has an edge in the matching incident upon it. For example, a complete matching in the bipartite graph of Figure 3.2 consists of the following edges:

$$\{(exogenous(\boldsymbol{x}), \boldsymbol{x}), (\boldsymbol{x} = \boldsymbol{y}, \boldsymbol{y}), (\boldsymbol{u} + \boldsymbol{v} = \boldsymbol{y}, \boldsymbol{u}), (\boldsymbol{u} - \boldsymbol{v} = 0, \boldsymbol{v})\}$$

From the above definitions, it follows that an onto causal mapping $F : ic(E) \rightarrow P(E)$ corresponds to a complete matching in the bipartite graph representing $E$, and vice versa. In particular, the complete matching

[3] "$|\cdot|$" returns the cardinality of a set. "\" is the set difference operator.

[4] This representation of the set of equations is due to Serrano and Gossard [Serrano and Gossard, 1987].

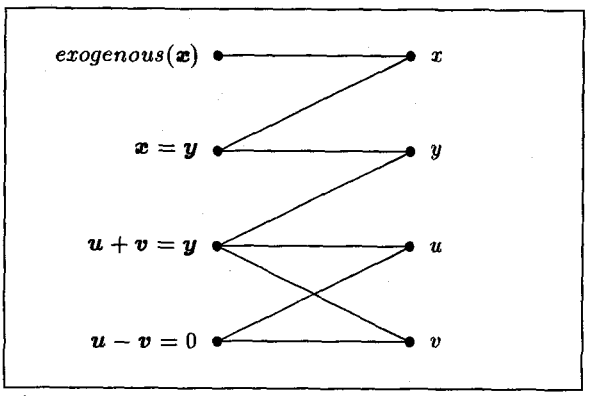



**Fig. 3.2.** Graph representing a set of equations

corresponding to an onto causal mapping $F : ic(E) \rightarrow P(E)$ is the following set of edges:

$$\{(e, F(e)) : e \in ic(E)\}$$

Hence, it follows that an onto causal mapping $F : ic(E) \rightarrow P(E)$ exists if and only if a complete matching exists for the bipartite graph representing $E$. However, Hall's theorem [Even, 1979, pages 137–138] tells us that a bipartite graph $G = (X, Y, R)$ contains a complete matching if and only if (a) $|X| = |Y|$; and (b) for every $A \subseteq X, |A| \leq |R(A)|$, where $R(A)$ denotes the set of nodes connected to the nodes in $A$ by edges in $R$. However, from Definition 3.5, condition (a) is equivalent to saying $|ic(E)| = |P(E)|$, and condition (b) is equivalent to saying that for every $S \subseteq ic(E), |S| \leq |P_c(S)|$. But, from Definition 3.4, this is equivalent to saying that $E$ is a complete set of equations. Hence, it follows that there exists an onto causal mapping $F : ic(E) \rightarrow P(E)$ if and only if $E$ is complete.

**Causal ordering algorithm.** The proof of the above lemma leads directly to the efficient causal ordering algorithm based on Serrano and Gossard's work [Serrano and Gossard, 1987]. This algorithm is shown in Figure 3.3.

In this algorithm, step 1 uses Definition 3.5 to construct the bipartite graph representing $E$. Step 2 constructs a maximum matching in this graph. A maximum matching is a matching with maximum cardinality. If $n$ is the number of nodes, and $e$ the number of edges, in a bipartite graph, a maximum matching in the graph can be constructed in $O(\sqrt{n}e)$ using algorithms for finding maximum flow in networks, e.g., see [Even, 1979, pages 135–138]. Appendix D gives a brief overview of this algorithm. Step 3 checks whether or not this matching is complete. Note that if a complete matching exists then it is a maximum matching. Conversely, if a complete matching exists then any maximum matching is a complete matching. If the matching is complete, it constructs the corresponding causal mapping and returns its transitive closure. If the matching is not complete, then no complete matching exists,

```
function find-causal-order(E)
  1. Using Definition 3.5 construct G, the
     bipartite graph representing E;
  2. Construct a maximum matching for G;
  3. if the above matching is complete then
       a. Let F be the corresponding onto causal mapping;
       b. return the transitive closure of the direct causal
                  dependencies entailed by F
     else
       c. return nil /* No onto causal mapping exists */
     endif
end
```

**Fig. 3.3.** Causal ordering algorithm

and the set of equations is not complete. Hence, the causal ordering is not well defined, and the above algorithm returns nil.

We now illustrate the above algorithm with an example. Figure 3.4 shows a set of equations describing the temperature gauge shown in Figure 1.1. This set of equations is exactly the same as the ones shown in Figure 1.3, except that here we have included knowledge of allowed causal orientations of each equation. Figure 3.5 shows the bipartite graph representing this set of equations. This figure also shows a maximum matching consisting of the thick edges with arrow heads at each end. One can see that this set of edges forms a complete matching. Figure 3.6 shows a graphical representation of the direct causal dependencies generated from the causal mapping corresponding to the above complete matching. Note, in particular, the cycle of dependencies between $i_t, i_w, V_w$, and $V_t$.

**Miscellaneous observations.** In practice, we modify step 3b of function *find-causal-order* to return $C_F$, the graph of direct causal dependencies generated by the causal mapping $F$ generated in step 3a, rather than its transitive closure, $C(E)$. For example, the function would then return graphs like the one shown in Figure 3.6. This has two important advantages:

1. Paths in this graph provide a causal explanation for *how* one parameter causally depends on another. For example, while the transitive closure of the graph shown in Figure 3.6 can tell us that $\theta_p$ causally depends on $T_t$, it is unable to say that this causal dependence is not a direct causal dependence, i.e., is not due to a single causal mechanism. On the other hand, the graph in Figure 3.6 can be used to give a detailed explanation for how $\theta_p$ depends on $T_t$ by identifying the different causal mechanisms that mediate this dependence.
2. We can use this graph to easily identify the minimal sets of causally interdependent parameters, without incurring the cost of generating the

| | |
|---|---|
| `Linkage(bms-1,ptr-1):` | $\theta_p = k_1 x_b$ |
| `Thermal-bms(bms-1):` | $x_b = k_2 T_b$ |
| `Heat-flow(bms-1,atm-1):` | $f_{ba} = k_3(T_b - T_a)$ |
| `Heat-flow(wire-1,bms-1):` | $f_{wb} = k_4(T_w - T_b)$ |
| `Constant-temperature(atm-1):` | $exogenous(T_a)$ |
| `Thermal-equilibrium(bms-1):` | $f_{ba} = f_{wb}$ |
| `Thermal-equilibrium(wire-1):` | $f_{wb} = f_w$ |
| `Resistor(wire-1):` | $V_w = i_w R_w$ |
| `Constant-resistance(wire-1):` | $exogenous(R_w)$ |
| `Thermal-resistance(wire-1):` | $f_w = V_w i_w$ |
| `Electrical-thermistor(thermistor-1):` | $V_t = i_t R_t;\ \ R_t = k_5 e^{k_6/T_t}$ |
| `Constant-voltage-source(battery-1):` | $exogenous(V_v)$ |
| `Kirchhoff's laws:` | $V_v = V_w + V_t;\ \ i_v = i_t;\ \ i_t = i_w$ |
| `Input:` | $exogenous(T_t)$ |

| | |
|---|---|
| $\theta_p$: Pointer angle | $x_b$: Bms deflection |
| $R_w$: Wire resistance | $R_t$: Thermistor resistance |
| $i_t$: Thermistor current | $V_t$: Thermistor voltage |
| $i_w$: Wire current | $V_w$: Wire voltage |
| $i_v$: Battery current | $V_v$: Battery voltage |
| $T_b$: Bms temperature | $T_a$: Atm temperature |
| $T_w$: Wire temperature | $T_t$: Thermistor temperature |
| $f_{ba}$: Heat flow (bms to atm) | $f_{wb}$: Heat flow (wire to bms) |
| $f_w$: Heat generated in wire | $k_j$: Exogenous constants |

**Fig. 3.4.** A possible model of the temperature gauge

transitive closure.[5] The minimal sets of causally interdependent parameters are precisely the *strongly connected components* of the graph. A strongly connected component of a directed graph is a maximal set of nodes in the graph such that there exists a directed path from each node in the set to every other node in the set. An efficient algorithm for generating the strongly connected components of a directed graph is found in [Even, 1979, pages 64–66]. For example, the set $\{i_t, i_w, V_w, V_t\}$ form a strongly connected component of the graph in Figure 3.6, and hence these parameters are causally interdependent.

If step 2 results in a maximum matching that is not complete, then the set of equations is either overconstrained, or incomplete, or both. Following [Serrano and Gossard, 1987], we state the following without proof:

1. If the maximum matching found in step 2 is such that a node corresponding to one of the equations in $ic(E)$ is not covered by an edge in the matching, then the set of equations is overconstrained.

[5] One can easily show that these minimal sets of causally interdependent parameters are the *minimal complete subsets* identified by the causal ordering algorithm in [Iwasaki and Simon, 1986b].

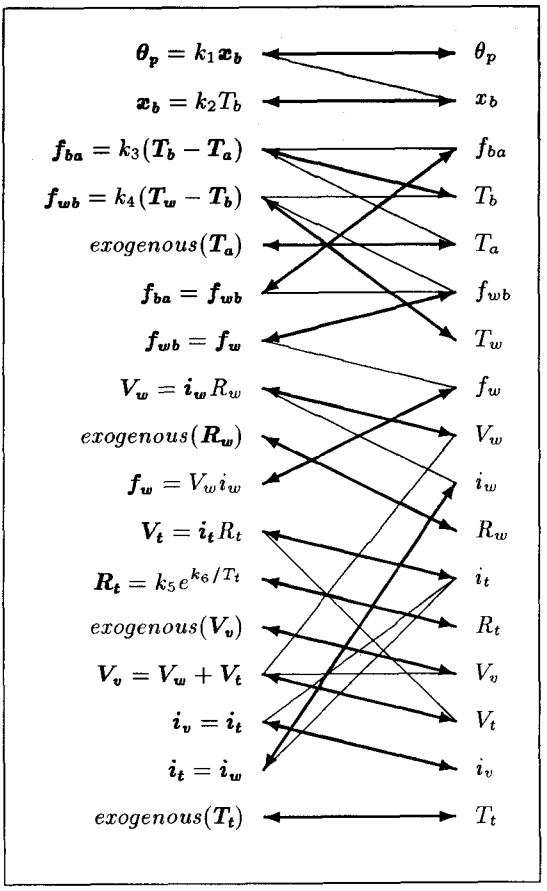



**Fig. 3.5.** Bipartite graph representing the equations in Figure 3.4. The set of thick edges with arrow heads at each end form a complete matching.

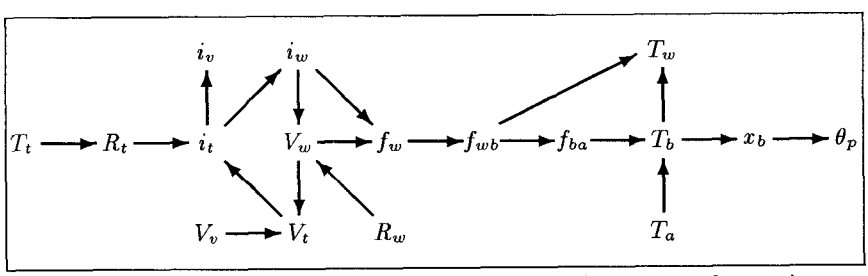



**Fig. 3.6.** The direct causal dependencies generated by the causal mapping corresponding to the complete matching shown in Figure 3.5.

2. If the maximum matching found in step 2 is such that a node corresponding to one of the parameters in $P(E)$ is not covered by an edge in the matching, then the set of equations is incomplete.

The proofs of the above statements are similar to the proof of Lemma 3.3.2.

This concludes our discussion of causal explanations and how they are generated from a model, i.e., from a set of equations. We now proceed to define the criteria that we use for model adequacy. The next section introduces the consistency and completeness of a model; adequate models are required to be consistent and complete. Section 3.5 introduces our representation for the phenomenon of interest; an adequate model must be able to provide a causal explanation for the phenomenon of interest. Sections 3.6 and 3.7 introduce constraints stemming from the structural and behavioral contexts of a physical system, that must be satisfied by an adequate model. Finally, Section 3.8 will introduce a simplicity ordering on the set of models, with an adequate model being a simplest model that satisfies all the above criteria.

## 3.4 Consistency and completeness of models

In this section we define the two notions of model consistency and model completeness. Recall from Chapter 2, that a model can be viewed in one of two ways: (a) as a set of model fragments (Section 2.3.2); and (b) as a set of equations (Section 2.1.2). Our definitions of model consistency and model completeness will be based on knowledge stemming from both these viewpoints.

### 3.4.1 Model consistency

Recall that when two model fragments make contradictory assumptions about the domain they are related by the *contradictory* relation (Section 2.3.4). Therefore, the use of contradictory model fragments in a model is undesirable. Similarly, in Definition 3.4 we defined the notion of an overconstrained set of equations. If a set of independent equations is overconstrained, then the equations have no solutions,[6] leading to a contradiction.

The above observations lead directly to our definition of a consistent model:

**Definition 3.6 (Consistent model).** *A model $M$ is said to be consistent if and only if the following two conditions are satisfied:*

1. $\forall m_1, m_2 \in M \ \neg contradictory(m_1, m_2)$, *i.e., the model does not contain mutually contradictory model fragments;*

[6] Being independent, the possibility of the equations being merely redundant is ruled out.

2. *The set of equations of M is not overconstrained.*

An immediate consequence of the above definition is that a consistent model can have at most one model fragment from each assumption class. Consistency is the first important property of an adequate model; a model that makes contradictory assumptions about the domain, or whose equations are inconsistent, is undesirable. Hence, we have:

– An adequate model must be consistent.

For example, any consistent model of the temperature gauge in Figure 1.1 cannot simultaneously model the wire both as an `Ideal-conductor` and as a `Resistor` because these two model fragment classes contradict each other. Similarly, no consistent model of the temperature gauge will model both the wire and the thermistor as `Ideal-conductors` and the battery as a `Constant-voltage-source`. This is because this set of modeling choices would lead to the following overconstrained set of equations:

$$\begin{aligned} &\boldsymbol{V_r} = 0 \\ &\boldsymbol{V_t} = 0 \\ &\mathit{exogenous}(\boldsymbol{V_v}) \\ &\boldsymbol{V_v} = \boldsymbol{V_r} + \boldsymbol{V_t} \end{aligned}$$

### 3.4.2 Model completeness

Recall that model fragments are partial descriptions of phenomena. Additional model fragments, drawn from the set of required assumption classes, are required to complete this description (Section 2.3.4). A complete model must include complete descriptions of all phenomena that are being modeled. Hence, a complete model must include model fragments from all required assumption classes. In addition, we will require that the equations of a complete model be able to causally determine all the parameters of the model. From Lemma 3.3.2 we know that when a set of equations is complete then all the parameters can be causally determined, and the causal ordering is well defined. These observations lead directly to our definition of a complete model:

**Definition 3.7 (Complete model).** *A model M is said to be complete if and only if the following two conditions are satisfied:*

1. $(\forall m \in M)$ *requires*$(m, A) \Rightarrow (\exists m' \in A)\ m' \in M$, *i.e., the model contains a model fragment from each required assumption class; and*
2. *The set of equations of M is complete.*

Completeness is the second important property of an adequate model; an adequate model must include complete descriptions of all phenomena that are being modeled, and the model's equations must be complete so that we can generate causal explanations for phenomena of interest. Hence, we have:

– An adequate model must be complete.

For example, the model shown in Figure 3.4 is complete.[7]

## 3.5 Representing the phenomenon of interest

Toward the end of Section 3.1 we stated that, in this thesis, we will define the adequacy of a model with respect to the task of generating causal explanations for a phenomenon of interest. Hence, the phenomenon of interest is a crucial input that focuses model selection. We call the phenomenon of interest the *expected behavior*. The expected behavior of a device is an abstract description of *what* the system does (but not *how* it does it). The causal explanation generated by a model is a description of how the expected behavior is achieved.

The expected behavior captures, in part, what is commonly referred to as the *function* of a device. For example, stating that the device in Figure 1.1 is a temperature gauge indicates that the device model must explain how the temperature of the thermistor determines the angular position of the pointer. Expected behaviors can also provide abstract descriptions of device behaviors that would not normally be considered the device's primary function. Such knowledge of the expected behavior is commonplace and almost always available either directly from the user, from the description of the problem to be solved, or from the context in which the device operates.

For example, a student wanting to understand how a device works can provide an intelligent tutoring system a description of the expected behavior that he or she wants explained. Or, for example, an automated diagnosis program that diagnoses faults in a device, must first be provided with a description of the what the correctly working device is supposed to do. Finally, device names, such as light bulb, vacuum cleaner, and disk drive are widely used and all are associated with expected behaviors. The most common expected behavior descriptions are input/output descriptions of device behavior.

Following our discussion of causal ordering in Section 3.3, we specify expected behaviors as a query that requests a causal explanation for how one parameter causally depends on another. For example, the expected behavior of the temperature gauge shown in Figure 1.1, representing its primary function, is:

$$causes(T_t, \theta_p)$$

where $T_t$ is the temperature of the thermistor and $\theta_p$ is the angular position of the pointer. This expected behavior requests a causal explanation for how

[7] Though we haven't shown the *requires* constraints, in fact they are all satisfied in this model.

the temperature of the thermistor causally determines the angular position of the pointer.

The expected behavior provides us with our most important criterion for model adequacy:

- An adequate model must explain the expected behavior, i.e., a model is adequate with respect to an expected behavior, *causes*$(p_1, p_2)$, if it is able to provide a causal explanation for how $p_2$ causally depends on $p_1$.

Given such an expected behavior, one can use the procedures described in Section 3.3 to check whether or not a device model is able to provide a explanation for how the second parameter causally depends on the first parameter. This procedure is briefly summarized in Figure 3.7.

```
function check-expected-behavior(M,p1,p2)
  /* M is a model, assumed to be consistent and complete */
  /* causes(p1,p2) is the expected behavior */
  1. Let E be the equations of M
     /* Section 2.3.3 describes how to do this */
  2. Compute C(E), the causal ordering generated from E
     /* Section 3.3 describes how to do this */
  3. if (p1,p2) ∈ C(E) then
       /* The expected behavior is satisfied */
       return true
       /* The causal explanation can also be returned (Section 3.3) */
     else
       /* The expected behavior is not satisfied */
       return false
     endif
end
```

**Fig. 3.7.** Algorithm for checking whether a model can explain the expected behavior.

For example, the model in Figure 3.4 is able to explain the expected behavior

$$causes(T_t, \theta_p)$$

since $\theta_p$ causally depends on $T_t$ in the causal ordering generated from this model, shown in Figure 3.6.

It must be noted that our language for expressing the expected behaviors is extremely simple; it only allows us to ask for explanations for causal dependencies between parameters. More expressive languages are, of course, desirable. We might want to include information about the directions of change, e.g., we might want to say that *increasing* $T_t$ causes $\theta_p$ to increase. Or we might want to include more information about the actual functional relationship, e.g., we might want to say that there is a linear relationship between $T_t$ and $\theta_p$.

However, the price we must pay for using more expressive languages for the expected behavior is that checking whether or not the expected behavior is satisfied becomes very expensive, and can often even be impossible. For example, deciding whether an increase in $T_t$ causes an increase or a decrease in $\theta_p$ with purely qualitative information is not possible when there are competing influences. Additional information about the relative magnitudes of these influences is necessary, which may or may not be available. Hence, we have chosen a simple, though useful, language for expressing the expected behavior, leading to an efficient algorithm for deciding whether or not a model satisfies the expected behavior.

Thus far we have said that an adequate model must be consistent and complete, and must be able to explain the expected behavior. In addition to these constraints, a domain expert might want to place additional domain-dependent constraints on model adequacy. We now investigate two important classes of such constraints, stemming from the *structural* and *behavioral contexts* of the device. These constraints are expressed using a first-order constraint language, and an adequate model must satisfy all such constraints. Symbols in these constraints that begin with "?" are variables. Constraints are all evaluated with respect to a component of interest, with the variable "`?object`" being bound to that component. All other variables in the constraints are assumed to be existentially quantified.

## 3.6 Constraints from the structural context

In this section we discuss the structural context, an important source of constraints on model adequacy. We will then discuss different types of constraints that stem from the structural context.

### 3.6.1 Structural context

The structural context of a device consists of the different aspects of the structure of the device. Informally, the structure of a device is a description of how the device is physically put together. It includes the *components* in the the device, the *physical* and *structural properties* of these components, and the *structural relations* between these components that describe how they are put together to form the device.

**Components.** The components that can be used to describe the structure of a device are drawn from a library of component types, like the one described in Section 2.5. The particular choice of components in a component library must reflect (a) the domain of interest; and (b) the most detailed level of granularity that needs to be reasoned about. (Section 2.4.2 shows how components at a coarser level of detail can be recognized automatically.) For example, the components used to describe electronic devices would differ

from the components used to describe chemical plants. Similarly, electronic devices can be described at multiple levels of detail, ranging from logic gates down to layers in semiconductor wafers.

**Physical and structural properties.** In addition to the types of the components in the device, the structure of the device can also specify various properties of these components. These properties can be broadly classified as physical and structural properties, and include properties such as shape, dimensions, mass, and material composition. As with the choice of component types, the choice of physical and structural properties of components depends on the domain and how it is conceptualized.

**Structural relations.** Structural relations are relations between components that describe how components are put together. The most commonly used structural relation is the `connected-to` relation, that says that two component terminals are connected to each other [de Kleer and Brown, 1984]. Other structural relations that we use include `coiled-around` (indicating that a wire is coiled around a component), `meshed` (indicating that a pair of gears mesh with each other), and `immersed-in` (indicating that a component is immersed in a fluid). As with components and their physical and structural properties, the set of structural relations is crucially dependent upon the domain, and how we choose to conceptualize it.

**Structural predicates.** Predicates that can be used to describe the structure of a device will be called *structural predicates*. In particular, component types will be unary structural predicates, structural and physical properties of components will be binary structural predicates (in fact, they will be unary functions), and structural relations will be general n-ary structural predicates. As discussed above, deciding which predicates are structural predicates is dependent upon the domain and how it is conceptualized.

**Miscellaneous observations.** The device structure provides an important bias for model selection. In particular, we have seen in Chapter 2 that the components specified in the device structure, in conjunction with a component library, defines the basic space of possible device models. The structural relations specified in the device structure constrains the space of component interactions. Hence, the bias provided by the device structure aids the search for device models by specifying the space of possible component models and the space of possible component interactions.

An alternate, though consistent, viewpoint is as follows: the description of device structure is already a model of the device which embodies some set of modeling decisions. Hence, the model selection algorithms discussed in this thesis can be viewed as making additional modeling decisions, given the modeling decisions made above. In other words, certain aspects of modeling have been automated, while other parts are still the purview of human experts. This division of labor is particularly useful since rudimentary structural models of devices are automatically available when human designers use CAD tools.

Finally, note that the structural context of a device is not fixed, but can change, even during the normal operation of the device: the components in a device can change as new components are created and old ones are destroyed (e.g., boiling water creates steam); the physical and structural properties of components can change (e.g., the magnetic strip on your credit card can get demagnetized); and the structural relations between components can change (e.g., the contact between the hammer and the dome of an electric bell constantly changes during the normal operation of the bell).

### 3.6.2 Constraints

Domain-dependent constraints that stem from the structural context are called *structural constraints*. Structural constraints are evaluated with respect to a structural context and a device model. Hence, as the structural context changes, different device models may be necessary to ensure that all the structural constraints are satisfied. We distinguish two types of structural constraints: *preconditions* and *coherence constraints*.

**Structural preconditions.** Structural preconditions are first-order constraints associated with model fragment classes which use only structural predicates. The structural preconditions associated with a model fragment class are constraints on the structural context that must be satisfied if a component is to be modeled by that model fragment class. For example, assuming that `composition` and `metal` are structural predicates, the precondition:[8]

```
(and (composition ?object ?material)
     (metal ?material))
```

in the `Electrical-conductor` model fragment class indicates that a component must be metallic for it to be modeled as an `Electrical-conductor`.

Structural preconditions are similar to process preconditions in QP theory [Forbus, 1984]. However, process preconditions are sufficient conditions, i.e., a process instance is created whenever the process preconditions are satisfied. On the other hand, structural preconditions are necessary conditions. Hence, the above constraint does not require that every metallic object be modeled as an `Electrical-conductor`. It only says that a component can be modeled as an `Electrical-conductor` only if it is metallic. We can express this precisely by rewriting the above constraint as follows:

```
(implies
   (Electrical-conductor ?object)
   (and (composition ?object ?material)
        (metal ?material)))
```

[8] Recall that the variable "`?object`" is bound to the component of interest, i.e., to the component that we want to model as an instance of this model fragment class.

More generally, if $C$ is a structural precondition associated with model fragment class $M$, then this is equivalent to the constraint $M(\texttt{?object}) \Rightarrow C$.

**Structural coherence constraints.** Structural coherence constraints are additional first-order constraints on the model fragment classes used to model one or more components. The predicates used in structural coherence constraints are either structural predicates or model fragment classes (which are unary predicates). As with structural preconditions, each structural coherence constraint is associated with a model fragment class, expressing the constraint that a component can be modeled by that model fragment class only if the corresponding constraint is satisfied.

For example, the following structural coherence constraint:

```
(implies
   (and (Wire ?object)
        (coiled-around ?object ?core)
        (magnetic-material ?core))
   (Magnet ?core))
```

associated with the `Electromagnet` model fragment class, requires that a wire coiled around a core made of magnetic material can be modeled as an electromagnet only if the core is modeled as a magnet. The justification for this domain dependent constraint is that such a core amplifies the wire's magnetic field by three or four orders of magnitude, converting the core into a powerful magnet. Hence, under these circumstances, an engineer would not consider the model to be adequate unless the core were modeled as a magnet.

Note that, like structural preconditions, structural coherence constraints are also associated with model fragment classes. Hence, the above constraint is more precisely written as:

```
(implies
   (and (Electromagnet ?object)
        (Wire ?object)
        (coiled-around ?object ?core)
        (magnetic-material ?core))
   (Magnet ?core))
```

where we have used the fact that $A \Rightarrow (B \Rightarrow C)$ is equivalent to $(A \wedge B) \Rightarrow C$).

In summary, an adequate model must satisfy all applicable structural constraints:

- A model fragment `M(c)` can be part of an adequate model only if all the structural preconditions and structural coherence constraints associated with model fragment class `M` are satisfied, with the variable `?object` bound to `c`.

## 3.7 Constraints from the behavioral context

In this section we discuss the behavioral context, another important source of constraints on model adequacy. We will then discuss different types of constraints that stem from the behavioral context.

### 3.7.1 Behavioral context

The behavioral context of a device is its behavior at a particular time. The behavior of a device at a particular time is just the values, at that time, of the parameters that can be used to model the device. Note that the behavioral context of a device is dependent upon the time at which the behavior snapshot is taken. Hence, the behavioral context changes with time, as the values of the parameters change. For example, the behavioral context of the temperature gauge in Figure 1.1 would include values for the current flowing in the circuit, the temperature of the bimetallic strip, and the magnetic field generated by the wire. As the values of these parameters change, the behavioral context also changes.

Ideally, we would like the behavioral context to refer to the actual behavior of the device, e.g., the values of the parameters are obtained by actual measurements on a physical prototype. However, the actual behavior of a device is usually unavailable. Rather, the behavior must be computed using the equations of a device model. Hence, the behavioral context can be computed only after a device model has been selected. Of course, different device models can predict different behaviors, each introducing different errors. Hence, it is essential that the behavior be computed with a device model that introduces an acceptably low error.

A component's behavioral context can provide modeling information not explicitly available in the structural context. This is because behavior generation explicates information that is implicit in equations. Consider modeling an air gap: if the voltage drop across it is large enough (as in a properly functioning spark plug), then it should be modeled as an electrical conductor; if the voltage drop across it is not large enough (as in a common electrical switch), it should be modeled as an electrical insulator. The value of the voltage drop across the air gap (a behavioral property) determines the appropriate model for it.

### 3.7.2 Constraints

Domain-dependent constraints that stem from the behavioral context are called *behavioral constraints*. Behavioral constraints are evaluated with respect to a behavioral context, a structural context, and a device model. Hence, as the behavioral context changes over time, different device models may be necessary to ensure that all the behavioral constraints are satisfied

(assuming that the structural context remains the same). As with structural constraints, we distinguish two types of behavioral constraints: *preconditions* and *coherence constraints*.

**Behavioral preconditions.** Behavioral preconditions are first-order constraints associated with model fragment classes which use only structural predicates and order relations between parameter values, i.e., they do not use model fragment classes. The behavioral preconditions associated with a model fragment class are constraints that must be satisfied if a component is to be modeled by that model fragment class. For example, the precondition:

```
(< (voltage-difference ?object)
   (voltage-difference-threshold ?object))
```

associated with the `Ideal-conductor` model fragment class indicates that a component can be modeled as an `Ideal-conductor` only if the voltage drop across it is less than some threshold. As with structural preconditions, behavioral preconditions are necessary conditions on the use of model fragment classes. Hence, the above constraint is more precisely written as:

```
(implies
   (Ideal-conductor ?object)
   (< (voltage-difference ?object)
      (voltage-difference-threshold ?object)))
```

Behavioral preconditions look superficially similar to quantity conditions in processes [Forbus, 1984]. However, behavioral preconditions are used to decide which model fragment classes in an assumption class can be used to model a component. In contrast quantity conditions in processes only control the activity of a process, but not the existence of the process. In essence, behavioral preconditions are modeling constraints, while quantity conditions are about the physics of the situation.

**Behavioral coherence constraints.** Behavioral coherence constraints are additional first-order constraints on the model fragment classes used to model one or more components. The predicates used in behavioral coherence constraints are either relations between parameter values, structural predicates, or model fragment classes (which are unary predicates). As with behavioral preconditions, each behavioral coherence constraint is associated with a model fragment class, expressing the constraint that a component can be modeled by that model fragment class only if the corresponding constraint is satisfied.

For example, the following behavioral coherence constraint:

```
(implies
  (>= (* (voltage-difference ?object)
         (current (electrical-terminal-one ?object)))
      (electrical-power-threshold ?object))
  (Thermal-resistor ?object))
```

in the **Resistor** model fragment class states that when a component is being modeled as a resistor, and if the dissipated power exceeds a threshold, then this dissipation must be explicitly modeled by modeling the component as a **Thermal-resistor**.

Note that, like behavioral preconditions, behavioral coherence constraints are also associated with model fragment classes. Hence, the above constraint is more precisely written as:

```
(implies
  (and (Resistor ?object)
       (>= (* (voltage-difference ?object)
              (current (electrical-terminal-one ?object)))
           (electrical-power-threshold ?object)))
       (Thermal-resistor ?object))
```

In summary, an adequate model must satisfy all applicable behavioral constraints:

- A model fragment **M(c)** can be part of an adequate model only if all the behavioral preconditions and behavioral coherence constraints associated with model fragment class **M** are satisfied, with the variable **?object** bound to **c**.

### 3.7.3 Thresholds in behavioral constraints

Behavioral constraints can be viewed as deciding whether or not particular phenomena are significant, and hence worth modeling. Behavioral constraints decide on the significance of phenomena by checking whether the values of certain parameters are high enough or low enough. Appropriately set thresholds decide whether or not the parameters values are high enough or low enough.

For example, the behavioral precondition shown above says that a component can be modeled as an **Ideal-conductor** only if the **voltage-difference** across the component is insignificant. This is checked by comparing the **voltage-difference** with the **voltage-difference-threshold**. Similarly, the behavioral coherence constraint shown above requires a **Resistor** to be modeled as a **Thermal-resistor** if the heat generated in the **Resistor** is significant, i.e., large enough. This is checked by comparing the actual amount of heat generated against the **electrical-power-threshold**.

Since the thresholds determine the significance of various phenomena, different threshold settings lead to models of differing accuracy, i.e., to models that include different sets of significant phenomena. Thresholds can be either preset or computed dynamically. A widely used preset threshold in the domain of fluid mechanics is a threshold of 2300 for Reynolds number, that distinguishes laminar fluid flow from turbulent fluid flow. Thresholds can also be preset by an engineer from common practice. For example, in the domain

of power distribution systems, where normal voltages are in the range of tens of thousands of volts, a voltage difference of up to 10 volts may be considered insignificant. On the other hand, in the domain of electronic circuits, voltages of only up to .01 volts may be considered insignificant.

While thresholds can be preset, a more interesting and robust method of setting the thresholds is to set them dynamically, based on knowledge of acceptable error tolerances on some parameters. These error tolerances can be propagated to set other thresholds. This propagation can be done using either propagation rules, or the equations of a device model. In this thesis, we do not explore this interesting line of work any further. See [Shirley and Falkenhainer, 1990; Nayak, 1991] for some initial work in this area.

## 3.8 Simplicity of models

Thus far, we have said that an adequate model must be consistent and complete, must be able to explain the expected behavior, and must satisfy all the domain-dependent constraints stemming from the structural and behavioral contexts. Typically a very large number of device models satisfy these criteria. Most of these models introduce irrelevant detail into the causal explanations they generate, either by modeling irrelevant phenomena, or by including needlessly complex models of relevant phenomena.

For example, assume that the model in Figure 3.4 satisfies all the above criteria. Other models that augment this model by modeling additional phenomena, such as the electromagnetic field generated by the wire, would also satisfy the above criteria. Similarly, models that use more accurate descriptions of phenomena that are already modeled, e.g., by modeling the wire as a temperature dependent resistor rather than a constant resistance resistor, would also satisfy the above criteria. Such models introduce irrelevant detail into the causal explanation of how the thermistor's temperature affects the pointer's angular position.

To address this problem we need a *simplicity ordering* on the models. Given such a simplicity ordering, we will say that an adequate model is a simplest model that satisfies all the above criteria, i.e., no simpler model satisfies the above criteria. The simplicity ordering we consider is a partial ordering of the models, and is based on the *approximation* relation between model fragments. This definition of simplicity is based on the following two intuitions: (a) a model is simpler if it models fewer phenomena; and (b) approximate descriptions are simpler than more accurate ones.

**Definition 3.8 (Simplicity of models).** *A model $M_2$ is simpler than a model $M_1$ (written $M_2 \leq M_1$) if for each model fragment $m_2 \in M_2$ either (a) $m_2 \in M_1$; or (b) there is a model fragment $m_1 \in M_1$ such that $m_2$ is an approximation of $m_1$, i.e., approximation$(m_1, m_2)$. $M_2$ is strictly simpler than $M_1$ (written $M_2 < M_1$) if $M_2 \leq M_1$ and $M_1 \not\leq M_2$.*

For example, a model simpler than the one shown in Figure 3.4 is one that removes the model fragment `Thermal-resistor(wire-1)`. A more complex model results from replacing `Constant-resistance(wire-1)` by the model fragment `Temperature-dependent-resistor(wire-1)`. A model that is incomparable to the one in Figure 3.4 is the one in which we remove `Thermal-resistor(wire-1)` and replace `Constant-resistance(wire-1)` by `Temperature-dependent-resistor(wire-1)`.

It is important to note that this definition of model simplicity is based purely on the intuitions mentioned above. In particular, the definition does not guarantee that a simpler model is more efficient. Nor does it guarantee that simpler models lead to simpler causal explanations of the expected behavior. However, while there are no such guarantees, we believe that the above definition of simplicity provides a good heuristic for identifying more efficient models, and for generating simpler causal explanations. In particular, it is common engineering practice to simplify models by disregarding irrelevant phenomena and by using all applicable approximations. In addition, in Chapter 5 we shall introduce a special class of approximations, called *causal approximations*, which will ensure that the above definition of simplicity will, in fact, lead to simpler causal explanations.

We will require that adequate models be as simple as possible, provided the rest of the criteria discussed in this chapter are satisfied:

- An adequate model is a simplest model that meets all the criteria discussed in this chapter.

## 3.9 Summary

The adequacy of models is closely linked to the task for which the model is to be used. In this thesis, we consider the adequacy of models with respect to the task of generating causal explanations for phenomena of interest. Causal explanations play an important role in reasoning about physical systems, not only as a vehicle for communicating with human users, but also to focus other tasks such as diagnosis, design, and simulation. A widely used class of causal explanations are based on causal dependencies between parameters. These causal dependencies between parameters, also called the causal ordering of the parameters, are derived from the equations comprising a device model.

Our definition of model adequacy is based on the following inputs:

1. The component library, described in Chapter 2, which is a description of the components, their possible models, and various relations between the possible models.
2. The expected behavior, which is the phenomenon for which the causal explanation is desired. The expected behavior is represented as a query, $causes(p_1, p_2)$, requesting a causal explanation for how one parameter, $p_2$, causally depends on another, $p_1$.

3. The structural context, which includes the different aspects of the structure of the device. The structural context defines the basic space of possible device models.
4. The behavioral context, which includes the values of parameters that can be used to model the device.
5. The structural constraints, which are a set of domain-dependent constraints that can be evaluated using the structural context and the device model.
6. The behavioral constraints, which are a set of domain-dependent constraints that can be evaluated using the behavioral context, the structural context, and the device model.

Given the above set of inputs, the adequacy of a device model is defined as follows:

1. An adequate model must be consistent, i.e., its equations must not be overdetermined and it must not include contradictory model fragments.
2. An adequate model must be complete, i.e., its equations must be complete and it must include model fragments from every required assumption class.
3. An adequate model must be able to explain the expected behavior, i.e., the causal ordering generated from the model's equations must subsume the causal dependency for which an explanation is requested.
4. An adequate model must satisfy all domain-dependent structural and behavioral constraints.
5. An adequate model is a simplest model that satisfies the above four conditions.

# 4. Complexity of model selection

In this chapter we analyze the complexity of the problem of finding adequate device models. In particular, we will show that this problem is NP-hard. We will provide three different proofs of this result, with each proof being based on a special case of the general problem. These special cases help to identify three different sources of the intractability of the problem of finding adequate device models. Informally, the three sources of intractability are: (a) deciding what phenomena to model; (b) deciding how to model the selected phenomena; and (c) ensuring that all domain-dependent constraints are satisfied.

In Section 4.1 we present a formalization of the problem of finding adequate models. In particular, we show how the elements of this formalization are derived from the inputs to the model selection problem discussed in the previous chapter. Section 4.2 contains the complexity analysis of the different special cases of the general problem of finding adequate models. In Section 4.3 we briefly discuss the complexity of some related problems. In particular, we show that the problem of finding just a consistent and complete model is intractable, and that finding adequate models remains intractable even if each equation can have exactly one causal orientation

## 4.1 Formalizing the problem

In this section we develop a formal statement of the problem of finding an adequate device model. We start by formalizing the input to the model selection problem, and then give a formal statement of the problem.

### 4.1.1 Formalizing the input

In the previous chapter we saw that the inputs to our definition of model adequacy are the following: the component library, the expected behavior, the structural context, the behavioral context, the structural constraints, and the behavioral constraints. The formalization we develop here will include representations of the first two and the last two of these inputs. However, the formalization will not include an explicit representation of the structural and behavioral contexts. Rather, we assume that these are given, and we use them

implicitly in formalizing the other inputs. This means that the complexity results of this chapter, and the algorithms developed in the next chapter, will have nothing to say about how the structural and behavioral contexts are computed. Chapters 7 and 8 will discuss this issue in more detail.

We formalize the input to the model selection problem as a tuple $\mathcal{I}$:

$$\mathcal{I} = (\mathcal{M}, \mathit{contradictory}, \mathit{approximation}, \mathcal{A}, \mathcal{C}, p, q) \tag{4.1}$$

where $\mathcal{M}$ is the set of all applicable model fragments, *contradictory* and *approximation* are binary relations on model fragments as discussed in Chapter 2, $\mathcal{A}$ is the set of all applicable assumption classes, $\mathcal{C}$ is a set of *propositional coherence constraints*, and $p$ and $q$ are parameters representing the fact that *causes*$(p, q)$ is the expected behavior. We now discuss each of these, focusing in particular on how the component library, the structural constraints, and the behavioral constraints are translated into elements of the above tuple. As a typographic convention, we will typeset all elements of the input using `typewriter` font, and all elements of our formalization using *italics* or calligraphic letters.

**Propositional coherence constraints.** We start by introducing *propositional coherence constraints*. A propositional coherence constraint is just a propositional formula in which the propositions are model fragments. A propositional coherence constraint is satisfied with respect to a set $M$ of model fragments just in case the corresponding propositional formula is satisfied by the interpretation that assigns *true* to a proposition if and only if the proposition is in $M$, and *false* otherwise. For example, the propositional coherence constraint

$$(m_1 \vee m_2) \Rightarrow m_3$$

is satisfied by the set $\{m_1, m_3\}$ of model fragments. It is also satisfied by the set $\{m_2, m_3\}$, and by the empty set.

As a convenient shorthand, we allow the use of assumption classes in propositional coherence constraints. Recall that an assumption class is a set of mutually contradictory model fragments. Hence, we use an assumption class as a shorthand for a disjunction of the model fragments in the assumption class. For example, if the assumption class $A$ contains the model fragments $m_1$ and $m_2$, then the propositional coherence constraint

$$m_3 \Rightarrow A$$

is equivalent to the propositional coherence constraint

$$m_3 \Rightarrow (m_1 \vee m_2)$$

Recall that a model is just a set of model fragments. $\mathcal{C}$ is the set of propositional coherence constraints that must be satisfied by any adequate model. As we shall see, the propositional coherence constraints in $\mathcal{C}$ will be defined using the structural and behavioral coherence constraints, and the required assumption classes of model fragments.

**The component library.** We formalize the component library as a set $\mathcal{M}'$ of *all* model fragments. (Later, when we discuss the structural and behavioral preconditions, we will introduce the set $\mathcal{M} \subseteq \mathcal{M}'$, which is the set of all *applicable* model fragments.) The model fragments in $\mathcal{M}'$ are constructed from the structural context, which specifies the components used in the device, and the component library, which specifies the possible models of each component class. $\mathcal{M}'$ contains a model fragment for each of the possible ways in which each component of the device can be modeled, i.e., if `c-1` is a component of the device, and if `c-1` is an instance of component class `C`, and if `M` is a model fragment class that is a possible model of `C`, then $\mathcal{M}'$ contains the model fragment `M(c-1)`. The possible models of a component class is just the transitive closure of the `possible-models` of the class.

$$\begin{aligned}\mathcal{M}' \quad = \quad \{&\texttt{M(c-1)} : \texttt{c-1} \text{ is a component of the device} \\ &\wedge\ \texttt{c-1} \text{ is an instance of component class } \texttt{C} \\ &\wedge\ \texttt{M} \text{ is a possible model of } \texttt{C}\}\end{aligned} \tag{4.2}$$

Note that the device components used in defining $\mathcal{M}'$ include the structural abstractions discussed in Section 2.4.2.

The component library also defines a number of important relations: generalization, `contradictory`, `approximation`, `assumption-class`, and `required-assumption-classes`. As discussed in Chapter 2, we use `contradictory` and `approximation` to define the *contradictory* and *approximation* relations between model fragments, respectively. In particular, if model fragment class `M1` specifies model fragment class `M2` as a `contradictory` class, and if `c` is a component such that `M1(c)` and `M2(c)` are model fragments in $\mathcal{M}'$, then we include the literal

$$contradictory(\texttt{M1(c)}, \texttt{M2(c)})$$

in our formalization. Similarly, if model fragment class `M1` specifies model fragment class `M2` as an `approximation`, and if `c` is a component such that `M1(c)` and `M2(c)` are model fragments in $\mathcal{M}'$, then we include the literal

$$approximation(\texttt{M1(c)}, \texttt{M2(c)})$$

in our formalization. The properties of *contradictory* and *approximation* are discussed in detail in Chapter 2. For convenience, we restate their most important properties here:

$$\neg contradictory(m_1, m_1) \tag{4.3}$$

$$contradictory(m_1, m_2) \Rightarrow contradictory(m_2, m_1) \tag{4.4}$$

$$\neg approximation(m_1, m_1) \tag{4.5}$$

$$approximation(m_1, m_2) \Rightarrow \neg approximation(m_2, m_1) \tag{4.6}$$

$$\begin{aligned}&approximation(m_1, m_2) \wedge approximation(m_2, m_3) \\ &\qquad \Rightarrow approximation(m_1, m_3)\end{aligned} \tag{4.7}$$

$$approximation(m_1, m_2) \Rightarrow contradictory(m_1, m_2) \tag{4.8}$$

We use the `assumption-classes` of model fragment classes to define the set $\mathcal{A}'$ of all assumption classes. In particular, if `c` is a component, and `M` is a model fragment class that specifies `A` as its `assumption-class`, and if `M(c)` is a model fragment in $\mathcal{M}'$, then we say that `A(c)` is an assumption class in $\mathcal{A}'$ that contains the model fragment `M(c)`.

We formalize the `required-assumption-classes` of model fragment classes using propositional coherence constraints. In particular, if a model fragment class `M` specifies `A` as a `required-assumption-class`, and if `c` is a component such that `M(c)` is a model fragment in $\mathcal{M}'$, then we add the propositional coherence constraint

$$\texttt{M(c)} \Rightarrow \texttt{A(c)}$$

to the set $\mathcal{C}$ of propositional coherence constraints. Since an adequate model must satisfy each constraint in $\mathcal{C}$, it follows that every adequate model will include a model fragment from each required assumption class.

Finally, we represent the generalization relation between model fragment classes using propositional coherence constraints. In particular, if a model fragment class `M1` is a generalization of a model fragment class `M2`, and if `M1(c)` and `M2(c)` are model fragments in $\mathcal{M}'$, then we add the propositional coherence constraint

$$\texttt{M2(c)} \Rightarrow \texttt{M1(c)}$$

to the set $\mathcal{C}$ of propositional coherence constraints. This ensures that every adequate model that models `c` as an instance of `M2` also models it as an instance of `M1`.

**Structural and behavioral preconditions.** The structural and behavioral preconditions associated with a model fragment class are necessary conditions for a component to be modeled by that class. Recall that structural and behavioral preconditions are constraints that use only structural predicates and order relations between parameters, i.e., they do not use model fragment classes. Hence, these preconditions are not evaluated with respect to a device model, but rather can be evaluated using only the structural and behavioral contexts of the device.

We use the structural and behavioral preconditions to define the set $\mathcal{M} \subseteq \mathcal{M}'$ of *applicable* model fragments, i.e., the set of model fragments for which the structural and behavioral preconditions are satisfied. More precisely, let `M` be a model fragment class and let `c` be a component such that `M(c)` is a model fragment in $\mathcal{M}'$. `M(c)` is in $\mathcal{M}$ if and only if all the structural and behavioral preconditions associated with model fragment class `M` are satisfied when the variable "`?object`" is bound to `c`. For example, the model fragment `Electrical-conductor(wire-1)` is in $\mathcal{M}$ only if the structural precondition

```
(and (composition ?object ?material)
     (metal ?material))
```

is satisfied when "`?object`" is bound to `wire-1`. Recall that the other variables in this constraint, like "`?material`," are existentially quantified.

Hence, from the structural and behavioral preconditions, the structural and behavioral contexts, and the the set $\mathcal{M}'$ of all model fragments, we define the set $\mathcal{M}$ of all applicable model fragments. Using $\mathcal{M}$ and the set $\mathcal{A}'$ of all assumption classes, it is straightforward to define the set $\mathcal{A}$ of all applicable assumption classes. Informally, $\mathcal{A}$ is the set of assumption classes that results from restricting the assumption classes in $\mathcal{A}'$ to contain only applicable model fragments. More precisely, if $A' \in \mathcal{A}'$ is an assumption class, then let $applicable(A')$ be the maximal subset of $A'$ that contains only applicable model fragments, i.e., model fragments from $\mathcal{M}$. Hence, we have:

$$\mathcal{A} = \{A : A = applicable(A') \ \wedge \ A' \in \mathcal{A}' \ \wedge \ A \neq \emptyset\} \tag{4.9}$$

**Structural and behavioral coherence constraints.** The structural and behavioral coherence constraints associated with model fragment classes are constraints that use structural predicates, order relations between parameters, *and* unary predicates representing model fragment classes. Hence, these coherence constraints are evaluated with respect to the structural context, the behavioral context, and a device model. We can remove the dependence of these coherence constraints on the structural and behavioral contexts by converting each of them into a set of propositional coherence constraints. Conceptually, this is achieved by instantiating each coherence constraint in all possible ways over the universe of all objects in the knowledge base.[1] Each resulting instantiated constraint can be converted into a propositional coherence constraint by replacing each ground literal in the constraint by *true*, *false*, or a model fragment, according to the following rules:

1. If the literal involves a structural predicate, use the structural context to decide whether the literal is *true* or *false*.
2. If the literal is an order relation between parameters, use the behavioral context to decide whether the literal is *true* or *false*.
3. If the literal involves a unary predicate representing a model fragment class, then check whether or not the corresponding model fragment is in $\mathcal{M}$. (The model fragment corresponding to the ground literal `(M c)` is, of course, `M(c)`.) If the corresponding model fragment is in $\mathcal{M}$, replace the literal by the model fragment, else replace the literal by *false*.

Using the above procedure, each instantiated coherence constraint can be converted into a propositional coherence constraint. All such propositional coherence constraints, except the ones that are vacuously *true*, are added to $\mathcal{C}$ as constraints that must be satisfied by any adequate model.[2]

[1] The universe of all objects in the knowledge base would include, among others, the components in the device, the components terminals, and the parameters.

[2] If there are any vacuously *false* propositional coherence constraints then that means that the corresponding coherence constraint can never be satisfied, and hence there is no adequate model.

For example, consider the following structural coherence constraint:

```
(implies
   (and (Electromagnet ?object)
        (Wire ?object)
        (coiled-around ?object ?core)
        (Magnetic-material ?core))
   (Magnet ?core))
```

One way to instantiate the above constraint is to bind "`?object`" to `wire-1`, and to bind "`?core`" to `bms-1` to get the following ground constraint:

```
(implies
   (and (Electromagnet wire-1)
        (Wire wire-1)
        (coiled-around wire-1 bms-1)
        (Magnetic-material bms-1))
   (Magnet bms-1))
```

To convert the above ground constraint into a propositional coherence constraint, let us assume that the structural context says that `wire-1` is a `Wire`, and that it is `coiled-around` `bms-1`, which is made of `Magnetic-material`. Hence, the above constraint reduces to the following propositional coherence constraint:

$$\texttt{Electromagnet(wire-1)} \Rightarrow \texttt{Magnet(bms-1)}$$

On the other hand, if we bind "`?core`" to `ptr-1`, then we get the following ground constraint:

```
(implies
   (and (Electromagnet wire-1)
        (Wire wire-1)
        (coiled-around wire-1 ptr-1)
        (Magnetic-material ptr-1))
   (Magnet ptr-1))
```

Since `wire-1` is not `coiled-around` `ptr-1`, the third conjunct in the antecedent of the above constraint gets replaced by *false*, and hence the propositional coherence constraint corresponding to the above ground constraint is vacuously *true*.

In summary, given the structural and behavioral contexts, the structural and behavioral coherence constraints can be converted into a set of propositional coherence constraints. Note that the above discussion does not imply that it is a good idea to convert all the structural and behavioral coherence constraints into propositional coherence constraints, or that the above is the best way to do it. The point of the discussion is to show that, given the structural and behavioral contexts, the structural and behavioral coherence

constraints can be viewed as a set of propositional coherence constraints. This will simplify the complexity analysis of this chapter, and the development of efficient algorithms in the next chapter.

### 4.1.2 Problem statement

Given the formalization of the input to the problem of finding an adequate model as the following tuple:

$$\mathcal{I} = (\mathcal{M}, \mathit{contradictory}, \mathit{approximation}, \mathcal{A}, \mathcal{C}, p, q)$$

we are in a position to give a precise statement of the problem itself. Before we do this we define three important types of models: coherent models, causal models, and adequate models.

**Coherent, causal, and adequate models.** Recall that a model is a set of model fragments. We will require that the model fragments in a model must be in $\mathcal{M}$, i.e., we will only consider models consisting of applicable model fragments. A *coherent* model is a complete, consistent model, that satisfies all the propositional coherence constraints in $\mathcal{C}$:

**Definition 4.1 (Coherent models).** *A model $M \subseteq \mathcal{M}$ is said to be a coherent model if and only if the following conditions are satisfied:*

1. *$M$ contains no mutually contradictory model fragments.*
2. *The equations of $M$ are complete (Definition 3.4).*
3. *All the constraints in $\mathcal{C}$ are satisfied by $M$.*

Conditions 1 and 2 together ensure that coherent models are consistent (Definition 3.6), since if the equations of $M$ are complete then the equations are not overconstrained. Conditions 2 and 3 together ensure that coherent models are complete (Definition 3.7), since $\mathcal{C}$ contains constraints that ensure that coherent models contain model fragments from all required assumption classes.

A *causal model* is a coherent model that also explains the expected behavior.

**Definition 4.2 (Causal model).** *A model $M \subseteq \mathcal{M}$ is a causal model, with respect to the expected behavior causes$(p, q)$, if and only if (a) $M$ is a coherent model; and (b) $q$ causally depends on $p$ in the causal ordering generated from the equations of $M$, i.e., $(p, q) \in C(E(M))$.*

Finally, an *adequate* model is just a minimal causal model.

**Definition 4.3 (Adequate model).** *A model $M \subseteq \mathcal{M}$ is an adequate model if and only if $M$ is a causal model and no coherent model strictly simpler than $M$ is a causal model, i.e., for all coherent models $M'$, such that $M' < M$, $M'$ is not a causal model. (Model simplicity is defined as in Definition 3.8.)*

**The minimal causal model problem.** We now give a formal statement of the problem of finding an adequate model. We call this problem the MINIMAL CAUSAL MODEL problem.

**Definition 4.4** (MINIMAL CAUSAL MODEL). *Let the input to the problem of finding an adequate model be the tuple $\mathcal{I}$:*

$$\mathcal{I} = (\mathcal{M}, \textit{contradictory}, \textit{approximation}, \mathcal{A}, \mathcal{C}, p, q)$$

*where the elements of the tuple are as in Equation 4.1. Find an adequate model with respect to $\mathcal{I}$, i.e., find a minimal, causal model with respect to $\mathcal{I}$.*

To help in analyzing the complexity of the MINIMAL CAUSAL MODEL problem, we introduce the CAUSAL MODEL problem, which is the decision problem corresponding to the MINIMAL CAUSAL MODEL problem. The CAUSAL MODEL problem asks whether or not there exists a causal model, without requiring this causal model to be minimal.

**Definition 4.5** (CAUSAL MODEL). *Let the input to the problem of finding an adequate model be the tuple $\mathcal{I}$:*

$$\mathcal{I} = (\mathcal{M}, \textit{contradictory}, \textit{approximation}, \mathcal{A}, \mathcal{C}, p, q)$$

*where the elements of the tuple are as in Equation 4.1. Does there exist a causal model with respect to $\mathcal{I}$?*

## 4.2 Complexity analysis

In this section we analyze the complexity of the CAUSAL MODEL problem and the MINIMAL CAUSAL MODEL problem. In particular, we will show that the CAUSAL MODEL problem is NP-complete. An immediate corollary of this is that the MINIMAL CAUSAL MODEL problem is NP-hard. Since it is strongly believed that P $\neq$ NP, these results imply that, in general, the problem of finding adequate device models is intractable, i.e., there is no polynomial time algorithm for finding adequate device models.

We prove that the CAUSAL MODEL problem is NP-complete by first showing that it is in NP, and then showing that three of its special cases are NP-hard. The three special cases will identify three sources for the intractability of the CAUSAL MODEL problem. Informally, the three sources are: (a) deciding what phenomena to model, i.e., deciding which assumption classes to use; (b) deciding how to model the chosen phenomena, i.e., selecting model fragments from chosen assumption classes; and (c) ensuring that causal models satisfy all the propositional coherence constraints. In the next chapter, we will use this knowledge to design special cases of the MINIMAL CAUSAL MODEL problem that can be solved in polynomial time.

### 4.2.1 Problem size

Before we start the complexity analysis, we define the *size* of the input to the CAUSAL MODEL and MINIMAL CAUSAL MODEL problems. The input to these problems is as defined in Equation 4.1, reproduced here for ease of reference:

$$\mathcal{I} = (\mathcal{M}, \mathit{contradictory}, \mathit{approximation}, \mathcal{A}, \mathcal{C}, p, q)$$

We define the size of the input as the sum of:

1. $|\mathcal{M}|$, the number of model fragments in $\mathcal{M}$;
2. $|\mathcal{C}|$, the number of constraints in $\mathcal{C}$;
3. $|E(\mathcal{M})|$, the number of equations in the model fragments in $\mathcal{M}$; and
4. $|P(\mathcal{M})|$, the number of parameters used in the equations of the model fragments in $\mathcal{M}$.

It is easy to see that the amount of space occupied by any reasonable encoding of $\mathcal{I}$ must be a polynomial function of the size of $\mathcal{I}$.[3] In particular, the number of tuples in the *contradictory* and *approximation* relations is bounded by a quadratic function of $|\mathcal{M}|$, and the number of assumption classes in $\mathcal{A}$ is bounded by $|\mathcal{M}|$. The complexity analyses in this chapter and the next chapter are with respect to the above definition of the size of a problem instance. In particular, the phrase "runs in polynomial time" will often be used to mean "runs in time polynomial in the size of $\mathcal{I}$," where the instance $\mathcal{I}$ will be clear from the context.

### 4.2.2 Preliminaries

We start the analysis by showing that the CAUSAL MODEL problem is in NP.

**Lemma 4.2.1.** *The* CAUSAL MODEL *problem is in NP.*

*Proof.* To show that the CAUSAL MODEL problem is in NP, we need to show that a nondeterministic algorithm can find a causal model in (nondeterministic) polynomial time. Since there are a finite number of models, each of which can be generated in polynomial time, it suffices to show that checking whether or not a model is a causal model can be done in time polynomial in the size of $\mathcal{I}$.

Given $M \subseteq \mathcal{M}$, it is easy to check in polynomial time whether or not $M$ contains mutually *contradictory* model fragments, and whether or not $M$ satisfies all the constraints in $\mathcal{C}$. From the algorithms given in the previous chapter, it is also possible to check in polynomial whether or not the equations

[3] Note that we have made the (reasonable) assumption that the amount of space used in encoding each equation is bounded by a polynomial function of the number of parameters used in the equation.

of $M$ are complete, and whether or not $q$ causally depends on $p$ in the causal ordering generated from the equations of $M$. Hence, the CAUSAL MODEL problem is in NP.

We now show that the CAUSAL MODEL problem is NP-hard. We will give three different proofs of this result. In each proof, we will introduce a subclass of the instances of the CAUSAL MODEL problem, and show that even if we restrict ourselves to solving just the problem instances in that subclass, the CAUSAL MODEL problem is NP-hard. This will allow us to identify three different sources of intractability. The NP-hardness of the general CAUSAL MODEL problem is, of course, an immediate consequence of the NP-hardness of any of the three subclasses.

In each of the subclasses of the CAUSAL MODEL problem we will restrict the *contradictory* relation to be a relation that partitions the set of model fragments into the set of assumption classes, i.e., two model fragments are in the same assumption class *if* and only if they are mutually *contradictory*:

$$\begin{array}{l}(\forall m_1, m_2 \in \mathcal{M}) \\ m_1 \neq m_2 \Rightarrow (\mathit{contradictory}(m_1, m_2) \equiv (\exists A \in \mathcal{A})\ m_1, m_2 \in A)\end{array} \tag{4.10}$$

A consequence of the above restriction is that we can conceptually view the problem of finding a causal model as one involving the following two steps: (a) selecting a set of assumption classes; and (b) selecting a single model fragment from each selected assumption class. Intuitively, this corresponds to deciding which phenomena to model (step (a)), and then deciding how to model the chosen phenomena (step (b)).

### 4.2.3 The SELECT MODEL FRAGMENTS problem

The first special case of the CAUSAL MODEL problem consists of those instances of the problem that satisfy the following two conditions: (a) the instance has no propositional coherence constraints; and (b) every causal model of the instance includes a model fragment from each assumption class. Hence, this special case allows us to identify the first source of intractability: choosing a model fragment from each assumption class in a set of selected assumption classes is intractable. More abstractly, even if we knew exactly which phenomena we wanted to model, deciding how to model the chosen phenomena is intractable.

**Definition 4.6** (SELECT MODEL FRAGMENTS). *This problem is the special case of the* CAUSAL MODEL *problem which includes exactly those instances of the* CAUSAL MODEL *problem in which (a) the contradictory relation partitions the set $\mathcal{M}$ of model fragments into the set $\mathcal{A}$ of assumption classes; (b) $\mathcal{C} = \emptyset$; and (c) every causal model of the instance includes a model fragment from each assumption class, i.e., if $M \subseteq \mathcal{M}$ is a causal model and $A \in \mathcal{A}$ is an assumption class, then $M \cap A \neq \emptyset$.*

We now show that the above special case is NP-hard. The proof of this lemma is based on a reduction from the ONE-IN-THREE 3SAT problem, a variation of the more common 3SAT problem in which an acceptable truth assignment must satisfy exactly one literal in each clause. Briefly, the reduction introduces a model fragment for each literal in an instance of ONE-IN-THREE 3SAT, with model fragments corresponding to complementary literals being placed in the same assumption class. The mapping between truth assignments and models is straightforward: a literal is true if and only if the corresponding model fragment is in the model. Equations are assigned to model fragments to ensure that a model is a causal model if and only if the corresponding truth assignment assigns exactly one true literal to each clause.

**Lemma 4.2.2.** *The* SELECT MODEL FRAGMENTS *problem is NP-hard.*

*Proof.* To show that the SELECT MODEL FRAGMENTS problem is NP-hard, we reduce an arbitrary instance of the ONE-IN-THREE 3SAT problem to an instance of the SELECT MODEL FRAGMENTS problem. An instance of the ONE-IN-THREE 3SAT problem is defined as follows:

**Definition 4.7** (ONE-IN-THREE 3SAT). *Let $U = \{u_1, \ldots, u_n\}$ be a set of $n$ boolean variables, and $C = \{c_1, \ldots, c_m\}$ a set of $m$ clauses over $U$, such that each clause $c_i \in C, 1 \leq i \leq m$, has $|c_i| = 3$. Is there a truth assignment for $U$ such that each clause in $C$ has exactly one true literal?*

The ONE-IN-THREE 3SAT problem is shown to be NP-complete in [Schaefer, 1978]. We now reduce an arbitrary instance

$$\mathcal{I}_1 = (U, C)$$

of the ONE-IN-THREE 3SAT problem to an instance

$$\mathcal{I}_2 = (\mathcal{M}, \mathit{contradictory}, \mathit{approximation}, \mathcal{A}, \emptyset, p, q)$$

of the SELECT MODEL FRAGMENTS problem as follows.

Introduce a model fragment $m_l$ for each literal $l$ in $\mathcal{I}_1$, and a model fragment $m$:

$$\mathcal{M} = \{m_{u_i} : 1 \leq i \leq n\} \cup \{m_{\bar{u}_i} : 1 \leq i \leq n\} \cup \{m\}$$

Let $m_l$ and $m_{\bar{l}}$ be *contradictory*, where $l$ and $\bar{l}$ are complementary literals:

$$\mathit{contradictory}(m_{u_i}, m_{\bar{u}_i}), \text{ for } 1 \leq i \leq n$$

Note that *contradictory* partitions $\mathcal{M}$ into a set of mutually consistent assumption classes, with $m$ being in its own assumption class. This defines $\mathcal{A}$, the set of assumption classes:

$$\mathcal{A} = \{\{m_{u_i}, m_{\bar{u}_i}\} : 1 \leq i \leq n\} \cup \{\{m\}\}$$

Let *approximation* be the empty relation, so that no model fragment is an *approximation* of any other model fragment, let $\mathcal{C} = \emptyset$.

Introduce the set $\mathcal{P}$ of $(m+n+3)$ parameters:

$$\mathcal{P} = \{P_0, P_1, \ldots, P_{m+n+2}\}$$

We let $p = P_0$ and $q = P_{m+n+2}$. Next, we introduce the set $E$ of $(3m+2n+3)$ equations:[4]

$$E = (\bigcup_{1 \le j \le m} E_j) \cup (\bigcup_{1 \le i \le n} F_i) \cup G$$

where $E_j$ contains an equation for each literal in clause $c_j$, $F_i$ contains an equation for literals $u_i$ and $\bar{u}_i$, and $G$ contains three equations, as follows:

$$\begin{aligned} E_j &= \{e_{jl} : l \text{ is a literal in clause } c_j\} \\ F_i &= \{f_{u_i}, f_{\bar{u}_i}\} \\ G &= \{g_1, g_2, g_3\} \end{aligned}$$

The parameters of the equations in $E$ are defined as follows:

$$\text{If } e \in E, \text{ then } P(e) = \begin{cases} \{P_j, P_{j+1}\} & \text{If } e \in E_j, 1 \le j \le m \\ \{P_{m+i}, P_{m+i+1}\} & \text{If } e \in F_i, 1 \le i \le n \\ \{P_0\} & \text{If } e = g_1 \\ \{P_0, P_1\} & \text{If } e = g_2 \\ \{P_{m+n+1}, P_{m+n+2}\} & \text{If } e = g_3 \end{cases}$$

For each $e \in E$, let $P_c(e) = P(e)$. The equations in the model fragments of $\mathcal{M}$ are defined as follows:

$$\begin{aligned} E(m_l) &= \{e_{jl} : \text{literal } l \text{ is in clause } c_j\} \cup \{f_l\} \\ E(m) &= \{g_1, g_2, g_3\} \end{aligned}$$

That completes the reduction. Clearly, the reduction can be done in polynomial time. We now show that $\mathcal{I}_2$ is, indeed, an instance of the SELECT MODEL FRAGMENTS problem. Since $\mathcal{C} = \emptyset$ in $\mathcal{I}_2$, we need only show that every causal model of $\mathcal{I}_2$ contains a model fragment from each assumption class.

Let $M$ be any causal model of $\mathcal{I}_2$. We first show that $P(M) = \mathcal{P}$. If $P(M) \neq \mathcal{P}$, then there exists some parameter $P_k \in \mathcal{P}$, $1 \le k \le (m+n+1)$, with $P_k \notin P(M)$ ($P_0$ and $P_{m+n+2}$ must, of course, be in $P(M)$). Since each equation in $E$, except $g_1$, relates two parameters whose subscripts differ by 1, it follows that no equation relates parameters with subscripts less than $k$ to parameters with subscripts greater than $k$. Hence, no parameter with subscript less than $k$ can be related to any parameter with subscript greater than $k$, and hence $P_0$ and $P_{m+n+2}$ are unrelated, contradicting the fact that $M$ is a causal model. Hence $P(M) = \mathcal{P}$.

Next, we show that $E(M)$ contains exactly one equation from each $E_j$, $1 \le j \le m$, exactly one equation from each $F_i$, $1 \le i \le n$, and the three

[4] The equations that we introduce in this proof, and in all the other proofs in this chapter, will not contain any differential equations. Hence, $E = ic(E)$.

equations in $G$. First, we show that $E(M)$ must contain at least one equation from $E_j$, $1 \leq j \leq m$. If $E(M)$ contains no equation from $E_j$ for some $j$, $1 \leq j \leq m$, then $E(M)$ contains no equation that relates a parameter with subscript less than or equal to $j$ to a parameter with subscript greater than or equal to $j+1$. This follows from the facts that all equations, except $g_1$, relate two parameters whose subscripts differ by 1, and the only equations that relate $P_j$ to $P_{j+1}$ are found in $E_j$. Hence, $P_0$ and $P_{m+n+2}$ are unrelated, violating the fact that $M$ is a causal model. Hence, $E(M)$ contains at least one equation from each $E_j$, $1 \leq j \leq m$. A similar argument shows that $E(M)$ contains at least one equation from each $F_i$, $1 \leq i \leq n$, and that $E(M)$ must contain $g_2$ and $g_3$ (and hence $g_1$). Hence $E(M)$ contains at least $(m+n+3)$ equations. But since $M$ is complete, $|E(M)| = |P(M)| = (m+n+3)$, and hence $E(M)$ contains exactly one equation from each $E_j$, $1 \leq j \leq m$, exactly one equation from each $F_i$, $1 \leq i \leq n$, and the three equations in $G$.

Recall that the assumption classes of $\mathcal{I}_2$ are the following:

$$\mathcal{A} = \bigcup_{1 \leq i \leq n} \{m_{u_i}, m_{\bar{u_i}}\} \cup \{m\}$$

Since $M$ contains the three equations in $G$, $M$ contains the model fragment $m$. Now we show that $M$ contains a model fragment from each of the other assumption classes, i.e., for each $i$, $1 \leq i \leq n$, $M$ contains one of $m_{u_i}$ or $m_{\bar{u_i}}$. Since $M$ is consistent, at most one of $m_{u_i}$ and $m_{\bar{u_i}}$ is in $M$. Since $E(M)$ contains an equation from $F_i$, at least one of $m_{u_i}$ and $m_{\bar{u_i}}$ is in $M$. Hence, exactly one of $m_{u_i}$ and $m_{\bar{u_i}}$ is in $M$. Hence, $\mathcal{I}_2$ is indeed an instance of the SELECT MODEL FRAGMENTS problem.

We now show that $\mathcal{I}_1$ has an acceptable truth assignment if and only if $\mathcal{I}_2$ has a causal model.

($\Rightarrow$) Suppose that $\mathcal{I}_1$ has an acceptable truth assignment on $U$. Let $M$ be the following model:

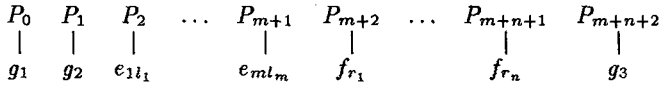
$$M = \{m_{u_i} : u_i \text{ is } \textit{true}\} \cup \{m_{\bar{u_i}} : u_i \text{ is } \textit{false}\} \cup \{m\}$$

We claim that $M$ is a causal model. First, $M$ contains no mutually contradictory model fragments, because if $M$ did contain mutually contradictory model fragments, then for some $i, 1 \leq i \leq n$, we have $m_{u_i}, m_{\bar{u_i}} \in M$. But this means that $u_i$ is both *true* and *false*, which is impossible.

To show that the set of equations of $M$ is complete, it suffices to show, by Lemma 3.3.2, that there is an onto causal mapping from $E(M)$ to $P(M)$. We start by claiming that $E(M)$ contains exactly $(m+n+3)$ equations, one from each $E_j, 1 \leq j \leq m$, one from each $F_i, 1 \leq i \leq n$, and the three equations in $G$. Since $m \in M$, it follows that $g_1, g_2, g_3 \in E(M)$. Since $u_i$, $1 \leq i \leq n$, is either *true* or *false*, $M$ contains exactly one of $m_{u_i}$ or $m_{\bar{u_i}}$, and hence $E(M)$ contains exactly one of $f_{u_i}$ or $f_{\bar{u_i}}$. Hence, $E(M)$ contains exactly one equation from each $F_i, 1 \leq i \leq n$. Finally, let $l_j$ be the single true literal in clause $c_j$, $1 \leq j \leq m$. This means that $m_{l_j} \in M$ and hence $e_{jl_j} \in E(M)$. Note that if $l'_j$ is a literal in $c_j$ that is not true, it follows that $m_{l'_j} \notin M$ and

hence $e_{jl'_j} \notin E(M)$. Hence, $E(M)$ contains exactly one equation from each $E_j, 1 \leq j \leq m$. Hence, $E(M)$ contains exactly $(m+n+3)$ equations.

We now create a 1-1 mapping from the $(m+n+3)$ equations of $E(M)$ to the parameters of $\mathcal{P}$. Note that if such a mapping is possible, $|P(M)| = (m+n+3)$, and hence $P(M) = \mathcal{P}$, in which case the mapping must be an onto mapping, and we are done. Map $g_1$ to $P_0$, $g_2$ to $P_1$, and $g_3$ to $P_{m+n+2}$. Map the representative of $E_j$ to $P_{j+1}$, $1 \leq j \leq m$. Map the representative of $F_i$ to $P_{m+i+1}$, $1 \leq i \leq n$. It is easy to verify that this mapping is a valid 1-1 mapping. Figure 4.1 shows this matching. Hence, $M$ is complete.




$l_j$ is the true literal in clause $c_j$, $1 \leq j \leq m$
$r_i$ is $u_i$ if $u_i$ is true, and $\bar{u}_i$ otherwise, $1 \leq i \leq n$

**Fig. 4.1.** Mapping of parameters to equations

Finally, we show that $M$ satisfies the expected behavior. The mapping in Figure 4.1 shows that (a) $P_1$ depends on $P_0$ because $P_1$ is matched to $g_2$ and $g_2$ uses $P_0$; (b) for each $j, 1 \leq j \leq m$, $P_{j+1}$ depends on $P_j$ because $P_{j+1}$ is mapped to $e_{jl_j}$ which uses $P_j$; (c) for each $i, 1 \leq i \leq n$, $P_{m+i+1}$ depends on $P_{m+i}$ because $P_{m+i+1}$ is mapped to $f_{r_i}$ which uses $P_{m+i}$; and (d) $P_{m+n+2}$ depends on $P_{m+n+1}$ because $P_{m+n+2}$ is matched to $g_3$ which uses $P_{m+n+1}$. We can put all these dependency links together to infer that $P_{m+n+2}$ $(= q)$ causally depends on $P_0$ $(= p)$, and hence $M$ satisfies the expected behavior. Hence, we have proved that $M$ is a causal model.

$(\Leftarrow)$ We now prove that if a causal model exists in $\mathcal{I}_2$, then there is an acceptable truth assignment in $\mathcal{I}_1$. Let $M$ be any causal model of $\mathcal{I}_2$. We use $M$ to construct an acceptable truth assignment for $\mathcal{I}_1$ as follows: for each $u_i \in U, 1 \leq i \leq n$, let $u_i$ be *true* if $m_{u_i} \in M$, and $u_i$ be *false* if $m_{\bar{u}_i} \in M$. This truth assignment assigns a unique truth value to each $u_i \in U, 1 \leq i \leq n$, since we saw earlier that $M$ contains exactly one of $m_{u_i}$ and $m_{\bar{u}_i}$.

Next, we prove that each clause $c_j \in C, 1 \leq j \leq m$, has exactly one true literal. To prove this, we prove that a literal $l$ in clause $c_j$ is true if and only if equation $e_{jl} \in E(M)$. If $e_{jl} \in E(M)$, it follows that $m_l \in M$, and hence $l$ is true. On the other hand, if $l$ is true, then $m_l \in M$, and hence $e_{jl} \in E(M)$. However, we know that $E(M)$ contains exactly one equation from $E_j$, and hence exactly one literal in $c_j$ is true. Hence, the truth assignment is acceptable.

Hence, we have shown that $\mathcal{I}_1$ has an acceptable truth assignment if and only if $\mathcal{I}_2$ has a causal model. Hence, the SELECT MODEL FRAGMENTS problem is NP-hard.

### 4.2.4 The SELECT ASSUMPTION CLASSES problem

The second special case of the CAUSAL MODEL problem consists of those instances of the problem that satisfy the following two conditions: (a) the instance still has no propositional coherence constraints; and (b) each assumption class has exactly one model fragment. Since each assumption class contains exactly one model fragment, any causal model can be viewed as merely selecting a set of assumption classes. Hence, this special case identifies the second source of intractability: deciding which assumption classes to select is intractable. More abstractly, deciding which phenomena we want to model is itself intractable.

**Definition 4.8** (SELECT ASSUMPTION CLASSES). *This is the special case of the* CAUSAL MODEL *problem which includes exactly those instances of the* CAUSAL MODEL *problem in which (a) the contradictory relation partitions the set $\mathcal{M}$ of model fragments into the set $\mathcal{A}$ of assumption classes; (b) $\mathcal{C} = \emptyset$; and (c) every assumption class in $\mathcal{A}$ contains exactly one model fragment, i.e., the contradictory relation is the empty relation.*

We now show that the SELECT ASSUMPTION CLASSES problem is NP-hard. The proof is a minor variation of the proof of Lemma 4.2.2.

**Lemma 4.2.3.** *The* SELECT ASSUMPTION CLASSES *problem is NP-hard.*

*Proof.* The proof is a minor variation of the proof of Lemma 4.2.2. In the proof of Lemma 4.2.2, we reduced an arbitrary instance $\mathcal{I}_1$ of the ONE-IN-THREE 3SAT problem to an instance $\mathcal{I}_2$ of the SELECT MODEL FRAGMENTS problem. Here we reduce $\mathcal{I}_1$ to an instance $\mathcal{I}_2'$ of the SELECT ASSUMPTION CLASSES problem. $\mathcal{I}_2'$ is the same as $\mathcal{I}_2$, except for the following differences.

In $\mathcal{I}_2$, model fragments corresponding to complementary literals were made mutually *contradictory*. In contrast, in $\mathcal{I}_2'$, we make the *contradictory* relation the empty relation, i.e., there are no mutually *contradictory* model fragments. Hence, each assumption class in $\mathcal{I}_2'$ contains exactly one model fragment.

The second difference is that, in $\mathcal{I}_2'$, we add an equation to each model fragment as follows. In particular, we introduce $n$ new parameters:

$$\mathcal{Q} = \{q_1, q_2, \ldots, q_n\}$$

and $2n$ new equations:

$$H = \{h_1, h_2, \ldots, h_n\} \cup \{\bar{h_1}, \bar{h_2}, \ldots, \bar{h_n}\}$$

The parameters of the equations in $H$ are defined as follows:

$$P(h_i) = P_c(h_i) = P(\bar{h_i}) = P_c(\bar{h_i}) = \{q_i\}, \text{ for } 1 \leq i \leq n$$

i.e., both $h_i$ and $\bar{h_i}$ use only the parameter $q_i$.

We add the equation $h_i$ to model fragment $m_{u_i}$, and the equation $\bar{h_i}$ to model fragment $m_{\bar{u_i}}$, for $1 \leq i \leq n$. As a consequence, no consistent model of

$\mathcal{I}_2'$ can include both $m_{u_i}$ and $m_{\bar{u_i}}$. This is because the equations of a model that includes both $m_{u_i}$ and $m_{\bar{u_i}}$ would include equations $h_i$ and $\bar{h_i}$, and the equations would be overconstrained (see Definition 3.4 with $S = \{h_i, \bar{h_i}\}$.)

Hence, effectively, model fragments $m_{u_i}$ and $m_{\bar{u_i}}$ behave as though they were *contradictory*. Hence, $\mathcal{I}_2$ and $\mathcal{I}_2'$ have the same causal models. Hence, $\mathcal{I}_1$ has an acceptable truth assignment if and only if $\mathcal{I}_2'$ has a causal model. Hence, the SELECT ASSUMPTION CLASSES problem is NP-hard.

Another view of the results of Lemmas 4.2.2 and 4.2.3 is that the fundamental source of intractability is that a causal model must choose at most one of the model fragments $m_{u_i}$ and $m_{\bar{u_i}}$: in Lemma 4.2.2, the choice is enforced by making $m_{u_i}$ and $m_{\bar{u_i}}$ mutually *contradictory*; in Lemma 4.2.3, the choice is enforced by assigning equations to $m_{u_i}$ and $m_{\bar{u_i}}$ such that a model that contains both of them becomes overconstrained. This suggests that other ways of enforcing such a choice would also lead to intractability. In particular, if $\mathcal{C}$ contained constraints of the form

$$\neg m_{u_i} \vee \neg m_{\bar{u_i}} \tag{4.11}$$

then any coherent model would have to choose at most one of $m_{u_i}$ and $m_{\bar{u_i}}$, leading to intractability. Hence, allowing propositional coherence constraints of the above form (also called negative clauses) is yet another source of intractability. In the next section, we show that other very simple types of propositional coherence constraints can lead to intractability.

### 4.2.5 The SATISFY CONSTRAINTS problem

The third special case of the CAUSAL MODEL problem consists of those instances of the problem that satisfy the following two conditions: (a) as in the first case, every causal model of the instance includes a model fragment from each assumption class; (b) model fragments in the same assumption class have the same sets of equations; and (c) $\mathcal{C}$ contains only definite horn clauses (a definite horn clause is a disjunction of literals with exactly one positive literal). Conditions (a) and (b) ensure that, if $\mathcal{C}$ were empty, then finding a causal model would be trivial: a causal model exists if and only if selecting an arbitrary model fragment from each assumption class leads to a causal model. Hence, the intractability of this problem stems from the causal model having to satisfy the constraints in $\mathcal{C}$, even when the constraints are restricted to be definite horn clauses.

**Definition 4.9** (SATISFY CONSTRAINTS). *This problem is the special case of the*
CAUSAL MODEL *problem which includes exactly those instances of the* CAUSAL MODEL *problem in which (a) the contradictory relation partitions the set $\mathcal{M}$ of model fragments into the set $\mathcal{A}$ of assumption classes; (b) every causal model of the instance includes a model fragment from each assumption*

*class; (c) model fragments in the same assumption class have the same sets of equations; and (d) $\mathcal{C}$ contains only definite horn clauses.*

We now show that the SATISFY CONSTRAINTS problem is NP-hard.

**Lemma 4.2.4.** *The* SATISFY CONSTRAINTS *problem is NP-hard.*

*Proof.* Once again the proof is based on a reduction from the ONE-IN-THREE 3SAT problem, i.e., we will reduce an arbitrary instance

$$\mathcal{I}_1 = (U, C)$$

of the ONE-IN-THREE 3SAT problem (see Definition 4.7) to an instance

$$\mathcal{I}_2 = (\mathcal{M}, \mathit{contradictory}, \mathit{approximation}, \mathcal{A}, \mathcal{C}, p, q)$$

of the SATISFY CONSTRAINTS problem. Introduce a model fragment $m_l$ for each literal $l$ in $\mathcal{I}_1$:

$$\mathcal{M} = \{m_{u_i} : 1 \leq i \leq n\} \cup \{m_{\bar{u_i}} : 1 \leq i \leq n\}$$

Let $m_l$ and $m_{\bar{l}}$ be *contradictory*, where $l$ and $\bar{l}$ are complementary literals:

$$\mathit{contradictory}(m_{u_i}, m_{\bar{u_i}}), \text{for } 1 \leq i \leq n$$

Note that *contradictory* partitions $\mathcal{M}$ into a set $\mathcal{A}$ of assumption classes:

$$\mathcal{A} = \{\{m_{u_i}, m_{\bar{u_i}}\} : 1 \leq i \leq n\}$$

Let *approximation* be the empty relation. Introduce the set $\mathcal{P}$ of $(n+1)$ parameters:

$$\mathcal{P} = \{P_0, P_1, \ldots, P_n\}$$

Let $p = P_0$, and $q = P_n$. Next, introduce the set $E$ of $n+1$ equations:

$$E = \{e_0, e_1, e_2, \ldots, e_n\}$$

The parameters of these equations are defined as follows:

$$P(e_0) = \{P_0\}$$
$$P(e_i) = P_c(e_i) = \{P_{i-1}, P_i\} \quad \text{for } 1 \leq i \leq n$$

i.e., each equation (except $e_0$) relates a pair of consecutively numbered parameters. Assign the equations to the model fragments as follows:

$$E(m_{u_1}) = E(m_{\bar{u_1}}) = \{e_0, e_1\}$$
$$E(m_{u_i}) = E(m_{\bar{u_i}}) = \{e_i\} \qquad \text{for } 2 \leq i \leq n$$

Note that model fragments in the same assumption class are assigned the same set of equations. Finally, we introduce the set $\mathcal{C}$ of $3m$ propositional coherence constraints. $\mathcal{C}$ will contain 3 constraints from each clause in $C$. Let us assume that the three literals in clause $c_j$ are named $l_{j1}$, $l_{j2}$, and $l_{j3}$, $1 \leq j \leq m$. The $3m$ constraints in $\mathcal{C}$ are defined as follows:

$$\mathcal{C} = \bigcup_{1 \le j \le m} \{ \begin{array}{l} (m_{l_{j1}^-} \wedge m_{l_{j2}^-}) \equiv m_{l_{j3}}, \\ (m_{l_{j1}^-} \wedge m_{l_{j3}^-}) \equiv m_{l_{j2}}, \\ (m_{l_{j2}^-} \wedge m_{l_{j3}^-}) \equiv m_{l_{j1}} \} \end{array} \tag{4.12}$$

where the literal $\bar{l}$ is the complement of the literal $l$, i.e., if $l$ is $\bar{u_i}$ then $\bar{l}$ is $u_i$ and vice versa. Note that a constraint of the form

$$(m_1 \wedge m_2) \equiv m_3$$

is equivalent to the following three definite horn clauses:

$$\neg m_1 \vee \neg m_2 \vee m_3$$
$$\neg m_3 \vee m_1$$
$$\neg m_3 \vee m_2$$

That completes the reduction. Clearly, the reduction can be done in polynomial time.

We now show that $\mathcal{I}_2$ is, in fact, an instance of the SATISFY CONSTRAINTS problem. Conditions (a), (c), and (d) in Definition 4.9 are straightforward to verify. Hence, we only need to show that every causal model of $\mathcal{I}_2$ contains a model fragment from each assumption class in $\mathcal{A}$. Let $M \subseteq \mathcal{M}$ be a causal model $\mathcal{I}_2$.

First, we show that $P(M) = \mathcal{P}$, i.e., the equations of $M$ contain all the parameters in $\mathcal{P}$. Since $M$ is a causal model, $P(M)$ must contain $P_0$ $(= p)$ and $P_n$ $(= q)$. Suppose $P(M)$ does not contain $P_i$, $0 < i < n$. Since equations in the model fragments of $\mathcal{M}$ relate only consecutively numbered parameters (except $e_0$ which contains only one parameter), it follows that no equation relates parameters numbered less than $i$ to parameters numbered greater than $i$. Hence, $P_0$ and $P_n$ are unrelated, and hence $M$ is not a causal model. Hence, $P(M)$ contains all the parameters in $\mathcal{P}$.

Since $P(M) = \mathcal{P}$, it follows that $|P(M)| = (n + 1)$. Since $M$ is complete, it follows that $|E(M)| = (n+1)$. However, note that each model fragment has exactly one equation, except $m_{u_1}$ and $m_{\bar{u_1}}$ which have two. Since $m_{u_1}$ and $m_{\bar{u_1}}$ are in the same assumption class, it follows that the only way $E(M)$ can have $(n + 1)$ equations is if $M$ contains a model fragment from each assumption class. Hence, every causal model of $\mathcal{I}_2$ contains a model fragment from each assumption class. Hence, $\mathcal{I}_2$ is an instance of the SATISFY CONSTRAINTS problem problem.

Now we show that $\mathcal{I}_1$ has an acceptable truth assignment if and only if $\mathcal{I}_2$ has a causal model.

($\Rightarrow$) Suppose that $\mathcal{I}_1$ has an acceptable truth assignment on $U$. Let $M$ be the following model:

$$M = \{m_{u_i} : u_i \text{ is } \textit{true}\} \cup \{m_{\bar{u_i}} : u_i \text{ is } \textit{false}\}$$

We claim that $M$ is a causal model. First, $M$ contains no mutually *contradictory* model fragments, because if $M$ did contain mutually *contradictory*

model fragments, then for some $i, 1 \leq i \leq n$, we have $m_{u_i}, m_{\bar{u_i}} \in M$. But this means that $u_i$ is both *true* and *false*, which is impossible.

Next, we show that $M$ contains a model fragment from each assumption class. This is a direct consequence of the fact that each $u_i, 1 \leq i \leq n$, is either *true* or *false*. Hence, either $m_{u_i}$ or $m_{\bar{u_i}}$ is in $M$. Hence, $M$ contains a model fragment from each assumption class.

Since $M$ contains a model fragment from each assumption class, it follows that $E(M) = E$. It is easy to verify that $E$ is a complete set of equations, and that $P_n$ causally depends on $P_0$ in the causal ordering generated from $E$.

Finally, we show that all the constraints in $\mathcal{C}$ are satisfied. Since, for any literal $l$, $m_l$ and $m_{\bar{l}}$ are *contradictory*, it is easy to see that the constraints in $\mathcal{C}$ are satisfied by $M$ if only if $M$ contains exactly one model fragment from each of the following sets:

$$\{m_{l_{j1}}, m_{l_{j2}}, m_{l_{j3}}\}, 1 \leq j \leq m$$

where $l_{j1}$, $l_{j2}$, and $l_j$ are the three literals in clause $c_j$. For example, if $M$ does not contain $m_{l_{j1}}$ and $m_{l_{j2}}$, for some $1 \leq j \leq m$, then the constraint

$$(m_{l_{j1}^-} \wedge m_{l_{j2}^-}) \equiv m_{l_{j3}}$$

is satisfied if and only if $m_{l_{j3}} \in M$. Similarly, if $M$ contains $m_{l_{j3}}$ then the above constraint is satisfied if and only $M$ does not contain $m_{l_{j1}}$ and $m_{l_{j2}}$. Since we started with an acceptable truth assignment, it follows that exactly one of the literals in

$$\{l_{j1}, l_{j2}, l_{j3}\}$$

for each $1 \leq j \leq m$, is *true*. Hence, for each $1 \leq j \leq m$, exactly one model fragment in

$$\{m_{l_{j1}}, m_{l_{j2}}, m_{l_{j3}}\}$$

is in $M$. Hence, all the constraints in $\mathcal{C}$ are satisfied. Hence, $M$ is a causal model.

($\Leftarrow$) We now prove that if $\mathcal{I}_2$ has a causal model, then there is an acceptable truth assignment for $\mathcal{I}_1$. Let $M$ be a causal model of $\mathcal{I}_2$. We use $M$ to construct a truth assignment for $\mathcal{I}_1$ as follows: for each $u_i \in U, 1 \leq i \leq n$, let $u_i$ be *true* if $m_{u_i} \in M$, and $u_i$ be *false* if $m_{\bar{u_i}} \in M$. We now prove that this is an acceptable truth assignment for $\mathcal{I}_1$.

First, we note that the truth assignment assigns exactly one truth value to every variable in $U$. This is a direct consequence of the fact that $M$ contains a model fragment from each assumption class in $\mathcal{A}$ (shown earlier). Hence, $M$ contains exactly one of $m_{u_i}$ and $m_{\bar{u_i}}$, for every $1 \leq i \leq n$. Hence, $u_i$ is assigned either *true* or *false* for each $u_i \in U$.

Next, we show that the truth assignment is such that each clause $c_j \in C$ has exactly one *true* literal. Let $c_j = \{l_{j1}, l_{j2}, l_{j3}\}, 1 \leq j \leq m$. We have

already seen that the constraints in $\mathcal{C}$ have been constructed to ensure that every causal model selects exactly one model fragment from the set

$$\{m_{l_{j1}}, m_{l_{j2}}, m_{l_{j3}}\}$$

for each $1 \leq j \leq m$. Hence, the corresponding truth assignment assigns *true* to exactly one literal in each

$$\{l_{j1}, l_{j2}, l_{j3}\}$$

for $1 \leq j \leq m$. Hence, the truth assignment is an acceptable truth assignment. Hence, if $\mathcal{I}_2$ contains a causal model, $\mathcal{I}_1$ contains an acceptable truth assignment.

Hence, $\mathcal{I}_1$ contains an acceptable truth assignment if and only if $\mathcal{I}_2$ contains a causal model. Hence, the SATISFY CONSTRAINTS problem is NP-hard.

### 4.2.6 The intractability of finding causal models

An immediate consequence of the above three lemmas is the intractability of the CAUSAL MODEL problem.

**Theorem 4.2.1.** *The* CAUSAL MODEL *problem is NP-complete.*

*Proof.* Since the SELECT MODEL FRAGMENTS problem, the SELECT ASSUMPTION CLASSES problem, and the SATISFY CONSTRAINTS problem are all special cases of the CAUSAL MODEL problem, an immediate corollary of any one of the above three lemmas (Lemmas 4.2.2, 4.2.3, 4.2.4) is that the CAUSAL MODEL problem is NP-hard. In conjunction with Lemma 4.2.1, we can immediately infer that the CAUSAL MODEL problem is NP-complete.

An immediate consequence of the above theorem is that finding an adequate model is NP-hard.

**Theorem 4.2.2.** *The* MINIMAL CAUSAL MODEL *problem is NP-hard.*

*Proof.* Since the set of all models is finite, it follows that a minimal causal model exists if and only if a causal model exists. Hence, an algorithm for finding a minimal causal model can be used to decide whether or not there exists a causal model. Hence, the CAUSAL MODEL problem is Turing-reducible[5] to the MINIMAL CAUSAL MODEL problem. Since the CAUSAL MODEL problem is NP-complete, it follows that the MINIMAL CAUSAL MODEL problem is NP-hard.

[5] Informally, a problem $\Pi_1$ is Turing-reducible to a problem $\Pi_2$ if there exists an algorithm $A_1$ that solves $\Pi_1$ using a hypothesized algorithm $A_2$ for solving $\Pi_2$, such that $A_1$ is a polynomial time algorithm if and only if $A_2$ is a polynomial time algorithm [Garey and Johnson, 1979].

## 4.3 Other complexity results

In the previous section, we investigated the complexity of finding causal models. In this section we will briefly investigate the complexity of two other cases. The first case is a restriction of the CAUSAL MODEL problem in which each equation has exactly one causal orientation. This is an interesting case because it is the same restriction as the one used in QP Theory [Forbus, 1984] and its derivatives. The second case is a variation of the CAUSAL MODEL problem in which we do not require that models be able to explain the expected behavior, i.e., we look for coherent models, rather than causal models. This is interesting because, while we may not always be interested in causal models, we will certainly insist that device models be coherent.

We will show that both the above cases are intractable. Our proofs are based on on the proof of Lemma 4.2.2, but the proofs could also have been based on the proofs of Lemmas 4.2.3 and 4.2.4.

### 4.3.1 Fixed causal orientations

We now show that, even if each equation is restricted to have a single causal orientation, the problem of finding a causal model remains intractable.

**Definition 4.10** (FIXED ORIENTATION). *This problem is the special case of the* CAUSAL MODEL *problem in which each equation has a fixed causal orientation:*

$$\forall e \in E(\mathcal{M}) \; |P_c(e)| = 1$$

**Lemma 4.3.1.** *The* FIXED ORIENTATION *problem is NP-complete*

*Proof.* The proof is a minor variation of the proof of Lemma 4.2.2. In the proof of Lemma 4.2.2, we reduced an arbitrary instance $\mathcal{I}_1$ of the ONE-IN-THREE 3SAT problem to an instance $\mathcal{I}_2$ of the SELECT MODEL FRAGMENTS problem. Here we reduce $\mathcal{I}_1$ to an instance $\mathcal{I}_2'$ of the FIXED ORIENTATION problem. $\mathcal{I}_2'$ is the same as $\mathcal{I}_2$, except that we modify the definition of $P_c$ as follows.

In the proof to Lemma 4.2.2, we had made $P_c(e) = P(e)$, i.e., each equation could causally determine any parameter in that equation. Here we restrict $P_c(e)$ as follows:

$$\text{If } e \in E, \text{ then } P_c(e) = \begin{cases} \{P_{j+1}\} & \text{If } e \in E_j, 1 \leq j \leq m \\ \{P_{m+i+1}\} & \text{If } e \in F_i, 1 \leq i \leq n \\ \{P_0\} & \text{If } e = g_1 \\ \{P_1\} & \text{If } e = g_2 \\ \{P_{m+n+2}\} & \text{If } e = g_3 \end{cases}$$

i.e., each equation has exactly one causal orientation.

We now show that $\mathcal{I}_2$ and $\mathcal{I}_2'$ have the same causal models. It is easy to see that any causal model of $\mathcal{I}_2'$ is also a causal model of $\mathcal{I}_2$. Hence, we need only show that a causal model of $\mathcal{I}_2$ is also a causal model of $\mathcal{I}_2'$.

In the proof of Lemma 4.2.2, we made the following claim about all the causal models of $\mathcal{I}_2$:

> ...and hence $E(M)$ contains exactly one equation from each $E_j$, $1 \leq j \leq m$, exactly one equation from each $F_i$, $1 \leq i \leq n$, and the three equations in $G$.

Furthermore, when a model is of the above form, we constructed a causal mapping as follows:

> ...Map $g_1$ to $P_0$, $g_2$ to $P_1$, and $g_3$ to $P_{m+n+2}$. Map the representative of $E_j$ to $P_{j+1}$, $1 \leq j \leq m$. Map the representative of $F_i$ to $P_{m+i+1}$, $1 \leq i \leq n$.

It is easy to see that the above mapping can still be done when we restrict $P_c$ as shown above. Hence, any causal model in the original proof, remains a causal model even with the restriction on $P_c$. Hence, every causal model of $\mathcal{I}_2$ is a causal model of $\mathcal{I}_2'$.

Since $\mathcal{I}_1$ has an acceptable truth assignment if and only if $\mathcal{I}_2$ has a causal model, it follows that $\mathcal{I}_1$ has an acceptable truth assignment if and only if $\mathcal{I}_2'$ has a causal model. Hence, the FIXED ORIENTATION problem is NP-hard.

The FIXED ORIENTATION problem is in NP because the CAUSAL MODEL problem is in NP. Hence, the FIXED ORIENTATION problem is NP-complete.

The above lemma shows that the problem of finding adequate device models remains intractable even when each equation is restricted to have a single causal orientation. Since the above proof is based on the proof of Lemma 4.2.2, it means that, even with the restriction on the causal orientations of equations, selecting model fragments from selected assumption classes remains intractable. A similar proof, based on the proof of Lemma 4.2.3, shows that, even with the restriction on the causal orientations of the equations, selecting a set of assumption classes remains intractable.

### 4.3.2 Finding coherent models

We now show that deciding whether or not there exists a coherent model, rather than a causal model, is also NP-complete.

**Definition 4.11** (COHERENT MODEL). *Let the input to the* COHERENT MODEL *problem be the tuple* $\mathcal{I}$*:*

$$\mathcal{I} = (\mathcal{M}, \textit{contradictory}, \textit{approximation}, \mathcal{A}, \mathcal{C})$$

*where the elements of the tuple are as in Equation 4.1. Does there exist a non-empty coherent model with respect to* $\mathcal{I}$*?*

**Lemma 4.3.2.** *The* COHERENT MODEL *problem is NP-complete.*

*Proof.* The proof is a variation of the proof of Lemma 4.2.2. In the proof of Lemma 4.2.2, we reduced an arbitrary instance $\mathcal{I}_1$ of the ONE-IN-THREE 3SAT problem to an instance $\mathcal{I}_2$ of the SELECT MODEL FRAGMENTS problem. Here we reduce $\mathcal{I}_1$ to an instance $\mathcal{I}_2'$ of the COHERENT MODEL problem. $\mathcal{I}_2'$ is the same as $\mathcal{I}_2$, except for some modifications. The net result of these modifications will be that every coherent model of $\mathcal{I}_2'$ will be a causal model of $\mathcal{I}_2$, and vice versa. Hence, $\mathcal{I}_1$ will have an acceptable truth assignment if and only if $\mathcal{I}_2'$ has a coherent model.

The modifications we make are as follows. We introduce the set $Q$ of $m$ parameters:

$$Q = \{q_1, q_2, \ldots, q_m\}$$

We also introduce the set $H$ of $3m$ equations:

$$H = \{h_{jl} : l \text{ is a literal in clause } c_j\}$$

i.e., there is a new equation corresponding to each literal in each clause. The parameters of the equations are defined as follows:

$$P(h_{jl}) = P_c(h_{jl}) = \{q_j\}, 1 \leq i \leq m$$

i.e., the equations corresponding to clause $c_j$ can determine $q_j$. We assign these new equations to the model fragments in the same way that we assigned the equations in $E$. In particular, we add equation $h_{jl}$ to model fragment $m_l$. Hence, the equations of model fragment $m_l$ are:

$$\begin{aligned} E(m_l) \quad = \quad & \{e_{jl} : \text{literal } l \text{ is in clause } c_j\} \cup \{f_l\} \cup \\ & \{h_{jl} : \text{literal } l \text{ is in clause } c_j\} \end{aligned}$$

That concludes the modifications to be made to $\mathcal{I}_2$ to get $\mathcal{I}_2'$. We now show that every coherent model of $\mathcal{I}_2'$ is a causal model of $\mathcal{I}_2$, and vice versa.

First, we show that every coherent model of $\mathcal{I}_2'$ contains at most one equation from $E_j, 1 \leq j \leq m$, and at most one equation from $F_i, 1 \leq i \leq n$.

Because of the equations that we added to $\mathcal{I}_2'$, if literals $l_1$ and $l_2$ are in the same clause, say $c_j$, then it means that model fragments $m_{l_1}$ and $m_{l_2}$ cannot be used in the same consistent model. This follows from the fact that, if they are used in the same model, then the model's equations would include both $h_{jl_1}$ and $h_{jl_2}$. But it is easy to verify that $\{h_{jl_1}, h_{jl_2}\}$ is an overconstrained set of equations. Hence, the model is not consistent. An immediate consequence of this observation is that a consistent model can contain at most one equation from each $E_j$, $1 \leq j \leq m$.

Similarly, recall that the two equations in each $F_i$, $1 \leq i \leq n$, are assigned to *contradictory* model fragments. Hence, it follows that that any consistent model can contain at most one equation from each $F_i$, $1 \leq i \leq n$.

Now we show that any coherent model of $\mathcal{I}_2'$ must contain exactly one equation from each $E_j$, $1 \leq j \leq m$, exactly one equation from each $F_i$,

$1 \leq i \leq n$, and the equations $g_1, g_2, g_3$. Let $M$ be any coherent model of $\mathcal{I}_2'$. Consider the following causal mapping (based on the mapping shown in Figure 4.1). If $g_1 \in E(M)$, then map $g_1$ to $P_0$; If $g_2 \in E(M)$, then map $g_2$ to $P_1$; If $g_3 \in E(M)$, then map $g_3$ to $P_{m+n+2}$; if $E(M)$ contains an equation from $E_j, 1 \leq j \leq m$, then map that equation to $P_{j+1}$; if $E(M)$ contains an equation from $F_i, 1 \leq i \leq n$, then map that equation to $P_{m+i+1}$. The last two are possible because we know that any consistent model can contain at most one equation from each $E_j$ and at most one equation from each $F_i$. If each parameter in $\mathcal{P}$ is matched to an equation by the above mapping, then this is the same mapping as the one shown in Figure 4.1, and the coherent model is also a causal model of $\mathcal{I}_2$.

If, on the other hand, some parameter is not matched by the above mapping, then we show that $M$ is not complete, i.e., there is a parameter in $E(M)$ that is not matched. Let $P_i$ be the parameter with the largest subscript such that $P_i$ is not matched but $P_{i+1}$ is matched. Such a $P_i$ must exist, for if $P_{m+n+2}$ is the unmatched parameter with the largest subscript, then $P_1$ must also be unmatched (since $g_2$ and $g_3$ belong to the same model fragment). Hence, either none of the parameters are matched (in which case the model is empty), or there is a parameter between $P_1$ and $P_{m+n+2}$ which is matched, and hence the required $P_i$ must exist.

Given such a $P_i$, it is easy to check that $P_{i+1}$ must be matched to an equation, $e$, with parameters $P(e) = \{P_i, P_{i+1}\}$. Since $e \in E(M)$, it follows that $P_i \in P(M)$. Hence, since $P_i$ is unmatched, it follows that $E(M)$ is incomplete. Hence, we have shown that if $M$ is coherent, then all the parameters in $\mathcal{P}$ are matched, and hence the coherent model of $\mathcal{I}_2'$ is also a causal model of $\mathcal{I}_2$.

It is easy to see that every causal model of $\mathcal{I}_2$ is also a coherent model of $\mathcal{I}_2'$. Hence, a model is a causal model of $\mathcal{I}_2$ if and only if it is also a coherent model of $\mathcal{I}_2'$. Since $\mathcal{I}_1$ contains an acceptable truth assignment if and only if $\mathcal{I}_2$ contains a causal model, it follows that $\mathcal{I}_1$ contains an acceptable truth assignment if and only if $\mathcal{I}_2'$ contains a coherent model. Hence, the COHERENT MODEL problem is NP-hard.

The COHERENT MODEL problem is clearly in NP (the proof is similar to the proof that the CAUSAL MODEL problem is in NP). Hence, it follows that the COHERENT MODEL problem is NP-complete.

Since the above proof is based on the proof of Lemma 4.2.2, it identifies a source of intractability of the COHERENT MODEL problem: even if we are interested only in coherent models, choosing model fragments from selected assumption classes is intractable. Similar proofs, based on the proofs to Lemmas 4.2.3 and 4.2.4, can be used to identify the other sources of intractability of the COHERENT MODEL problem.

## 4.4 Summary

In this chapter we analyzed the complexity of the problem of finding adequate device models. We started by developing a formal statement of the problem. This development showed how the component library, consisting of model fragment classes and first-order constraints, can be converted into a set of model fragments, a set of relations between model fragments, and a set of propositional coherence constraints. This conversion is done with the help of the structural and behavioral contexts.

We then showed that the problem of finding adequate device models is intractable. We gave three different proofs for this result, which helped us identify three different sources of the intractability. Informally, intractability arises in (a) deciding what phenomena to model; (b) deciding how to model selected phenomena; and (c) satisfying any domain-dependent constraints.

We then showed that certain related problems are also intractable. We first showed that, even if equations are restricted to have fixed causal orientations, the problem of finding adequate device models remains intractable. We also showed that the problem of finding coherent models, rather than causal models, is also intractable.

# 5. Causal approximations

In the previous chapter we showed that the problem of finding adequate device models (the MINIMAL CAUSAL MODEL problem) is intractable. This means that there is no efficient, polynomial time algorithm for finding adequate models—any algorithm for finding such models will, in the worst case, take an exponential amount of time. To put it another way, any algorithm for finding adequate models will be forced to search a significant portion of the exponentially large space of possible device models. Unfortunately, even for fairly simple devices, the space of of possible models is prohibitively large. Searching any significant portion of such a huge space is unthinkable.

However, the apparent intractability of finding adequate models seems to directly contradict the informal observation that trained engineers are remarkably good at providing parsimonious causal explanations for phenomena. One way to resolve this apparent contradiction is to assume that trained engineers are not solving the general MINIMAL CAUSAL MODEL problem. Rather, the problem instances that they normally encounter are drawn from a subclass of the MINIMAL CAUSAL MODEL problem which can, in fact, be solved efficiently. In this chapter, we identify such an efficiently solvable subclass. We believe that commonly encountered instances of the MINIMAL CAUSAL MODEL problem are, in fact, drawn from this subclass.

Since this chapter is quite long, we give a detailed road map of its sections. Section 5.1 introduces the basic idea underlying the efficiently solvable subclass of the MINIMAL CAUSAL MODEL problem. It introduces the upward failure property, and shows that if an instance of the MINIMAL CAUSAL MODEL problem satisfies the upward failure property, and if the immediate simplifications of a coherent model can be generated in polynomial time, then a minimal causal model can be found in polynomial time. Unfortunately, in general, it is difficult to decide whether or not the upward failure property is satisfied, and whether or not coherent models have a polynomial number of immediate simplifications. Hence, the rest of the chapter focuses on finding efficient characterizations of these properties.

Section 5.2 introduces a set of preliminary restrictions on the MINIMAL CAUSAL MODEL problem. Sections 5.3 and 5.4 introduce a set of restrictions on the MINIMAL CAUSAL MODEL problem that ensure that the upward failure property is satisfied. Section 5.3 introduces a special class of

approximations, called *causal approximations*, and shows that when all the approximations are causal approximations, the causal relations entailed by a model decrease monotonically as model fragments are replaced by their approximations. Section 5.4 generalizes these results to the case in which models are also simplified by dropping model fragments.

Section 5.5 and 5.6 focus on the problem of efficiently generating the immediate simplifications of a coherent model. Section 5.5 shows that the model fragments of a coherent model can be approximated one at a time. Section 5.6 introduces a syntactic restriction on the expressive power of the propositional coherence constraints in $\mathcal{C}$. This restriction ensures that models can be efficiently simplified.

Finally, Section 5.7 puts all the restrictions together, and presents an efficient algorithm for finding a minimal causal model. Throughout the chapter, we will also discuss the reasonableness of the restrictions.

The discussion in this chapter is restricted to models that do not contain differential equations. Differential equations are discussed in Chapter 6. In the rest of this chapter, we follow the terminology introduced in the previous chapter and let the tuple

$$\mathcal{I} = (\mathcal{M}, \mathit{contradictory}, \mathit{approximation}, \mathcal{A}, \mathcal{C}, p, q) \tag{5.1}$$

be an arbitrary instance of the MINIMAL CAUSAL MODEL problem, where the elements of the tuple are as in Equation 4.1.

## 5.1 Upward failure property

Intuitively, the reason that the MINIMAL CAUSAL MODEL problem is intractable seems to be that knowing whether a particular model is, or is not, a causal model tells us very little about which other models are, or are not, causal models. This means that there is no "clever" way to organize the search for adequate models, that allows us to rule out "large" parts of the search space by explicitly checking only a "small" part of the search space. With this intuition in mind, we introduce the upward failure property.

The upward failure property is based on the intuition that if a model is unable to explain the phenomenon of interest, there is little reason to believe that a simpler model will be able to explain that phenomenon. We make this precise with the following definition, which is similar in spirit to the one given in [Weld and Addanki, 1991]:

**Definition 5.1 (Upward failure property).** *An instance $\mathcal{I}$ of the* MINIMAL CAUSAL MODEL *problem is said to satisfy the upward failure property if and only if for all coherent models $M \subseteq \mathcal{M}$, if $M$ is not a causal model, then no strictly simpler model is a causal model, i.e., no model $M' \subseteq \mathcal{M}$ and $M' < M$ is a causal model.*

In essence, the upward failure property property says that the simpler the model, the less it can explain. Of course, it is by no means obvious that simpler models explain fewer phenomena. However, it does seem to be standard engineering practice that models that account for more phenomena are more complex by our definition, i.e., modeling more phenomena more accurately leads to models that can explain more. This is, of course, not an argument for claiming that the upward failure property is satisfied by all commonly encountered instances of the MINIMAL CAUSAL MODEL problem. Rather, it merely provides a motivation for our definition of the upward failure property.

### 5.1.1 Efficient model selection algorithm

Earlier we said that the reason that the MINIMAL CAUSAL MODEL problem is intractable seems to be that knowing whether a particular model is, or is not, a causal model tells us very little about which other models are, or are not, causal models. The upward failure property addresses exactly this problem: knowing that a coherent model is not a causal model allows us to rule out all simpler models as possibly adequate models. Hence, we can exploit this property to develop an efficient, polynomial time algorithm for solving instances of the MINIMAL CAUSAL MODEL problem that satisfy the upward failure property. The algorithm we develop has two parts: (a) finding an initial causal model; and (b) finding an adequate model by simplifying the initial causal model. We start by discussing how a causal model can be simplified, and then discuss how we find an initial causal model.

**Simplifying a model.** A causal model can be simplified to a minimal causal model using the function *find-minimal-causal-model* shown in Figure 5.1. This function takes two arguments: (a) $\mathcal{I}$, an instance of MINIMAL CAUSAL MODEL; and (b) a coherent model $M$. It returns an adequate model (i.e., a minimal causal model) that is simpler than $M$. If there is more than one such adequate model, it returns the first one it finds. If no such model exists, it returns nil.

The *simplifications* function, used in *find-minimal-causal-model*, when applied to a coherent model $M$, returns the set of coherent models that are immediate simplifications of $M$. A coherent model $M'$ is an immediate simplification of $M$ if and only if $M' < M$ and there does not exist a coherent model $M''$ such that $M' < M'' < M$.

$$\begin{aligned} &\textit{simplifications}(M, \mathcal{I}) = \\ &\quad \{M' : \ M' \text{ is coherent wrt } \mathcal{I} \\ &\qquad \wedge\, M' < M \\ &\qquad \wedge\, (\forall M'')\ M' < M'' < M \Rightarrow M'' \text{ is not coherent wrt } \mathcal{I}\} \end{aligned} \tag{5.2}$$

*Find-minimal-causal-model*$(M, \mathcal{I})$ works by systematically searching the immediate simplifications of $M$, until it finds a causal model $M'$ such that all the

```
function find-minimal-causal-model(M,I)
  /* I is assumed to satisfy the upward failure property */
  /* M is assumed to be coherent */
  if M is not a causal model then
    /* Since no simpler model can be a causal model */
    return nil
  else
    for each M' ∈ simplifications(M,I) do
      result := find-minimal-causal-model(M',I)
      if result ≠ nil then
        /* A simpler causal model has been found */
        return result
      endif
    endfor
    /* No simplification is a causal model, but M is */
    return M
  endif
end
```

**Fig. 5.1.** Function *find-minimal-causal-model*

immediate simplifications of $M'$ causal models. The upward failure property then assures us that $M'$ is a minimal causal model.

The following two lemmas establish the correctness and efficiency of this function. The proofs will be by induction. This is possible because every recursive call made by *find-minimal-causal-model*$(M,\mathcal{I})$ replaces $M$ by a model that is strictly simpler than $M$. Since there are a finite number of models, the recursive calls are guaranteed to bottom out, and hence we can use induction in our proofs. We first prove the correctness of *find-minimal-causal-model.*

**Lemma 5.1.1.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies the upward failure property, and let $M \subseteq \mathcal{M}$ be a coherent model. Then find-minimal-causal-model$(M,\mathcal{I})$ returns an adequate model (i.e., a minimal causal model) of $\mathcal{I}$ that is simpler than $M$, if it exists, and nil otherwise.*

*Proof.* We prove this lemma by induction. There are two base cases:

1. First, $M$ is not a causal model. Since, $\mathcal{I}$ satisfies the upward failure property, it follows that there is no causal model simpler than $M$, and hence no adequate model simpler than $M$. Since $M$ is not a causal model, the condition of the first **if** statement succeeds, and *find-minimal-causal-model*$(M,\mathcal{I})$ returns nil. Hence, the lemma is true for this base case.
2. Second, $M$ is a causal model, but it has no immediate simplifications. Hence, $M$ is an adequate model. *find-minimal-causal-model*$(M,\mathcal{I})$ enters the **else** clause, does not enter the **for** loop, and immediately returns $M$. Hence, the lemma is also true for this base case.

Now, for the inductive step, assume that $M$ is a causal model with immediate simplifications. There are two cases:

1. For every $M' \in$ *simplifications*$(M, \mathcal{I})$, *find-minimal-causal-model*$(M', \mathcal{I})$ returns nil. By induction, this means that there is no adequate model that is simpler than any of the immediate simplifications of $M$. But every coherent model that is strictly simpler than $M$ is simpler than some immediate simplification of $M$. Hence, it follows that there is no adequate model that is strictly simpler than $M$. Since $M$ is a causal model, it follows that $M$ is an adequate model. When every $M' \in$ *simplifications*$(M, \mathcal{I})$ is such that *find-minimal-causal-model*$(M', \mathcal{I})$ returns nil, *find-minimal-causal-model*$(M, \mathcal{I})$ exits the **for**, and returns $M$. Hence, the lemma is satisfied for this case of the inductive step.
2. $M' \in$ *simplifications*$(M, \mathcal{I})$ is such that *find-minimal-causal-model*$(M', \mathcal{I})$ returns a non-nil value. Let $M'$ be the first such model encountered in the **for** loop. Hence, *find-minimal-causal-model*$(M, \mathcal{I})$ can be seen to return *find-minimal-causal-model*$(M', \mathcal{I})$. By induction, the value returned by *find-minimal-causal-model*$(M', \mathcal{I})$ is an adequate model that is simpler than $M'$. Since $M' < M$, it follows that this model is also an adequate model simpler than $M$. Hence, the lemma is satisfied for this case of the inductive step.

Hence, if $M$ is a coherent model, then *find-minimal-causal-model*$(M, \mathcal{I})$ returns an adequate model of $\mathcal{I}$ that is simpler than $M$, if it exists, and nil otherwise.

Next, we show that if the immediate simplifications of a coherent model can be computed in polynomial time, then *find-minimal-causal-model* also runs in polynomial time.

**Lemma 5.1.2.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem, and let $M \subseteq \mathcal{M}$ be a coherent model. If the immediate simplifications of every coherent model of $\mathcal{I}$ can be computed in time polynomial in the size of $\mathcal{I}$, then find-minimal-causal-model$(M, \mathcal{I})$ terminates in time polynomial in the size of $\mathcal{I}$.*

*Proof.* Excluding the recursive calls, the only significant work done by *find-minimal-causal-model*$(\mathcal{I}, M)$ is to check if $M$ is a causal model, and to generate the immediate simplifications of $M$ (if necessary). Lemma 4.2.1 tells us that the former can be done in polynomial time, while the latter can be done in polynomial time by assumption. Hence, we need only show that there are a polynomial number of recursive calls to *find-minimal-causal-model*. One can see that if *find-minimal-causal-model* returns nil, it makes no recursive calls. Hence, one can verify that, of the recursive calls made by *find-minimal-causal-model*, at most one can itself make recursive calls. Hence, in conjunction with the fact that every call to *find-minimal-causal-model* can make at most a polynomial number of recursive calls, it follows that the total

number of recursive calls is polynomial if and only if the maximum depth of the recursion is polynomial. Now, every recursive call made by *find-minimal-causal-model*$(\mathcal{I}, M)$ replaces the model $M$ by an immediate simplification of $M$, constructed by dropping and/or approximating some set of model fragments in $M$. Hence, once a model fragment, $m$, is removed from $M$, no deeper recursive call uses a model containing $m$. Hence, the depth of the recursion is bounded by the number of model fragments in $\mathcal{M}$. Hence, the total number of recursive calls is polynomial.

**Finding an initial causal model.** Given any causal model $M$, the *find-minimal-causal-model* function can be used to find a minimal causal model that is simpler than $M$. Hence, to find a minimal causal model of $\mathcal{I}$, we must first find some causal model of $\mathcal{I}$. In Chapter 8, we will discuss a heuristic method for finding such an initial causal model. Here, however, we discuss an alternate method of finding an initial causal model, that does not rely on heuristics.

Let us introduce a fictitious model $M_T$, representing a model that is more accurate than every model of $\mathcal{I}$.[1] Hence, the immediate simplifications of $M_T$ are just those models of $\mathcal{I}$ that are not strictly simpler than any other models of $\mathcal{I}$:

$$\begin{aligned} &\textit{simplifications}(M_T, \mathcal{I}) = \\ &\quad \{M : \ M \text{ is coherent wrt } \mathcal{I} \\ &\qquad \wedge \neg(\exists M') \ M' \text{ is coherent wrt } \mathcal{I} \\ &\qquad\qquad \wedge M < M'\} \end{aligned} \tag{5.3}$$

We can then use *find-minimal-causal-model* to find an adequate model of $\mathcal{I}$ by assuming that $M_T$ is, by fiat, a causal model. In that case,

*find-minimal-causal-model*$(M_T, \mathcal{I})$

returns an adequate model of $\mathcal{I}$, if it exists, or returns $M_T$ if $\mathcal{I}$ has no causal models.

Note that, to find an adequate model of $\mathcal{I}$ in polynomial time, using the above function call, we will require that *simplifications*$(M_T, \mathcal{I})$ return in polynomial time, i.e., $\mathcal{I}$ must have a polynomial number of most accurate models, all of which can be generated in polynomial time. Hence, we have the following theorem:

**Theorem 5.1.1.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies the upward failure property. If the most accurate models of $\mathcal{I}$ can be generated in time polynomial in the size of $\mathcal{I}$, and if the immediate simplifications of any coherent model of $\mathcal{I}$ can be generated in time polynomial in the size of $\mathcal{I}$, then an adequate model of $\mathcal{I}$ can be found in time polynomial in the size of $\mathcal{I}$.*

[1] Note that $M_T$ isn't really a set of model fragments. It is just a fictitious model that is assumed to be more accurate than every model.

*Proof.* Immediate consequence of Lemmas 5.1.1 and 5.1.2 and the above discussion.

### 5.1.2 Discussion

We have seen that the upward failure property is useful because it leads to an efficient algorithm for finding an adequate model. However, it has a major drawback: it is very difficult to decide whether or not a particular instance of the MINIMAL CAUSAL MODEL problem satisfies the upward failure property. For example, a straightforward use of Definition 5.1 requires us to check every model in the space of possible models. Since the space of possible models is exponentially large, any such check is unthinkable. In fact, the upward failure property was suggested as a way around having to check the whole space of possible models. Unfortunately, it does not seem to have succeeded in helping us to circumvent this problem.

This drawback of the upward failure property stems from the fact that it is a *global* property, i.e., a property of the whole space of possible models. What we want is a *local* property that entails the upward failure property, i.e., a property of the encoding of $\mathcal{I}$ that can be checked efficiently, that will ensure that $\mathcal{I}$ satisfies the upward failure property.

In the next few sections we present some local properties of $\mathcal{I}$ that ensure that $\mathcal{I}$ satisfies the upward failure property. In particular, we will go back to the sources of intractability identified in the previous chapter, and place appropriate restrictions on $\mathcal{I}$ that will make the MINIMAL CAUSAL MODEL problem tractable: (a) we will introduce a new class of *approximations*, called *causal approximations*, that will address the problem of selecting model fragments from selected assumption classes; (b) we will add additional constraints to $\mathcal{C}$, called *ownership constraints*, that will address the problem of selecting assumption classes; and (c) we will restrict the expressive power of constraints in $\mathcal{C}$.

## 5.2 Preliminary restrictions

In this section we introduce three preliminary restrictions on $\mathcal{I}$. First, we assume that the *contradictory* relation partitions the set of model fragments into the set of assumption classes, i.e., model fragments are in the same assumption class *if* and only if they are mutually contradictory (see Definition 4.10). Recall that an assumption class is a set of different descriptions of the same phenomena, i.e., a set of mutually contradictory model fragments describing the same phenomena. Hence, the above assumption is based on the intuition that there is little reason for descriptions of different phenomena to be mutually contradictory.

Second, we assume that each assumption class has a single, most accurate model fragment:

$$(\forall A \in \mathcal{A})(\exists m \in A)(\forall m' \in A)\ m \neq m' \Rightarrow \mathit{approximation}(m, m') \quad (5.4)$$

In other words, we assume that each phenomena has a single best description. This is a reasonable assumption as long as we only model fairly well understood phenomena, i.e., where there is broad consensus amongst the domain experts about how best to model the phenomena.

Note that the above restriction appears to be a problem when a given phenomena can be modeled with multiple ontologies. In such cases, it may not be possible to say that one ontology is more accurate than the other, leading to multiple most accurate model fragments in an assumption class. However, this does not pose a problem if the different ontologies are not mutually contradictory, so that model fragments that use different ontologies are in different assumption classes. This is often the case, since different ontologies are often used for different purposes.

For example, magnetism can be modeled either as magnetic fields [Halliday and Resnick, 1978], or as magnetic circuits [Coren, 1989]. However, these different ontologies are used for different purposes. In particular, the magnetic field ontology is suitable for reasoning about the interactions of magnets with externally applied magnetic fields, e.g., in generators and motors. On the other hand, the magnetic circuits ontology is useful for reasoning about the interactions of magnets with magnetic materials, e.g., in electric relays and door bells. Hence, it is perfectly acceptable for a model to include both ontologies. Hence, we can meet the requirement that each assumption class has a single most accurate model fragment by placing model fragments using the different ontologies in different assumption classes.

An important consequence of the above restriction is that it leads to $\mathcal{I}$ having a single most accurate model: the most accurate model of $\mathcal{I}$ is just the set of most accurate model fragments of the assumption classes of $\mathcal{I}$. This brings us to our third assumption: we assume that the most accurate model of $\mathcal{I}$ is coherent, i.e., we assume that the most accurate model is complete and that it satisfies all the domain-dependent structural and behavioral coherence constraints.

The second and third assumptions together imply that the immediate simplifications of $M_T$ contains exactly one model: the most accurate model of $\mathcal{I}$. It is easy to see that the most accurate model of $\mathcal{I}$ can be constructed very easily (in polynomial time), and hence the immediate simplifications of $M_T$ can be computed in polynomial time.

In summary, the preliminary restrictions that we place on $\mathcal{I}$ are the following:

- The *contradictory* relation partitions the set of model fragments in $\mathcal{M}$ into the set $\mathcal{A}$ of assumption classes.
- Each assumption class in $\mathcal{A}$ contains a single most accurate model fragment.
- The most accurate model of $\mathcal{I}$, which is the set of most accurate model fragments of the assumption classes in $\mathcal{A}$, is coherent.

## 5.3 Causal Approximations

We now introduce an important class of approximations, called *causal approximations*, that are commonly found in modeling the physical world. The basic idea underlying the definition of causal approximations is that more approximate descriptions often tend to involve fewer parameters. Furthermore, more accurate descriptions tend to explain more about a phenomenon than more approximate descriptions.

For example, Figure 2.6 shows different descriptions of electrical conduction in a wire. Figure 2.7 shows the *approximation* relation between these descriptions. Note that the parameters in the equations of the more approximate descriptions ($V_w = 0$ and $i_w = 0$) are a subset of the parameters in the equations of the more accurate description ($V_w = i_w R_w$). Furthermore, only `Resistor(wire-1)` is able to explain the relationship between $V_w$, $i_w$, and $R_w$.

In this section, we make the above idea precise, and investigate its consequences. As mentioned earlier, we restrict our discussion to models that do not include differential equations. Differential equations will be discussed in Chapter 6.

### 5.3.1 Definitions

We start by defining *local* parameters. A local parameter is a parameter that can be causally determined only by equations of model fragments in a single assumption class.

**Definition 5.2 (Local parameters).** *A parameter $p$ is said to be local to a model fragment $m \in \mathcal{M}$ if and only if $p$ can be causally determined by the equations of $m$, but not by the equations of any model fragment that does not contradict $m$:*

$$p \in P_c(m) \wedge (\forall m' \in \mathcal{M})\ m \neq m' \wedge p \in P_c(m') \Rightarrow \mathit{contradictory}(m, m')$$

*A parameter is said to be shared if it is not local to any model fragment.*

Using the above definition, we now define causal approximations. The idea underlying this definition is that if $m_2$ is a causal approximation of $m_1$, then any causal orientation of the equations of $m_2$ can be extended to a causal orientation of the equations of $m_1$, such that the latter causal orientation entails a superset of causal relations, i.e., $m_1$ can explain more than $m_2$:

**Definition 5.3 (Causal approximations).** *A model fragment $m_2$ is said to be a causal approximation of a model fragment $m_1$ if and only if:*

1. *$m_2$ is an approximation of $m_1$;*
2. *There exists a 1-1 mapping $G : m_2 \rightarrow m_1$ such that for each $e \in m_2$, $P(e) \subseteq P(G(e))$, and $P_c(e) \subseteq P_c(G(e))$. $G$ is called a correspondence mapping, and $e$ and $G(e)$ are said to be corresponding equations; and*

*3. Let $E^*$ denote the equations of $m_1$ that have no corresponding equations in $m_2$, and let $P^*$ denote the set of parameters that are local to $m_1$, but not local to $m_2$. Then there exists an onto causal mapping $L : E^* \rightarrow P^*$. $L$ is called a local causal mapping with respect to correspondence mapping $G$.*

Condition 1 ensures that causal approximations are approximations. Condition 2 ensures that for any causal orientation of an equation $e \in m_2$, there exists a causal orientation of $G(e) \in m_1$ which entails a superset of causal relations. Condition 3 ensures that additional equations in $m_1$ can be oriented to causally determine newly introduced local parameters.

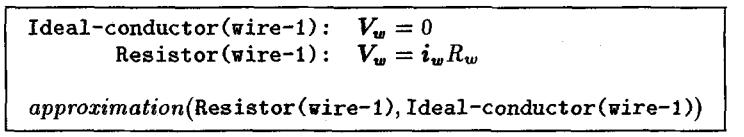



**Fig. 5.2.** Model fragments describing electrical conduction in `wire-1`

For example, the approximation relation between `Resistor(wire-1)` and `Ideal-conductor(wire-1)` shown in Figure 5.2 is a causal approximation. In particular, $V_w = 0$ and $V_w = i_w R_w$ are corresponding equations with:

$$P(V_w = 0) \subseteq P(V_w = i_w R_w)$$
$$P_c(V_w = 0) \subseteq P_c(V_w = i_w R_w)$$

Similarly, the approximation relation between `Temperature-dependent-resistance(wire-1)` and `Constant-resistance(wire-1)` shown in Figure 5.3 is a causal approximation if we assume that $R_{w0}, \alpha_w$, and $T_{w0}$ are local parameters of `Temperature-dependent-resistance(wire-1)`.

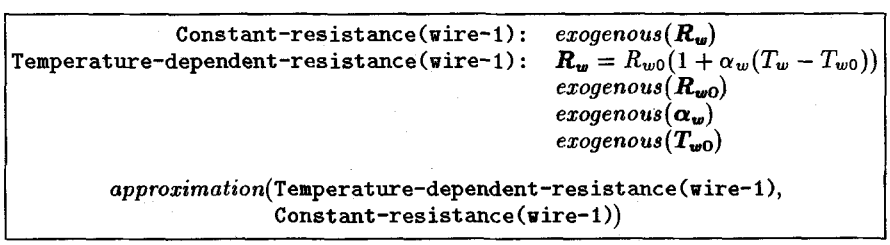



**Fig. 5.3.** Model fragments describing a wire's resistance.

In particular, $exogenous(R_w)$ and $R_w = R_{w0}(1 + \alpha_w(T_w - T_{w0}))$ are corresponding equations with:

$$P(exogenous(R_w)) \subseteq P(R_w = R_{w0}(1 + \alpha_w(T_w - T_{w0})))$$
$$P_c(exogenous(R_w)) \subseteq P_c(R_w = R_{w0}(1 + \alpha_w(T_w - T_{w0})))$$

and the local causal mapping, $L$, with respect to this correspondence mapping is:

$$\begin{aligned} L(\mathit{exogenous}(\boldsymbol{R_{w0}})) &= R_{w0} \\ L(\mathit{exogenous}(\boldsymbol{\alpha_w})) &= \alpha_w \\ L(\mathit{exogenous}(\boldsymbol{T_{w0}})) &= T_{w0} \end{aligned}$$

One can show that the causal approximation relation between model fragments is transitive. Hence, to check that all the approximations are causal approximations, it is sufficient to check that the immediate approximations of each model fragment are causal approximations.

It is worth noting that the restriction that local parameters in a model fragment cannot be causally determined by equations of model fragments in other assumption classes is not a serious one. It is easy to convert a local parameter into a shared parameter by defining a new assumption class. For example, to convert $\alpha_w$ (Figure 5.3) into a shared parameter, we would (a) define a new assumption class with one model fragment $m = \{\mathit{exogenous}(\boldsymbol{\alpha_w})\}$; and (b) remove $\mathit{exogenous}(\boldsymbol{\alpha_w})$ from the equations of `Temperature-dependent-resistance(wire-1)`. After this conversion, $\alpha_w$ is not necessarily local to any assumption class, and hence can be used in multiple assumption classes.

### 5.3.2 Causal approximations and the upward failure property

Causal approximations plays a key role in ensuring that the upward failure property is satisfied. In particular, we will show that when all the approximations are causal approximations, the causal relations entailed by a model decrease monotonically as we simplify models without dropping assumption classes. This means that if a model does not explain the expected behavior, then a simpler model that uses the same assumption classes also does not explain the expected behavior. It is easy to see that this is just a restricted version of the upward failure property.

To prove the above important result, we first introduce *local* and *global extensions* of causal mappings. We will then use the properties of these extensions to show that a causal mapping of a simpler model can be extended to a causal mapping of a more accurate model, such that the latter causal mapping entails a superset of causal relations.

**Local extensions.** A *local extension* of a causal mapping $H_2$, defined on a model fragment $m_2$, is a causal mapping $H_1$, defined on a more accurate model fragment $m_1$, that orients corresponding equations in the same way.

**Definition 5.4 (Local extension).** *Let $m_1, m_2 \in \mathcal{M}$ be model fragments such that $m_2$ is a causal approximation of $m_1$. Let $H_1 : m_1 \rightarrow P(m_1)$ and $H_2 : m_2 \rightarrow P(m_2)$ be causal mappings, and let $G : m_2 \rightarrow m_1$ be a correspondence mapping. $H_1$ is said to be a local extension of $H_2$ if and only if for each equation $e \in m_1$:*

1. *if* $G(e') = e$, *for some* $e' \in m_2$, *then* $H_1(e) = H_2(e')$;
2. *otherwise* $H_1(e)$ *is local to* $m_1$, *but not local to* $m_2$.

In other words, $H_1$ and $H_2$ orient corresponding equations in the same way, and $H_1$ orients the remaining equations in $m_1$ to causally determine parameters local to $m_1$ but not in $m_2$. An immediate consequence of this is that the range of $H_1$ contains only parameters that are either in the range of $H_2$ or that are local to $m_1$.

The following lemma tells us that the local extension of a causal mapping like $H_2$ always exists:

**Lemma 5.3.1 (Existence of local extension).** *Let* $m_1, m_2 \in \mathcal{M}$ *be model fragments such that* $m_2$ *is a causal approximation of* $m_1$. *Let* $H_2 : m_2 \rightarrow P(m_2)$ *be a causal mapping. Then there exists a causal mapping* $H_1 : m_1 \rightarrow P(m_1)$ *such that* $H_1$ *is a local extension of* $H_2$.

*Proof.* Let $G : m_2 \rightarrow m_1$ be a correspondence mapping, and let $L$ be a local causal mapping with respect to $G$. $G$ and $L$ must exist because $m_2$ is a causal approximation of $m_1$. We define $H_1$ as follows. For each $e \in m_1$, if there exists $e' \in m_2$ such that $G(e') = e$, then let $H_1(e) = H_2(e')$. This is possible because $G$ is a correspondence mapping and hence $P_c(e') \subseteq P_c(e)$. Otherwise, let $H_1(e) = L(e)$.

$H_1$ is well defined because the range of $L$ contains only parameters that are not local to $m_2$, and hence not in the range of $H_2$. $H_1$ is a causal mapping because both $H_1$ and $L$ are causal mappings. Finally, $H_1$ is a local extension of $H_2$ because $H_1$ and $H_2$ agree on corresponding equations, and on the remaining equations $H_1$ agrees with $L$, and hence maps these equations to parameters local to $m_1$ but not local to $m_2$.

**Global extensions.** *Global extensions* are similar to local extensions, except that instead of considering causal mappings of the equations of model fragments, we consider causal mappings of the equations of models.

**Definition 5.5 (Global extension).** *Let* $\mathcal{I}$ *be an instance of the* MINIMAL CAUSAL MODEL *problem such that all the approximation relations are causal approximations. Let* $M_1, M_2 \subseteq \mathcal{M}$ *be models, such that* $M_1$ *and* $M_2$ *are not overconstrained,* $M_1$ *and* $M_2$ *have model fragments from the same assumption classes, and* $M_2 < M_1$. *Let* $H_1 : E(M_1) \rightarrow P(M_1)$ *and* $H_2 : E(M_2) \rightarrow P(M_2)$ *be causal mappings.* $H_1$ *is said to be a global extension of* $H_2$ *if for each* $m_2 \in M_2$:

1. *If* $m_2 \in M_1$, *then for each* $e \in m_2$, $H_2(e) = H_1(e)$.
2. *If* $m_2 \notin M_1$, *then let* $m_1 \in M_1$ *be such that approximation*$(m_1, m_2)$. *Then,* $H_1$ *restricted to* $m_1$ *is a local extension of* $H_2$ *restricted to* $m_2$.

Hence, as with local extensions, $H_1$ is a global extension of $H_2$ if both $H_1$ and $H_2$ causally orient corresponding equations in the same way. It is also

easy to check that the range of $H_1$ contains only parameters that are in the range of $H_2$ or that are local to the model fragments in $M_1$. Finally, global extensions are guaranteed to exist:

**Lemma 5.3.2 (Existence of global extension).** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem such that all the approximation relations are causal approximations, and the contradictory relation partitions the set $\mathcal{M}$ of model fragments into the set $\mathcal{A}$ of assumption classes. Let $M_1, M_2 \subseteq \mathcal{M}$ be models such that $M_1$ and $M_2$ are not overconstrained, $M_1$ and $M_2$ have model fragments from the same assumption classes, and $M_2 < M_1$. Let $H_2 : E(M_2) \rightarrow P(M_2)$ be a causal mapping. Then there exists a causal mapping $H_1 : E(M_1) \rightarrow P(M_1)$ such that $H_1$ is a global extension of $H_2$.*

*Proof.* For the equations of each model fragment $m \in M_1$, define $H_1$ as follows:

1. If $m \in M_2$, then define $H_1$ to be identical to $H_2$ for each equation in $E(m)$;
2. Otherwise, there exists a unique $m' \in M_2$ such that $m'$ is a causal approximation of $m$. Define $H_1$ on $m$ to be the local extension of $H_2$ on $m'$. This is always possible because Lemma 5.3.1 tells us that such a local extension must exist. In addition, parameters in the range of $H_1$ restricted to $m$ are either in the range of $H_2$ restricted to $m'$, or are local to $m$. Hence, this extension does not overlap with the definition of $H_1$ on other model fragments.

It is easy to verify that $H_1$ is, in fact, a global extension of $H_2$.

The importance of global extensions stems from the fact that if $H_1$ is a global extension of $H_2$, then the direct causal dependencies entailed by $H_2$ are a subset of the direct causal dependencies entailed by $H_1$, i.e., $C_{H_2} \subseteq C_{H_1}$:

**Lemma 5.3.3.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem such that all the approximation relations are causal approximations. Let $M_1, M_2 \subseteq \mathcal{M}$ be models such that $M_1$ and $M_2$ are not overconstrained, $M_1$ and $M_2$ have model fragments from the same assumption classes, and $M_2 < M_1$. Let $H_1 : E(M_1) \rightarrow P(M_1)$ and $H_2 : E(M_2) \rightarrow P(M_2)$ be causal mappings such that $H_1$ is a global extension of $H_2$. Then $C_{H_2} \subseteq C_{H_1}$.*

*Proof.* Let $(p_1, p_2) \in C_{H_2}$. From Equation 3.5, this means that there is an equation $e \in E(M_2)$ such that $H_2(e) = p_2$ and $p_1 \in P(e)$. Let $m \in M_2$ be the model fragment such that $e \in m$. Now there are two cases:

1. If $m \in M_1$, then since $H_1$ is a global extension of $H_2$, $H_1(e) = H_2(e)$, and hence $(p_1, p_2) \in C_{H_1}$.

2. Otherwise, there is a model fragment $m' \in M_1$ such that $m$ is a causal approximation of $m'$. Let $G : m \rightarrow m'$ be the correspondence mapping. Let $e' = G(e)$. Since $H_1$ is a global extension of $H_2$, it follows that $H_1$ restricted to $m'$ is a local extension of $H_2$ restricted to $m$. Hence, $H_1(e') = H_2(e) = p_2$. Since $G$ is a correspondence mapping, it follows that $P(e) \subseteq P(e')$. Hence, since $p_1 \in P(e)$, it follows that $p_1 \in P(e')$. Hence, it follows that $(p_1, p_2) \in C_{H_1}$.

Hence, in either case, if $(p_1, p_2) \in C_{H_2}$, then $(p_1, p_2) \in C_{H_1}$. Hence, it follows that $C_{H_2} \subseteq C_{H_1}$.

**Monotonicity of causal relations.** The main theorem of this section is an immediate consequence of the above lemma: if we simplify a model without dropping any assumption classes, then the entailed causal relations decrease monotonically.

**Theorem 5.3.1.** *Let $\mathcal{I}$ be an instance of* MINIMAL CAUSAL MODEL *such that all the approximation relations are causal approximations, and the contradictory relation partitions the set $\mathcal{M}$ of model fragments into the set $\mathcal{A}$ of assumption classes. Let $M_1, M_2 \subseteq \mathcal{M}$ be complete models such that $M_1$ and $M_2$ contain model fragments from the same assumption classes, and $M_2 \leq M_1$. The causal relations entailed by the equations of $M_2$ are a subset of the causal relations entailed by the equations of $M_1$, i.e., $C(E(M_2)) \subseteq C(E(M_1))$.*

*Proof.* Let $H_2 : E(M_2) \rightarrow P(M_2)$ be an onto causal mapping. Such an onto causal mapping must exist because $M_2$ is complete. Let $H_1 : E(M_1) \rightarrow P(M_1)$ be the global extension of $H_2$. Since $M_1$ is complete, and hence not overconstrained, Lemma 5.3.2 tells us that such a global extension must exist. Since $H_1$ is defined on each equation in $E(M_1)$, and $M_1$ is complete, it follows that $H_1$ is an onto causal mapping.

Lemma 5.3.3 tells us that $C_{H_2} \subseteq C_{H_1}$. Hence, $tc(C_{H_2}) \subseteq tc(C_{H_1})$, and hence $C(E(M_2)) \subseteq C(E(M_1))$, i.e., the causal relations entailed by the equations of $M_2$ are a subset of the causal relations entailed by the equations of $M_1$.

Hence, if a coherent model does not explain the expected behavior, it follows that no simpler coherent model that uses the same set of assumption classes can explain the expected behavior. Hence, when all the approximations are causal approximations, a restricted version of the upward failure property is satisfied. Note that, unlike the upward failure property, it is easy to decide whether or not all the approximations are causal approximations. In particular, one can easily check whether a particular approximation is a causal approximation, and the transitivity of the causal approximation relation tells us that we need only check that all the immediate approximations of a model fragment are causal approximations.

### 5.3.3 Causal approximations are common

Causal approximations are particularly useful, because they are commonly found in modeling the physical world. The following is a partial list of commonly used approximations that are causal approximations.

1. Disregarding the translational and rotational inertia of an object
2. Disregarding relativistic effects
3. Rigid bodies
4. Frictionless motion
5. Zero or uniform gravitational fields
6. Elastic collisions
7. Ideal gas law
8. Ideal thermal conductors and insulators
9. Constant thermal conductance
10. Ideal electrical conductors and insulators
11. Constant electrical resistance and resistivity
12. Ideal heat engine
13. Inviscid flow
14. Disregarding thermal expansion

The details of the above causal approximations, including the actual equations used, can be found in Appendix A. The ubiquity of causal approximations suggests that we have identified an important property of commonly occurring instances of the MINIMAL CAUSAL MODEL problem.

In summary, we have shown that when all the approximation relations are causal approximations, if we restrict ourselves to models that select a model fragment from each assumption class in a fixed set of assumption classes, then the upward failure property is satisfied. Hence, the basic restriction that we place on the instance $\mathcal{I}$ of the MINIMAL CAUSAL MODEL problem is:

- All the approximation relations between model fragments are causal approximations.

## 5.4 Selecting assumption classes

In the previous section, we investigated the case where all models were required to have a model fragment from each assumption class in some fixed set of assumption classes. We showed that when all the approximation relations were causal approximations, then the causal relations entailed by a simpler model were a subset of the causal relations entailed by a more complex model. In this section we extend this result to all models, i.e., where models can also be simplified by dropping model fragments.

### 5.4.1 Causal approximations are not enough

A simple example illustrates that causal approximations alone are not sufficient. Let $A_1 = \{m_{11}, m_{12}\}$ and $A_2 = \{m_2\}$ be assumption classes, and let the equations of model fragments $m_{11}$, $m_{12}$, and $m_2$ be defined as follows:

$$\begin{aligned} m_{11} &= \{x = y, y = z\} \\ m_{12} &= \{x = y, exogenous(y)\} \\ m_2 &= \{exogenous(x)\} \end{aligned}$$

Furthermore, let $m_{12}$ be an approximation of $m_{11}$:

$$approximation(m_{11}, m_{12})$$

It is easy to verify that $m_{12}$ is a causal approximation of $m_{11}$. Let $M_1 = \{m_{11}, m_2\}$ and $M_2 = \{m_{12}\}$ be two models. Assuming that there are no propositional coherence constraints, it is easy to verify that both $M_1$ and $M_2$ are coherent models, and that $M_2 < M_1$.

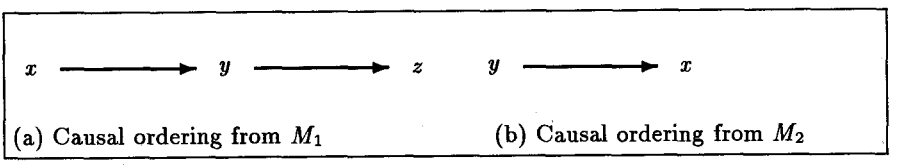



**Fig. 5.4.** Causal orderings generated from $M_1$ and $M_2$

Figure 5.4 shows the causal orderings generated from these two models. In particular, $y$ causally depends on $x$ in the causal ordering generated from $M_1$, while $x$ causally depends on $y$ in the causal ordering generated from $M_2$. Hence, in simplifying $M_1$ to $M_2$, the causal relations have not decreased monotonically. This means that the upward failure property is not satisfied.

On the other hand, if we replace $m_2$ with $m_2'$, where:

$$m_2' = \{exogenous(z)\}$$

then the causal ordering generated using $M_1' = \{m_{11}, m_2'\}$ is shown in Figure 5.5. In this case, it is easy to verify that the causal relations have decreased monotonically in going from $M_1'$ to $M_2$.

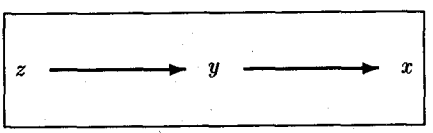



**Fig. 5.5.** Causal ordering generated from $M_1'$

Intuitively, the difference between the two cases can be summarized as follows. In the first case, $M_2$ did not include all phenomena that were possibly

"relevant" to its parameters. In particular, $M_2$ used the parameter $x$, but did not include $m_2$, even though an equation in $m_2$ could causally determine $x$. On the other hand, in the second case, $M_2$ included models of all phenomena relevant to $x$ and $y$—the equations of $m_2'$ can only determine $z$.

We can use the above intuition to ensure that the causal relations entailed by coherent models decrease monotonically as models become simpler, even when assumption classes can be dropped. We formalize this intuition by first introducing the *ownership of parameters* by assumption classes, and then introducing a set of *ownership constraints* that will ensure that coherent models include all possibly "relevant" phenomena.

### 5.4.2 Parameter ownership

The parameters owned by an assumption class are the parameters that can be causally determined by some equation of some model fragment in the assumption class.

**Definition 5.6 (Parameter ownership).** *The parameters owned by an assumption class $A$, denoted by $owns(A)$, are the parameters that can be causally determined by the equations of model fragments of $A$:*

$$owns(A) = \bigcup_{m \in A} P_c(m)$$

One can view an assumption class as being possibly "relevant" to the parameters that it *owns*. For example, the `Electrical-conductor(wire-1)` assumption class, shown in Figure 5.6, *owns* $V_w$ and $i_w$, but not $R_w$. Hence, model fragments from this assumption class are possibly "relevant" to causally determining $V_w$ and $i_w$, but not $R_w$.

| | |
|---|---|
| `Ideal-conductor(wire-1):` | $V_w = 0$ |
| `Ideal-insulator(wire-1):` | $i_w = 0$ |
| `Resistor(wire-1):` | $V_w = i_w R_w$ |

**Fig. 5.6.** Model fragments in the `Electrical-conductor(wire-1)` assumption class

### 5.4.3 Ownership constraints

We can ensure that coherent models will contain model fragments from all possibly "relevant" assumption classes, by adding constraints of the form

$$m \Rightarrow A$$

to the set $\mathcal{C}$ of propositional coherence constraints, whenever assumption class $A$ *owns* a parameter that can be causally determined by an equation in $m$,

i.e., when $P_c(m) \cap owns(A)$ is not empty. This will ensure that whenever a coherent model contains model fragment $m$, it will also contain a model fragment from $A$. We call the above set of constraints *ownership constraints:*

**Definition 5.7 (Ownership constraints).** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem. The set $\mathcal{O}$ of ownership constraints of $\mathcal{I}$ are defined as follows:*

$$\mathcal{O} = \{m \Rightarrow A : m \in \mathcal{M} \ \wedge \ A \in \mathcal{A} \ \wedge P_c(m) \cap owns(A) \neq \emptyset\}$$

### 5.4.4 Monotonicity of causal relations

When $\mathcal{C}$ contains all the ownership constraints, so that coherent models contain model fragments from all possibly "relevant" assumption classes, we can extend Theorem 5.3.1 to all coherent models. We start with a lemma that states that if $E_1$ and $E_2$ are complete sets of equations such that $C(E_2) \subseteq C(E_1)$, then removing the same set of *exogenous* equations from both $E_1$ and $E_2$ preserves this property.

**Lemma 5.4.1.** *Let $E_1$ and $E_2$ be complete sets of equations. Let $Q = \{q_1, \ldots, q_n\}$ be a set of parameters, and let $D = \cup_{q \in Q} exogenous(q)$ be a set of equations such that $D \subseteq E_1$ and $D \subseteq E_2$. Let $H_1' : E_1 \rightarrow P(E_1)$ and $H_2' : E_2 \rightarrow P(E_2)$ be onto causal mappings. Let $H_1$ be $H_1'$ restricted to the equations in $E_1 \setminus D$, and let $H_2$ be $H_2'$ restricted to the equations in $E_2 \setminus D$. If $tc(C_{H_2'}) \subseteq tc(C_{H_1'})$, then $tc(C_{H_2}) \subseteq tc(C_{H_1})$.*

*Proof.* We first show that $tc(C_{H_1'}) = tc(C_{H_1})$. For any equation $exogenous(q_i)$ in $D$, $H_1'(exogenous(q_i)) = q_i$. Hence, there is no parameter $p$ such that $(p, q_i) \in C_{H_1'}$. Hence, there is no parameter $p$ such that $(p, q_i) \in tc(C_{H_1'})$. Hence, if $(p, q) \in tc(C_{H_1'})$, it follows that there is no parameter $q_i \in Q$ such that $(p, q_i) \in tc(C_{H_1'})$ and $(q_i, q) \in tc(C_{H_1'})$. Hence, any causal path from $p$ to $q$ using $H_1'$ does not involve any equation in $D$. Hence, this causal path must exist in the causal ordering using $H_1$. Hence, it follows that $tc(C_{H_1'}) = tc(C_{H_1})$. Similarly, $tc(C_{H_2'}) = tc(C_{H_2})$. Hence, if $tc(C_{H_2'}) \subseteq tc(C_{H_1'})$, it follows that $tc(C_{H_2}) \subseteq tc(C_{H_1})$.

Using the above lemma we can generalize Theorem 5.3.1 to all coherent models.

**Theorem 5.4.1.** *Let $\mathcal{I}$ be an instance of* MINIMAL CAUSAL MODEL *such that all the approximation relations are causal approximations, and the contradictory relation partitions the set $\mathcal{M}$ of model fragments into the set $\mathcal{A}$ of assumption classes. Let $\mathcal{C}$ contain all the ownership constraints of $\mathcal{I}$. Let $M_1, M_2 \subseteq \mathcal{M}$ be coherent models such that $M_2 \leq M_1$. The causal relations entailed by the equations of $M_2$ are a subset of the causal relations entailed by the equations of $M_1$, i.e., $C(E(M_2)) \subseteq C(E(M_1))$.*

*Proof.* Let $H_1 : E(M_1) \rightarrow P(M_1)$ and $H_2 : E(M_2) \rightarrow P(M_2)$ be onto causal mappings. $H_1$ and $H_2$ must exist because $M_1$ and $M_2$ are coherent models. We will now use $H_1$ and $H_2$ to construct an onto causal mapping $H : E(M_1) \rightarrow P(M_1)$ such that $C_{H_2} \subseteq tc(C_H)$.

Let us partition $M_1$ into two mutually disjoint sets $M_{11}$ and $M_{12}$ such that $M_{11}$ and $M_2$ have model fragments from the same assumption classes, and $M_{12}$ and $M_2$ have no model fragments from the same assumption classes. Let $H_{11} : E(M_{11}) \rightarrow P(M_{11})$ be $H_1$ restricted to the the equations in $E(M_{11})$, and let $H_{12} : E(M_{12}) \rightarrow P(M_{12})$ be $H_1$ restricted to the equations in $E(M_{12})$.

Let $P_{12}$ be the parameters in the range of $H_{12}$. From the definition of $H_{11}$ and $H_{12}$, it is easy to see that they have mutually disjoint ranges. Hence, no parameter in $P_{12}$ is in the range of $H_{11}$. In addition, since $M_2$ is a coherent model, it follows that all the ownership constraints are satisfied. Hence, since $M_2$ contains no model fragments from the assumption classes used in $M_{12}$, it follows that no parameter in $P(M_2)$ is owned by an assumption class used in $M_{12}$. Hence, no parameter in the range of $H_2$ is in $P_{12}$.

Let $m$ be a new model fragment consisting of the following equations:

$$m = \{\mathit{exogenous}(\boldsymbol{p}) : p \in P_{12}\}$$

i.e., $m$ makes each parameter in $P_{12}$ exogenous. One can see that the equations of $m$ are complete.

Since there is no overlap between $P_{12}$ and $P(M_2)$, and since $M_2$ is complete, it follows that $M_2' = M_2 \cup \{m\}$ is a complete model. Let $H_2' : E(M_2') \rightarrow P(M_2')$ be an onto causal mapping, such that $H_2'$ restricted to $E(M_2)$ is the same as $H_2$.

Now consider the set $M_{11}' = M_{11} \cup \{m\}$. We show that $M_{11}'$ is complete by defining a causal mapping $H_{11}' : E(M_{11}') \rightarrow P(M_{11}')$. Let $H_{11}'$ restricted to $E(M_{11})$ be identical to $H_{11}$. For any equation $\mathit{exogenous}(\boldsymbol{p})$ in $m$, let $H_{11}'(\mathit{exogenous}(\boldsymbol{p})) = p$. This is possible because the range of $H_{11}$ and the parameters in $P_{12}$ do not overlap. Clearly, $H_{11}'$ is defined on each equation in $E(M_{11}')$.

Finally, $H_{11}'$ is complete because every parameter in $P(M_{11}')$ is either in $P(M_{11})$ or in $P_{12}$. Every parameter in $P(M_{11})$ is either in the range of $H_{11}$ or in the range of $H_{12}$. Hence, every parameter in $P(M_{11}')$ is either in the range of $H_{11}$ or in $P_{12}$. Hence, every parameter in $P(M_{11}')$ is in the range of $H_{11}'$. Hence, $H_{11}'$ is an onto causal mapping and hence $M_{11}'$ is complete.

But $M_{11}'$ and $M_2'$ have model fragments from the same set of assumption classes (assuming that $m$ is in an assumption class by itself), and both these models are complete. In addition, since $M_2 \leq M_1$, it follows that $M_2' \leq M_{11}'$. Hence, Theorem 5.3.1 tells us that $C(E(M_2')) \subseteq C(E(M_{11}'))$. Hence, $tc(C_{H_2'}) \subseteq tc(C_{H_{11}'})$.

Applying Lemma 5.4.1, with $D = m$ and $Q = P_{12}$, we have $tc(C_{H_2}) \subseteq tc(C_{H_{11}})$. But $H_{11}$ is just $H_1$ restricted to $E(M_{11})$. Hence, it follows that $tc(C_{H_2}) \subseteq tc(C_{H_1})$. Hence, $C(E(M_2)) \subseteq C(E(M_1))$.

In summary, to ensure that the upward failure property property is satisfied, even when models can be simplified by dropping model fragments, we must place the following restriction on $\mathcal{I}$:

- The set $\mathcal{C}$ of propositional coherence constraints of $\mathcal{I}$ must contain all the ownership constraints of $\mathcal{I}$, as defined in Definition 5.7.

### 5.4.5 Discussion

The above theorem tells us that, with the addition of the ownership constraints, the causal relations entailed by models decrease monotonically as models become simpler. But how reasonable are the ownership constraints? On the face of it, they seem quite restrictive. However, under certain circumstances, we get the ownership constraints for free. In particular, consider the situation in which each equation can causally determine exactly one parameter. This situation is found in QP Theory [Forbus, 1984] and its derivatives, e.g., [Falkenhainer and Forbus, 1991]. When each equation can causally determine exactly one parameter, one can see that all the parameters are local to some assumption class. This means that no model fragment can causally determine a parameter owned by a different assumption class. Hence, under this situation, there are *no* ownership constraints!

However, as we have argued earlier, the constraint that each equation can causally determine exactly one parameter is also restrictive. In the absence of this constraint, the ownership constraints appear to be necessary to guarantee that the upward failure property is satisfied. However, in practice, we have found that the upward failure property is satisfied even in the absence of the ownership constraints. In other words, the pathological cases, like the one shown at the beginning of this section, do not appear to occur naturally. We conjecture that the reason for this is that most equations describing the physical world do seem to have a natural causal orientation (as required in QP Theory), and the few equations that do allow multiple causal orientations do not lead to pathological situations. Hence, in the modeling program that we describe in Chapter 8, no ownership constraints are included.

## 5.5 Individually approximating model fragments

The last two sections introduced two local properties of $\mathcal{I}$, causal approximations and ownership constraints, that ensure that the global upward failure property is satisfied. In this section, and in the next section, we turn to the other important element of the efficient model selection algorithm: the efficient generation of the immediate simplifications of coherent models.

A coherent model $M$ can be simplified by (a) replacing one or more model fragments in $M$ by their approximations; and/or (b) dropping one or more model fragments from $M$. In this section we prove a very important property

of the simplifications of $M$ which states that the model fragments in $M$ can be approximated one at a time. More precisely, let $m \in M$ be any model fragment, and let $m'$ be any immediate approximation of $m$. We show that if the model resulting from replacing $m$ by $m'$ is not complete, then no coherent model simpler than $M$ can contain $m'$ or any of its approximations. Hence, if the model fragments of $M$ cannot be approximated one at a time, there is no point approximating them two or more at a time. This property will be exploited in Section 5.7 to efficiently simplify a causal model.

**Theorem 5.5.1.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem such that all the approximations are causal approximations and $C$ contains all the ownership constraints. Let $M \subseteq \mathcal{M}$ be a coherent model. Let $m \in M$ be any model fragment, and let $m' \in \mathcal{M}$ be an immediate approximation of $m$. Let $M'$ be the result of replacing $m$ by $m'$ in $M$, i.e., $M' = (M \setminus \{m\}) \cup \{m'\}$. Let $m'' \in \mathcal{M}$ be any model fragment such that either $m'' = m'$ or $m''$ is one of the approximations of $m'$. If $E(M')$ is not complete, then every model $M'' < M$, such that $m'' \in M''$, is not coherent.*

*Proof.* The proof is by contradiction. Assume that there exists $M'' < M$ such that $m'' \in M''$ and $M''$ is coherent. We will now show that $E(M')$ is complete by showing that there exists an onto causal mapping $F' : E(M') \rightarrow P(M')$.

Let $F : E(M) \rightarrow P(M)$ and $F'' : E(M'') \rightarrow P(M'')$ be onto causal mappings. $F$ and $F''$ must exist because $M$ and $M''$ are coherent. We use $F$ and $F''$ to define $F'$ as follows.

Partition $M'$ into two sets $M'_1$ and $M'_2$, such that $M'_1$ contains all the model fragments in $M'$ that are in the same assumption class as a model fragment in $M''$, and $M'_2$ contains all the model fragments in $M'$ that are *not* in the same assumption class as any model fragment in $M''$. Hence, $m' \in M'_1$, and $M'_2 \subseteq M$.

We define $F'$ by first defining it on equations in $E(M'_1)$, and then on equations in $E(M'_2)$.

Let $m'_1$ be a model fragment in $M'_1$, and let $m''_1$ be the model fragment in $M''$ that is in the same assumption class as $m'_1$. It is easy to see that $m''_1$ is either identical to $m'_1$, or is an approximation of $m'_1$. If $m''_1$ is identical to $m'_1$, then define $F'$ on each equation in $m'_1$ to be identical to $F''$. If $m''_1$ is an approximation of $m'_1$, then use Lemma 5.3.1 to define $F'$ restricted to $m'_1$ to be the local extension of $F''$ restricted to $m''_1$. In either case, every parameter in the range of $F'$, when restricted to $m'_1$, is either in the range of $F''$ restricted to $m''_1$, or is local to $m'_1$. Hence, $F'$ maps equations in different model fragments in $M'_1$ to different parameters. Hence, the above mapping maps each equation in $E(M'_1)$ to a unique parameter.

Define $F'$ on equations in $E(M'_2)$ to be identical to $F$. This is possible because each model fragment in $M'_2$ is also in $M$. In addition, since $M''$ is coherent, it satisfies all the ownership constraints. Hence, it follows that $P_c(M'')$ and $P_c(M'_2)$ have no parameters in common. Hence, the range of $F''$

and the range of $F$ restricted to $M_2'$ are disjoint. Hence, the above extension of $F'$ to equations in $E(M_2')$ is well defined.

The above definition of $F'$ maps each equation in $E(M')$ to a parameter in $P(M')$. Hence, $|E(M')| \leq |P(M')|$. However, in going from $M'$ to $M$, we replace $m'$ by $m$, which introduces at least as many new local parameters as equations. Since $M$ is coherent, it follows that $|E(M)| = |P(M)|$, and hence $|E(M')| = |P(M')|$. Hence, $F'$ is an onto causal mapping, and hence $M'$ is complete, which is a contradiction.

## 5.6 Expressivity of constraints

Using the above theorem, it is possible to show that the immediate simplifications of any coherent model can be generated in polynomial time, as long as the only constraints in $\mathcal{C}$ are the ownership constraints. The complexity of generating the immediate simplifications of a coherent model in the presence of additional constraints is critically dependent upon the expressive power of these constraints. In particular, in Chapter 4, we showed that the MINIMAL CAUSAL MODEL problem becomes intractable if $\mathcal{C}$ has (a) negative clauses, i.e., clauses with all negative literals; or (b) definite horn clauses, i.e., clauses with exactly one positive literal. Hence, allowing such constraints will defeat any hopes of efficiently generating the immediate simplifications of a coherent model.

Fortunately, there is a class of constraints, only slightly different from horn clauses, that does not lead to intractability. In particular, we will allow constraints of the form:

$$m_1 \wedge m_2 \wedge \ldots \wedge m_n \Rightarrow A \tag{5.5}$$

where $m_1$, $m_2$, ..., $m_n$ are model fragments, for some $n \geq 0$,[2] and $A$ is an assumption class. Recall that using an assumption class in a propositional coherence constraint is just a shorthand for the disjunction of the model fragments in that assumption class. Hence, the consequents of constraints that have the above form are restricted to be the disjunction of all model fragments in an assumption class.

In practice, the restricted expressivity of the propositional coherence constraints has not proved to be a limitation. This is because our focus on the task of generating parsimonious causal explanations has made the expected behavior a central criterion for defining model adequacy. This has decreased the importance of the propositional coherence constraints in defining model adequacy, and hence the restricted expressive power has not proved to be problematic.

[2] When $n = 0$, there are no model fragments in the antecedent of the constraint, and the antecedent is assumed to be *true*.

It is worth noting that the ownership constraints, introduced above, and the *requires* constraints, introduced in Chapter 2, are special cases of the above type of constraint with $n = 1$. However, it is also worth noting that this restriction limits the set of model fragment classes that can have specializations: only model fragment classes that do not contradict any model fragment class (so that they are the only members of their assumption class) can have specializations.

In the rest of this chapter, we assume that all the constraints in $\mathcal{C}$ have the above form. That is, we place the following restriction on the instance $\mathcal{I}$ of the MINIMAL CAUSAL MODEL problem:

- The form of each constraint in $\mathcal{C}$ is required to be as shown in Equation 5.5.

In the next section we show that restricting the expressivity of constraints in this way does not lead to intractability.

## 5.7 Efficiently simplifying a coherent model

In this section we use the restrictions discussed thus far to develop an efficient algorithm for finding an adequate model. To summarize these restrictions, we assume that the instance

$$\mathcal{I} = (\mathcal{M}, \mathit{contradictory}, \mathit{approximation}, \mathcal{A}, \mathcal{C}, p, q)$$

of the MINIMAL CAUSAL MODEL problem satisfies the following restrictions:

**Definition 5.8 (Restrictions on $\mathcal{I}$).** *The restrictions on $\mathcal{I}$ introduced in this chapter are as follows:*

1. *The contradictory relation partitions the set of model fragments in $\mathcal{M}$ into the set $\mathcal{A}$ of assumption classes.*
2. *Each assumption class in $\mathcal{A}$ contains a single most accurate model fragment.*
3. *The most accurate model of $\mathcal{I}$ is coherent.*
4. *All the approximation relations are causal approximations.*
5. *$\mathcal{C}$ contains all the ownership constraints of $\mathcal{I}$.*
6. *The form of each constraint in $\mathcal{C}$ is as shown in Equation 5.5.*

The algorithm for efficiently finding an adequate model is a two step procedure. In the first step, the most accurate model is simplified using the function *find-minimal-causal-model*, shown in Figure 5.1, with the *simplifications* function being restricted to simplifying a model by approximating it, i.e., by replacing a model fragment by one of its immediate approximations. In the second step, the resulting model is simplified by dropping all unnecessary model fragments. We will show that the resulting model is, indeed, a minimal causal model.

### 5.7.1 Simplifying a model by approximating

The first step of simplification simplifies the most accurate model by replacing model fragments with their immediate approximations. Simplifying a model by replacing a model fragment with an immediate approximation is called *simplifying by approximating*:

**Definition 5.9 (Simplifying by approximating).** *Let $M \subseteq \mathcal{M}$ be a coherent*
*model, and let $m \in M$ by any model fragment. Let $m'$ be an immediate approximation of $m$. Let $M' = (M \setminus \{m\}) \cup \{m'\}$, i.e., $M'$ is the result of replacing $m$ by $m'$ in $M$. If $M'$ is coherent, then $M'$ is said to be an immediate simplification of $M$ by approximating.*

**Properties of immediate simplifications by approximating.** The following four lemmas state the important properties of immediate simplifications by approximating. The first two lemmas show that the only immediate simplifications of a coherent model that do not drop any assumption classes are the immediate simplifications by approximating. The next two lemmas show that the immediate simplifications by approximating can be generated in polynomial time.

It is easy to see that if $M'$ is an immediate simplification of $M$ by approximating, then $M'$ is one of $M$'s immediate simplifications:

**Lemma 5.7.1.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies the restrictions in Definition 5.8, and let $M, M' \subseteq \mathcal{M}$ be coherent models such that $M'$ is an immediate simplification of $M$ by approximating. Then $M' \in$ simplifications$(M, \mathcal{I})$.*

*Proof.* From the definition of model simplicity, it is easy to see that there is no model that is strictly between $M$ and $M'$ in the simplicity partial ordering.

We now show that the only immediate simplifications of $M$ that do not drop any assumption classes are the immediate simplifications of $M$ by approximating. This is a straightforward consequence of Theorem 5.5.1.

**Lemma 5.7.2.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies all the restrictions in Definition 5.8, and let $M \subseteq \mathcal{M}$ be a coherent model that contains a model fragment from every assumption class in $\mathcal{A}$. Let $M'$ be an immediate simplification of $M$, i.e., $M' \in$ simplifications$(M)$, and let $M$ and $M'$ have model fragments from the same assumption class. Then $M'$ is an immediate simplification of $M$ by approximating.*

*Proof.* The proof is by contradiction. Assume that $M'$ is not an immediate simplification by approximating. Let $m' \in M'$ be a model fragment that is not in $M$. Since $M$ and $M'$ have model fragments from the same assumption

classes, and since $M' < M$, let $m \in M$ be such that $m'$ is an *approximation* of $m$. Let $m''$ be a model fragment such that $m''$ is an immediate *approximation* of $m$, and $m'$ is either identical to $m''$ or $m'$ is an *approximation* of $m''$. Let $M'' = (M \setminus \{m\}) \cup \{m''\}$. Since $M' < M$, it is easy to see that $M' \leq M''$. There are now three cases:

1. If $M'$ is the same as $M''$, then $M'$ is an immediate simplification of $M$ by approximating, which is a contradiction.
2. If $M' < M''$, and $M''$ is coherent, then $M'$ is not an immediate simplification of $M$, which is a contradiction.
3. If $M' < M''$, and $M''$ is not coherent, then the equations of $M''$ are not complete. This follows from the fact that $M$, and hence $M''$, contains a model fragment from each assumption class, and hence every constraint in $\mathcal{C}$ must be satisfied. Hence, the only reason that $M''$ is not coherent is because the equations of $M''$ are not complete. But since $M'$ contains a model fragment, $m'$, that is either identical to $m''$ or is an *approximation* of $m''$, Theorem 5.5.1 tells us that $M'$ is not coherent, which is a contradiction.

Hence, in all three cases, we encounter a contradiction. Hence, all the immediate simplifications of $M$, that use a model fragment from each assumption class in $M$, are immediate simplifications by approximating.

```
function simp-by-approximating(M,I)
  /* Returns the immediate simplifications of M by approximating */
  result ← nil
  for every m ∈ M do
    for every m' that is an immediate approximation of m do
      M' ← (M \ {m}) ∪ {m'}
      if M' is coherent then
        /* M' is an immediate simplification of M by approximating */
        Add M' to result
      endif
    endfor
  endfor
  return result
end
```

**Fig. 5.7.** The function *simp-by-approximating*.

Figure 5.7 shows the *simp-by-approximating* function, which returns the immediate simplifications of a model by approximating. It is easy to see that this function returns all the immediate simplifications of $M$ by approximating.

**Lemma 5.7.3.** *The simp-by-approximating function returns all the immediate simplifications of a model by approximating.*

*Proof.* Immediate from the definition of immediate simplification by approximating.

In addition, one can see that the immediate simplifications of $M$ by approximating can be computed in polynomial time.

**Lemma 5.7.4.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies all the restrictions in Definition 5.8, and let $M \subseteq \mathcal{M}$ be a coherent model. Then simp-by-approximating$(M, \mathcal{I})$ terminates in time polynomial in the size of $\mathcal{I}$.*

*Proof.* Lemma 4.2.1 tells us that checking whether or not $M'$ is coherent can be done in polynomial time. The number of times this check has to be made is equal to the number of immediate approximations of the model fragments in $M$. Since the model fragments in $M$ are in different assumption classes, it follows that this number is bounded by the number of model fragments in $\mathcal{M}$. Hence, *simp-by-approximating*$(M, \mathcal{I})$ terminates in time polynomial in the size of $\mathcal{I}$.

**Finding a simplest causal model by approximating.** The above lemmas, in conjunction with a modified version of the *find-minimal-causal-model* function in Section 5.1, allow us to efficiently find a *simplest causal model by approximating*. A simplest causal model by approximating is a causal model that contains a model fragment from each assumption class, such that no causal model that contains a model fragment from each assumption class is strictly simpler than it.

**Definition 5.10 (Simplest causal model by approximating).** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem and let $M \subseteq \mathcal{M}$ be a causal model. $M$ is said to be a simplest causal model by approximating if and only if (a) $M$ contains a model fragment from each assumption class in $\mathcal{A}$; and (b) if $M'$ is a causal model of $\mathcal{I}$ that contains a model fragment from each assumption class in $\mathcal{A}$, then $M \leq M'$.*

Intuitively, a simplest causal model by approximating models each phenomena as approximately as possible. A simplest causal model by approximating can be identified using a modified version of the *find-minimal-causal-model* function, in which the call to the *simplifications* function is replaced by a call to the *simp-by-approximating* function. The correctness of this modified function follows from

1. Lemma 5.7.2, which tells us that the only immediate simplifications of a model that do not drop any assumption classes are the immediate simplifications of the model by approximating;
2. Lemma 5.7.3, which tells us that the *simp-by-approximating* function returns all the immediate simplifications of a model by approximating;

3. Theorem 5.4.1, which tells us that $\mathcal{I}$ satisfies the upward failure property;[3]
4. Lemma 5.1.1, which proves the correctness of the *find-minimal-causal-model* function.

In addition, this modified function finds a simplest causal model by approximating in polynomial time. This follows from:

1. Lemma 5.7.4, which tells us that the immediate simplifications of a model by approximating can be computed in polynomial time; and
2. Lemma 5.1.2, which tells us that *find-minimal-causal-model* terminates in polynomial time if the immediate simplifications can be computed in polynomial time.

Hence, we have the following lemma:

**Lemma 5.7.5.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies all the restrictions in Definition 5.8. A simplest causal model by approximating of $\mathcal{I}$ can be found in time polynomial in the size of $\mathcal{I}$.*

*Proof.* Immediate from the above discussion.

### 5.7.2 Simplifying a model by dropping model fragments

We now consider the second step of the model simplification procedure, that involves simplifying a model by dropping model fragments. We start by showing the second important consequence of Theorem 5.5.1. In particular, we show that if $M$ is a simplest causal model by approximating, then a minimal causal model simpler than $M$ is a subset of $M$.

**Lemma 5.7.6.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies all the restrictions in Definition 5.8, and let $M \subseteq \mathcal{M}$ be a simplest causal model by approximating. Let $M' \subseteq \mathcal{M}$ be a minimal causal model of $\mathcal{I}$ such that $M' \leq M$. Then $M' \subseteq M$.*

*Proof.* Since $M$ is a simplest causal model by approximating, it follows that replacing any model fragment in $M$ by an immediate *approximation* results in a model that is not complete. Hence, Theorem 5.5.1 tells us that no coherent model that is simpler than $M$ can have a model fragment that is an *approximation* of one of the model fragments in $M$. Hence, since $M' \leq M$, it follows that $M' \subseteq M$.

Let $M$ be any simplest causal model by approximating. We will now construct a set $\mathcal{H}$ of propositional horn constraints that must be satisfied by any causal model that is simpler than $M$. $\mathcal{H}$ will allow us to construct a minimal causal model that is simpler than $M$.

[3] Note that, Theorem 5.3.1 is sufficient, since we are only simplifying by approximating.

**Definition 5.11.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies all the restrictions in Definition 5.8, and let causes$(p, q)$ be the expected behavior. Let $M \subseteq \mathcal{M}$ be any simplest causal model by approximating, and let $F : E(M) \rightarrow P(M)$ be any onto causal mapping. Let $\mathcal{H}$ be a set of propositional horn constraints defined as follows:*

1. *For each constraint of the form*

   $$m_1 \wedge m_2 \wedge \ldots \wedge m_n \Rightarrow A$$

   *in $\mathcal{C}$, where each $m_i \in M$, $1 \leq i \leq n$, $\mathcal{H}$ contains the horn constraint*

   $$m_1 \wedge m_2 \wedge \ldots \wedge m_n \Rightarrow m$$

   *where $m = A \cap M$, i.e., $m$ is the model fragment that is both in $A$ and $M$;*
2. *Let $p_1, p_2 \in P(M)$ be parameters and let $e_1, e_2 \in E(M)$ be equations such that $F(e_1) = p_1$ and $F(e_2) = p_2$. Let $m_1, m_2 \in M$ be model fragments such that $e_1 \in m_1$ and $e_2 \in m_2$. If $p_1 \in P(e_2)$, so that $p_2$ directly causally depends on $p_1$, then $\mathcal{H}$ contains the constraint*

   $$m_2 \Rightarrow m_1$$

3. *Let $e_q \in E(M)$ be an equation such that $F(e_q) = q$, and let $m_q \in M$ be a model fragment such that $e_q \in m_q$. Then $\mathcal{H}$ contains the constraint*

   $$m_q$$

We now show that any subset of $M$ is a causal model if and only if it satisfies all the constraints in $\mathcal{H}$.

**Lemma 5.7.7.** *Let $M$, $F$, $\mathcal{H}$, $p$, $q$, $e_q$, and $m_q$ be as in Definition 5.11. Let $M' \subseteq M$ be any model. Then $M'$ is a causal model if and only if $M'$ satisfies all the constraints in $\mathcal{H}$.*

*Proof.* ($\Rightarrow$) First, let us assume that $M'$ is a causal model. We show that $M'$ satisfies all the constraints in $\mathcal{H}$.

Since $M'$ is a causal model, it must satisfy all the constraints in $\mathcal{C}$. Hence, since $M' \subseteq M$, it follows that $M'$ must satisfy all the constraints in $\mathcal{H}$ defined under point 1 of Definition 5.11.

Since $M' \subseteq M$, it follows that $E(M') \subseteq E(M)$. Hence, let $F' : E(M') \rightarrow P(M')$ be the restriction of $F$ to the equations in $E(M')$. Since $M'$ is a causal model, it follows that $F'$ is an onto causal mapping.

Let $p_1$, $p_2$, $e_1$, $e_2$, $m_1$, and $m_2$ be as in point 2 of Definition 5.11. If $m_2 \in M'$, then it follows that $e_2 \in E(M')$. Hence, since $p_1 \in P(e_2)$, it follows that $p_1 \in P(M')$. Since $F'$ is an onto causal mapping, and since $F'$ is a restriction of $F$, it follows that $e_1 \in E(M')$, with $F'(e_1) = p_1$. Hence, it follows that $m_1 \in M'$. Hence, the constraint $m_2 \Rightarrow m_1$ in $\mathcal{H}$ is satisfied. Hence, the constraints in $\mathcal{H}$, defined under point 2 of Definition 5.11, are satisfied.

Since $M'$ is a causal model, $q \in P(M')$. Since $F'$ is an onto causal mapping, and since $F'$ is a restriction of $F$, it follows that $e_q \in E(M')$. Hence, it follows that $m_q \in M'$. Hence, the constraint in $\mathcal{H}$, defined under point 3 of Definition 5.11, is satisfied.

Hence, $M'$ satisfies all the constraints in $\mathcal{H}$.

($\Leftarrow$) Let us now assume that $M'$ satisfies all the constraints in $\mathcal{H}$. We now show that $M'$ is a causal model.

Since $M'$ satisfies all the constraints in $\mathcal{H}$ defined under point 1, it follows that $M'$ satisfies all the constraints in $\mathcal{C}$.

We now show that the equations of $M'$ are complete. Since $M' \subseteq M$, it follows that $E(M') \subseteq E(M)$. Hence, let the causal mapping $F' : E(M') \rightarrow P(M')$ be the restriction of $F$ to the equations in $E(M')$. We now show that $F'$ is an onto causal mapping.

Let $p_1 \in P(M')$ be a parameter. We show that there is an equation $e_1 \in E(M')$, such that $F'(e_1) = p_1$. Let $e_2 \in E(M')$ be an equation such that $p_1 \in P(e_2)$, and let $m_2 \in M'$ be a model fragment such that $e_2 \in m_2$. Since $M' \subseteq M$, it follows that $e_2 \in E(M)$, and $m_2 \in M$. Since $F$ is an onto mapping, let $e_1 \in E(M)$ be an equation such that $F(e_1) = p_1$, and let $m_1 \in M$ be a model fragment such that $e_1 \in m_1$. Hence, from point 2 in Definition 5.11, $\mathcal{H}$ contains the constraint $m_2 \Rightarrow m_1$. Since $M'$ satisfies all the constraints in $\mathcal{H}$, and since $m_2 \in M'$, it follows that $m_1 \in M'$. Hence, $e_1 \in E(M')$ with $F'(e_1) = p_1$. Hence, $M'$ is a complete model.

It is easy to see from the above discussion that if $p_2$ directly causally depends on $p_1$ according to $F$, i.e., $(p_1, p_2) \in C_F$, and if $p_2 \in P(M')$, then $p_1 \in P(M')$ and $(p_1, p_2) \in C_{F'}$. We use this observation to show that $M'$ is a causal model, i.e., $(p, q) \in C(E(M'))$.

Since $M$ is a causal model, it follows that $(p, q) \in C(E(M))$. Hence, there exists a sequence of parameters $p_0, p_1, \ldots, p_n \in P(M)$ such that $p_0 = p$, $p_n = q$, and $(p_i, p_{i+1}) \in C_F$, $0 \leq i \leq (n-1)$. Since $M'$ satisfies the constraint under point 3 of Definition 5.11, it means that $m_q \in M'$, and hence $q \in P(M')$. Hence, from the observations in the previous paragraph, a simple inductive argument shows that $(p_i, p_{i+1}) \in C_{F'}$, $0 \leq i \leq (n-1)$. Hence, by transitivity, $(p, q) \in C(E(M'))$.

Hence, $M'$ is a causal model. Hence, $M'$ is a causal model if and only if $M'$ satisfies all the constraints in $\mathcal{H}$.

In conjunction with Lemma 5.7.6, the above lemma tells us that, if $M$ is a simplest causal model by approximating, then a minimal causal model that is simpler than $M$ is just a smallest subset of $M$ that satisfies all the constraints in $\mathcal{H}$. Since all the constraints in $\mathcal{H}$ are horn clauses, it is easy to show that there is exactly one such minimal causal model.[4] Given the set $\mathcal{H}$ of propositional horn constraints, the minimal causal model can be computed

[4] This result is just a special case of the more general result from logic programming, that a consistent set of horn clauses has a unique minimal Herbrand model [van Emden and Kowalski, 1976].

using a procedure that is completely analogous to *boolean constraint propagation* (BCP) [McAllester, 1980]. Figure 5.8 describes an algorithm, based on BCP, for finding a minimal causal model that is simpler $M$.

```
function simplify-by-dropping(M,I)
  /* M is assumed to be a simplest causal model by approximating */
  Construct H from M and I as described in Definition 5.11
  result ← nil
  while there exists a constraint h ∈ H such that
          the model fragments in the antecedent of h are in result and
          the model fragment in the consequent of h is not in result do
      Add the model fragment in the consequent of h to result
  endwhile
  return result
end
```

**Fig. 5.8.** Simplifying a model by dropping model fragments.

BCP runs in time linear in the number of constraints [McAllester, 1980]. Hence, we have the following:

**Lemma 5.7.8.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies all the restrictions in Definition 5.8, and let $M \subseteq \mathcal{M}$ be a simplest causal model by approximating. Then simplify-by-dropping$(M,\mathcal{I})$ returns a minimal causal model of $\mathcal{I}$ in time polynomial in the size of $\mathcal{I}$.*

*Proof.* It is easy to see that *simplify-by-dropping*$(M,\mathcal{I})$ returns a model that satisfies all the constraints in $\mathcal{H}$. Hence, it returns a causal model. It is also easy to see that the returned model is a minimal causal model, because a model fragment is added to *result* if and only if it *has* to be added to satisfy some constraint in $\mathcal{H}$.

From the definition of $\mathcal{H}$ (Definition 5.11), it is easy to see that $\mathcal{H}$ can be computed from $M$ and $\mathcal{I}$ in time polynomial in the size of $\mathcal{I}$. The **while** loop in *simplify-by-dropping* is identical to BCP, and hence terminates in time linear in the number of constraints in $\mathcal{H}$ [McAllester, 1980]. Hence, *simplify-by-dropping*$(M,\mathcal{I})$ returns a minimal causal model of $\mathcal{I}$ in time polynomial in the size of $\mathcal{I}$.

Hence, we can efficiently identify a minimal causal model by first finding a simplest causal modely by approximating, as described in Section 5.7.1, and then finding a minimal causal model using *simplify-by-dropping*.

**Theorem 5.7.1.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem that satisfies all the restrictions in Definition 5.8. A minimal causal model of $\mathcal{I}$ can be found in time polynomial in the size of $\mathcal{I}$.*

*Proof.* Lemma 5.7.5 tells us that a simplest causal model by approximating can be found in polynomial time. Lemma 5.7.8 tells us that this simplest

causal model by approximating can be used to find a minimal causal model in polynomial time. Hence, a minimal causal model of $\mathcal{I}$ can be found in time polynomial in the size of $\mathcal{I}$.

## 5.8 Summary

In this chapter we identified a special case of the MINIMAL CAUSAL MODEL problem for which a minimal causal model could be found efficiently. We started, in Section 5.1, by introducing the upward failure property, which states that if a coherent model is not a causal model, then no simpler model is a causal model. We showed that if (a) the upward failure property is satisfied; and (b) the immediate simplifications of a coherent model can be generated in polynomial time; then a minimal causal model can be found in polynomial time using the function *find-minimal-causal-model* shown in Figure 5.1. Unfortunately, in general, it is difficult to decide whether or not the upward failure property is satisfied, and whether or not coherent models have a polynomial number of immediate simplifications. Hence, the rest of the chapter focuses on finding efficient characterizations of these properties.

Section 5.3 introduced a new class of approximations, called causal approximations, that are commonly found in modeling the physical world. When all the *approximations* are causal approximations, the causal relations entailed by a model decrease monotonically as model fragments are replaced by their *approximations*. Hence, if a model does not explain the expected behavior, the simpler model, with a subset of causal relations, also does not explain the expected behavior. In addition, causal approximations are particularly useful because they are commonly found in modeling the physical world. Appendix A gives a list of commonly used approximations, all of which are causal approximations.

Section 5.4 introduced the ownership constraints. These constraints ensure that coherent models have model fragments from all possibly relevant assumption classes. The ownership constraints, in conjunction with causal approximations, are sufficient to ensure that the upward failure property is satisfied. Section 5.5 then showed that if the model fragments of a coherent model cannot be individually approximated, then there is no point approximating them two or more at a time.

Section 5.6 introduced a syntactic restriction on the expressive power of the propositional coherence constraints in $\mathcal{C}$. The constraints are restricted to have the form specified in Equation 5.5.

Finally, Section 5.7 put all the restrictions together, and presented an efficient algorithm for finding a minimal causal model. Definition 5.8 summarizes all the restrictions introduced in this chapter. The efficient algorithm for finding a minimal causal model is a two step procedure. First, a simplest model is found by simplifying the most accurate model as much as possible, without dropping any assumption classes. Second, a minimal causal model is

found by dropping all unnecessary model fragments from the model identified in the first step.

# 6. Differential equations

In this chapter we generalize the results of the previous chapter to include models involving differential equations. The complication introduced by the use of differential equations is that the causal ordering is not generated from the set $E$ of equations, but rather from the set $ic(E)$, the integration completion of $E$. The results of the previous chapter do not take into account the additional *int* equations in $ic(E)$. We remedy that situation in this chapter.

The central result of the previous chapter was the efficient model selection algorithm developed in Section 5.7. A careful analysis of the proofs of the theorems and lemmas of Section 5.7 reveals that their correctness is based on the results of Section 5.1, Theorem 5.4.1, and Theorem 5.5.1. Of these results, only Theorems 5.4.1 and 5.5.1 were based on the assumption that the models did not involve differential equations. (The former proves that the upward failure property is satisfied, while the latter proves that model fragments can be individually approximated.)

A further analysis of the proofs of these theorems reveals that Theorem 5.4.1 depends on Theorem 5.3.1, which proves the restricted version of the upward failure property, and Theorem 5.5.1 depends on Lemma 5.3.1, which guarantees the existence of a local extension. Furthermore, the assumption that the models do not involve differential equations is restricted to these two results. Hence, in this chapter, we will generalize Theorem 5.3.1 and Lemma 5.3.1, so that the efficient model selection algorithm developed in Section 5.7 can be used for models involving differential equations.

Section 6.1 introduces a canonical form for differential equations, and discusses its consequences. This canonical form is similar to the one commonly used in numerical integration. Section 6.2 discusses the different ways of approximating a differential equation: *exogenizing* and *equilibrating*. These two approximation methods are discussed in detail in Sections 6.3 and 6.4, respectively. The latter section concludes with an updated definition of a causal approximation. Section 6.5 uses this updated causal approximation definition to generalize Theorem 5.3.1 to models with differential equations.

Finally, Section 6.6 generalizes Lemma 5.3.1. It first shows that, in general, a coherent model can have an exponential number of immediate simplifications that use model fragments from the same assumption classes. It then introduces *locally self-regulating* parameters, and shows that when all

the parameters are locally self-regulating, Lemma 5.3.1 can be generalized to model fragments with differential equations.

## 6.1 Canonical form

For many purposes, e.g., numerical integration [Dahlquist *et al.*, 1974] and causal ordering as defined in [Iwasaki, 1988], sets of differential equations are required to be in *canonical form*. A set of first-order differential equations is in canonical form if each derivative occurs in exactly one equation. For our purposes, we weaken this condition slightly. We shall say that a set of first-order differential equations is in canonical form if each derivative can be causally determined by exactly one equation. Hence, we allow a derivative to occur in more than one equation, though exactly one equation can causally determine it. To ensure that the equations of all device models are in canonical form, we assume that the set of model fragments under consideration are in canonical form:

**Definition 6.1 (Canonical form).** *A set of model fragments is said to be in canonical form if and only if the following conditions are satisfied:*

1. *All derivatives are local parameters; and*
2. *If derivative $dp/dt$ is local to model fragment $m$, then $dp/dt$ can be causally determined by exactly one equation in $m$.*

Condition 1 ensures that derivatives can be determined by the equations of model fragments in exactly one assumption class, while condition 2 ensures that exactly one equation in each such model fragment can determine it. Hence, the above restrictions ensure that the equations of all device models are in canonical form. Hence, we place the following restriction on $\mathcal{I}$:

– The set of model fragments in $\mathcal{M}$ are in canonical form.

An important consequence of the above restriction is as follows. Let $dp/dt$ be a derivative that is local to model fragment $m$, and let $e$ be the equation of $m$ that can causally determine $dp/dt$. The integration completion of any set of equations that includes $e$ will introduce the equation $int(\boldsymbol{p}, dp/dt)$. Since $dp/dt$ is local to $m$, this is *exactly equivalent* to augmenting the equations of $m$ with the equation $int(\boldsymbol{p}, dp/dt)$. In fact, if we do this for all derivatives in all model fragments, there is no need to explicitly apply the integration completion operator to a set of equations—the model fragments would already include the equations introduced by the integration completion.

The advantage of the above viewpoint is as follows. Let $m_1$ and $m_2$ be model fragments such that $m_2$ is a causal approximation of $m_1$, as in Definition 5.3. Suppose that every derivative that is local to $m_1$ is also local to $m_2$. If we were to augment the equations of $m_1$ and $m_2$ with the *int* equations, as above, it is easy to see that the same set of equations are added to

both $m_1$ and $m_2$. Hence, it is straightforward to extend the correspondence mapping between the equations of $m_1$ and $m_2$, to include these additional equations: if $dp/dt$ is a derivative local to both $m_1$ and $m_2$, the correspondence mapping is extended to map the equation $int(\boldsymbol{p}, dp/dt)$ in $m_2$ to the equation $int(\boldsymbol{p}, dp/dt)$ in $m_1$. It is easy to see that this extension satisfies all the conditions of a correspondence mapping. Hence, if the equations of $m_1$ and $m_2$ are augmented as above, $m_2$ is still a causal approximation of $m_1$.

In summary, if the same set of derivatives are local to $m_1$ and $m_2$, whenever $m_2$ is a causal approximation of $m_1$, then $m_2$ remains a causal approximation of $m_1$ even after we augment the equations of all model fragments as above. This means that all the results that we have proved in the previous chapter remain true, even if we include differential equations!

However, it may not always be the case that the same set of derivatives are local to $m_1$ and $m_2$. We now discuss this important case.

## 6.2 Approximating differential equations

Let $m_1$ and $m_2$ be model fragments such that $m_2$ is an approximation of $m_1$. Let $dp/dt$ be a derivative that is local to $m_1$, but not local to $m_2$. Intuitively, $m_1$ describes a phenomenon using a dynamic model, i.e., a model involving differential equations, while $m_2$ approximates this description by describing the phenomena using a static, or equilibrium, model. We will now consider two types of approximations, called *exogenizing* and *equilibrating*, that convert a dynamic description of a phenomenon into an equilibrium description. These two types of approximations were identified by Iwasaki in [Iwasaki, 1988].

The basic idea behind these approximation techniques is that, when viewed at the right time-scale, a dynamic system can appear to be in equilibrium. Iwasaki considers two cases:

1. The dynamic behavior of a system is slow compared to the time-scale of interest. For example, car brakes wear out over a period of years, while the time-scale of interest may be only a matter of hours or days. Hence, at this time scale, the thickness of the brake pads can be assumed to be constant. Assuming that a parameter does not change, because its dynamic behavior is much slower than the time-scale of interest, is called *exogenizing*.
2. The dynamic behavior of a system is fast compared to the time-scale of interest. For example, the dynamic behavior of the temperature of a small object lasts only a few minutes, after which it reaches thermal equilibrium with its environment. Hence, at a time-scale of hours, the object's temperature can be assumed to "instantaneously" track the environment's temperature. Assuming that a parameter is always in equilibrium, because its dynamic behavior is much faster than the time-scale of interest, is called *equilibrating*.

Let us now consider the effect that exogenizing and equilibrating have on the differential equations describing a dynamic phenomenon. Let

$$\frac{dp}{dt} = f(\bar{P}) \tag{6.1}$$

be a differential equation, where $f$ is some function of the set $\bar{P}$ of parameters, such that $\bar{P}$ does not contain $dp/dt$. Exogenizing the equation involves assuming that, at the time scale of interest, the value of $p$ does not change significantly. Hence, the above equation is replaced by

$$exogenous(p)$$

Equilibrating Equation 6.1 involves assuming that $p$ "quickly" reaches equilibrium, and hence $dp/dt$ is always 0. Hence, we replace Equation 6.1 by

$$0 = f(\bar{P})$$

More generally, we have the following definitions of exogenizing and equilibrating:

**Definition 6.2 (Exogenizing and equilibrating).** *Let $e$ be a differential equation that can causally determine the derivative $dp/dt$, i.e., $dp/dt \in P_c(e)$.*

- *Exogenizing $e$ involves replacing it with the equation $exogenous(p)$.*
- *Equilibrating $e$ involves replacing it with an equation $e'$ such that (a) $dp/dt \notin P(e')$; (b) $P(e') \subseteq P(e)$; and (c) $P_c(e') \subseteq P_c(e)$.*

Note that, in both exogenizing and equilibrating, the resulting equation does not contain $dp/dt$. Note, also, the slight generalization in our definition of equilibration—we do not require $e'$ to be the result of modifying $e$ by replacing $dp/dt$ with 0.

For example, the equation describing the dynamic behavior of the temperature of an object is:

$$\frac{dT}{dt} = CF \tag{6.2}$$

where $T$ is the object's temperature, $C$ is its heat capacity, and $F$ is the net heat flow into the object. Exogenizing this equation results in:

$$exogenous(T)$$

which states that the temperature is constant. Equilibrating that equation results in:

$$F = 0$$

which states that the object's temperature always changes to ensure that the net heat flow into the object is 0.

We now investigate the effect that these types of approximations have on the results of the previous chapter.

## 6.3 Exogenizing differential equations

Let $m_1$ and $m_2$ be model fragments such that $m_2$ is an approximation of $m_1$. Let $dp/dt$ be a derivative that is local to $m_1$, but not local to $m_2$. Let $e \in m_1$ be the equation that can causally determine $dp/dt$. Assume that $e$ has been exogenized in $m_2$, i.e., $e$ has been replaced by $exogenous(\boldsymbol{p})$ in $m_2$.

Now, let us assume that, if $m_1$ had not contained $e$ and $m_2$ had not contained $exogenous(\boldsymbol{p})$, then $m_2$ would have been a causal approximation of $m_1$ according to Definition 5.3. Let $G$ be the correspondence mapping of this hypothetical causal approximation, and let $L$ be the local causal mapping with respect to $G$.

We now claim that $m_2$ (with $exogenous(\boldsymbol{p})$) is a causal approximation of $m_1$ (with $e$), if we augment the equations of $m_1$ to include the equation $int(\boldsymbol{p}, dp/dt)$. In particular, the correspondence mapping $G$ can be extended to map $exogenous(\boldsymbol{p})$ in $m_2$ to $int(\boldsymbol{p}, dp/dt)$ in $m_1$, and the local causal mapping $L$ can be extended to map the equation $e$ to the parameter $dp/dt$ which is local to $m_1$, but not local to $m_2$. But we have already seen that $m_1$ can be viewed as containing the equation $int(\boldsymbol{p}, dp/dt)$. Hence, $m_2$ is a causal approximation of $m_1$.

More generally, we can modify Definition 5.3 so that the domain and range of the correspondence mapping are not the equations of the model fragments, but the integration completion of the equations of the model fragments. With this modifications, it is easy to see from the above argument that, exogenizing of differential equations is a causal approximation. Hence, all the results of the previous chapter continue to hold, even when we allow differential equations to be approximated by exogenizing.

In the next section, where we discuss equilibration, we will give an updated definition of a causal approximation that incorporates the above change.

## 6.4 Equilibrating differential equations

Let $m_1$ and $m_2$ be model fragments such that $m_2$ is an approximation of $m_1$. Let $dp/dt$ be a derivative that is local to $m_1$, but not local to $m_2$. Let $e_1 \in m_1$ be the equation that can causally determine $dp/dt$. Assume that $e_1$ has been equilibrated in $m_2$, and let $e_2$ be the equilibrated version of $e_1$.

Let us now assume that, if $m_1$ had not contained $e_1$ and $m_2$ had not contained $e_2$, then $m_2$ would have been a causal approximation of $m_1$ according to Definition 5.3. Let $G$ be the correspondence mapping of this hypothetical causal approximation, and let $L$ be the local causal mapping with respect to $G$.

It is now tempting to proceed as we did in the case of exogenizing. In particular, we can argue that, when $m_1$ does contain $e_1$ (and hence implicitly $int(\boldsymbol{p}, dp/dt)$) and $m_2$ does contain $e_2$, we can extend $G$ by mapping $e_2$ to $e_1$. Similarly, we can extend $L$ by mapping $int(\boldsymbol{p}, dp/dt)$ to $p$.

But $L$ can map $int(\mathbf{p}, dp/dt)$ to $p$ only if $p$ is local to $m_1$, but not local to $m_2$. This means that $p$ cannot be determined by equilibrium equations—$p$ can only be determined by an *int* equation, or $p$ can be constant as a result of exogenizing equations like $e$. This is an undesirable state of affairs. For example, the equilibrium temperature of an object cannot be determined by equilibrating Equation 6.2.

Hence, we do not take the above approach. Rather, we show that even if we do not extend $L$ to map $int(\mathbf{p}, dp/dt)$ to $p$, but we do extend $G$ to map $e_2$ to $e_1$, then Theorem 5.3.1 continues to remain true. Before we do this, we update the definition of a causal approximation, incorporating the various changes that we have discussed.

**Definition 6.3 (Causal approximation with differential equations).** *A model fragment $m_2$ is a causal approximation of a model fragment $m_1$ if and only if:*

1. *$m_2$ is an approximation of $m_1$;*
2. *if $dp/dt$ is a derivative that is local to $m_1$ but not local to $m_2$, then the equation that can causally determine $dp/dt$ in $m_1$ is either exogenized or equilibrated in $m_2$;*
3. *There exists a 1-1 mapping $G : ic(m_2) \rightarrow ic(m_1)$ that satisfies the following properties:*
   a) *for each $e \in ic(m_2)$, $P(e) \subseteq P(G(e))$, and $P_c(e) \subseteq P_c(G(e))$;*
   b) *if $m_1$ contains a differential equation $e$ that can causally determine derivative $dp/dt$, and if $exogenous(\mathbf{p}) \in m_2$ is the exogenized version of $e$, then $G(exogenous(\mathbf{p})) = int(\mathbf{p}, dp/dt)$; and*
   c) *if $m_1$ contains a differential equation $e$ that can causally determine derivative $dp/dt$, and if $e' \in m_2$ is the equilibrated version of $e$, then $G(e') = e$.*

   *$G$ is called a correspondence mapping, and $e$ and $G(e)$ are called corresponding equations.*
4. *Let $E^*$ be the set of equations in $m_1$ that have no corresponding equations according to $G$. Let $P^*$ denote the set of parameters such that $q \in P^*$ if and only if*
   a) *$q$ is local to $m_1$;*
   b) *$q$ is not local to $m_2$; and*
   c) *if $p$ is a parameter, $dp/dt$ is its derivative, and the equation that can causally determine $dp/dt$ in $m_1$ has been equilibrated in $m_2$, then $q$ is neither $p$ nor $dp/dt$.*

   *Then there exists an onto causal mapping $L : E^* \rightarrow P^*$. $L$ is called a local causal mapping with respect to $G$.*

It is easy to verify that when $m_1$ and $m_2$ have no differential equations, the above definition reduces to Definition 5.3. The changes are as follows. Condition 2 ensures that the only way differential equations can be approximated is by either exogenizing them or equilibrating them. Condition 3b

incorporates the change to $G$ discussed in Section 6.3, while Condition 3c incorporates the restriction to $G$ discussed earlier in this section.

Condition 4 ensures that the "tempting" extension to $L$, discussed earlier in this section, is disallowed. In particular, note that $E^*$ contains only equations in $m_1$. Hence, equations of the form $int(\boldsymbol{p}, dp/dt)$ are not in the domain of $L$.

Condition 4c ensures that parameter $p$ and derivative $dp/dt$, where the equation $e$ that can causally determine $dp/dt$ has been equilibrated in $m_2$, are not in the co-domain of $L$. This is necessary because $e$ is the only equation in $m_1$ that can causally determine $dp/dt$, and $G$ already matches $e$ to its equilibrated version. Hence, no equation in the domain of $L$ can causally determine $dp/dt$. Similarly, if $p$ were local to $m_1$, but not local to $m_2$, then it would be okay to be "tempted" as discussed earlier, and extend $L$ to map $int(\boldsymbol{p}, dp/dt)$ to $p$. However, since $int(\boldsymbol{p}, dp/dt)$ is not in the domain of $L$, it makes sense to leave $p$ out of the co-domain of $L$.

## 6.5 Monotonicity of causal relations

In this section we show that Theorem 5.3.1 remains true with the above updated definition of a causal approximation. We start by proving a number of subsidiary lemmas.

Recall that a causal mapping $H : E \rightarrow P$ is said to be *partial* if and only if $H$ is not defined for each equation in $E$. In the next two lemmas, we define a condition under which a partial causal mapping can be extended to a causal mapping defined over more equations, such that the resulting causal mapping entails a superset of causal relations. We will first motivate these two lemmas with an example.

Let $E$ be a set of equations:

$$E = \{\boldsymbol{du/dt} = \boldsymbol{v}, \boldsymbol{dv/dt} = \boldsymbol{u}, \boldsymbol{dw/dt} = \boldsymbol{w}\} \tag{6.3}$$

and let $H : ic(E) \rightarrow P(E)$ and $H' : ic(E) \rightarrow P(E)$ be partial causal mappings as defined in Figure 6.1.

| | |
|---|---|
| $H(\boldsymbol{du/dt} = \boldsymbol{v}) = v$ | $H'(\boldsymbol{du/dt} = \boldsymbol{v}) = du/dt$ |
| $H(\boldsymbol{dv/dt} = \boldsymbol{u}) = u$ | $H'(\boldsymbol{dv/dt} = \boldsymbol{u}) = dv/dt$ |
| $H(\boldsymbol{dw/dt} = \boldsymbol{w}) = w$ | $H'(\boldsymbol{dw/dt} = \boldsymbol{w}) = w$ |
| | $H'(int(\boldsymbol{u}, du/dt)) = u$ |
| | $H'(int(\boldsymbol{v}, dv/dt)) = v$ |

**Fig. 6.1.** Partial causal mappings $H$ and $H'$.

Note that $H'$ is defined over more equations than $H$. Figure 6.2 shows the bipartite graphs representing $E$, and the two causal mappings $H$ and $H'$. The bold lines with arrowheads at each end represent the two causal mappings.

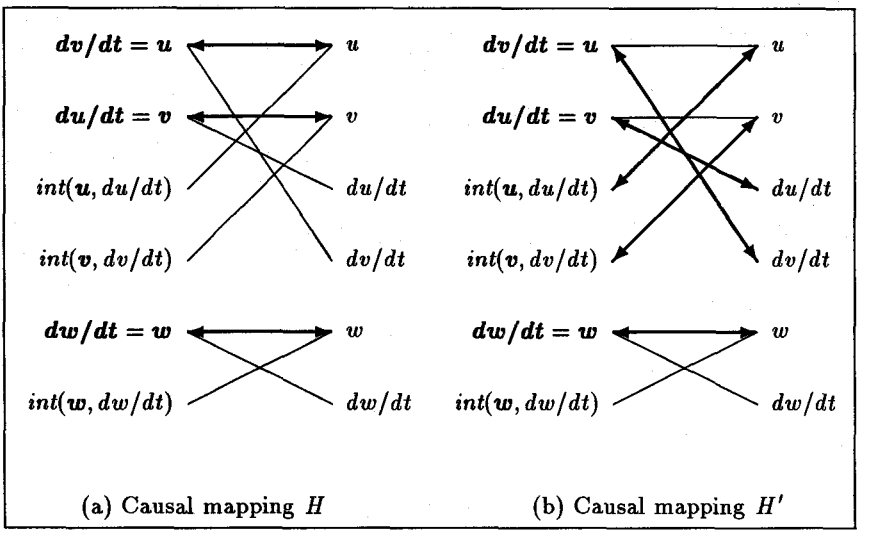



**Fig. 6.2.** A motivating example

Now consider $S$, an alternating sequence of equations and parameters, defined as follows:

$$S = \{int(\boldsymbol{v}, dv/dt), v, \boldsymbol{du/dt = v}, du/dt, int(\boldsymbol{u}, du/dt), u, \\ \boldsymbol{dv/dt = u}, dv/dt\} \tag{6.4}$$

One can verify that the partial causal mapping $H'$ can be derived from the partial causal mapping $H$ using $S$ as follows:

- If equation $e$ is followed by parameter $p$ in the sequence $S$, then $H'(e) = p$. For example, $int(\boldsymbol{v}, dv/dt)$ is followed by $v$, and hence $H'(int(\boldsymbol{v}, dv/dt)) = v$.
- If equation $e$ is not in the sequence $S$, and if $H$ is defined for $e$, then $H'(e) = H(e)$.

i.e., if we view $S$ as a path in Figure 6.2a, then Figure 6.2b is a result of the following operations: (a) if an edge in $S$ is bold with arrows, then convert it into a light edge without arrows; and (b) if an edge in $S$ is light without arrows, then convert it into a bold edge with arrows.

One can also verify the following two properties: (a) according to the partial causal mapping $H'$, the parameters appearing in the sequence $S$ (i.e., $v$, $du/dt$, $u$, and $dv/dt$) are causally dependent upon each other; and (b) the direct causal dependencies entailed by $H$ are a subset of all the causal dependencies entailed by $H'$, i.e., $C_H \subseteq tc(C_{H'})$. In the next two lemmas, we give a general characterization of such partial causal mappings $H$ and $H'$, and sequence $S$, and show that the above two properties hold. These lemmas will provide us with a mechanism to extend a partial causal mapping like $H$.

**Lemma 6.5.1.** *Let $E$ be a complete set of equations, and let $H : ic(E) \rightarrow P(E)$ be a partial causal mapping. Let $S$ be an alternating sequence of equations and parameters:*

$$S = \{e_1, p_1, e_2, p_2, \ldots, e_n, p_n\}, \text{ for some } n \geq 1$$

*such that*

1. *No equation or parameter is repeated in $S$;*
2. *$p_i \in P_c(e_i)$, for $1 \leq i \leq n$;*
3. *$H$ is undefined for $e_1$;*
4. *$p_n$ is not in the range of $H$, i.e., there is no $e \in ic(E)$ such that $H$ is defined for $e$ and $H(e) = p_n$;*
5. *for each $e_i$, $2 \leq i \leq n$, if $H$ is defined for $e_i$, then $H(e_i) = p_{i-1}$. If $H$ is not defined for $e_i$, then $p_{i-1} \in P(e_i)$; and*
6. *$p_n \in P(e_1)$.*

*Let $H' : ic(E) \rightarrow P(E)$ be a partial causal mapping defined as follows:*

$$H'(e) = \begin{cases} p_i & \text{if } e \text{ is } e_i,\ 1 \leq i \leq n \\ H(e) & \text{if } e \text{ is not } e_i,\ 1 \leq i \leq n, \text{ and } H \text{ is defined for } e \\ \text{undefined} & \text{otherwise} \end{cases}$$

*Then, according to $H'$, each parameter in $S$ is causally dependent on every other parameter in $S$, i.e., for every $p_i, p_j$, $1 \leq i, j \leq n$, $(p_i, p_j) \in tc(C_{H'})$.*

*Proof.* Let $p_k, p_{k+1}$, $1 \leq k \leq n-1$, be any two consecutive parameters in the sequence $S$. We first show that

$$(p_k, p_{k+1}) \in C_{H'}, \text{ for } 1 \leq k \leq n-1 \tag{6.5}$$

i.e., according to $H'$, $p_{k+1}$ directly causally depends on $p_k$. From the definition of $H'$, we know that $H'(e_{k+1}) = p_{k+1}$. Hence, we need only show that $p_k \in P(e_{k+1})$.

Condition 5 in the statement of the lemma tells us that if $H$ is defined for $e_{k+1}$, then $H(e_{k+1}) = p_k$. Since $H$ is a causal mapping, it follows that $p_k \in P(e_{k+1})$. If $H$ is not defined for $e_{k+1}$, then condition 5 tells us that $p_k \in P(e_{k+1})$. Hence, in either case $p_k \in P(e_{k+1})$. Hence, $(p_k, p_{k+1}) \in C_{H'}$.

Now, condition 6 in the statement of the lemma tells us that $p_n \in P(e_1)$. Since $H'(e_1) = p_1$, it follows that $(p_n, p_1) \in C_{H'}$. Hence, using transitivity and the result shown in Equation 6.5, we can conclude that for every $p_i, p_j$, $1 \leq i, j \leq n$, $(p_i, p_j) \in tc(C_{H'})$.

**Lemma 6.5.2.** *Let $E$, $S$, $H$, and $H'$ be as in Lemma 6.5.1. Then $C_H \subseteq tc(C_{H'})$, i.e., the direct causal dependencies entailed by $H$ are a subset of the transitive closure of the direct causal dependencies entailed by $H'$.*

*Proof.* Let $(q_1, q_2) \in C_H$. To prove this lemma, we need only show that $(q_1, q_2) \in tc(C_{H'})$.

Since $(q_1, q_2) \in C_H$, it follows that there is an equation $e \in ic(E)$, such that $H(e) = q_2$ and $q_1 \in P(e)$. If $e$ is not an equation in the sequence $S$, then by the definition of $H'$, it follows that $H'(e) = H(e) = q_2$. Therefore, $(q_1, q_2) \in C_{H'}$, and hence $(q_1, q_2) \in tc(C_{H'})$.

On the other hand, if $e$ is an equation in the sequence $S$, then let $H'(e) = p$, where $p$ is some parameter in the sequence $S$. Since $H$ is defined for $e$, and $H(e) = q_2$, condition 5 of Lemma 6.5.1 (and hence of this lemma) tells us that $q_2$ is also a parameter in the sequence $S$. Hence, Lemma 6.5.1 tells us that, according to $H'$, $p$ and $q_2$ are causally interdependent. In particular, we have

$$(p, q_2) \in tc(C_{H'}) \tag{6.6}$$

Now, since $q_1 \in P(e)$, we have two cases: either $p$ is $q_1$, or $p$ is not $q_1$. If $p$ is $q_1$, then Equation 6.6 directly tells us that $(q_1, q_2) \in tc(C_{H'})$. If $p$ is not $q_1$, then since $q_1 \in P(e)$, and $H'(e) = p$, it follows that $(q_1, p) \in C_{H'}$. Hence, in conjunction with Equation 6.6, we have $(q_1, q_2) \in tc(C_{H'})$.

Hence, in every case, we have $(q_1, q_2) \in tc(C_{H'})$. Hence, it follows that $C_H \subseteq tc(C_{H'})$.

The above two lemmas show that, under certain conditions, a partial causal mapping $H$, and an alternating sequence of equations and parameters $S$, can be used to extend $H$ to a causal mapping $H'$ that entails a superset of causal dependencies. We now investigate conditions on $H$ which ensure that that a sequence like $S$ exists, so that $H$ can be extended to $H'$. In particular, we introduce *augmentable* causal mappings:

**Definition 6.4 (Augmentable causal mapping).** *Let $E$ be a complete set of equations, and let $H : ic(E) \rightarrow P(E)$ be a partial causal mapping. $H$ is said to be augmentable with respect to $E$ if and only if*

1. *every equation $e \in ic(E)$, for which $H$ is not defined, is of the form $int(\boldsymbol{q}, dq/dt)$, for some $q$; and*
2. *every parameter $p \in P(E)$ not in the range of $H$, is of the form $dq/dt$, such that $int(\boldsymbol{q}, dq/dt) \in ic(E)$ and $H$ is not defined for $int(\boldsymbol{q}, dq/dt)$.*

*i.e., $H$ is not defined for some int equations, and the corresponding derivatives are the only parameters not in the range of $H$.*

Hence, an augmentable causal mapping $H$ is undefined only for equations of the form $int(\boldsymbol{q}, dq/dt)$, and the corresponding derivatives are the only parameters not in the range of $H$. One can verify that both the causal mappings $H$ and $H'$ shown in Figure 6.2 are augmentable causal mappings. We now show that if $H$ is an augmentable causal mapping, then there exists a sequence $S$ satisfying the conditions of the above two lemmas. This will ensure that $H$ can be extended to $H'$, as discussed earlier.

**Lemma 6.5.3.** *Let $E$ be a complete set of equations, and let $H : ic(E) \rightarrow P(E)$ be an augmentable causal mapping with respect to $E$. Then there exists an alternating sequence of equations and parameters:*

$$S = \{e_1, p_1, e_2, p_2, \ldots, e_n, p_n\}, \text{ for some } n \geq 1$$

*that satisfies conditions 1–6 in the statement of Lemma 6.5.1, with respect to the causal mapping $H$. In addition, let $int(\boldsymbol{q}, dq/dt)$ be any equation in $ic(E)$ for which $H$ is not defined. Then $int(\boldsymbol{q}, dq/dt)$ occurs in the sequence $S$ if and only if the parameter $dq/dt$ occurs in the sequence $S$.*

*Proof.* The proof of this lemma is based on an understanding of the algorithm for finding maximum matchings in bipartite graphs described in Appendix D.

Let $E_1 \subseteq ic(E)$ be the set of equations for which $H$ is defined, and let $P_1$ be the range of $H$. Let $E_2 = ic(E) \setminus E_1$ and let $P_2 = P(E) \setminus P_1$. Since $H$ is augmentable, it follows that $E_2$ and $P_2$ are not empty, and have the following form (where $m = |E_2| = |P_2|$):

$$E_2 = \bigcup_{1 \leq i \leq m} \{int(\boldsymbol{q_i}, dq_i/dt)\}$$

$$P_2 = \bigcup_{1 \leq i \leq m} \{dq_i/dt\}$$

Let $G = (X, Y, R)$ be the bipartite graph representing the equations in $ic(E)$ (see Definition 3.5). Since $E$ is complete, it follows that any maximum matching in $G$ is complete. Let $U$ be a matching in $G$ defined using $H$:

$$U = \{(e, H(e)) : e \in ic(E) \wedge H(e) \text{ is defined}\}$$

$U$ is clearly not complete because $H$ is defined only on the equations in $E_1$, and not on the equations in $E_2$. In addition, since the range of $H$ contains no parameters in $P_2$, it follows that no edge in $U$ is incident on any equation in $E_2$ or any parameter in $P_2$. In particular, no edge in $U$ is incident on the equation $int(\boldsymbol{q}_1, dq_1/dt)$ and on the parameter $dq_1/dt$.

For example, the set $U$ corresponding to the causal mapping $H$ defined in Figure 6.1 is the set

$$\{(\boldsymbol{dv/dt} = \boldsymbol{u}, u), (\boldsymbol{du/dt} = \boldsymbol{v}, v), (\boldsymbol{dw/dt} = \boldsymbol{w}, w)\}$$

This is shown graphically in Figure 6.2a, where $U$ is the set of bold edges with arrows at both ends.

Define a second bipartite graph $G' = (X, Y, R')$ that is exactly like $G$, except that $G'$ contains the following additional set of edges:

$$W = \bigcup_{2 \leq i \leq m} \{(int(\boldsymbol{q_i}, dq_i/dt), dq_i/dt)\}$$

i.e., the edges in $W$ connect *int* equations for which $H$ is undefined to the corresponding derivatives, but $W$ does not connect $int(\boldsymbol{q}_1, dq_1/dt)$ to $dq_1/dt$. Since $G'$ has more edges than $G$, but is otherwise identical to $G$, it follows

that any matching of $G$ is also a matching of $G'$. Hence, since $G$ contains a complete matching, it follows that $G'$ contains a complete matching.

For example, Figure 6.3 shows the graph $G'$ corresponding to the graph shown in Figure 6.2a, with $q_1$ being the parameter $v$. The bold edges with arrows correspond to the set $U$, while the bold edges without arrows correspond to the set $W$.

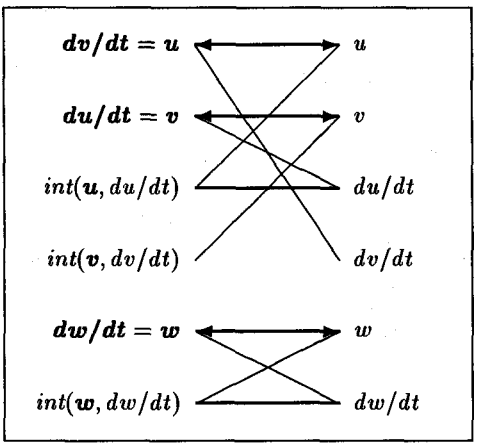



**Fig. 6.3.** Example of the graph $G'$

Let $V$ be the union of $U$ and $W$. Since no edge in $U$ is incident on an equation in $E_2$ or a parameter in $P_2$, it is easy to verify that $V$ is a matching in $G'$. However, $V$ is not a complete matching. In fact, the only two nodes in $G'$ that $V$ does not match are the nodes corresponding to equation $int(q_1, dq_1/dt)$ and parameter $dq_1/dt$. Since $G'$ contains a complete matching, it follows that there is an augmenting path in $G'$, with respect to $V$ (augmenting paths are defined in Definition D.3). This is a direct consequence of Lemma D.0.2.

For example, the following sequence is an augmenting path in the graph shown in Figure 6.3:

$$\{int(\boldsymbol{v}, dv/dt), v, \boldsymbol{du/dt = v}, du/dt, int(\boldsymbol{u}, du/dt), u, \\ \boldsymbol{dv/dt = u}, dv/dt\} \tag{6.7}$$

This sequence is the same as the one shown in Equation 6.4.

In general, let

$$S = (e_1, p_1, e_2, p_2, \ldots, e_n, p_n), \text{for some } n \geq 1$$

be such an augmenting path, where each $e_i$ is (an equation) in $X$ and each $p_i$ is (a parameter) in $Y$. We claim that $S$ satisfies conditions 1–6 in Lemma 6.5.1:

1. Since $S$ is an augmenting path, no node is repeated in $S$. Hence, no equation or parameter is repeated in $S$.

2. Since $S$ is an augmenting path, it follows that $(e_i, p_i)$, $1 \leq i \leq n$, is an edge in $R'$, such that $(e_i, p_i)$ is not in $V$. Since $W$ is the set of edges in $R'$ that are not in $R$, and $W \subseteq V$, it follows that $(e_i, p_i)$ is an edge in $R$. Hence, using Definition 3.5, we conclude that $p_i \in P_c(e_i)$, for $1 \leq i \leq n$.
3. Since $S$ is an augmenting path, no edge in $V$ is incident on $e_1$. Hence, $H$ is undefined for $e_1$.
4. Since $S$ is an augmenting path, no edge in $V$ is incident on $p_n$. Hence, $p_n$ is not in the range of $H$.
5. Since $S$ is an augmenting path, it follows that $(e_i, p_{i-1})$, $2 \leq i \leq n$, is an edge in $V$. Hence, either $H(e_i) = p_{i-1}$, or $(e_i, p_{i-1}) \in W$. If $(e_i, p_{i-1}) \in W$, then $H$ is not defined for $e_i$. Hence, $e_i$ is of the form $int(\boldsymbol{q}, dq/dt)$ for some $q$, and $p_{i-1}$ is $dq/dt$. Hence, $p_{i-1} \in P(e_i)$.
   Hence, for each $e_i$, $2 \leq i \leq n$, if $H$ is defined for $e_i$, then $H(e_i) = p_{i-1}$. If $H$ is not defined for $e_i$, then $p_{i-1} \in P(e_i)$.
6. Since $int(\boldsymbol{q}_1, dq_1/dt)$ and $dq_1/dt$ are the only two nodes on which an edge in $V$ is not incident, it follows that $e_1$ is $int(\boldsymbol{q}_1, dq_1/dt)$ and $p_n$ is $dq_1/dt$. Hence, $p_n \in P(e_1)$.

Hence, the augmenting path $S$ is an alternating sequence of equations and parameters that satisfies conditions 1–6 in Lemma 6.5.1.

Now we show that for any equation $int(\boldsymbol{q}, dq/dt)$ for which $H$ is not defined, the equation $int(\boldsymbol{q}, dq/dt)$ occurs in $S$ if and only if $dq/dt$ occurs in $S$. This clearly holds for $q = q_1$, since $int(\boldsymbol{q}_1, dq_1/dt)$ and $dq_1/dt$ are both in $S$. Now suppose that the equation $int(\boldsymbol{q}_i, dq_i/dt)$, $2 \leq i \leq m$, occurs in $S$, and let it be the equation $e_j$, for some $2 \leq j \leq n$. Since $S$ is an augmenting path, it follows that $(e_j, p_{j-1}) \in V$. But this is only possible if $p_{j-1}$ is $dq_i/dt$, since no edge in $U$ is incident on $int(\boldsymbol{q}_i, dq_i/dt)$. Hence $dq_i/dt$ occurs in $S$. A symmetric argument shows that if $dq_i/dt$ occurs in $S$ then $int(\boldsymbol{q}_i, dq_i/dt)$ occurs in $S$. Hence, if $H$ is not defined for $int(\boldsymbol{q}, dq/dt)$, then $int(\boldsymbol{q}, dq/dt)$ occurs in $S$ if and only if $dq/dt$ occurs in $S$.

The above lemma shows that if $H$ is augmentable, then a sequence $S$, with the right properties, exists such that $H$ can be extended. The next lemma shows that the resulting causal mapping is either complete or is itself augmentable.

**Lemma 6.5.4.** *Let $E$ be a complete set of equations, and let $H : ic(E) \rightarrow P(E)$ be an augmentable causal mapping. Then there exists a causal mapping $H' : ic(E) \rightarrow P(E)$ such that*

1. *$H'$ is defined on more equations in $ic(E)$ than $H$;*
2. *$C_H \subseteq tc(C_{H'})$; and*
3. *either $H'$ is complete, or $H'$ is augmentable with respect to $E$.*

*$H'$ is called an augmentation of $H$.*

*Proof.* For any augmentable causal mapping $H$, Lemma 6.5.3 tells us that there exists an alternating sequence of equations and parameters

$$S = \{e_1, p_1, e_2, p_2, \ldots, e_n, p_n\}$$

that satisfies conditions 1–6 of Lemma 6.5.1. Let $H'$ be defined from $H$ and $S$ in the same way that it was in the statement of Lemma 6.5.1:

$$H'(e) = \begin{cases} p_i & \text{if } e \text{ is } e_i,\ 1 \leq i \leq n \\ H(e) & \text{if } e \text{ is not } e_i,\ 1 \leq i \leq n, \text{ and } H \text{ is defined for } e \\ \text{undefined} & \text{otherwise} \end{cases}$$

Lemma 6.5.2 tells us that $C_H \subseteq tc(C_{H'})$.

One can check from the above definition that $H'$ is defined on every equation on which $H$ is defined, on every equation that occurs in $S$, and no other. Similarly, one can check that every parameter in the range of $H$ is in the range of $H'$, every parameter that occurs in $S$ is in the range of $H'$, and no other parameters are in the range of $H'$.

In addition to $H'$ being defined on every equation that $H$ is defined, $H'$ is also defined on $e_1$, but $H$ is not. Hence, $H'$ is defined on more equations in $ic(E)$ than $H$.

Since $H'$ is defined on every equation on which $H$ is defined, and $H$ is augmentable, it follows that the only equations on which $H'$ may not be defined are of the form $int(\boldsymbol{q}, dq/dt)$. Lemma 6.5.3 tells us $int(\boldsymbol{q}, dq/dt)$ occurs in $S$ if and only if $dq/dt$ occurs in $S$. Hence, $H'$ is defined on $int(\boldsymbol{q}, dq/dt)$ if and only if $dq/dt$ is in the range of $H'$. Hence, if $H'$ is not defined on some equation, it follows that $H'$ is augmentable with respect to $E$. Otherwise, $H'$ is a complete causal mapping.

For example, augmenting the causal mapping $H$ shown in Figure 6.2a using the sequence shown in Equation 6.7, results in the causal mapping $H'$ shown in Figure 6.2b. One can easily verify that $H'$ is augmentable.

We can now recursively apply the above lemma to $H'$, until we are left with an onto causal mapping. For example, applying the above lemma to the causal mapping $H'$ of Figure 6.2b results in the onto causal mapping $H''$ shown in Figure 6.4.

The following lemma merely formalizes the above recursive procedure.

**Lemma 6.5.5.** *Let $E$ be a complete set of equations and let $H : ic(E) \rightarrow P(E)$ be an augmentable causal mapping with respect to $E$. Then there exists an onto causal mapping $H' : ic(E) \rightarrow P(E)$ such that $C_H \subseteq tc(C_{H'})$, i.e., the direct causal dependencies entailed by $H$ are a subset of the transitive closure of the direct causal dependencies entailed by $H'$.*

*Proof.* To prove this lemma, we construct a finite sequence of partial causal mappings $H_i : ic(E) \rightarrow P(E)$, $1 \leq i \leq k$, such that:

$$H_1 = H$$
$$H_k = H'$$
$$H_{i+1} \text{ is an augmentation of } H_i \quad 1 \leq i \leq k-1$$

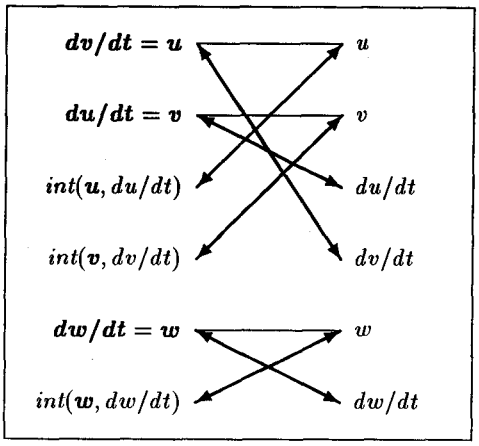



**Fig. 6.4.** The resulting onto causal mapping $H''$

This sequence is well defined because Lemma 6.5.4 tells us that for every augmentable causal mapping, there exists an augmentation which is either complete or augmentable. Furthermore, Lemma 6.5.4 also tells us that if $H_{i+1}$ is an augmentation of $H_i$, then $H_{i+1}$ is defined on more equations than $H_i$. Hence, $H_{i+1}$ is undefined on fewer equations than $H_i$. Hence, the above sequence must be finite, as the number of equations on which the augmentations are undefined decreases monotonically to 0, at which point the augmentation is complete.

Finally, Lemma 6.5.4 also tells us that if $H_{i+1}$ is an augmentation of $H_i$, then $C_{H_i} \subseteq tc(C_{H_{i+1}})$. Hence, by transitivity, it follows that $C_H = C_{H_1} \subseteq tc(C_{H_k}) = tc(C_{H'})$.

We are now in a position to prove a version of Theorem 5.3.1 that includes differential equations. Given complete models $M_1$ and $M_2$, such that $M_2 \leq M_1$, the idea is to use an onto causal mapping on $ic(E(M_2))$ to construct an augmentable causal mapping on $ic(E(M_1))$. The above lemma is then invoked to construct an onto causal mapping on $ic(E(M_1))$.

**Theorem 6.5.1.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem such that all the approximation relations are causal approximations (as in Definition 6.3), and the contradictory relation partitions the set $\mathcal{M}$ of model fragments into the set $\mathcal{A}$ of assumption classes. Let $M_1, M_2 \subseteq \mathcal{M}$ be complete models such that $M_1$ and $M_2$ contain model fragments from the same assumption classes, and $M_2 \leq M_1$. The causal relations entailed by the equations of $M_2$ are a subset of the causal relations entailed by the equations of $M_1$, i.e., $C(E(M_2)) \subseteq C(E(M_1))$.*

*Proof.* Let $H_2 : ic(E(M_2)) \rightarrow P(M_2)$ be an onto causal mapping. $H_2$ must exist because $M_2$ is complete. We now use $H_2$ to construct an onto causal mapping $H_1 : ic(E(M_1)) \rightarrow P(M_1)$, such that $C_{H_2} \subseteq tc(C_{H_1})$.

Let *same-ac*$(m, M)$ denote the model fragment $m' \in M$ such that $m$ and $m'$ are in the same assumption class. Since $M_1$ and $M_2$ have model fragments from the same assumption class, it follows that for any $m_1 \in M_1$ and $m_2 \in M_2$, the expressions *same-ac*$(m_1, M_2)$ and *same-ac*$(m_2, M_1)$ are well defined.

For any model fragment $m_1 \in M_1$, let *not-equil*$(m_1)$ denote the set of equations in $m_1$ that are not equilibrated in *same-ac*$(m_1, M_2)$, and let *not-equil*$(M_1)$ denote the union of the *not-equil* equations in the model fragments of $M_1$. Similarly, for any model fragment $m_2 \in M_2$, let *not-equil*$(m_2)$ denote the set of equations in $m_2$ that are not equilibrated versions of equations in *same-ac*$(m_2, M_1)$, and let *not-equil*$(M_2)$ denote the union of the *not-equil* equations in the model fragments of $M_2$.

Let $m_1 \in M_1$ be any model fragment, and let $m_2 \in M_2$ be such that $m_2 =$ *same-ac*$(m_1, M_2)$. If $m_1$ and $m_2$ are not identical, then it follows that $m_2$ is a causal approximation of $m_1$. Suppose that $m_1$ and $m_2$ are not identical, and hence $m_2$ is a causal approximation of $m_1$. From the discussion at the end of Sections 6.1 and 6.3, it follows that, if we let the equations of $m_1$ be *ic*(*not-equil*$(m_1)$), and the equations of $m_2$ be *ic*(*not-equil*$(m_2)$), then $m_2$ is a causal approximation of $m_1$ according to Definition 5.3. Hence, it follows that, if we let the equations of each model fragment $m \in M_1 \cup M_2$ be *ic*(*not-equil*$(m)$), then Lemma 5.3.2 applies, i.e., any causal orientation of the equations in *ic*(*not-equil*$(M_2)$) can be globally extended to a causal orientation of the equations in *ic*(*not-equil*$(M_1)$).

Using the above observation, we define a partial causal mapping $H$ : *ic*$(E(M_1)) \rightarrow P(M_1)$ as follows. Let $H$ restricted to the equations in *ic*(*not-equil*$(M_1)$) be the global extension of $H_2$ restricted to the equations in *ic*(*not-equil*$(M_2)$). Lemma 5.3.3 tells us that the direct causal dependencies entailed by $H$ restricted to the equations in *ic*(*not-equil*$(M_1)$) is a superset of the direct causal dependencies entailed by $H_2$ restricted to the equations in *ic*(*not-equil*$(M_2)$).

Now, consider any equation $e_2$ in *ic*$(E(M_2))$ but not in *ic*(*not-equil*$(M_2)$). One can see that, by definition, $e_2$ is an equilibrated version of some equation $e_1$ that is in $E(M_1)$ but not in *ic*(*not-equil*$(M_1)$). Hence, we can extend $H$ to $e_1$ by letting $H(e_1) = H_2(e_2)$. Since $e_2$ is an equilibrated version of $e_1$, this extension preserves the fact that the direct causal dependencies entailed by $H$ are a superset of the direct causal dependencies entailed by $H_2$. Since we have extended the causal orientation of each of the equations in *ic*$(E(M_2))$, it follows that $C_{H_2} \subseteq C_H$.

We will now construct the onto causal mapping $H_1$ from $H$. If $H$ is an onto causal mapping, then we make $H_1$ identical to $H$, and hence $C_{H_2} \subseteq C_{H_1}$, and hence $C_{H_2} \subseteq tc(C_{H_1})$.

On the other hand, suppose that $H$ is a partial causal mapping. We now show that $H$ is an augmentable causal mapping with respect to $E(M_1)$ (see Definition 6.4). First, note that the equations in *ic*$(E(M_1))$ can be partitioned

into three subsets: (a) the integration completion of the equations that are not equilibrated; (b) the equations that are equilibrated; and (c) the *int* equations corresponding to the equations that are equilibrated. It is easy to verify that $H$ is undefined only on the equations listed under (c), i.e., $H$ is undefined only on the *int* equations corresponding to the equations that are equilibrated.

Second, if $H$ is undefined for $int(\boldsymbol{q}, dq/dt)$, then $dq/dt$ cannot be in the range of $H$. This follows from the following facts. Let $e$ be the equation that can causally determine $dq/dt$ in $E(M_1)$. From (c) above, we know that $e$ has been equilibrated in $E(M_2)$. Let the equilibrated version of $e$ be $e'$, so that $H(e) = H_2(e')$. Since $e'$ is the equilibrated version of $e$, it follows that $H_2(e') \neq dq/dt$, and hence $H(e) \neq dq/dt$. Since $e$ is the only equation in $E(M_1)$ that can causally determine $dq/dt$, it follows that $dq/dt$ is not in the range of $H$. Hence, whenever $H$ is undefined for $int(\boldsymbol{q}, dq/dt)$, it follows that $dq/dt$ is not in the range of $H$. Finally, since $E(M_1)$ is complete and hence $|ic(E(M_1))| = |P(M_1)|$, it follows that every other parameter in $P(M_1)$ is in the range of $H$.

Hence, from the above observations, and from Definition 6.4, it follows that $H$ is augmentable with respect to $E(M_1)$. But Lemma 6.5.5 tells us that there exists an onto causal mapping $H_1 : ic(E(M_1)) \rightarrow P(M_1)$ such that $C_H \subseteq tc(C_{H_1})$. Since $C_{H_2} \subseteq C_H$, it follows that $C_{H_2} \subseteq tc(C_{H_1})$.

Hence, whether or not $H$ is onto, it follows that $C_{H_2} \subseteq tc(C_{H_1})$, and hence $C(E(M_2)) \subseteq C(E(M_1))$.

Hence, we have succeeded in generalizing Theorem 5.3.1 to models containing differential equations. This generalization required an updated definition of causal approximations. Hence, we require the following restriction on $\mathcal{I}$:

- All the approximation relations are causal approximations as in Definition 6.3.

## 6.6 Efficiently equilibrating differential equations

In the previous section, we introduced an updated definition of a causal approximation, and used this definition to generalize Theorem 5.3.1 to models that involve differential equations. Unfortunately, even with this updated definition, a coherent model $M$ can have an exponential number of immediate simplifications, all of which use model fragments from the same assumption classes as $M$. In Section 6.6.1, we illustrate this with an example, and show that the source of the problem is that we have placed no restrictions on the equilibrations of differential equations. We then address this problem by introducing *locally self-regulating* parameters, and show that when all the parameters are locally self-regulating, Lemma 5.3.1 can be generalized to

model fragments involving differential equations. This allows us to generalize Theorem 5.5.1, ensuring that model fragments of a coherent model can be approximated one at a time.

### 6.6.1 Equilibrating differential equations can be hard

Consider the two assumption classes $A_1$ and $A_2$, and the six model fragments $m_1$, $m_{11}$, $m_{12}$, $m_2$, $m_{21}$, and $m_{22}$, shown in Figure 6.5. Note that all the approximation relations are causal approximations.

$$A_1 = \{m_1, m_{11}, m_{12}\}$$
$$A_2 = \{m_2, m_{21}, m_{22}\}$$

| | | | | | |
|---|---|---|---|---|---|
| $m_1$ | = | $\{\boldsymbol{dy/dt = x}\}$ | $m_2$ | = | $\{\boldsymbol{dx/dt = y}\}$ |
| $m_{11}$ | = | $\{exogenous(\boldsymbol{x})\}$ | $m_{21}$ | = | $\{exogenous(\boldsymbol{y})\}$ |
| $m_{12}$ | = | $\{exogenous(\boldsymbol{x})\}$ | $m_{22}$ | = | $\{exogenous(\boldsymbol{y})\}$ |

$approximation(m_1, m_{11})$ $approximation(m_2, m_{21})$
$approximation(m_1, m_{12})$ $approximation(m_2, m_{22})$

**Fig. 6.5.** Assumption classes and model fragments

Consider the model $M = \{m_1, m_2\}$. Assuming that there are no propositional coherence constraints, it is easy to see that $M$ is complete, and hence coherent. Now consider the immediate simplifications of $M$. Replacing $m_1$ by either of its immediate approximations leads to an overconstrained model. For example, the model $M_1 = \{m_{11}, m_2\}$ is overconstrained because $ic(E(M_1))$ contains the equations $int(\boldsymbol{x}, dx/dt)$ from the integration completion of $m_2$, and $exogenous(\boldsymbol{x})$ from $m_{11}$. This is clearly overconstrained. Similarly, replacing $m_2$ by either of its immediate approximations leads to an overconstrained model.

The only way to get a coherent simplification is to replace *both* $m_1$ and $m_2$ by one of their immediate approximations. For example, $M_2 = \{m_{11}, m_{21}\}$ is a coherent model that is one of the immediate simplifications of $M$. Since both $m_1$ and $m_2$ have two immediate approximations, it follows that $simplifications(M)$ has four models. This example can be trivially generalized to have $n$ assumption classes, so that the number of models in the immediate simplifications of the most accurate model is $2^n$, i.e., if we allow arbitrary differential equations to be equilibrated, then coherent models can have an exponential number of immediate simplifications.

The fundamental problem underlying the above example is that equations cannot, in general, be equilibrated individually, i.e., to ensure that the resulting model is coherent, a set of differential equations may have to be equilibrated simultaneously. Hence, if each equation that has to be equilibrated has multiple equilibrations, it follows that there will be an exponential number of immediate simplifications.

In addition, it is not clear how we can efficiently identify a minimal set of differential equations, such that simultaneously equilibrating each equation in that set results in a coherent model.[1] Hence, even if each differential equation has just one equilibration, we would still be unable to generate the immediate simplifications of a coherent model efficiently.

### 6.6.2 Self-regulating parameter

We can circumvent the above problem if we restrict ourselves to only those differential equations that can be equilibrated if and only if they can be equilibrated individually. Hence, at most one differential equation needs to be equilibrated in any immediate simplification of a model, and the problem discussed above disappears.

Let us now understand how we can enforce this restriction. We start by defining a *self-regulating* parameter:

**Definition 6.5 (Self-regulating parameter).** *A parameter $p$ is said to be self-regulating with respect to a coherent model $M$ if and only if $dp/dt$ causally depends on $p$ in the causal ordering of the equations of $M$, i.e., $(p, dp/dt) \in C(E(M))$.*

Since $dp/dt$ determines $p$ by integration, it follows that a self-regulating parameter regulates its own variation over time. We now prove an easy consequence of the lemmas in Section 6.5. In particular, we show that a differential equation can be equilibrated only if the corresponding parameter is self-regulating.

**Lemma 6.6.1.** *Let $M$ be a coherent model, and let $e \in E(M)$ be a differential equation that can causally determine derivative $dp/dt$. Let $M' < M$ be a coherent model such that $e$ has been equilibrated in $M'$. Then $p$ is self-regulating with respect to $M$.*

*Proof.* We provide only a sketch of this proof. Recall from the proof of Theorem 6.5.1 that an onto causal mapping $F' : ic(E(M')) \rightarrow P(M')$ is extended to an onto causal mapping $F : ic(E(M)) \rightarrow P(M)$ using the following steps:[2]

1. First, $F$ is defined as the global extension of $F'$. One can check that, at this stage, $dp/dt$ is not in the range of $F$.
2. Then $F$ is successively augmented, using the alternating sequence of equations and parameters defined in Lemma 6.5.3. $dp/dt$ is introduced into the range of $F$ when $dp/dt$ occurs in such a sequence.

[1] We suspect that the problem of identifying such a minimal set of differential equations is intractable. However, we do not, as yet, have a proof of this hypothesis.

[2] This assumes that $M$ and $M'$ have model fragments from the same assumption classes. However, this proof can be generalized straightforwardly, using the proof of Theorem 5.4.1.

However, Lemma 6.5.3 tells us that $int(p, dp/dt)$ occurs in the sequence if and only if $dp/dt$ occurs in the sequence. But if $int(p, dp/dt)$ occurs in the sequence, it is easy to see that $p$ must occur in the sequence (using condition 2 of Lemma 6.5.1). Hence, both $p$ and $dp/dt$ occur in some sequence during the augmentation. Hence, using Lemma 6.5.1, we can infer that $p$ and $dp/dt$ are causally dependent upon each other. Hence, $p$ is self-regulating with respect to $M$.

Intuitively, the causal ordering from the equations of $M$ has a causal path from $p$ to $dp/dt$, and back to $p$ via integration, like the following:

$$p \rightarrow p_1 \rightarrow \cdots \rightarrow p_n \rightarrow dp/dt \xrightarrow{i} p$$

Equilibrating the equation that causally determines $dp/dt$ can be viewed as removing the last edge in this path, and making the rest of the edges point in the opposite direction:

$$p \leftarrow p_1 \leftarrow \cdots \leftarrow p_n$$

Of course, this is only possible if the path can be inverted in the simpler model. This is not always possible. For example, if one of the edges is an integration link:

$$p \rightarrow \cdots \rightarrow p_k \rightarrow dq/dt \xrightarrow{i} q \rightarrow \cdots \rightarrow p_n \rightarrow dp/dt \xrightarrow{i} p$$

then the integration link from $dq/dt$ to $q$ cannot be inverted. The only way to equilibrate the equation with this path is to also equilibrate the equation that causally determines $dq/dt$:

$$p \leftarrow \cdots \leftarrow p_k$$
$$q \leftarrow \cdots \leftarrow p_n$$

The above observations provide us with the condition necessary to ensure that differential equations can be individually equilibrated. In particular, if there is a causal path from $p$ to $dp/dt$ that can be inverted, and that contains no integration links, then the equation that causally determines $dp/dt$ can be individually equilibrated. Let us say that a parameter $p$ is *statically* self-regulating, with respect to a coherent model $M$, if there exists an invertible causal path from $p$ to $dp/dt$ that contains no integration links. Hence, if in every coherent model, every self-regulating parameter is also statically self-regulating, then differential equations can be individually equilibrated.

Unfortunately, we have no efficient method of ensuring that in every coherent model, every self-regulating parameter is also statically self-regulating. Instead, we restrict ourselves to *locally* self-regulating parameters. Informally, $p$ is locally self-regulating if all the parameters in the causal path from $p$ to $dp/dt$ are local to a model fragment. More precisely, we have the following definition:

**Definition 6.6 (Locally self-regulating parameter).** *Let $p$ be a parameter, and let $m$ be any model fragment that contains an equation $e$ that can causally determine $dp/dt$, and that has a causal approximation that equilibrates $e$. $p$ is said to be locally self-regulating if and only if every such model fragment $m$ has a subset $R \subseteq m$ of equations, called the self-regulating subset of $m$ with respect to $p$, such that*

1. $e \in R$;
2. $|ic(R)| = |P_c(ic(R))|$;
3. *if $p'$ is any other locally self-regulating parameter, and if $R'$ is the self-regulating subset of $m$ with respect to $p'$, then $R$ and $R'$ are disjoint, i.e., $R \cap R' = \emptyset$.*

Condition 2 assures us that if a coherent model contains $m$, then the preimage of every parameter in $P_c(ic(R))$, under any onto causal mapping, will be an equation in $ic(R)$. Hence, every parameter in $P_c(R)$ behaves as if it were local to $m$. In conjunction with condition 3, this also means that $P_c(ic(R))$ and $P_c(ic(R'))$ are disjoint.

For example, Figure 6.6 shows a model fragment that describes the velocity of a falling raindrop [Halliday and Resnick, 1978, page 95]. Figure 6.7 shows a causal approximation of this model fragment, which equilibrates the first equation, and hence describes the raindrop's terminal velocity.

$$\begin{aligned} m_r \boldsymbol{dv_r/dt} &= m_r g - \boldsymbol{d_r} \\ \boldsymbol{d_r} &= k\boldsymbol{v_r} \end{aligned}$$

$v_r$ : Velocity of the raindrop
$m_r$ : Mass of the raindrop
$g$ : Acceleration due to gravity
$d_r$ : Drag
$k$ : Coefficient of drag

**Fig. 6.6.** Model fragment describing the velocity of a falling raindrop

$$\begin{aligned} 0 &= m_r g - \boldsymbol{d_r} \\ \boldsymbol{d_r} &= k\boldsymbol{v_r} \end{aligned}$$

**Fig. 6.7.** Model fragment describing the raindrop's terminal velocity

It is easy to verify that $v_r$ is locally self-regulating with respect to this set of model fragments. In particular, we can use

$$R = \{m_r \boldsymbol{dv_r/dt} = m_r g - \boldsymbol{d_r}, \boldsymbol{d_r} = k\boldsymbol{v_r}\}$$

to verify the conditions in Definition 6.6.

Note that, in Definition 6.6, there is no guarantee that the parameter $p$ is self-regulating with respect to every model that contains $m$. However, we can show that if the equation $e$ can be successfully equilibrated, then $dp/dt$ causally depends on $p$ in any coherent model that includes $m$. In such cases,

the causal dependence of $dp/dt$ on $p$ is mediated by parameters in $P_c(ic(R))$. ($e$, $m$, and $R$ are as in Definition 6.6).

Instead of proving the above claim, we show that Lemma 5.3.1 can be generalized to model fragments involving differential equations. This will ensure that Theorem 5.5.1 will apply to models involving differential equations, so that the efficient model selection algorithm developed in Section 5.7 can be used. We start by proving some preliminary properties of the above definition.

**Lemma 6.6.2.** *Let $p$, $m$, $e$, and $R$ be as in Definition 6.6. Let $m'$ be a causal approximation of $m$, and let $G : ic(m') \rightarrow ic(m)$ be a correspondence mapping, and let $L$ be the local causal mapping with respect to $G$. Let $R' \subseteq m'$ be the pre-image of $R$ under $G$*

$$R' = \{e' : G(e') \in R\}$$

*Then we have the following:*

1. $|ic(R')| = |P_c(ic(R'))|$;
2. *parameters in $P_c(ic(R))$ are either in $P_c(ic(R'))$, or they are local to $m$; and*
3. *if $e_l$ is an equation in the domain of $L$, but $e_l \notin ic(R)$, then $L(e_l) \notin P_c(ic(R))$*

*Proof.* First, we show that $|ic(R')| = |P_c(ic(R'))|$. Let $|ic(R)| - |ic(R')| = k_e$ and let $k_{eq}$ be the number of equations in $R$ that are equilibrated in $R'$.[3] This means that $ic(R)$ contains $k_{eq}$ *int* equations that are not found in $ic(R')$. However, this also means that $P_c(ic(R))$ contains $k_{eq}$ derivatives that are not in $P_c(ic(R'))$ (since they have been equilibrated).

Let $k_l = k_e - k_{eq}$. Hence, $k_l$ is the number of equations in $R$ that have no corresponding equations in $R'$. Since $m'$ is a causal approximation of $m$, it follows that there exists a local causal mapping that maps these $k_l$ equations to $k_l$ parameters that are local to $m$ but not local to $m'$. Hence, these $k_l$ local parameters must be in $P_c(R)$, but not in $P_c(R')$. Hence, $|P_c(R)| - |P_c(R')| \geq k_l$.

Hence, from the above two paragraphs, we have the following:

$$\begin{aligned} |ic(R)| - |ic(R')| &= k_{eq} + k_l \\ |P_c(ic(R))| - |P_c(ic(R'))| &\geq k_{eq} + k_l \end{aligned}$$

Since $|ic(R)| = |P_c(ic(R))|$, it follows that

$$|ic(R')| \geq |P_c(ic(R'))|$$

But $|ic(R')|$ cannot be greater than $|P_c(ic(R'))|$, for otherwise $m'$ would be overconstrained. Hence, $|ic(R')| = |P_c(ic(R'))|$.

We now show that parameters in $P_c(ic(R))$ are either in $P_c(ic(R'))$, or they are local to $m$. From the above argument, we can see that the parameters in $P_c(ic(R))$ can be partitioned into three sets: (a) $k_e$ derivatives; (b) $k_l$

[3] In fact, condition 3 in Definition 6.6 allows us to show that $k_{eq} = 1$.

local parameters; and (c) parameters that are in $P_c(ic(R'))$. Since the derivatives are local to $m$, it follows that every parameter in $P_c(ic(R))$ is either in $P_c(ic(R'))$, or local to $m$.

Finally, we show that if $e_l$ is an equation in the domain of $L$, but $e_l \notin ic(R)$, then $L(e_l) \notin P_c(ic(R))$. As argued above, $P_c(ic(R))$ contains $k_l$ parameters that are local to $m$ but not local to $m'$, and that $ic(R)$ contains $k_l$ equations that are not in the range of $G$. Hence, it follows that $L$ must map these $k_l$ equations to the $k_l$ parameters. Since $L$ is 1-1, it follows that if $e_l \notin ic(R)$, $L(e_l) \notin P_c(ic(R))$.

The above properties can be used to guarantee the existence of local extensions, i.e., we prove an extended version of Lemma 5.3.1.

**Lemma 6.6.3.** *Let $\mathcal{I}$ be an instance of the* MINIMAL CAUSAL MODEL *problem in which all the approximation relations are causal approximations, and all the parameters are locally self-regulating. Let $m, m' \in \mathcal{M}$ be model fragments and let $m'$ be an approximation of $m$. Let $F' : ic(m') \rightarrow P(ic(m'))$ be a causal mapping. Then there exists a causal mapping $F : ic(m) \rightarrow P(ic(m))$ such that every parameter in the range of $F$ is either in the range of $F'$ or is local to $m$.*

*Proof.* Let $e_1, e_2, \ldots, e_n \in m$, for some $n \geq 0$, be the differential equations that can be equilibrated by some approximation of $m$ (if $n = 0$ then no equations can be equilibrated). Let equation $e_i$, $1 \leq i \leq n$, causally determine derivative $dp_i/dt$. Since all the parameters are locally self-regulating, let $R_i$, $1 \leq i \leq n$, be the self-regulating subset of $m$ with respect to $p_i$. We know that these self-regulating subsets are mutually disjoint, i.e., $R_i \cap R_j = \emptyset$, for $1 \leq i, j \leq n$ and $i \neq j$.

Partition the equations in $m$ into $(n+1)$ subsets $R_0, R_1, \ldots, R_n$, where $R_0$ contains all the equations in $m$ that are not in any of the other subsets.

Let $G : ic(m') \rightarrow ic(m)$ be a correspondence mapping, and let $L$ be the local causal mapping with respect to $G$. Using $G$ and the above partition of $ic(m)$, partition the set $m'$ into the $(n+1)$ subsets $R'_0, R'_1, \ldots, R'_n$ as follows:

$$R'_i = \{e : G(e) \in R_i\}, 0 \leq i \leq n$$

i.e., $R'_i$ contains the pre-image of $R_i$ under $G$.

Let $F'' : ic(m) \rightarrow P(ic(m))$ be any causal mapping. We now use $F'$ and $F''$ to construct the desired causal mapping $F$.

For each equation in $ic(R_i)$, $1 \leq i \leq n$, let $F$ be identical to $F''$. As shown in Lemma 6.6.2, every parameter in $P_c(ic(R_i))$ is either in $P_c(ic(R'_i))$, or is local to $m$. In addition, Lemma 6.6.2 also showed that $|ic(R'_i)| = |P_c(ic(R'_i))|$, so that every element in $P_c(ic(R'_i))$ is in the range of $F'$ restricted to $ic(R'_i)$. Hence, every element in the range of $F$, when restricted to $ic(R_i)$, is either in the range of $F'$ restricted to $ic(R'_i)$, or is local to $m$.

That only leaves equations in $ic(R_0)$. The equations in $R_0$ have no differential equations that can be equilibrated. Hence, if we restrict the equations

of $m$ to $ic(R_0)$ and the equations of $m'$ to $ic(R_0')$, it follows that $m'$ is a causal approximation of $m$ according to Definition 5.3. Hence, the results of the previous chapter apply to $ic(R_0)$ and $ic(R_0')$. In particular, Lemma 5.3.1 tells us that any causal mapping of the equations of $ic(R_0')$ can be locally extended to a causal mapping of the equations of $ic(R_0)$.

Define $F$ on $ic(R_0)$ to be the local extension of $F'$ on $ic(R_0')$. Hence, the parameters in the range of this extension are either in the range of $F'$ restricted to $ic(R_0')$, or are local to $m$. From Lemma 6.6.2, the local parameters used in the range of this extension are not in $P_c(ic(R_i'))$, $1 \leq i \leq n$. Hence, this extension of $F$ is well defined.

Hence, we have defined the causal mapping $F$ such that the parameters in the range of $F$ are either in the range of $F'$, or are local to $m$.

The above generalization of Lemma 5.3.1 allows us to generalize Theorem 5.5.1 to models with differential equations. Hence, the efficient model selection algorithms of Section 5.7 can be used when all the parameters are locally self-regulating. Hence, we have the following restriction on $\mathcal{I}$:

- All the parameters of $\mathcal{I}$ must be locally self-regulating as defined in Definition 6.6.

### 6.6.3 Discussion

Iwasaki defines a closely related notion of a self-regulating *equation* [Iwasaki, 1988]. In her definition, a differential equation that can causally determine $dp/dt$ is self-regulating if the equation can also causally determine $p$. It is easy to verify that this is just a special case of $p$ being locally self-regulating. In particular, the causal path from $p$ to $dp/dt$ is not mediated by any additional parameters, local or otherwise.[4]

Not all parameters that are encountered in modeling the physical world are locally self-regulating. For example, consider the case of two objects connected by a heat path. Figure 6.8 shows the three assumption classes that describe this situation. Figure 6.9a shows the causal ordering generated from the most accurate model of this situation. Note that both $\boldsymbol{dT_1/dt} = C_1\boldsymbol{f}$ and $\boldsymbol{dT_2/dt} = -C_2\boldsymbol{f}$ can be individually equilibrated. For example, Figure 6.9b shows the causal ordering resulting from replacing the first equation by $\boldsymbol{f} = 0$.

It is easy to verify from Figure 6.9a that, even though both $T_1$ and $T_2$ are self-regulating with respect to this model, neither of them is locally self-regulating ($f$ is not local to any assumption class). However, note that both $T_1$ and $T_2$ are statically self-regulating with respect to the most accurate model, so that individual equilibration is guaranteed. In practice, the differential equations that we have encountered can be individually equilibrated. Hence, even though our efficient model selection algorithm is based on the restriction

[4] Hence, one can call this *direct* self-regulation.

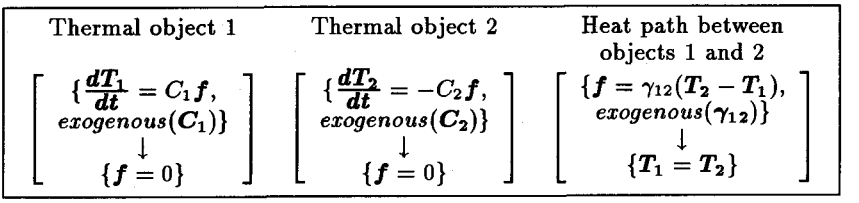



**Fig. 6.8.** Assumption classes, and their model fragments, describing the temperature of objects 1 and 2, and the heat path between them. The arrows denote the approximation relation between the model fragments.

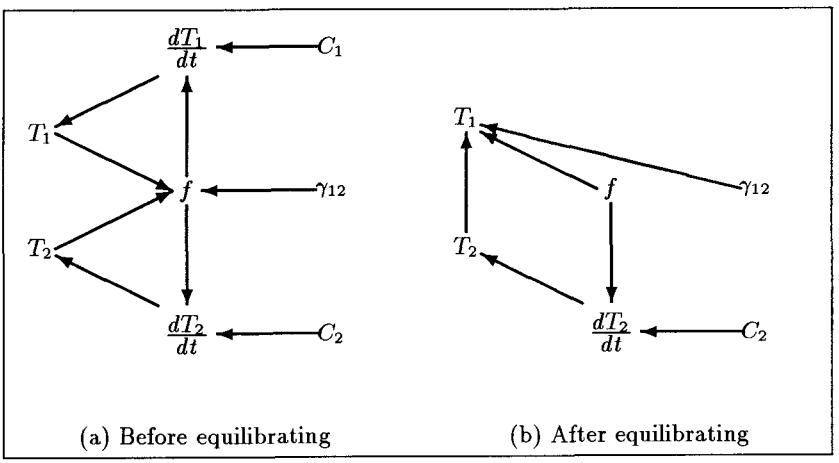



**Fig. 6.9.** Causal ordering before and after equilibration

that all the parameters are locally self-regulating, the program described in Chapter 8 does not place this restriction. Instead, it implicitly assumes that all the parameters are statically self-regulating, and hence assumes that they can be equilibrated if and only if they can be individually equilibrated.

## 6.7 Summary

In this chapter, we generalized the results of Chapter 5 to models with differential equations. We started by requiring that all model fragments be in canonical form. This means that all derivatives were required to be local to some model fragment. We then discussed two important methods of approximating differential equations: exogenizing and equilibrating. Exogenizing a differential equation is equivalent to assuming that the dynamic behavior is much slower than the time-scale of interest. Equilibrating a differential equation is equivalent to assuming that the dynamic behavior is much faster than the time-scale of interest. We introduced an updated definition of a causal ap-

proximation, and used this definition to generalize Theorem 5.3.1 to models with differential equations.

We then showed that, in the worst case, a coherent model with differential equations can have an exponential number of immediate simplifications. The root of this problem was traced to the lack of any restrictions on how differential equations could be equilibrated. We addressed this problem by introducing locally self-regulating parameters. We then showed that when all the parameters are locally self-regulating, a generalized version of Lemma 5.3.1 can be proved. This generalizes Theorem 5.5.1, ensuring that model fragments can be approximated if and only if they can be individually approximated. Together with the generalization of Theorem 5.3.1, this means that the efficient model selection techniques developed in Section 5.7 can be used for models with differential equations.

# 7. Order of magnitude reasoning

In Chapter 3 we said that the behavioral context of a device is its behavior at a particular time, i.e., the values, at that time, of the parameters used to model the device. Ideally, we would like the behavioral context to refer to the actual behavior of the device, e.g., the values of the parameters obtained by actual measurements on a physical prototype. However, since the actual behavior is usually unavailable, we content ourselves with computing the behavior from the equations of a device model.

Different techniques can be used to generate the behavior from the equations of a device model. At one extreme, purely numerical techniques can be used to solve a set of equations [Press *et al.*, 1989]. The advantage of such techniques is that predictions can be made with high precision. The primary disadvantage is that such techniques require exact numerical values for exogenous parameters. Exact numerical values are not always available, specially during conceptual design, making such methods largely inapplicable. At the other extreme, purely qualitative techniques can be used for behavior generation [Bobrow, 1984; Weld and de Kleer, 1990]. The advantage of such techniques is that they work with weak qualitative information, e.g., signs of parameters, and qualitative functional relationships. However, a primary disadvantage is that the predictions lack the precision of numerical techniques.

In this chapter we discuss a novel order of magnitude reasoning technique for generating the behavior from the equations of a device model. In this technique, the order of magnitude of a parameter is defined on a logarithmic scale, and a set of rules are used to propagate orders of magnitude through equations. A novel feature of the set of propagation rules is that they allow us to effectively handle non-linear simultaneous equations, using linear programming in conjunction with backtracking. This technique has been implemented in a program called NAPIER.[1]

The order of magnitude technique embodied in NAPIER is at the right level of detail. On the one hand, it does not require exact numerical values for exogenous parameters; a more qualitative order of magnitude is enough. On the other hand, unlike purely qualitative techniques, it provides valuable numerical information.

[1] John Napier (1550–1617), a Scottish nobleman, is credited with the first discovery of logarithms.

Section 7.1 presents a motivating example that has been used by others working on order of magnitude reasoning. Section 7.2 presents the basic order of magnitude reasoning technique, and Section 7.3 analyzes its complexity. Since we show that order of magnitude reasoning is, in general, intractable, Section 7.4 develops and empirically evaluates an approximate reasoning technique for order of magnitude reasoning. Finally, Section 7.5 estimates the error introduced by some of the order of magnitude rules introduced in Section 7.2, and Section 7.6 discusses related work.

## 7.1 Motivating example

Consider the following example, previously discussed in [Bennett, 1987; Raiman, 1991], from the domain of acid-base chemistry. An important task in this domain is to find the concentration of $H^+$ ions in a solution. The concentration of ions in solution depends on the dynamic equilibrium resulting from competing chemical reactions. Consider dissolving an acid, $AH$, in water. The two reversible reactions that occur, corresponding to the ionization of $AH$ and $H_2O$, are shown in Figure 7.1.

$$\begin{array}{lcl} AH & \rightleftharpoons & H^+ + A^- \\ H_2O & \rightleftharpoons & H^+ + OH^- \end{array}$$

**Fig. 7.1.** Ionization reactions that occur on dissolving $AH$ in water

The equilibrium concentrations of the three ions ($H^+, OH^-, A^-$) and the acid ($AH$) are determined by Equations 7.1–7.4. (Square brackets denote concentrations; $C_a$ is the initial concentration of the acid; $K_w$ is the ion product of water; and $K_a$ is the ionization constant of the acid.)

$$\text{Charge balance:} \quad [H^+] = [A^-] + [OH^-] \tag{7.1}$$

$$\text{Mass balance:} \quad C_a = [A^-] + [AH] \tag{7.2}$$

$$\text{Acid ionization equilibrium:} \quad K_a[AH] = [A^-][H^+] \tag{7.3}$$

$$\text{Water ionization equilibrium:} \quad K_w = [OH^-][H^+] \tag{7.4}$$

As has been pointed out in [Bennett, 1987; Raiman, 1991], solving this set of equations analytically for $[H^+]$ results in a cubic equation which is difficult to solve. In fact, in problems involving *polyprotic* acids, i.e., acids that can yield more than one $H^+$ ion, the closed form solution for $[H^+]$ can involve equations of degree five or higher, making the solution significantly harder.

An alternative to the above approach is to approximate the equations, and hence simplify them. For example, a chemist might guess that the acid is strong, so that $[A^-] \gg [OH^-]$ and $[A^-] \gg [AH]$. This justifies reducing the

first equation to $[H^+] = [A^-]$ and the second equation to $C_a = [A^-]$, leading to a straightforward solution.

The reasoning following the assumptions that $[A^-] \gg [OH^-]$ and $[A^-] \gg [AH]$ is very nicely formalized in [Raiman, 1991]. But how are these assumptions justified? In [Bennett, 1987], Bennett suggests that such assumptions are justified by *domain specific* inference rules. A much better, *domain-independent* method for justifying such assumptions is embodied in NAPIER. NAPIER can propagate the order of magnitude of exogenous parameters like $C_a$, $K_w$, and $K_a$ through a set of equations like Equations 7.1–7.4 to compute orders of magnitudes of the remaining parameters like $[H^+]$, $[OH^-]$, $[A^-]$, and $[AH]$. The computed orders of magnitude of $[A^-]$, $[OH^-]$, and $[AH]$ can be used to justify the above assumptions and simplify the equations.

We now describe the order of magnitude reasoning technique embodied in NAPIER.

## 7.2 Order of magnitude reasoning in NAPIER

Order of magnitude reasoning in NAPIER is a form of interval reasoning. The *order of magnitude* of a parameter $q$ (denoted $om(q)$) is defined as follows:

$$om(q) = \lfloor \log_b |q| \rfloor \tag{7.5}$$

where the base, $b$, of the logarithm is chosen to be the smallest number that can be considered to be "much larger" than 1. The choice of $b$ is clearly domain and task dependent. Here we assume that $b = 10$. Note that the order of magnitude of a parameter is an integer, with $om(q) = n$ being equivalent to:

$$b^n \leq |q| < b^{n+1}$$

Similarly, $n_1 \leq om(q) \leq n_2$ is equivalent to

$$b^{n_1} \leq |q| < b^{n_2+1}$$

Note also that the order of magnitude of a parameter is independent of its sign, and hence $om(q) = om(-q)$. In what follows, we assume that the signs of all parameters have been determined, to the extent possible, prior to any reasoning about orders of magnitude using standard constraint satisfaction techniques.[2]

[2] This assumption is unnecessarily strong. For example, if $a$ and $b$ are positive, constraint satisfaction alone is unable to deduce the sign of $a - b$. However, if $om(a) > om(b)$, then $a - b$ can be deduced to be positive.

### 7.2.1 Inference rules in NAPIER

Given the orders of magnitude of $q_1$ and $q_2$, NAPIER computes bounds on the orders of magnitude of arithmetic expressions involving $q_1$ and $q_2$, using the rules shown in Figure 7.2. The rules for $(q_1 + q_2)$ and $(q_1 - q_2)$ assume that $q_1$ and $q_2$ have the same sign, so that the magnitudes of $q_1$ and $q_2$ are actually being added or subtracted, respectively. The rule for $(q_1 \pm q_2)$ is applicable to a sum or difference of $q_1$ and $q_2$ when the sign at least one of $q_1$ and $q_2$ is unknown.

| | | | |
|---|---|---|---|
| 1. | | $om(q_1) + om(q_2) \leq om(q_1 * q_2) \leq om(q_1) + om(q_2) + 1$ | |
| 2. | | $om(q_1) - om(q_2) - 1 \leq om(q_1/q_2) \leq om(q_1) - om(q_2)$ | |
| | a) | $om(q_1) \leq om(q_1 + q_2) \leq om(q_1) + 1$ | if $om(q_1) = om(q_2)$ |
| 3. | b) | $om(q_1 + q_2) = om(q_1)$ | if $om(q_1) > om(q_2)$ |
| | c) | $om(q_1 + q_2) = om(q_2)$ | if $om(q_1) < om(q_2)$ |
| | a) | $om(q_1 - q_2) \leq om(q_1)$ | if $om(q_1) = om(q_2)$ |
| 4. | b) | $om(q_1 - q_2) = om(q_1)$ | if $om(q_1) > om(q_2)$ |
| | c) | $om(q_1 - q_2) = om(q_2)$ | if $om(q_1) < om(q_2)$ |
| | a) | $om(q_1 \pm q_2) \leq om(q_1) + 1$ | if $om(q_1) = om(q_2)$ |
| 5. | b) | $om(q_1 \pm q_2) = om(q_1)$ | if $om(q_1) > om(q_2)$ |
| | c) | $om(q_1 \pm q_2) = om(q_2)$ | if $om(q_1) < om(q_2)$ |

**Fig. 7.2.** Rules for order of magnitude reasoning. In rules 3 and 4, $q_1$ and $q_2$ are assumed to have the same sign. Rule 5 assumes that the sign of at least one of $q_1$ or $q_2$ is unknown.

The rules for $(q_1 * q_2)$ and $(q_1/q_2)$ (rules 1 and 2) follow directly from Equation 7.5 and the rules of interval arithmetic [Moore, 1979]. For example, if $om(q_1) = n_1$ and $om(q_2) = n_2$, it follows that $b^{n_1} \leq |q_1| < b^{n_1+1}$ and $b^{n_2} \leq |q_2| < b^{n_2+1}$. Using interval arithmetic, we get $b^{n_1+n_2} \leq |q_1 * q_2| < b^{n_1+n_2+2}$, and hence $n_1 + n_2 \leq om(q_1 * q_2) \leq n_1 + n_2 + 1$.

Like rules 1 and 2, rules 3a and 4a are also based on Equation 7.5 and interval arithmetic. Note, however, that these rules predict larger intervals for $(q_1 + q_2)$ and $(q_1 - q_2)$ than interval arithmetic predicts under the same restrictions on $q_1$ and $q_2$. For example, if $om(q_1) = om(q_2) = n$, then interval arithmetic predicts that $(q_1 + q_2)$ is bounded by $2b^n$ and $2b^{n+1}$, while NAPIER predicts the bounds $b^n$ and $b^{n+2}$. This is a consequence of NAPIER being able to represent only intervals whose end points are integer powers of the chosen base. Further note that rules 3a and 4a are correct only if the base is greater than 2. This is reasonable given our heuristic for selecting the base (viz., 2 is unlikely to be considered to be "much larger" than 1).

Unlike the rules discussed thus far, rules 3b, 3c, 4b, and 4c are not guaranteed to be correct, but are heuristic rules. They are all based on the intuition that adding or subtracting a "small" parameter from a "large" parameter does not significantly affect the larger parameter. Since the base in Equation 7.5 is chosen as the smallest number that can be considered to be "much

larger" than 1, the above intuition justifies these rules; the order of magnitude of a parameter is not affected by adding or subtracting parameters of a smaller order of magnitude. The inclusion of these heuristic order of magnitude rules differentiates NAPIER from standard interval reasoners. In section 7.5, we estimate the error introduced by the use of these heuristic rules.

Finally, rule set 5 merely encompasses both rule sets 3 and 4. It is used to infer the order of magnitude of a sum or difference of two parameters when the signs of at least one of the two parameters is not known. To determine the order of magnitude of a sum or difference of two parameters, NAPIER selects the appropriate rule set from rule sets 3, 4, and 5, depending on the operation (sum or difference) and the signs of the two parameters. For example, consider the equation $q_3 = q_1 + q_2$. If $q_1$ and $q_2$ have the same sign, then rule set 3 is used to infer $om(q_3)$; if $q_1$ and $q_2$ have opposite signs, then rule set 4 is used to infer $om(q_3)$, since then the magnitude of $q_3$ is really the difference of the magnitudes of $q_1$ and $q_2$; and if the signs of at least one of $q_1$ and $q_2$ is unknown, then rule set 5 is used to infer $om(q_3)$.

### 7.2.2 Set of simultaneous equations

Until now, we have focussed exclusively on how NAPIER uses a single equation to propagate orders of magnitudes, i.e., how $om(q_1\ op\ q_2)$ is computed from $om(q_1)$ and $om(q_2)$. However, the rules in Figure 7.2 can also be used to compute orders of magnitudes of parameters related by a set of (possibly non-linear) simultaneous equations. NAPIER uses these rules to convert a set of simultaneous equations into a set of constraints, where each constraint is a disjunction of a set of linear inequalities. Each equation in the set of simultaneous equations contributes a constraint as follows:

1. Product and quotient terms contribute a single set of linear inequalities according to rules 1 and 2, respectively. For example, $q_3 = q_1 * q_2$ contributes the following set:

$$\{om(q_1) + om(q_2) \leq om(q_3),\\ om(q_3) \leq om(q_1) + om(q_2) + 1\}$$

2. Sum and difference terms contribute a disjunction of three sets of linear inequalities, using rule sets 3, 4, or 5, as applicable. Each disjunct corresponds to one of the rules (a, b, or c) in the applicable rule set. For example, assuming that $q_1$ and $q_2$ have the same sign, the equation $q_3 = q_1 - q_2$ contributes the following disjunction, corresponding to rules 4a, 4b, and 4c, respectively:[3]

[3] Since the order of magnitudes are integral, $om(q_1) > om(q_2)$ is equivalent to $om(q_1) \geq om(q_2) + 1$.

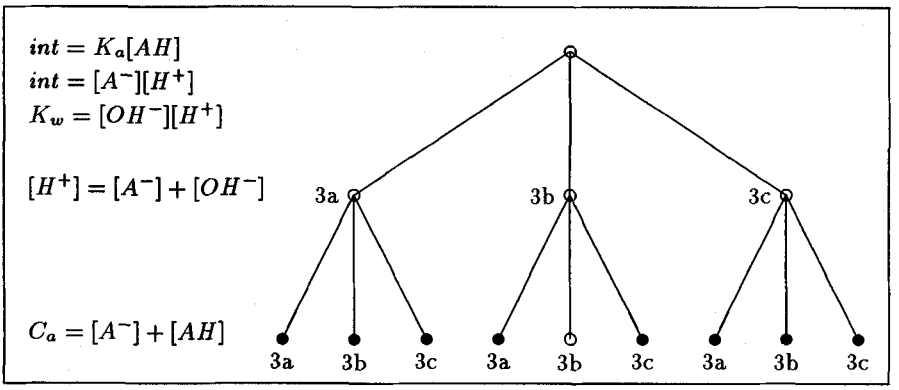



**Fig. 7.3.** A backtrack tree.

$$\{om(q_3) \leq om(q_1), om(q_1) = om(q_2)\}$$
$$\vee$$
$$\{om(q_3) = om(q_1), om(q_1) \geq om(q_2) + 1\}$$
$$\vee$$
$$\{om(q_3) = om(q_2), om(q_1) \leq om(q_2) - 1\}$$

NAPIER uses this set of constraints to compute bounds on the orders of magnitudes of the parameters. Since all the inequalities in the constraints are linear inequalities, NAPIER uses linear programming [Hillier and Lieberman, 1980], in conjunction with backtracking, to compute order of magnitude bounds. Backtracking is necessary to handle the disjunctions. We describe this algorithm next.

### 7.2.3 Backtracking algorithm

Let $E$ denote the set of simultaneous equations being processed. NAPIER's backtracking procedure is best visualized as a depth-first traversal of a backtrack tree. Each level in the tree (except the root level) corresponds to one of the sum or difference terms in $E$. The root level corresponds to all the product and quotient terms in $E$. Each internal node has three children, corresponding to the three disjuncts in the constraint contributed by the sum or difference term at the level of the node's children. Each node in the tree has an associated set of linear inequalities defined as follows:

1. The set of inequalities at the root node consists of the union of the sets of inequalities contributed by each product and quotient term in $E$.
2. The set of inequalities at each non-root node consists of the union of (a) the inequalities at the node's parent; and (b) the inequalities in the disjunct associated with that node.

Starting at the root node, NAPIER traverses the backtrack tree in a depth-first manner. At each node it checks the consistency of the inequalities at that

node. If the set is inconsistent, it immediately backtracks to the node's parent. If the set is consistent and it is a non-leaf node, it continues its depth-first traversal. If the set is consistent and it is a leaf node, it uses the inequalities to find the maximum and minimum values of the order of magnitude of each parameter. The bounds computed at each of the consistent leaf nodes are combined so that the lower bound of each parameter is the least lower bound and the upper bound is the greatest upper bound.

Since the inequalities at each node are linear, NAPIER uses the Simplex linear programming algorithm [Hillier and Lieberman, 1980; Press *et al.*, 1989] to check their consistency, and to compute the order of magnitude bounds at leaf nodes. However, from Equation 7.5 it follows that the order of magnitude of a parameter is integral. Hence, instead of using linear programming, NAPIER should use integer programming [Hillier and Lieberman, 1980]. Unfortunately, integer programming is known to be intractable [Karp, 1972], which leads to severe restrictions on the number of equations and the size of the backtrack tree that can be handled. Hence, to avoid such restrictions, NAPIER uses linear programming.

It is important to note that, while bounds computed by linear programming are not guaranteed to be tight,[4] they are guaranteed to be correct: upper bounds will be greater than or equal to integer programming upper bounds, and lower bounds will be less than or equal integer programming lower bounds. In addition, we have found that, in practice, linear programming bounds are usually integral, in which case there is no loss of solution quality.

### 7.2.4 Example

We now illustrate the above procedure using Equations 7.1–7.4. Let us assume that the exogenous orders of magnitude are as follows: $om(K_w) = -14$, $om(K_a) = -2$, $om(C_a) = -5$. This corresponds to a moderately strong solution of a strong acid. The backtrack tree resulting from these equations is shown in Figure 7.3. The equations associated with each level are shown on the left of the tree. Note that Equation 7.3 had to be split into two product terms, with the introduction of an intermediate variable *int*. The rules associated with each non-root node are displayed near each node. Nodes that are filled in are the inconsistent nodes. For example, the left most leaf node can be seen to be inconsistent using the following line of reasoning. Applying rule 3a to Equations 7.1 and 7.2, we get

$$om([OH^-]) = om([A^-]) = om([AH])$$
$$om([A^-]) \leq om(C_a) \leq om([A^-]) + 1$$
$$om([A^-]) \leq om([H^+]) \leq om([A^-]) + 1$$

[4] A bound $b_1$, interpreted as an interval, is said to be *tight* with respect to a bound $b_2$ if $b_1$ and $b_2$ are identical. $b_1$ is *looser* than $b_2$ if $b_1$ contains $b_2$.

Since $om(C_a) = -5$, it follows that the least value of $om([A^-])$ is $-6$. Hence the least values of $om([OH^-])$ and $om([H^+])$ are also $-6$, and hence the least value of $om([OH^-][H^+])$ is $-12$. But rule 1 applied to Equation 7.4 requires that:

$$om([OH^-][H^+]) = om(K_w) = -14$$

which leads to a contradiction.

Of course, NAPIER doesn't need the above line of reasoning to infer inconsistencies; it reaches the same conclusion using linear programming.

The only consistent set of inequalities at the leaf nodes is the middle most leaf node, corresponding to assuming that $om([A^-]) > om([OH^-])$ and $om([A^-]) > om([AH])$. The parameter bounds calculated at this node are as follows:[5]

$$om([H^+]) = -5 \qquad om([OH^-]) = (-10, -9)$$
$$om([AH]) = (-9, -7) \qquad om([A^-]) = -5$$

Since $[A^-]$ is at least two orders of magnitude greater than $[AH]$, and at least four orders of magnitude greater than $[OH^-]$, a chemist is justified in making the assumptions that $[A^-] \gg [OH^-]$ and $[A^-] \gg [AH]$. These assumptions can then be used to simplify the equations, as discussed earlier.

A slight variation of the above example illustrates the importance of having such justifications. Suppose that, instead of having $om(C_a) = -5$, we had $om(C_a) = -8$. This corresponds to a weak solution of the same strong acid. Using this new value for $om(C_a)$, NAPIER predicts the following bounds on the orders of magnitude:

$$om([H^+]) = -7 \qquad om([OH^-]) = (-8, -7)$$
$$om([AH]) = (-14, -12) \qquad om([A^-]) = -8$$

These values justify the assumption that $[A^-] \gg [AH]$, but the other assumption, $[A^-] \gg [OH^-]$, is seen to be completely unjustified. This means that only Equation 7.2 can be simplified. Hence, NAPIER is a useful tool in justifying the order of magnitude assumptions that scientists and engineers make in simplifying equations.

In addition to its role in justifying order of magnitude assumptions, NAPIER's predictions can also be used directly. For example, if all the chemist is interested in is the approximate pH of the solution,[6] then NAPIER's predictions can be used directly: in the first case, the pH is between 5 and 4; in the second case, the pH is between 7 and 6. Note that NAPIER was able to make these predictions using approximate values of $C_a$, $K_w$, and $K_a$. This feature makes it particularly useful during conceptual design.

---

[5] $om(q) = (l, u)$ represents the fact that $l \leq om(q) \leq u$

[6] The pH of a solution is defined to be $-\log_{10}[H^+]$.

## 7.3 Order of magnitude reasoning is intractable

The backtracking algorithm described in the previous section, generates a tree whose worst case size is exponential in the number of sum and difference expressions. In this section we show that order of magnitude reasoning using the rules in Figure 7.2 is intractable. Clearly, one source of intractability is that the order of magnitude of a parameter is integral, so that consistency checks and bounds computations require integer programming. However, we now show that order of magnitude reasoning remains intractable even if orders of magnitude are not required to be integral. An immediate consequence of this result is that NAPIER can do little better than generate a backtrack tree whose worst case size is exponential.

We start by defining the decision problem corresponding to finding the maximum order of magnitude of a parameter:

**Definition 7.1.** *(*ORDER OF MAGNITUDE REASONING*) Let $E$ be a set of equations, and let $V$ be the set of parameters used in $E$. Let $X \subseteq V$ be the set of exogenous parameters, with known orders of magnitude. Let $q \in V$ be a parameter and let $B$ be an integer. Let $s : V \rightarrow \{+, -, unknown\}$ be a function that assigns signs to the parameters in $V$. (Parameters with unknown signs are assigned "unknown.") Assuming that the order of magnitude of a parameter is not required to be integral, is the maximum value of $om(q)$, derived using the rules in Figure 7.2 on the set $E$, greater than or equal to $B$?*

We now show that the above problem is NP-complete. The proof of this theorem is based on a reduction from an arbitrary instance of 3SAT. Briefly, the reduction introduces a parameter for each literal in the instance of 3SAT. Equations are added to ensure that parameters corresponding to complementary literals have the property that the order of magnitude of one of them must be 0 and the order of magnitude of the other one must be 1. The mapping between truth assignments and orders of magnitudes is straightforward: a literal is true if and only if the corresponding parameter's order of magnitude is 1. Additional equations involving the above parameters and the special parameter $q$ are then introduced, and the bound $B$ is defined to ensure that the maximum value of $om(q)$ is greater than or equal to $B$ if and only if all the clauses are satisfied. The details of this proof are as follows:

**Theorem 7.3.1.** *The* ORDER OF MAGNITUDE REASONING *problem is NP-complete.*

*Proof.* It is easy to see that the ORDER OF MAGNITUDE REASONING problem is in NP since a non-deterministic algorithm can proceed by (a) for each sum and difference term in $E$, guessing a rule (a, b, or c) from rule sets 3, 4, or 5, as applicable; and (b) use linear programming on the resulting set

of inequalities to see if the maximum value of $om(q)$ exceeds $B$. Since linear programming is known to be in P [Khachian, 1979], it follows that the ORDER OF MAGNITUDE REASONING problem is in NP.

To show that the ORDER OF MAGNITUDE REASONING problem is NP-hard, we reduce an arbitrary instance of 3SAT to an instance of the ORDER OF MAGNITUDE REASONING problem. Let $\mathcal{I}_1$ be an arbitrary instance of 3SAT consisting of a set $U = \{u_1, \ldots, u_n\}$ of boolean variables, and a set $C = \{c_1, \ldots, c_m\}$ of three literal clauses. We now reduce $\mathcal{I}_1$ to an instance, $\mathcal{I}_2$, of the ORDER OF MAGNITUDE REASONING problem.

For each boolean variable $u_i \in U$, $1 \leq i \leq n$, add the following 6 equations to $E$, and the corresponding parameters to $V$:

$$v_i = x_{i1} * x_{i2} \tag{7.6}$$

$$\bar{v_i} = \bar{x_{i1}} * \bar{x_{i2}} \tag{7.7}$$

$$y_{i1} = v_i * \bar{v_i} \tag{7.8}$$

$$y_i = y_{i1} + y_{i2} \tag{7.9}$$

$$z_i = v_i - \bar{v_i} \tag{7.10}$$

$$z_i = (z_{i1} + z_{i2}) * z_{i3} \tag{7.11}$$

Add $x_{i1}, x_{i2}, \bar{x_{i1}}, \bar{x_{i2}}, y_i, z_{i1}$, and $z_{i3}$ to the set $X$ of exogenous parameters. Define the orders of magnitudes of these parameters as follows:

$$om(x_{i1}) = om(x_{i2}) = om(\bar{x_{i1}}) = om(\bar{x_{i2}}) = om(z_{i3}) = 0 \tag{7.12}$$

$$om(y_i) = om(z_{i1}) = 1 \tag{7.13}$$

For each clause $c_j \in C$, $1 \leq j \leq m$, with literals $l_{j1}$, $l_{j2}$, and $l_{j3}$, add the equation

$$(((g_j + f_{j1}) + f_{j2}) + f_{j3}) = h_j \tag{7.14}$$

where the parameter $f_{jk}$ is $v_i$ if $l_{jk}$ is $u_i$, and $\bar{v_i}$ if $l_{jk}$ is $\bar{u_i}$, for some $1 \leq i \leq n$. Add $g_j$ and $h_j$ to $V$. Add $g_j$ to $X$, and define its order of magnitude as follows:

$$om(g_j) = 1 \tag{7.15}$$

Add the following equation to $E$:

$$h_1 * h_2 * \ldots * h_m = q \tag{7.16}$$

Let $s$ be such that all the parameters in $V$, except $z_i$ and $z_{i3}$ ($1 \leq i \leq n$), are positive, and let the signs of $z_i$ and $z_{i3}$ be unknown. Let $B$ be $3m - 1$.

That completes the reduction. Clearly, it can be done in polynomial time. We now show that any assignment of orders of magnitudes to the parameters of $\mathcal{I}_2$ that satisfies Equations 7.6–7.13, according to the rules in Figure 7.2, assigns the order of magnitude 1 to exactly one of $v_i$ and $\bar{v_i}$, for each $1 \leq i \leq n$, and 0 to the other.

From rule 1 applied to Equation 7.6 we have:

$$om(x_{i1}) + om(x_{i2}) \leq om(v_i) \leq om(x_{i1}) + om(x_{i2}) + 1 \tag{7.17}$$

Substituting the orders of magnitudes of $x_{i1}$ and $x_{i2}$ (Equation 7.12) into the above equation, we have

$$0 \leq om(v_i) \leq 1 \tag{7.18}$$

In a similar way, rule 1 applied to Equation 7.7 implies that:

$$0 \leq om(\bar{v_i}) \leq 1 \tag{7.19}$$

Hence, $om(v_i)$ and $om(\bar{v_i})$ are either 0 or 1. Applying rule 1 to Equation 7.8 leads to:

$$om(v_i) + om(\bar{v_i}) \leq om(y_{i1}) \leq om(v_i) + om(\bar{v_i}) + 1 \tag{7.20}$$

Equation 7.9 implies that $om(y_i) \geq om(y_{i1})$. This fact follows from the observation that in rule 3:

$$om(q_1 + q_2) \geq q_1 \tag{7.21}$$

$$om(q_1 + q_2) \geq q_2 \tag{7.22}$$

under all three conditions. Since $om(y_i) = 1$ (Equation 7.13), it follows that

$$om(y_{i1}) \leq 1 \tag{7.23}$$

Hence, from Equations 7.20 and 7.23 it follows that:

$$om(v_i) + om(\bar{v_i}) \leq 1 \tag{7.24}$$

Hence, $om(v_i)$ and $om(\bar{v_i})$ cannot both be 1.

Now, since $om(z_{i1}) = 1$ (Equation 7.13), it follows from Equation 7.21 that

$$om(z_{i1} + z_{i2}) \geq 1 \tag{7.25}$$

Rule 1 applied to Equation 7.11 leads to:

$$om(z_{i1} + z_{i2}) + om(z_{i3}) \leq om(z_i) \tag{7.26}$$

Hence, from Equations 7.25 and 7.12, it follows that:

$$om(z_i) \geq 1 \tag{7.27}$$

Now, rule 4 implies that

$$om(q_1 - q_2) \leq maximum\{om(q_1), om(q_2)\} \tag{7.28}$$

Hence, from Equations 7.10, 7.18, 7.19, and 7.28, it follows that at least one of $om(v_i)$ and $om(\bar{v_i})$ must be 1. But earlier we had inferred that at most one of $om(v_i)$ and $om(\bar{v_i})$ can be 1. Hence, exactly one of $om(v_i)$ and $om(\bar{v_i})$ can be 1, with the other being 0.

We now show that $\mathcal{I}_1$ has a satisfying truth assignment if and only if $\mathcal{I}_2$ is such that the maximum value of $om(q)$ is greater than or equal to $B$.

($\Rightarrow$) Let $\mathcal{I}_1$ have a satisfying truth assignment. We now assign order of magnitudes to the parameters of $\mathcal{I}_2$ such that Equations 7.6–7.16 are satisfied according to the rules of Figure 7.2. For $1 \leq i \leq n$, if $u_i$ is true, then let $om(v_i)$

be 1 and $om(\bar{v_i})$ be 0; otherwise let $om(v_i)$ be 0 and $om(\bar{v_i})$ be 1. Since exactly one of $om(v_i)$ and $om(\bar{v_i})$ is 1 and the other is 0, this assignment of orders of magnitude will satisfy Equations 7.6–7.13.

Since the truth assignment satisfies every clause $c_j = \{l_{j1}, l_{j2}, l_{j3}\}, 1 \leq j \leq m$, it follows that at least one of $l_{j1}, l_{j2}$, or $l_{j3}$ is true. Hence, at least one of $f_{j1}, f_{j2}$, or $f_{j3}$ has an order of magnitude of 1. Since $om(g_j) = 1$ (Equation 7.15), it follows from Equation 7.14 and rule 3 that the maximum value of $om(h_j)$ is 2. Hence, from Equation 7.16 and rule 1, it follows that the maximum value of $q$ is $3m - 1$ ($2m$ from each of the $om(h_j)$, and $m - 1$ from the product of $m$ factors). Hence, the maximum value of $om(q)$ is greater than or equal to $B$.

($\Leftarrow$) Let us now assume that the maximum value of $om(q)$ is greater than or equal to $B$. Consider the assignment of order of magnitudes to parameters that supports $om(q)$ taking on its maximum value. For each variable $u_i$, $1 \leq i \leq n$, let $u_i$ be true if and only the order of magnitude of parameter $v_i$ is 1 in the above assignment. This gives us a well defined truth assignment since we have already shown that exactly one of $om(v_i)$ and $om(\bar{v_i})$ is 1. To show that this truth assignment satisfies very clause, we proceed as follows.

Since each $f_{jk} (1 \leq j \leq m, 1 \leq k \leq 3)$ is either $v_i$ or $\bar{v_i}$, for some $i$, Equations 7.18 and 7.19 tell us that the maximum value of $om(f_{jk})$ is 1. Hence, using Equation 7.14 and rule 3, the maximum value of $om(h_j)$ can be 2. However, for the maximum value of $om(q)$ to be greater than or equal to $B(= 3m - 1)$, it follows that $om(h_j)$ must be 2. Hence, at least one of the $f_{jk}$ must have an order of magnitude of 1. Hence, at least one of the $l_{jk}$ will be true, and hence the truth assignment constructed above will satisfy each clause.

Hence, we have shown that $\mathcal{I}_1$ has a satisfying truth assignment if and only if $\mathcal{I}_2$ is such that the maximum value of $om(q)$ is greater than or equal to $B$. Hence, the ORDER OF MAGNITUDE REASONING problem is NP-hard.

The intractability of the ORDER OF MAGNITUDE REASONING problem tells us that, in the worst case, NAPIER will have to generate a backtrack tree whose size is exponential in the number of sum and difference terms. Unfortunately, the exponential blow up does occur in practice. Table 7.1 summarizes NAPIER's performance on models of ten different devices (these devices are described in Appendix B).

The second column in this table shows the total number of equations in each example, while the third column shows the the total number of sum and difference terms. The fourth column shows the time it took NAPIER to run its backtracking algorithm on the complete set of equations. (The fifth column will be discussed in the next section.) NAPIER was given a maximum of one hour to solve each example; a "–" entry in column four denotes that NAPIER could not solve the example in an hour. As is clear from the table, only the two smallest examples could be solved in under an hour, each taking over 40 minutes. Hence, NAPIER appears to be quite impractical, except for

**Table 7.1.** NAPIER's run times on an Explorer II, with and without causal ordering.

| Device | Total # of eqns. | # of +/− terms | Time (sec) | |
|---|---|---|---|---|
| | | | All eqns. | With causal ordering |
| Bimetallic strip temperature gauge | 28 | 11 | 2733 | 2.0 |
| Bimetallic strip thermostat | 31 | 11 | 2435 | 1.0 |
| Flexible link temperature gauge | 45 | 14 | – | 2.9 |
| Electromagnetic relay thermostat | 60 | 24 | – | 2.7 |
| Galvanometer temperature gauge | 80 | 25 | – | 37.2 |
| Electric bell | 110 | 32 | – | 35.9 |
| Magnetic sizing device | 111 | 32 | – | 94.6 |
| Carbon pile regulator | 119 | 35 | – | 20.4 |
| Tachometer | 145 | 43 | – | 45.2 |
| Car distributor system | 163 | 50 | – | 21.0 |

the smallest examples. To make it practical, we now develop an approximate reasoning scheme for NAPIER that trades off accuracy for speed.

## 7.4 Approximation algorithms in NAPIER

The backtrack tree developed by NAPIER is, in the worst case, exponential in the number of sum and difference terms in the set of equations under consideration. Hence, to make NAPIER practically useful, it is important to decrease the number of sum and difference terms that are handled at any one time. We now discuss a method for doing this, based on a *dependency ordering* of the equations.

### 7.4.1 Ordering the equations

The dependency ordering of equations that we consider is the *causal ordering*, described in Chapter 3. The causal ordering specifies the order in which equations are to be solved, and identifies minimal sets of equations that *must* be solved simultaneously. The causal ordering can be viewed as a directed acyclic graph. Each node in the graph consists of a minimal set of equations that must be solved simultaneously. There is an edge from node $n_1$ to node $n_2$ if the equations at $n_2$ use a parameter whose value is determined by the equations at $n_1$.

NAPIER processes the equation sets in the order specified by the causal ordering: equation sets earlier in the ordering are processed first. NAPIER bounds the orders of magnitudes of the parameters used in an equation set, and uses these bounds as exogenous bounds for equation sets later in the ordering.

The use of the above dependency ordering has a significant computational advantage. A large set of equations, with many sum and difference terms, can

often be broken down into many small sets of equations, with each equation set having very few sum and difference terms. Hence, NAPIER can process each equation set in the dependency ordering very fast. Column five in Table 7.1 shows the time it took NAPIER to solve the ten examples using causal ordering. It takes NAPIER from a few seconds to under two minutes to solve each of these examples, showing that causal ordering has made NAPIER practical for large sets of equations.

### 7.4.2 Loss of accuracy

The drawback of using the dependency ordering is that global constraints can be lost, leading to excessively loose bounds on the orders of magnitudes. Consider, for example, the set $\{y_1 = x_1 * x_2, y_2 = x_3/y_1, y_3 = y_1 * y_2\}$, and let $x_1, x_2$, and $x_3$ be exogenous with orders of magnitude 0. The dependency ordering generated from this set of equations is:

$$\begin{array}{ccc} & \{y_2 = x_3/y_1\} & \\ \nearrow & & \searrow \\ \{y_1 = x_1 * x_2\} & \longrightarrow & \{y_3 = y_1 * y_2\} \end{array}$$

Using this dependency ordering, NAPIER computes the order of magnitude of $y_3$ as follows: from the first equation it computes $om(y_1)$ to be between 0 and 1; from the second equation, and the calculated bound on $om(y_1)$, it computes $om(y_2)$ to be between $-2$ and 0; and from the third equation and the calculated bounds on $om(y_1)$ and $om(y_2)$, it computes $om(y_3)$ to be between $-2$ and 2. However, if all three equations were considered simultaneously, NAPIER computes $om(y_3)$ to be between $-1$ and 1.

The reason for the looser bound in the first case stems from not enforcing some global constraints. For example, the lower bound of $om(y_3)$ can be $-2$ only when $om(y_1) = 0$ and $om(y_2) = -2$. However, when $om(y_1)$ is 0, the second equation dictates that the lowest that $om(y_2)$ can be is $-1$. This fact is lost when the third equation is processed by itself.

More generally, the above problem occurs when a parameter, like $y_3$, depends on two or more parameters, like $y_1$ and $y_2$, whose values have been determined by equations that are earlier in the causal ordering. In using these previously determined values, NAPIER disregards any additional constraints that might hold between those values. Hence, bounds computed based on these values may not be as tight as possible.

NAPIER can partially address this problem by combining adjacent sets of equations in the dependency ordering. This allows more equations to be handled simultaneously, so that more global constraints can be incorporated. However, combining adjacent sets of equations can lead to an increase in the number of sum and difference terms that must be handled simultaneously. Hence, adjacent sets are combined only when the number of sum and difference terms in the resulting set does not increase beyond a threshold (call this threshold $\Delta$).

**Table 7.2.** Maximum value of $\Delta$ for each example.

| Device | $\Delta_{max}$ |
|---|---|
| Bimetallic strip temperature gauge | 11 |
| Bimetallic strip thermostat | 11 |
| Flexible link temperature gauge | 10 |
| Electromagnetic relay thermostat | 9 |
| Galvanometer temperature gauge | 9 |
| Electric bell | 12 |
| Magnetic sizing device | 7 |
| Carbon pile regulator | 9 |
| Tachometer | 7 |
| Car distributor system | 9 |

Combining adjacent sets of equations, as described above, also allows us to partially empirically evaluate the effect of causal ordering on accuracy. We ran NAPIER a number of times on each of our examples, using increasing values of $\Delta$, allowing a maximum of one hour per run. Table 7.2 shows the maximum value of $\Delta$ used for each example. We then compared the bounds that were computed without combining adjacent sets with the bounds that were computed with the maximum setting of $\Delta$. Interestingly, we found that there was *no* loss of accuracy—the bounds computed with and without combining adjacent sets were identical.

To understand the reason for this somewhat surprising result, we now analyze the source of the additional constraints on previously determined values. Let us assume that $om(p_3)$ is computed using previously computed values of $om(p_1)$ and $om(p_2)$. Additional constraints on the values of $om(p_1)$ and $om(p_2)$ stem from one of two sources: (a) $om(p_1)$ and $om(p_2)$ are determined simultaneously; and (b) the value of $om(p_1)$ is used in computing the value of $om(p_2)$, i.e., the values of one of these parameters depends on the value of the other. Point (a) manifests itself as a node in the causal ordering which contains more than one equation. Point (b) manifests itself as multiple paths between two nodes in the causal ordering.

Hence, if the causal ordering, viewed as a graph, satisfies the following two properties:

1. each node contains exactly one equation; and
2. there is at most one path between any two nodes;

then we can show that there will be *no* additional constraints between previously determined values. Hence, there is no loss of accuracy in using the causal ordering.

Table 7.3 shows how closely the causal orderings generated from our examples match the above two properties. The second and third columns of this table show the maximum and average number of equations per node, respectively. One can see that, in all cases, the average number of equations per node is very close to 1. The fourth column shows the minimum number

**Table 7.3.** Properties of the causal ordering graph

| Device | Equations per node | | # of extra edges |
|---|---|---|---|
| | Maximum | Average | |
| Bimetallic strip temperature gauge | 7 | 1.27 | 1 |
| Bimetallic strip thermostat | 7 | 1.24 | 0 |
| Flexible link temperature gauge | 7 | 1.15 | 1 |
| Electromagnetic relay thermostat | 1 | 1.00 | 0 |
| Galvanometer temperature gauge | 12 | 1.29 | 1 |
| Electric bell | 18 | 1.29 | 2 |
| Magnetic sizing device | 17 | 1.26 | 6 |
| Carbon pile regulator | 9 | 1.25 | 2 |
| Tachometer | 18 | 1.21 | 3 |
| Car distributor system | 16 | 1.10 | 0 |

of edges that must be removed from the causal ordering to ensure that there is at most one path between any two nodes. One can see that, in most cases these numbers are very small. Hence, the above analysis provides us with insight into the reasons underlying the fact that, in our examples, the bounds computed with and without combining adjacent sets are identical.

## 7.5 Error estimation

In this section, we estimate the error introduced by the use of the heuristic rules introduced in section 7.2.1. We then analyze some alternate order of magnitude rules, that seem intuitively plausible, and show that these rules introduce unacceptably large errors. The analysis is done using probability theory and is based on interpreting each parameter as a *random variable*.[7] The analysis also uses two assumptions, and we conclude with a discussion of the validity of these assumptions.

### 7.5.1 Estimating the error of heuristic rules

In this section we analyze the error introduced by the heuristic order of magnitude rules 3b, and 4b. (Rules 3c and 4c are similar to rules 3b and 4b, respectively, and are not discussed.) The remaining rules do not introduce errors in the sense that the bounds predicted by them are guaranteed to be conservative, i.e., correct though not necessarily tight.

We start by analyzing rule 3b. Let $Q, Q_1$, and $Q_2$ be parameters such that $Q = Q_1 + Q_2$. Let $f_{Q_1}$ and $f_{Q_2}$ be the probability density functions of $Q_1$ and $Q_2$, respectively, and let $f_{Q_1,Q_2}$ be their joint probability density function. (Briefly, $f_{Q_1}(q_1)$ is the probability that $Q_1$ lies between $q_1$ and $q_1 + dq_1$, and

[7] See [Davenport, 1970] for an introduction to probability theory and random variables.

$f_{Q_1,Q_2}(q_1, q_2)$ is the probability that $Q_1$ lies between $q_1$ and $q_1 + dq_1$, and $Q_2$ lies between $q_2$ and $q_2 + dq_2$.) Since $Q = Q_1 + Q_2$, it follows that the probability that $Q$ lies between $l$ and $u$, for any values $l$ and $u$, is:

$$Prob\{l \leq Q < u\} = \int_{-\infty}^{\infty} \int_{l-q_1}^{u-q_1} f_{Q_1,Q_2}(q_1, q_2) dq_2 dq_1 \tag{7.29}$$

Let us now assume that $om(Q_1) = n_1$ and $om(Q_2) = n_2$, with $n_1 > n_2$. Under these conditions, rule 3b states that $om(Q) = n_1$, i.e., $b^{n_1} \leq Q < b^{n_1+1}$. To estimate the error, $\epsilon$(Rule 3b), in rule 3b, we must calculate the probability that $Q$ lies outside the region from $b^{n_1}$ to $b^{n_1+1}$:

$$\epsilon(\text{Rule 3b}) = 1 - Prob\{b^{n_1} \leq Q < b^{n_1+1}\} \tag{7.30}$$

$$= 1 - \int_{-\infty}^{\infty} \int_{b^{n_1}-q_1}^{b^{n_1+1}-q_1} f_{Q_1,Q_2}(q_1, q_2) dq_2 dq_1 \tag{7.31}$$

To evaluate this integral, we make the following assumptions:

Assumption 1: $Q_1$ and $Q_2$ are independent random variables. Hence, the joint probability density of $Q_1$ and $Q_2$ is just the product of the individual probability densities:

$$f_{Q_1,Q_2}(q_1, q_2) = f_{Q_1}(q_1) f_{Q_2}(q_2) \tag{7.32}$$

Assumption 2: $Q_1$ and $Q_2$ are uniformly distributed on intervals $[b^{n_1}, b^{n_1+1})$ and $[b^{n_2}, b^{n_2+1})$, respectively:

$$f_{Q_1}(q_1) = \begin{cases} \frac{1}{b^{n_1+1}-b^{n_1}} & \text{if } b^{n_1} \leq q_1 < b^{n_1+1} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{Q_2}(q_2) = \begin{cases} \frac{1}{b^{n_2+1}-b^{n_2}} & \text{if } b^{n_2} \leq q_2 < b^{n_2+1} \\ 0 & \text{otherwise} \end{cases}$$

Hence, from Equations 7.31 and 7.32, we get:

$$\epsilon(\text{Rule 3b}) = 1 - \int_{-\infty}^{\infty} \int_{b^{n_1}-q_1}^{b^{n_1+1}-q_1} f_{Q_1}(q_1) f_{Q_2}(q_2) dq_2 dq_1 \tag{7.33}$$

We now use Assumption 2 to split the above integral into two integrals, such that the integrand in both integrals is a non-zero constant throughout the region of integration:

$$\begin{aligned} \epsilon(\text{Rule 3b}) &= 1 - \int_{b^{n_1}}^{b^{n_1+1}-b^{n_2+1}} \int_{b^{n_2}}^{b^{n_2+1}} \frac{1}{b^{n_1+n_2}(b-1)^2} dq_2 dq_1 \\ &\quad - \int_{b^{n_1+1}-b^{n_2+1}}^{b^{n_1+1}-b^{n_2}} \int_{b^{n_2}}^{b^{n_1+1}-q_1} \frac{1}{b^{n_1+n_2}(b-1)^2} dq_2 dq_1 \\ &= 1 - \int_{b^{n_1}}^{b^{n_1+1}-b^{n_2+1}} \frac{b^{n_2+1} - b^{n_2}}{b^{n_1+n_2}(b-1)^2} dq_1 \end{aligned} \tag{7.34}$$

$$-\int_{b^{n_1+1}-b^{n_2+1}}^{b^{n_1+1}-b^{n_2}} \frac{b^{n_1+1}-b^{n_2}-q_1}{b^{n_1+n_2}(b-1)^2}dq_1 \tag{7.35}$$

$$= \frac{b+1}{2b^{n_1-n_2}(b-1)} \tag{7.36}$$

Hence, under Assumptions 1 and 2, the error in rule 3b is maximum when $(n_1 - n_2)$ is minimum, i.e., $(n_1 - n_2) = 1$, which occurs when parameters of consecutive orders of magnitude are being added. When $b = 10$, the maximum error is 6.11%. For larger values of $(n_1 - n_2)$, the errors are even smaller. For example, with $b = 10$, and $(n_1 - n_2) = 2$ (i.e., adding a parameter that is two orders of magnitude smaller) the estimated error is only 0.61%.

The error in rule 4b can also be shown to be $(b+1)/2b^{n_1-n_2}(b-1)$ in a similar way. In particular, if $Q = Q_1 - Q_2$, $om(Q_1) = n_1$, $om(Q_2) = n_2$, and $n_1 > n_2$, then rule 4b predicts that $om(Q) = n_1$. Using Assumptions 1 and 2, $\epsilon$(Rule 4b) can be calculated as follows:

$$\epsilon(\text{Rule 4b}) = 1 - Prob\{b^{n_1} \leq Q < b^{n_1+1}\} \tag{7.37}$$

$$= 1 - \int_{-\infty}^{\infty}\int_{q_1-b^{n_1}+1}^{q_1-b^{n_1}} f_{Q_1,Q_2}(q_1,q_2)dq_2dq_1 \tag{7.38}$$

$$= 1 - \int_{-\infty}^{\infty}\int_{q_1-b^{n_1}+1}^{q_1-b^{n_1}} f_{Q_1}(q_1)f_{Q_2}(q_2)dq_2dq_1 \tag{7.39}$$

$$= 1 - \int_{b^{n_1}+b^{n_2}}^{b^{n_1}+b^{n_2+1}}\int_{b^{n_2}}^{q_1-b^{n_1}} \frac{1}{b^{n_1+n_2}(b-1)^2}dq_2dq_1 - \int_{b^{n_1}+b^{n_2+1}}^{b^{n_1+1}}\int_{b^{n_2}}^{b^{n_2+1}} \frac{1}{b^{n_1+n_2}(b-1)^2}dq_2dq_1 \tag{7.40}$$

$$= 1 - \int_{b^{n_1}+b^{n_2}}^{b^{n_1}+b^{n_2+1}} \frac{q_1-b^{n_1}-b^{n_2}}{b^{n_1+n_2}(b-1)^2}dq_1 - \int_{b^{n_1}+b^{n_2+1}}^{b^{n_1+1}} \frac{b^{n_2+1}-b^{n_2}}{b^{n_1+n_2}(b-1)^2}dq_1 \tag{7.41}$$

$$= \frac{b+1}{2b^{n_1-n_2}(b-1)} \tag{7.42}$$

Hence, under Assumptions 1 and 2, the maximum error in the heuristic rules of section 7.2.1 is 6.11%.

### 7.5.2 Alternate order of magnitude rules

The above error estimation techniques can also be used to analyze alternate rules for order of magnitude reasoning. In particular, we analyze the three inference rules shown in Figure 7.4. These rules were our first attempt at modeling an engineers order of magnitude reasoning. Rule 1′ and 2′ were

| | |
|---|---|
| 1′ | $om(q_1 * q_2) = om(q_1) + om(q_2)$ |
| 2′ | $om(q_1/q_2) = om(q_1) - om(q_2)$ |
| 3a′ | $om(q_1 + q_2) = om(q_1)$ if $om(q_1) = om(q_2)$ |

**Fig. 7.4.** Alternate rules for order of magnitude reasoning.

meant to model reasoning like: "If the resistance, $R$, is about $10^{-1}$ ohms, and the current, $i$, is about $10^{-2}$ amps, then the voltage drop, $V$ $(= iR)$, is about $10^{-3}$ volts." The idea was that the order of magnitude of a product or quotient was the sum or difference, respectively, of the orders of magnitudes of the arguments. Rule $3a'$ was meant to model the intuition that adding parameters of the same order of magnitude results in a parameter of the same order of magnitude. However, we now show that, while these rules may appear intuitively appealing, they are also unacceptably error-prone.

We start by estimating the error in rule 1′. Let $om(Q_1) = n_1$, $om(Q_2) = n_2$, and $Q = Q_1 * Q_2$. Rule 1′ predicts that $om(Q) = n_1 + n_2$. Using Assumptions 1 and 2, $\epsilon$(Rule 1′) can be calculated as follows:

$$\epsilon(\text{Rule } 1') = 1 - Prob\{b^{n_1+n_2} \leq Q < b^{n_1+n_2+1}\} \tag{7.43}$$

$$= 1 - \int_{-\infty}^{\infty} \int_{\frac{b^{n_1+n_2}}{q_1}}^{\frac{b^{n_1+n_2+1}}{q_1}} f_{Q_1,Q_2}(q_1, q_2) dq_2 dq_1 \tag{7.44}$$

$$= 1 - \int_{-\infty}^{\infty} \int_{\frac{b^{n_1+n_2}}{q_1}}^{\frac{b^{n_1+n_2+1}}{q_1}} f_{Q_1}(q_1) f_{Q_2}(q_2) dq_2 dq_1 \tag{7.45}$$

$$= 1 - \int_{b^{n_1}}^{b^{n_1+1}} \int_{b^{n_2}}^{\frac{b^{n_1+n_2+1}}{q_1}} \frac{1}{b^{n_1+n_2}(b-1)^2} dq_2 dq_1 \tag{7.46}$$

$$= 1 - \int_{b^{n_1}}^{b^{n_1+1}} \left(\frac{b^{n_1+n_2+1}}{q_1} - b^{n_2}\right) \frac{1}{b^{n_1+n_2}(b-1)^2} dq_1 \tag{7.47}$$

$$= 1 - \frac{b \ln b - b + 1}{(b-1)^2} \tag{7.48}$$

Substituting $b = 10$ in Equation 7.48, the error in rule 1′, under Assumptions 1 and 2, is 82.68%.

Next, we estimate the error in rule 2′. Let $om(Q_1) = n_1$, $om(Q_2) = n_2$, and $Q = Q_1/Q_2$. Rule 2′ predicts that $om(Q) = n_1 - n_2$. Using Assumptions 1 and 2, $\epsilon$(Rule 2′) can be calculated as follows:

$$\epsilon(\text{Rule } 2') = 1 - Prob\{b^{n_1-n_2} \leq Q < b^{n_1-n_2+1}\} \tag{7.49}$$

$$= 1 - \int_{-\infty}^{\infty} \int_{\frac{q_1}{b^{n_1-n_2+1}}}^{\frac{q_1}{b^{n_1-n_2}}} f_{Q_1,Q_2}(q_1, q_2) dq_2 dq_1 \tag{7.50}$$

$$= 1 - \int_{-\infty}^{\infty} \int_{\frac{q_1}{b^{n_1-n_2+1}}}^{\frac{q_1}{b^{n_1-n_2}}} f_{Q_1}(q_1) f_{Q_2}(q_2) dq_2 dq_1 \tag{7.51}$$

$$= 1 - \int_{b^{n_1}}^{b^{n_1+1}} \int_{b^{n_2}}^{\frac{q_1}{b^{n_1-n_2}}} \frac{1}{b^{n_1+n_2}(b-1)^2} dq_2 dq_1 \quad (7.52)$$

$$= 1 - \int_{b^{n_1}}^{b^{n_1+1}} (\frac{q_1}{b^{n_1-n_2}} - b^{n_2}) \frac{1}{b^{n_1+n_2}(b-1)^2} dq_2 dq_1 \quad (7.53)$$

$$= \frac{1}{2} \quad (7.54)$$

Hence, under Assumptions 1 and 2, the error in rule 2′ is 50%.

Finally, we estimate the error in rule 3a′. Let $om(Q_1) = om(Q_2) = n$, and $Q = Q_1 + Q_2$. Rule 3a′ predicts that $om(Q) = n$. Using Assumptions 1 and 2, $\epsilon$(Rule 3a′) can be calculated as follows:

$$\epsilon(\text{Rule 3a}') = 1 - Prob\{b^n \leq Q < b^{n+1}\} \quad (7.55)$$

$$= 1 - \int_{-\infty}^{\infty} \int_{b^n - q_1}^{b^{n+1}-q_1} f_{Q_1,Q_2}(q_1, q_2) dq_2 dq_1 \quad (7.56)$$

$$= 1 - \int_{-\infty}^{\infty} \int_{b^n - q_1}^{b^{n+1}-q_1} f_{Q_1}(q_1) f_{Q_2}(q_2) dq_2 dq_1 \quad (7.57)$$

$$= 1 - \int_{b^n}^{b^{n+1}-b^n} \int_{b^n}^{b^{n+1}-q_1} \frac{1}{b^{2n}(b-1)^2} dq_2 dq_1 \quad (7.58)$$

$$= 1 - \int_{b^n}^{b^{n+1}-b^n} \frac{b^{n+1} - q_1 - b^n}{b^{2n}(b-1)^2} dq_1 \quad (7.59)$$

$$= 1 - \frac{(b-2)^2}{2(b-1)^2} \quad (7.60)$$

Substituting $b = 10$ into Equation 7.60, the error in rule 3a′, under Assumptions 1 and 2, is 60.49%.

Hence, we have shown that, under Assumptions 1 and 2, the error introduced by the alternate rules shown in Figure 7.4 are greater than or equal to 50%. We believe that these errors are unacceptably large, and hence have chosen not to include these rules in NAPIER.

### 7.5.3 Discussion

The error estimation results presented above depend crucially on Assumptions 1 and 2. We now discuss the validity and scope of these assumptions.

Assumption 1 assumes that the two parameters being combined, $Q_1$ and $Q_2$, are independent random variables. This assumption is reasonable if $Q_1$ and $Q_2$ are exogenous parameters. It is also reasonable if the set of exogenous parameters used to calculate the order of magnitude of $Q_1$ is disjoint from the set of exogenous parameters used to calculate the order of magnitude of $Q_2$. However, if the orders of magnitudes of $Q_1$ and $Q_2$ depend on the order of magnitude of a common exogenous parameter, then $Q_1$ and $Q_2$ are not independent.

Assumption 2 assumes that the two parameters being combined, $Q_1$ and $Q_2$, are uniformly distributed random variables. In the absence of any additional information, this assumption is reasonable for exogenous parameters. However, it breaks down for derived parameters. For example, if $Q_1$ and $Q_2$ are uniformly distributed random variables, and if $Q = Q_1 \ op \ Q_2$ (where $op$ is one of $+, -, *,$ or $/$), then $Q$ is *not* uniformly distributed. Hence, when the order of magnitude of $Q$ is used to calculate the orders of magnitudes of other parameters, Assumption 2 is not valid.

The above discussion implies that our error estimation technique has limited applicability. In particular, the errors estimated in this section cannot be directly used to estimate the error introduced in predictions based on a set of equations. Nonetheless, these techniques have proved useful in helping us select a reasonable set of order of magnitude reasoning rules, while alerting us to the possibility of large errors introduced by alternate rules.

## 7.6 Related work

Order of magnitude reasoning has been widely studied in AI. Murthy [Murthy, 1988] was the first to propose the use of a logarithmic scale for the order of magnitude of a parameter. In that paper, he also provides rules of inference to infer new orders of magnitude from old ones. Some of these rules are similar to ours. For example, he includes rules 3b, 3c, 4b, and 4c. However, instead of 1, he proposes the rule $om(q_1 * q_2) = om(q_1) + om(q_2)$ (which is rule 1′), and instead of rule 3a, he proposes the rule $om(q_1 + q_2) = om(q_1)$ when $om(q_1) = om(q_2)$ (which is rule 3a′). As we saw in section 7.5.2, the estimated error in these rules is too large, and hence we have chosen not to include them in NAPIER. Unlike our work, Murthy provides no analysis of how his inference rules can be used to find the order of magnitudes of parameters related by sets of simultaneous equations. In addition, we also analyze the complexity of order of magnitude inference, and present an approximate reasoning technique that works well in practice.

Raiman [Raiman, 1991; Raiman, 1986] explores the foundations of symbolic order of magnitude reasoning. He defines a variety of order of magnitude scales, such as *Close* and *Comparable*, built out of the basic order of magnitude granularities, *Small* and *Rough*. He introduces ESTIMATES, a system to solve order of magnitude equations. The primary difference between NAPIER and ESTIMATES is one of emphasis: NAPIER can be viewed as providing justifications for making order of magnitude assumptions; ESTIMATES can be viewed as a formalization of the use of such order of magnitude assumptions to symbolically manipulate and simplify equations.

Order of magnitude reasoning in the O(M) formalism [Mavrovouniotis and Stephanopolous, 1987] uses a parameter $e$ to represent the largest parameter that can be considered to be "much smaller" than 1. This is analogous to the parameter $b$ in NAPIER (i.e., $b = 1/e$). However, there are a number of

differences between O(M) and NAPIER. First, the O(M) formalism is based on order of magnitude relations *between* parameters. Hence, it works best when equations involve only *links* (links are ratios of parameters). NAPIER, on the other hand, is based on the order of magnitudes of the parameters themselves, and hence works with any algebraic equations. This is advantageous because it is not always possible to convert equations into equations involving only links. Second, O(M) requires equations to be converted into *assignments*, which allow a new relation or range to be inferred from already known relations. This is a serious restriction since equations can be converted to assignments only in the absence of simultaneous equations. As we have seen, NAPIER does not have this restriction.

NAPIER is also related to interval reasoning discussed in [Moore, 1979; Simmons, 1986; Sacks, 1987]. NAPIER can be viewed as interval reasoning in which the end points of the interval are restricted to a particular set of points of the form $b^n$, with specified base $b$, and any integer $n$. The drawback of this restriction is that under certain conditions, compared to interval reasoning, the bounds inferred by NAPIER are unnecessarily loose (e.g., see the discussion of rule 3a in section 7.2.1). The advantage of this restriction is that, unlike traditional interval reasoners, NAPIER is able to use sets of non-linear simultaneous equations to infer parameter bounds. In addition, the ability to simultaneously process all the equations in a set allows NAPIER to exploit global constraints to compute tighter bounds (see section 7.4). Another distinguishing characteristic of NAPIER, which classifies it as an order of magnitude reasoning system rather than just an interval reasoner, is the use of heuristic rules (e.g, rule 3b).

## 7.7 Summary

In this chapter we described an implemented order of magnitude reasoning system called NAPIER. NAPIER defines the order of magnitude of a parameter on a logarithmic scale and uses a set of rules to propagate order of magnitudes through equations. A novel feature of NAPIER is its handling of non-linear simultaneous equations. Since the order of magnitude reasoning rules are all disjunctions of linear inequalities, NAPIER is able to use linear programming, in conjunction with backtracking, to find bounds on the order of magnitudes of parameters related by sets of non-linear simultaneous equations.

We also showed that order of magnitude reasoning using NAPIER's rules is intractable. Hence, NAPIER uses an approximate reasoning technique, based on causal ordering, leading to a practically useful system. This approximate reasoning technique trades off accuracy for speed, though in practice there does not appear to be any loss of accuracy.

Some of NAPIER's rules are heuristic rules, and we have estimated the error introduced by the use of these rules. We have also shown that intuitively appealing alternate heuristic rules lead to large estimated errors.

# 8. Model selection program and results

In this chapter we describe an implemented model selection program based on the algorithms developed in the previous two chapters. The program assumes that the knowledge base of component and model fragment classes satisfies all the restrictions introduced in Chapters 5 and 6. However, the actual knowledge base that we have constructed does not satisfy two of the restrictions: (a) the knowledge base does not include all the ownership constraints (justified in Section 5.4.5); and (b) parameters are not required to be locally self-regulating (justified in Section 6.6.3).

In addition to the knowledge base of component and model fragment classes, the program has the following inputs:

1. The structure of the device, which includes a description of the components of the device, the physical and structural properties of these components, and the structural relations between these components. As discussed in Section 3.6.1, this is the structural context of the device.
2. The expected behavior of the device.
3. Orders of magnitudes of initial values and exogenous values of parameters, which are used in generating the behavioral context of the device. Initial values are used when a parameter is determined by integration, while exogenous values are used when a parameter is assumed to be exogenous. Section 8.1.2 discusses this in detail.
4. Orders of magnitudes of the thresholds used in behavioral constraints. This allows us to check the behavioral constraints.

The program produces an adequate model by first finding an initial causal model and then simplifying this causal model using the variant of *find-minimal-causal-model* discussed in Section 5.7. Section 8.1 discusses a heuristic method for finding an initial causal model that is simpler than the most accurate model, and demonstrates this method on the temperature gauge shown in Figure 1.1. Section 8.2 illustrates the simplification procedure of Section 5.7 on the initial causal model of the above temperature gauge. Finally, Section 8.3 describes the results of running this model selection program on a variety of electromechanical devices.

## 8.1 Finding an initial causal model

The model selection algorithm developed in Section 5.1.2 was based on the fact that a causal model exists if and only if the most accurate model of the device is a causal model. Hence, a minimal causal model can be found by simplifying the most accurate model of the device. However, this is often undesirable because the most accurate device model can be unnecessarily complex, so that simplifying it can take a long time. In this section we introduce a heuristic technique for finding an initial causal model. This initial causal model is a subset of, and hence simpler than, the most accurate model. The heuristic technique is applicable only if the knowledge base satisfies the following restriction:

- If a model satisfies all the structural and behavioral coherence constraints, then any simpler consistent and complete model that uses model fragments from the same assumption classes also satisfies all the structural and behavioral coherence constraints.

This restrictions ensures that any causal model, not just the most accurate model, can be simplified using the techniques developed in Chapter 5. We assume that our knowledge base satisfies the above restriction.

The heuristic technique for finding an initial causal model is based on the *component interaction heuristic*. Section 8.1.1 introduces this heuristic, and Section 8.1.2 describes the heuristic technique.

### 8.1.1 Component interaction heuristic

A device model can be viewed as having two major parts: (a) models of individual components; and (b) models of interactions between components. Components can interact with each other only when appropriate structural relations hold between them. For example, in our knowledge base, the `connected-to` relation between terminals supports electrical interactions between the connected terminals. Hence, two components can electrically interact with each other if a terminal of one component is `connected-to` a terminal of the other component. As another example, the `coiled-around` relation between wires and physical objects supports thermal interactions between the wire and the physical object that it is `coiled-around`.

In addition to requiring appropriate structural relations, components can interact only when the component models are compatible with the type of interaction under consideration. For example, a wire can electrically interact with a battery if one of the wire's terminals is `connected-to` one of the battery's terminals. However, this interaction can take place only if both the wire and the battery are being modeled as electrical components, e.g., modeling the wire as an electrical conductor, and the battery as a voltage source is compatible with the electrical interaction.

Hence, components can interact with each other if the following conditions are satisfied: (a) the components are related by the structural relations that support the interaction; and (b) the component models are compatible with the interaction. The *component interaction heuristic* is based directly on the above observations. It states that if a set of components are related by one or more structural relations that support an interaction, and if one of the component models is compatible with this interaction, then the remaining component models must be augmented to be compatible with this interaction. This allows the components in the set to interact with each other via that interaction. Note that if none of the component models is compatible with the interaction, then no augmentations are necessary.

The component interaction heuristic is implemented as a set of heuristic coherence constraints. Each such constraint is a version of the component interaction heuristic that is specialized for a particular type of interaction and a particular set of structural relations. Heuristic coherence constraints are expressed as horn rules, and like structural and behavioral constraints, are associated with model fragment classes. For example, the following heuristic coherence constraint

```
(implies
   (and (terminals ?object ?term1)
        (voltage-terminal ?term1)
        (connected-to ?term1 ?term2)
        (terminal-of ?term2 ?comp2))
   (electrical-component ?comp2))
```

in the `electrical-component` model fragment class[1] says that if a component is being modeled as an `electrical-component`, and one of its voltage terminals is connected to a terminal of another component, then the other component must also be modeled as an `electrical-component`. This allows the two components to interact by sharing voltages at the connected terminals.

As another example, a heuristic coherence constraint associated with the `thermal-object` model fragment class is the following:

```
(implies
   (and (wire ?object)
        (coiled-around ?object ?core))
   (thermal-object ?core))
```

which implements the component interaction heuristic for the thermal interaction between a wire and an object around which it is coiled.

We will require that the initial causal model must satisfy all applicable heuristic coherence constraints. We now show how these constraints are used to find an initial causal model.

[1] Hence, "`?object`" is bound to a component being modeled as an `electrical-component`.

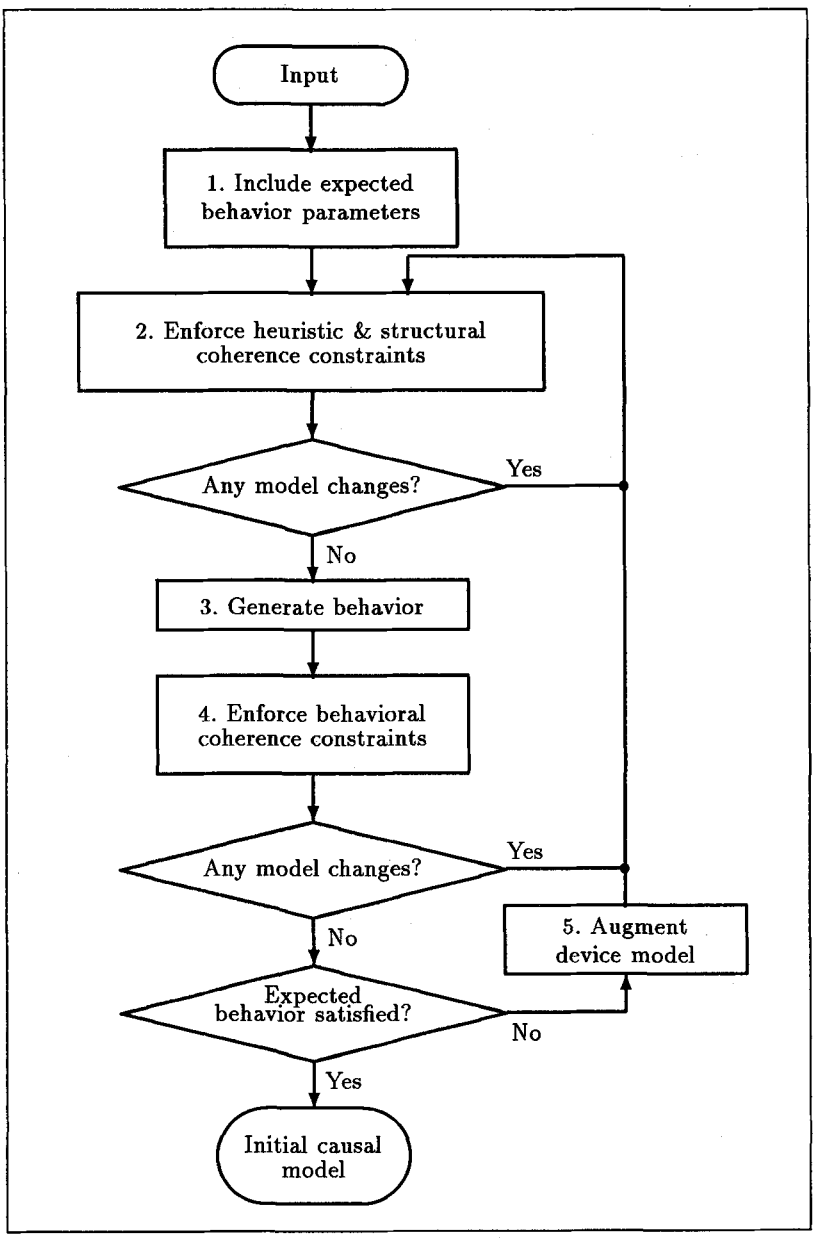



**Fig. 8.1.** Algorithm for finding an initial causal model

### 8.1.2 Finding an initial causal model

Figure 8.1 shows a flowchart describing our algorithm for finding an initial causal model. The input to the algorithm is as described earlier: (a) the structure of the device; (b) the expected behavior of the device; (c) orders of magnitudes of initial values and exogenous values of parameters; and (d) orders of magnitudes of thresholds. In addition, each component can have a set of model fragment classes preselected for it—these correspond to modeling decisions made by the user.

There are five major steps in this algorithm, numbered 1–5 in the rectangular boxes of Figure 8.1. Each step, except step 3 entitled "Generate behavior," can modify the device model by adding one or more model fragments to it. Whenever a step adds a model fragment to the device model, it also adds the most accurate model fragment from every assumption class required by the model fragment. Hence, at the end of every step, all the *requires* constraints are satisfied.

We now describe the details of the five steps, and the flow of control between them. We will illustrate the algorithm on the temperature gauge in Figure 1.1. Figure 8.2 shows some, though not all, of the components in the temperature gauge, together with their component classes. Note, in particular, the last four components, which correspond to structural abstractions that are automatically created from the device description (see Section 2.4.2). `bms-wire` is an abstraction representing `wire-1 coiled-around bms-3`, while the remaining three abstractions represent `pointer-2`, `bms-3`, and `battery-5`, respectively, `immersed-in atm-6`.

| Component | Component classes |
|---|---|
| thermistor-1 | Thermistor |
| pointer-2 | Pointer |
| bms-3 | Bimetallic-strip |
| wire-4 | Wire |
| battery-5 | Battery |
| atm-6 | Atmosphere |
| bms-wire | Coil-structure |
| atm-pointer | Immersion-structure |
| atm-bms | Immersion-structure |
| atm-battery | Immersion-structure |

**Fig. 8.2.** Components and their initial models.

**Include expected behavior parameters.** In the first step, the device model is augmented to ensure that it contains all the parameters in the expected behavior. This is a good place to start because every causal model must contain every parameter in the expected behavior.

Recall that a parameter represents a numerical attribute of some component. The relationship between a component and a parameter representing

one of its numerical attributes is mediated by parameter functions: applying the parameter function to the component returns the parameter. In Chapter 2 we saw that the the `attributes` clause in the definition of a model fragment class defines the parameter functions that can be used on instances of that class. Hence, a device model contains a parameter if and only if the component corresponding to the parameter is an instance of the model fragment class that defines the corresponding parameter function.

More precisely, if $p$ is a parameter, let $c_p$ denote the component of which $p$ is a numerical attribute, let $f_p$ denote the parameter function such that $f_p(c_p) = p$, and let $M_p$ be the model fragment class that defines $f_p$. A device model contains the parameter $p$ if and only if $c_p$ is modeled as an instance of $M_p$.

If the expected behavior contains a parameter $p$ such that the device model does not contain $p$, then it means that $c_p$ is not being modeled as an instance of $M_p$. This situation can be rectified by modeling $c_p$ as an instance of some model fragment class $M$ such that:

1. $M$ is a specialization of $M_p$;
2. $M$ is in the transitive closure of the `possible-models` of the component class of $c_p$; and
3. $M(c_p)$ is the most accurate applicable model fragment in its assumption class.

The first condition ensures that $c_p$ is an instance of $M_p$, so that $p$ becomes part of the device model. The second condition ensures that $M$ is a possible way of modeling $c_p$. The third condition ensures that we only consider model fragments that are parts of the most accurate model of the device, so that the initial causal model that we create will be a subset of the most accurate device model.

Note that $M(c_p)$ must be an applicable model fragment so that all the structural and behavioral preconditions of $M$ must be satisfied with `?object` bound to $c_p$. However, since the device's behavior has not yet been generated, behavioral preconditions cannot be evaluated. Hence, any model fragment classes with behavioral preconditions are inapplicable. This is not a serious problem because almost none of the most accurate model fragment classes in our library have behavioral preconditions—behavioral preconditions primarily control the use of approximations. In addition, we assume that if a most accurate model fragment class has behavioral preconditions, then all the other model fragment classes in that assumption class also have behavioral preconditions, so that none of them are applicable at this stage. This ensures that we only consider most accurate model fragment classes at this stage. Most accurate model fragment classes with behavioral preconditions will be considered in the fourth step, after the behavior is generated.

In general, for any parameter $p$ and component $c_p$, there can be more than one model fragment class that satisfies the above three conditions. Let $M_1$ and $M_2$ be model fragment classes that satisfy the above three conditions,

and let $M_2$ be a specialization of $M_1$. Hence, modeling $c_p$ as an instance of $M_2$ will also model it as an instance of $M_1$. However, modeling $c_p$ as an instance of $M_1$ will not model it as an instance of $M_2$. Hence, to keep the initial causal model as simple as possible, we make $c_p$ an instance of the most general model fragment class that satisfies the above three conditions. The use of the other model fragment classes that satisfy the above three conditions will be discussed later.

Let us illustrate this step on the temperature gauge in Figure 1.1. Let us assume that the expected behavior of this temperature gauge is:

```
(causes (temperature thermistor-1)
        (angular-position pointer-2))
```

This means that the parameters representing the temperature of `thermistor-1` and the angular position of `pointer-2` must be part of the device model. The `temperature` parameter function is defined in the model fragment class `Temperature-model`. A search of the `possible-models` of `Thermistor` reveals that `Thermal-object` and `Thermal-thermistor` are the most accurate model fragment classes that are specializations of `Temperature-model`. Since, `Thermal-object` is a generalization of `Thermal-thermistor`, we model `thermistor-1` as a `Thermal-object`, i.e., the device model is augmented with the model fragment `Thermal-object(thermistor-1)`. Similarly, `pointer-2` is modeled as an instance of `Rotating-object`.

In addition, as discussed earlier, the first step is not complete until all the *requires* constraints are satisfied. In particular, if the device model contains a model fragment $M(c)$, where $M$ is a model fragment class and $c$ is a component, and if the `required-assumption-classes` of $M$ specifies assumption class $A$, then the device model must contain a model fragment from assumption class $A(c)$. Once again, to ensure that the resulting model is a subset of the most accurate device model, we augment the device model with the most accurate applicable model fragment in $A(c)$, i.e., if $M_A$ is the most accurate model fragment class of $A$, we augment the device model with model fragment $M_A(c)$. ($M_A$ is assumed to be a `possible-model` of the component class of $c$.)

In the current example, `Thermal-object` specifies `Thermal-model-class` as one of its `required-assumption-classes`. The most accurate model fragment class of `Thermal-model-class` is `Dynamic-thermal-model` (see Equation 6.2). Hence, to satisfy this *requires* constraint, `thermistor-1` is made an instance of `Dynamic-thermal-model`. The resulting device model is shown in Figure 8.3.

**Enforcing heuristic and structural coherence constraints.** Once the device model contains all the parameters in the expected behavior, the algorithm checks all applicable structural and heuristic coherence constraints. If all these constraints are satisfied, the algorithm merely proceeds to the next

| Component | Model |
|---|---|
| thermistor-1 | Thermal-object<br>Dynamic-thermal-model |
| pointer-2 | Rotating-object |
| bms-3 | |
| wire-4 | |
| battery-5 | |
| atm-6 | |
| bms-wire | |
| atm-pointer | |
| atm-bms | |
| atm-battery | |

**Fig. 8.3.** Component models after the expected behavior parameters have been included.

step. However, if a constraint is not satisfied, it augments the device model as described below.

Recall that a structural coherence constraint is like a horn rule, except that the consequent of the rule is a disjunction of all the model fragments in an assumption class. Hence, if a structural coherence constraint is not satisfied, it means that the current device model does not include a model fragment from that assumption class. This situation is exactly analogous to the case where one of the *requires* constraints is not satisfied. Hence, it is rectified in the same way: the device model is augmented with the most accurate applicable model fragment in the assumption class. Once again, by choosing the most accurate model fragment, we ensure that the resulting device model continues to be a subset of the most accurate device model.

Heuristic coherence constraints, on the other hand, are just horn rules, i.e., the consequent of the rule is of the form $M(c)$, where $M$ is a model fragment class and $c$ is a component. Hence, if a heuristic coherence constraint is not satisfied, it means that the current device model does not include the model fragment in the consequent of the rule, i.e., $c$ is not an instance of $M$. This situation is exactly analogous to the case, discussed earlier, where $c_p$ had to be an instance of $M_p$ to ensure that the device model contained parameter $p$. Hence, it is rectified in exactly the same way, i.e., by making $c$ an instance of the most general specialization of $M$ that is a `possible-model` of the component class of $c$.

For example, `pointer-2` was modeled as a `Rotating-object` in the first step. Because of the linkage connecting `pointer-2` to the free end of bimetallic strip `bms-3`, a heuristic coherence constraint requires a kinematic interaction between `pointer-2` and `bms-3`. This constraint can be satisfied by modeling `bms-3` as an instance of `Thermal-bimetallic-strip`, which models the deflection of the free end of the bimetallic strip as a function of its temperature.

As in the first step, all *requires* constraints are also enforced. In particular, `Thermal-bimetallic-strip` is a specialization of `Thermal-object`, which

*requires* the use of a model fragment class in the **Thermal-model-class** assumption class. As before, the most accurate model fragment is used, so that the device model is augmented with **Dynamic-thermal-model(bms-3)**.

If there are any changes to the device model in this step, the step is repeated to check whether additional structural or heuristic coherence constraints have been violated. This repetition continues until all such constraints are satisfied, at which point the algorithm proceeds to the next step.

For example, we just modeled **bms-3** as an instance of **Thermal-object**. Since **bms-3** is **immersed-in** **atm-6**, a thermal interaction is possible between them. Hence, a heuristic coherence constraint requires that **atm-6** should be modeled as a **Thermal-object** and **atm-bms** should be modeled as a **Thermal-conductor**. Similarly, since **wire-4** is **coiled-around** **bms-3**, a thermal interaction is possible between them, and hence a heuristic coherence constraint requires that **wire-4** should be modeled as a **Thermal-object** and **wire-bms** as a **Thermal-conductor**. **Resistive-thermal-conductor** is the most general specialization of **Thermal-conductor** that is also a **possible-model** of both **Immersion-structure** and **Coil-structure**. Hence, **atm-bms** and **bms-wire** are both made instances of **Resistive-thermal-conductor**.

Modeling the atmosphere as a **Thermal-object** means that a thermal interaction is possible with all components **immersed-in** it. Hence, a heuristic coherence constraint requires that both **pointer-2** and **battery-5**, which are **immersed-in** the atmosphere, must be modeled as **Thermal-objects**, and the corresponding **Immersion-structures** are modeled as **Thermal-conductors**. Finally, to satisfy all the *requires* constraints, all the **Thermal-objects** are also modeled as instances of **Dynamic-thermal-model**. Once this is done, all the structural and heuristic coherence constraints are satisfied, and the algorithm continues to the next step. The resulting model is shown in Figure 8.4.

**Generating the behavior.** In the third step, the algorithm uses the device model constructed in the previous steps to generate the behavior. This involves calculating the orders of magnitudes of all the parameters, using the techniques developed in Chapter 7. The order of magnitude of parameters assumed to be exogenous in the device model are found in the input to the algorithm. Hence, the input must specify an exogenous value for every parameter that can be assumed to be exogenous in some device model.

Recall that the techniques of Chapter 7 were applicable only to sets of algebraic equations. However, the device model constructed above contains differential equations. We address this mismatch by only computing the behavior at a particular point in time. The particular point in time is defined by the values of the parameters being integrated. One can think of these values as being analogous to initial values specified for numerical integration. These initial values are specified in the input to the algorithm, and hence the input must specify an initial value for every parameter that can be integrated in some device model.

| Component | Model |
|---|---|
| thermistor-1 | Thermal-object<br>Dynamic-thermal-model |
| pointer-2 | Rotating-object<br>Thermal-object<br>Dynamic-thermal-model |
| bms-3 | Thermal-bimetallic-strip<br>Dynamic-thermal-model |
| wire-4 | Thermal-object<br>Dynamic-thermal-model |
| battery-5 | Thermal-object<br>Dynamic-thermal-model |
| atm-6 | Thermal-object<br>Dynamic-thermal-model |
| bms-wire | Resistive-thermal-conductor |
| atm-pointer | Resistive-thermal-conductor |
| atm-bms | Resistive-thermal-conductor |
| atm-battery | Resistive-thermal-conductor |

**Fig. 8.4.** Component models after the heuristic coherence constraints have been repeatedly satisfied.

Consider the following subset of the equations of the device model in Figure 8.4. This subset corresponds to the thermal interaction between battery-5 and atm-6:

$$dT_b/dt = C_b f_{ab}$$
$$dT_a/dt = C_a f_a$$
$$f_{ab} = \gamma_{ab}(T_a - T_b)$$
$$exogenous(C_b)$$
$$exogenous(C_a)$$
$$exogenous(\gamma_{ab})$$

where the parameters have the following denotations:

$T_b$:Temperature of battery-5
$C_b$:Heat capacity of battery-5
$T_a$:Temperature of atm-6
$C_a$:Heat capacity of atm-6
$f_{ab}$:Heat flow from atm-6 to battery-5
$f_a$:Net heat flowing into atm-6
$\gamma_{ab}$:Thermal conductance of atm-battery

Since $C_a$, $C_b$, and $\gamma_{ab}$ are assumed to be exogenous in this model, the program looks up their exogenous values from the input. Since $T_a$ and $T_b$ are determined by integrating $dT_a/dt$ and $dT_b/dt$, respectively, the program looks up their initial values from the input. Then, using the exogenous value of $\gamma_{ab}$, and the initial values of $T_a$ and $T_b$, the program calculates the order of magnitude of $f_{ab}$, using the techniques developed in Chapter 7. Similarly, the program calculates the orders of magnitudes of all the parameters in the device model.

**Enforcing behavioral coherence constraints.** In the fourth step, the algorithm uses the behavior generated above to enforce all the behavioral coherence constraints. This step is exactly analogous to the way structural coherence constraints were enforced in the second step. If there are any changes to the device model, the algorithm loops back to the second step, to ensure that all the structural and heuristic coherence constraints continue to be satisfied. The algorithm loops through steps two, three, and four, until all the heuristic, structural, and behavioral coherence constraints are satisfied. Once this happens, the algorithm proceeds to check whether the current device model satisfies the expected behavior. If the expected behavior is satisfied, we have an initial causal model, and the algorithm terminates. However, if the expected behavior is not satisfied, the algorithm proceeds to the fifth step.

The device model constructed above satisfies all the behavioral coherence constraints, but does not satisfy the expected behavior. Hence, the algorithm proceeds to the fifth step with that model.

**Augmenting the device model.** In the fifth step, the algorithm augments the device model as follows. Recall that in the first step, to ensure that the device model contained the parameter $p$, the algorithm had to make $c_p$ an instance of $M_p$. To do this it chose the most general specialization of $M_p$ that was also a `possible-model` of the component class of $c_p$. However, other more specific model fragment classes could also have been used to satisfy this constraint. Similarly, the algorithm chose the most general way to satisfy the heuristic coherence constraints in the second step. Once again, more specific model fragment classes could have been used to satisfy these constraints.

In the fifth step, the algorithm augments the device model with one of the more specific ways of satisfying the constraints in the first and second step. The algorithm then loops back to the second step, and continues looping through steps two, three, four, and five, until a causal model is found.

If no causal model is found, and there are no additional ways of satisfying the constraints in the first and second step, then the algorithm terminates with failure, reporting that there is no causal model. This is justified because the component interaction heuristic guarantees that the component models used in the final device model cannot interact with any other components, and hence no augmentation of the device model can lead to a causal model.

For example, the first step satisfied the constraint that `thermistor-1` be modeled as an instance of `Temperature-model` by modeling `thermistor-1` as an instance of `Thermal-object`. A more specific way of satisfying this constraint is to model `thermistor-1` as a `Thermal-thermistor`. Since the device model constructed until now is not a causal model, the fifth step augments the device model with the `Thermal-thermistor(thermistor-1)` model fragment, and loops back to the second step.

Since `Thermal-thermistors` are also `Electrical-components`, a heuristic coherence constraint requires that all components `connected-to` any of

thermistor-1's voltage terminals must be modeled as Electrical-components. Hence, wire-4 and battery-5 should be modeled as Electrical-components. This is achieved by modeling wire-4 as an Electrical-conductor, and battery-5 as a Voltage-source.

To complete the description of electrical conduction (i.e., to satisfy the *requires* constraint associated with Electrical-conductor), wire-4 is modeled as an instance of Resistor, and to complete the description of resistance wire-4 is further modeled as a Temperature-dependent-resistance. Similarly, to complete the description of a voltage source, battery-5 is modeled as a Voltage-source-with-internal-resistance.

| Component | Model |
|---|---|
| thermistor-1 | Thermal-object<br>Dynamic-thermal-model<br>Thermal-thermistor |
| pointer-2 | Rotating-object<br>Thermal-object<br>Dynamic-thermal-model |
| bms-3 | Thermal-bimetallic-strip<br>Dynamic-thermal-model |
| wire-4 | Thermal-object<br>Dynamic-thermal-model<br>Electrical-conductor<br>Resistor<br>Temperature-dependent-resistance<br>Thermal-resistor |
| battery-5 | Thermal-object<br>Dynamic-thermal-model<br>Voltage-source<br>Voltage-source-with-internal-resistance |
| atm-6 | Thermal-object<br>Dynamic-thermal-model |
| bms-wire | Resistive-thermal-conductor |
| atm-pointer | Resistive-thermal-conductor |
| atm-bms | Resistive-thermal-conductor |
| atm-battery | Resistive-thermal-conductor |

**Fig. 8.5.** The initial causal model.

Using this model, the order of magnitude behavior is generated, and the behavioral coherence constraints are checked. Assuming that the heat generated in wire-4 is greater than the electrical-power-threshold, a behavioral coherence constraint requires that wire-4 be modeled as a Thermal-resistor. The resulting model, shown in Figure 8.5, satisfies all the heuristic, structural, and behavioral coherence constraints. One can show that this model satisfies the expected behavior, and hence it is the initial causal model generated by the algorithm.

## 8.2 Simplifying the model

The initial causal model identified above is then simplified in two stages using the techniques developed in Section 5.7. The behavior generated using this initial causal model is used to evaluate all behavioral preconditions and behavioral coherence constraints. We now illustrate the simplification procedure on the model in Figure 8.5.

In the first stage of simplification, model fragments are replaced by their approximations, until no more simplification by approximation is possible. `Dynamic-thermal-model` has two immediate approximations: `Constant-temperature-model` and `Equilibrium-thermal-model`. These correspond to exogenizing and equilibrating the differential equation in `Dynamic-thermal-model`. `Constant-resistance` is the only approximation of `Temperature-dependent-resistance`, `Constant-voltage-source` is the only approximation of `Voltage-source-with-internal-resistance`, and `Ideal-thermal-conductor` and `Ideal-thermal-insulator` are the two approximations of `Resistive-thermal-conductor`.

The second column in Figure 8.6 shows one possible result of approximating the initial causal model's model fragments as much as possible, while retaining the causal model property (Section 5.7.1 called this the "simplest causal model by approximating"). In fact, in this case, the behavioral preconditions dictate that this is the only such causal model. For example, `Resistive-thermal-conductor(atm-battery)` cannot be approximated by `Ideal-thermal-insulator(atm-battery)` because the thermal resistance of `atm-battery` is not large enough. In addition, model fragments like `Resistive-thermal-conductor(bms-wire)` cannot be approximated because then the model ceases to be a causal model.

The above simplest causal model by approximating is then simplified further by retaining only the relevant model fragments (as described in Section 5.7.2). The resulting minimal causal model is shown in the third column of Figure 8.6. Irrelevant model fragments describing the thermal properties of `pointer-2` and `battery-5` have been dropped, as have the model fragments describing the heat conduction properties of `atm-pointer` and `atm-battery`.

## 8.3 Implementation and results

We have implemented the above model selection algorithm in Common Lisp, and tested it on a variety of electromechanical devices. We now give an overview of this implementation.

### 8.3.1 Overview of the knowledge base

We have constructed a library of 20 different types of components including wires, bimetallic strips, springs, and permanent magnets. The library of

| Component | Simplest causal model by approximating | Minimal causal model |
|---|---|---|
| thermistor-1 | Thermal-object<br>Constant-temperature-model<br>Thermal-thermistor | Thermal-object<br>Constant-temperature-model<br>Thermal-thermistor |
| pointer-2 | Rotating-object<br>Thermal-object<br>Equilibrium-thermal-model | Rotating-object |
| bms-3 | Thermal-bimetallic-strip<br>Equilibrium-thermal-model | Thermal-bimetallic-strip<br>Equilibrium-thermal-model |
| wire-4 | Thermal-object<br>Equilibrium-thermal-model<br>Electrical-conductor<br>Resistor<br>Constant-resistance<br>Thermal-resistor | Thermal-object<br>Equilibrium-thermal-model<br>Electrical-conductor<br>Resistor<br>Constant-resistance<br>Thermal-resistor |
| battery-5 | Thermal-object<br>Equilibrium-thermal-model<br>Voltage-source<br>Constant-voltage-source | <br><br>Voltage-source<br>Constant-voltage-source |
| atm-6 | Thermal-object<br>Constant-temperature-model | Thermal-object<br>Constant-temperature-model |
| bms-wire | Resistive-thermal-conductor | Resistive-thermal-conductor |
| atm-pointer | Ideal-thermal-conductor | |
| atm-bms | Resistive-thermal-conductor | Resistive-thermal-conductor |
| atm-battery | Ideal-thermal-conductor | |

**Fig. 8.6.** The two stages of simplifying the initial causal model. The boxed model fragments in the second column are the approximate model fragments that have replaced more accurate model fragments in the initial causal model.

model fragment classes consists of approximately 150 different types of model fragment classes including descriptions of electricity, magnetism, heat, elasticity, and the kinematics and dynamics of one-dimensional motion (including both rotation and translation). Each component class has an average of 30 model fragment classes describing different aspects of its behavior.

### 8.3.2 Overview of the devices

The model selection program has been tested on ten electromechanical devices drawn from [Artobolevsky, 1980; Macaulay, 1988; van Amerongen, 1967]. Table 8.1 shows the names of these ten devices, and the number of components in each of them. The number of components in each device is the sum of the number of components in the original device description and the number of structural abstractions identified by the system (see Section 2.4.2). One can see that the devices range in complexity from only 10 components

in the bimetallic strip thermostat, to 54 components in the car distributor system.

Some of these devices can operate in more than one operating region. Each operating region corresponds to a different set of inputs to the model selection program. Hence, different operating regions can have different device structures, expected behaviors, initial and exogenous values, and thresholds. Table 8.1 shows the number of operating regions of each device that our model selection program was run on.

**Table 8.1.** Number of components and operating regions used in each device.

| Device name | Number of components | Number of operating regions |
|---|---|---|
| Bimetallic strip temperature gauge | 12 | 1 |
| Bimetallic strip thermostat | 10 | 2 |
| Flexible wire temperature gauge | 13 | 1 |
| Galvanometer temperature gauge | 19 | 1 |
| Electric bell | 22 | 2 |
| Magnetic sizing device | 22 | 1 |
| Carbon pile regulator | 26 | 1 |
| Electromagnetic relay thermostat | 30 | 3 |
| Tachometer | 34 | 1 |
| Car distributor system | 54 | 1 |

We now give a brief description of each of these devices, highlighting their most important aspects from the modeling perspective. Detailed descriptions of these devices can be found in Appendix B. These devices have been selected primarily to demonstrate the fact that similar components in different devices can be modeled differently. In addition, device descriptions also include irrelevant information. In all cases, our model selection program correctly identifies the right component models, and disregards irrelevant information.

Bimetallic strip temperature gauge: This is just the temperature gauge in Figure 1.1. To understand how it works, one must model the heat generated in the wire due to current flow, and the deflection of the bimetallic strip due to temperature changes.

Bimetallic strip thermostat: This device contains a bimetallic strip which regulates the temperature of a room by turning on a heater when the room becomes too cold. To understand how it works, one must model both the deflection of the bimetallic strip due to temperature changes and the electrical conductivity of the bimetallic strip.

Flexible wire temperature gauge: This temperature gauge is very similar to the one in Figure 1.1, except that the pointer's angular position is determined by the length of a wire, rather than the deflection of a bimetallic strip. Since the wire's length depends on its temperature, it follows that

to understand how this temperature gauge works, we must model not only the heat generated in the wire due to current flow, but also the dependence of the wire's length on its temperature.

Galvanometer temperature gauge: This temperature gauge is also similar to the one if Figure 1.1, except that the current in the circuit is measured using a galvanometer, rather than measuring the deflection of the bimetallic strip. A galvanometer works by measuring the magnetic field generated by the current flowing in a coil of wire. Hence, to understand how this temperature gauge works, we must model the magnetic field generated by the coil of wire, but need not model the heat generated in the wire.

Electric bell: This device consists of a hammer and a bell. The device goes through two major operating regions. In the first operating region, an electric circuit is completed through the hammer, activating an electromagnet which attracts the hammer, causing the hammer to strike the bell. Hence, to understand the electric bell's operation in this operating region, we must model the electric and magnetic properties of the bell. When the hammer strikes the bell, it breaks the electric circuit, thereby deactivating the electromagnet, and hence allowing the hammer to return to its original position. To understand why this happens, we must model the hammer as a spring that restores the hammer's position when the external electromagnetic force is removed.

Magnetic sizing device: This device is used to measure the size of workpieces in a factory. It works on the principle that the magnetic flux in a magnetic circuit is dependent on the length of the air gaps in the circuit. The device is constructed to make the length of the air gaps proportional to the size of the workpiece. The important point to note here is that, unlike in a galvanometer, to understand how this device works, we must model magnetism using a magnetic circuit ontology (see the discussion in Section 5.2). In addition, the air gaps must be modeled as magnetic flux conductors.

Carbon pile regulator: This device allows manual regulation of the voltage supplied to another device. The principle of its operation is that the resistance of a carbon pile is proportional to the compressive force acting on it. Hence, to understand its operation, we must model the dependence of the carbon pile's resistance on the compressive force.

Electromagnetic relay thermostat: This thermostat is similar to the bimetallic strip thermostat. The primary difference is that in the bimetallic strip thermostat the bimetallic strip directly turned on the heater, while in this device the bimetallic strip turns on an electromagnetic relay which turns on the heater. Understanding the operation of the electromagnetic relay requires the magnetic circuit ontology mentioned above.

Tachometer: This device, which measures angular speed, is very interesting because it consists of two similar structures, each of which consists of a coil of wire wound around an iron core and embedded in an external

magnetic field. The interesting part is that, though these structures are very similar, they are modeled very differently. The first behaves as an electric generator: the rotation of the coil in the magnetic field causes an induced voltage. The second behaves as a galvanometer: the current flowing in the coil causes the coil to deflect in the external field.

Car distributor system: This is just a description of the distributor system in a car, including the spark plugs. The interesting part of this device is the air gaps in the spark plugs which are modeled as electric conductors because of the high voltage drops across them.

### 8.3.3 Results

Table 8.2 shows a summary of our experimental results on the devices described above. As mentioned above, the model selection program was run on more than one operating region for some of the devices. In such cases, the numbers in this table correspond to the totals over all the runs for that device.

**Table 8.2.** Summary of experimental results

| Device name | Estimated space | Generated space | Time (sec) on Explorer II |
|---|---|---|---|
| Bimetallic strip temperature gauge | 3.8e16 | 46 | 28.4 |
| Bimetallic strip thermostat | 5.4e12 | 85 | 43.7 |
| Flexible wire temperature gauge | 2.6e20 | 78 | 59.9 |
| Galvanometer temperature gauge | 6.1e31 | 120 | 149.8 |
| Electric bell | 6.6e40 | 117 | 262.4 |
| Magnetic sizing device | 2.1e51 | 117 | 456.0 |
| Carbon pile regulator | 1.5e49 | 115 | 262.5 |
| Electromagnetic relay thermostat | 8.7e49 | 293 | 472.7 |
| Tachometer | 6.8e58 | 195 | 503.6 |
| Car distributor system | 9.9e72 | 160 | 352.6 |

The second column displays the total number of models that (a) have at most one model fragment from each assumption class; and (b) have a model fragment from each required assumption class. This number is easy to calculate from the knowledge base, and it provides a rough estimate of the total number of consistent and complete models of each device.[2] As one can see, these numbers are very large, ranging from about $10^{12}$ to over $10^{72}$. This means that any sort of brute-force search for adequate models, that searches any significant fraction of this space, is completely hopeless.

The third column shows the total number of models actually examined by the program. This is the sum of the number of models examined during

[2] Of course, calculating the exact number of (a) consistent and complete models; (b) coherent models; or (c) causal models, of each device can only be done by explicitly checking all the models—a completely impractical task.

the first phase, when the program constructs an initial causal model, and the number of models examined during the second phase, when the program simplifies the model. These numbers range from a minimum of 46 to a maximum of 293. As one can see, these numbers are orders of magnitude smaller than the numbers in the second column, making model selection practical. This clearly demonstrates the utility of the restrictions introduced in Chapters 5 and 6, including the use of causal approximations.

The fourth column shows the actual run time on an Explorer II. These times range from a little less than half a minute for the bimetallic strip temperature gauge to a little over eight minutes for the tachometer. Significantly faster runs have been observed on different machines using different Lisp implementations. For example, the tachometer example has been run in a little over a minute and a half on a Sparc Station 2 under Lucid Lisp version 4.1 [Jon L White, personal communication].

**Table 8.3.** Number of model fragments in the most accurate model, the initial causal model, and the minimal causal model constructed by the program.

| Device name | Number of model fragments | | |
|---|---|---|---|
| | Most accurate model | Initial causal model | Minimal causal model |
| Bimetallic strip temperature gauge | 75 | 36 | 27 |
| Bimetallic strip thermostat | 54 | 38 | 14 |
| | 54 | 39 | 31 |
| Flexible link temperature gauge | 94 | 60 | 25 |
| Galvanometer temperature gauge | 154 | 98 | 28 |
| Electric bell | 177 | 7 | 6 |
| | 177 | 108 | 45 |
| Magnetic sizing device | 202 | 122 | 43 |
| Carbon pile regulator | 211 | 122 | 51 |
| Electromagnetic relay thermostat | 211 | 117 | 31 |
| | 211 | 119 | 36 |
| | 211 | 74 | 14 |
| Tachometer | 285 | 170 | 44 |
| Car distributor system | 348 | 178 | 28 |

Table 8.3 shows the number of model fragments in the most accurate model, the initial causal model, and the minimal causal model of each device. The initial causal model and the minimal causal model are, of course, the ones constructed by the program using the methods described above. Multiple entries for a single device correspond to running the program on the different operating regions of that device.

One can see from this table that the number of model fragments in the initial causal model is significantly less than the number in the most accurate model. In fact, on the average, the ratio of the number of model fragments in the initial causal model to the number in the most accurate model is 0.52.

This shows that the heuristic method is effective in finding an initial causal model that is significantly simpler than the most accurate model.

The table also shows that, in most cases, the minimal causal model is significantly simpler than the initial causal model. In fact, on the average, the ratio of the number of model fragments in the minimal causal model to the number in the initial causal model is 0.33. This shows that the heuristic method of finding an initial causal model is not, by itself, sufficient to find a minimal causal model, or even a model that is close to being a minimal causal model: the techniques developed in Chapter 5 are still necessary.

**Table 8.4.** Number of model fragments that were dropped and approximated in simplifying the initial causal model.

| Device name | Simplifications to the initial causal model | |
|---|---|---|
| | Dropped | Approximated |
| Bimetallic strip temperature gauge | 9 | 7 |
| Bimetallic strip thermostat | 24 | 3 |
| | 8 | 9 |
| Flexible link temperature gauge | 35 | 5 |
| Galvanometer temperature gauge | 70 | 6 |
| Electric bell | 1 | 0 |
| | 63 | 10 |
| Magnetic sizing device | 79 | 6 |
| Carbon pile regulator | 71 | 12 |
| Electromagnetic relay thermostat | 86 | 10 |
| | 83 | 5 |
| | 60 | 4 |
| Tachometer | 126 | 9 |
| Car distributor system | 150 | 9 |

Finally, Table 8.4 shows the number of model fragments that were dropped and approximated in simplifying the initial causal model to get the minimal causal model. Note that the number in the third column corresponds only to the model fragments that were approximated in the first phase of simplification, but were not dropped in the second phase of simplification. The number in the second column corresponds to the total number of model fragments that were dropped in the second phase of simplification.

## 8.4 Summary

In this chapter we described our implemented model selection program, and presented some experimental results. The model selection program takes four inputs: (a) the structure of the device; (b) the expected behavior; (c) initial values and exogenous values; and (d) threshold values. Using this input, the

program first finds an initial causal model using a heuristic technique based on the *component interaction heuristic.*

The component interaction heuristic states that if a set of components are related by one or more structural relations that support an interaction, and if one of the component models is compatible with this interaction, then the remaining component models must be augmented to be compatible with this interaction. This allows the components in the set to interact with each other via that interaction. The component interaction heuristic is implemented as a set of heuristic coherence constraints. The initial causal model is required to satisfy all such heuristic coherence constraints.

The algorithm for finding the initial causal model essentially enforces the following constraints: (a) the device model must include all parameters in the expected behavior; (b) structural coherence constraints; (c) heuristic coherence constraints; and (d) behavioral coherence constraints. If the device model resulting from enforcing these constraints is not a causal model, the algorithm augments the device model in a focussed manner, until a causal model is found.

Finally, the initial causal model is simplified using the techniques developed in Section 5.7.

The model selection program has been tested on ten different electromechanical devices. These devices have been selected primarily to demonstrate that similar components in different devices are modeled differently. The results of our experiments show that

1. brute-force search is indeed hopeless—the space of possible device models is just too large;
2. the restrictions introduced in Chapters 5 and 6 allow the model selection program to explore a tiny fraction of the enormous search space, making model selection practical;
3. the heuristic technique for finding an initial causal model does result in causal models that are significantly simpler than the most accurate model; and
4. the heuristic technique is insufficient for finding adequate models—the initial causal model still needs to be significantly simplified to find a minimal causal model.

# 9. Related work

In this chapter we compare our work to other work in automated modeling. Chapter 7 includes a comparison of our work on order of magnitude reasoning to other work in that area, and hence we will not repeat it here.

One of the earliest discussions of the importance of selecting adequate models for efficient problem solving is found in [Amarel, 1968]. Much work in planning with abstractions, starting with [Sacerdoti, 1974], has shown how the use of multiple abstractions can speed up planning. Patil *et al.* show how medical diagnosis can be done with multiple models of a patient [Patil *et al.*, 1981], and Sussman and Steele show how multiple views of an electronic circuit, called *slices*, can lead to tractable reasoning [Sussman and Steele Jr., 1980]. A growing body of literature is also focussed on creating new representations and abstractions from existing representations, e.g., see [Korf, 1980; Subramanian and Genesereth, 1987; Van Baalen and Davis, 1988; Unruh and Rosenbloom, 1989; Christensen, 1990; Knoblock, 1991] and the articles in [Ellman, 1990; Ellman, 1992].

Instead of reviewing this large body of work, we will instead focus on the work related to automatically selecting an adequate model from a space of possible models. This is in contrast to (a) manual selection of adequate models, e.g., [Sussman and Steele Jr., 1980]; (b) the work on creating new representations and abstractions; and (c) cases where all, or most, of the models are used synergistically in problem solving, e.g., much of the work on abstraction planning. Much of this review will focus on the work in the domain of physical systems. This is the topic of Section 9.1. Section 9.2 reviews some recent work in logical approaches to the same problem.

## 9.1 Automated modeling of physical systems

In this section we will review the recent work on automated modeling of physical systems.

### 9.1.1 Compositional modeling

The work most similar to ours is the work on *compositional modeling* [Falkenhainer and Forbus, 1991; Falkenhainer and Forbus, 1988]. In this work, as

in ours, device models are constructed by composing a set of model fragments. Each model fragment is conditioned on a set of *modeling assumptions* which explicate the approximations, perspectives, granularity, and operating assumptions underlying the model fragment. Mutually contradictory assumptions are organized into assumption classes, and a set of domain-independent and domain-dependent constraints are used to govern the use of modeling assumptions. A user *query* focuses the selection of adequate device models by requiring that every adequate model must contain the terms mentioned in the query. Hence, an adequate device model is a simplest model that contains all the terms mentioned in the query, and uses only model fragments that are entailed by a set of mutually consistent assumptions satisfying all the domain-independent and domain-dependent constraints. An adequate model is constructed using a variant of constraint satisfaction called *dynamic constraint satisfaction* [Mittal and Falkenhainer, 1990], and then validated using either qualitative or numerical simulation. If the validation discovers any inconsistencies, the process is repeated with this additional information.

There are many similarities between our work and theirs. First, our use of structural and behavioral preconditions and coherence constraints is very similar to their use of assumptions and constraints on the use of assumptions. Second, the first step of our heuristic algorithm for finding an initial causal model (see Figure 8.1) is similar to their requirement that an adequate model must contain the terms in the query. Third, our loop between the second, third, and fourth steps in the algorithm for finding an initial causal model (see Figure 8.1) is similar to their loop between finding a model and validating it with behavior generation.

However, there are a number of differences. The most important difference is in the definition of model adequacy: they have no counterpart of the expected behavior. Our focus on the task of causal explanation has allowed us to use the expected behavior as a central constraint on model adequacy, thereby decreasing the importance of structural and behavioral coherence constraints. Furthermore, because of the importance of causal explanations to other tasks (see Section 3.2.1), the expected behavior can also provide important constraints on model adequacy for other tasks. On the other hand, in compositional modeling, the constraints on the use of assumptions play a central role in defining model adequacy, and any task focus has to be embedded in these constraints. Embedding such a task focus is, in general, not easy. For example, it is not clear how the expected behavior of a device can be expressed as a set of declarative constraints.

A second difference is that we exploit the restrictions introduced in Chapter 5, especially the use of causal approximations, to develop a polynomial time algorithm for finding adequate models. Note that the abovementioned decrease in the importance of coherence constraints means that the restriction on their expressive power (see Section 5.6) has not proved to be serious. On the other hand, the constraints on the use of assumptions play a central role

in compositional modeling, so no such restriction in expressive power is possible. Hence, their model selection algorithm is based on dynamic constraint satisfaction, which can, in the worst case, take exponential time.

A third difference is that, while their system uses either qualitative simulation or numerical simulation for behavior generation, ours uses order of magnitude reasoning. At the beginning of Chapter 7 we discussed the advantages of order of magnitude reasoning over purely qualitative or purely numerical methods. The primary disadvantage of using our order of magnitude reasoning technique, compared to qualitative or numerical simulation, is that it is currently restricted to generating the behavior at a fixed point in time.

A fourth difference lies in the handling of multiple operating regions of a device. While they generate a single model for all operating regions of a device, we can generate different models for each operating region. However, both techniques have their drawbacks. The drawback with their technique is that the single model may not be the most appropriate model in all operating regions. The drawback with our technique is that we need a new set of inputs for each new operating region.

### 9.1.2 Graphs of models

The work on *graphs of models* [Addanki *et al.*, 1991] discusses a technique for selecting models of acceptable accuracy. A graph of models is a graph in which the nodes are models and the edges are assumptions that have to be changed in moving from one model to another. A model in this graph has acceptable accuracy if its predictions are free of *conflicts*. Conflicts are detected either *empirically* or *interally*. Empirical conflicts are detected by experimentally verifying a model's predictions, while internal conflicts are detected by checking the model's predictions against a set of *consistency* rules that capture the model's assumptions. When a conflict is detected, a set of domain-dependent *parameter change* rules help to select a more accurate model, and the above process is repeated. Analysis begins with the simplest model in the graph of models, and terminates when an accurate enough model has been found.

An important difference between their work and ours is the representation of the space of models. They use an explicit representation of this space as a graph of models, while we have an implicit representation as a set of model fragments that can be combined in different ways to produce an exponentially large number of device models. Our approach leads to greater flexibility in tailoring models to specific situations. To get comparable flexibility in the graphs of models approach requires an explicit representation of an exponentially large space of device models, which is quite impractical. An advantage of their approach is that each model in the graph can have a specialized problem solver, while we must have a general purpose problem solver that is applicable to all models.

The consistency rules used to verify a model's predictions are similar to our behavioral preconditions and coherence constraints. However, we do not validate a model's predictions empirically, and we have not explicitly addressed the problem of switching to a more accurate model in light of a conflict. Our techniques are best viewed as providing an intelligent method for selecting an initial model. Since they always start the analysis with the simplest model, making no effort to identify a better starting model, our techniques are complementary to theirs: select an initial model using our technique, and do model switching using theirs (but see the next section for an alternative model switching technique).

### 9.1.3 Fitting approximations

In [Weld, 1990], Weld introduces an interesting class of approximations called *fitting approximations*. Informally, a model $M_2$ is a fitting approximation of a model $M_1$ if $M_1$ contains an exogenous parameter, called a fitting parameter, such that the predictions using $M_1$ approach the predictions using $M_2$, as the fitting parameter approaches a limit. Weld shows that when all the approximations are fitting approximations, the domain-dependent parameter change rules discussed above can be replaced by a domain-independent technique for model switching.

Fitting approximations and causal approximations are fundamentally incomparable because the former talks about behavior differences, while the latter talks about causal dependencies. However, in practice, it appears that fitting approximations are also causal approximations. For example, all the fitting approximations given in [Weld, 1991] are also causal approximations. This means that his domain-independent technique for model switching can be easily incorporated into our system.

### 9.1.4 Critical abstractions

In [Williams, 1991a], Williams introduces the notion of a *critical abstraction*, which is a parsimonious description of a device relative to a set of questions. Given a device model, he constructs a critical abstraction in three steps: (a) eliminating superfluous interactions; (b) aggregating interactions that are local to a single mechanism using symbolic algebra; and (c) further abstracting the aggregated interactions.

His motivations for creating critical abstractions are very similar to our motivations for finding minimal causal models—we are both striving to find parsimonious descriptions of how a device works. Furthermore, his abstraction process is similar to our model simplification procedure. In fact, the first step of his abstraction process, which eliminates superfluous interactions, is similar to the last step of our simplification procedure, which drops all irrelevant model fragments. The primary difference between our approaches is one

of emphasis: we have focussed on the problem of selecting approximations from a prespecified space of possible approximations, while he has focussed on finding techniques for automatically abstracting a base model.

### 9.1.5 Model-based diagnosis with multiple models

One of the original inspirations for the work described in this thesis was Davis's work on model-based diagnosis [Davis, 1984]. In that work, Davis presents a diagnostic method based on tracing paths of causal interactions. He argues that the power of the approach stems not from the specific diagnostic method, but from the model which specifies the allowed paths of causal interaction. He shows that efficient diagnosis, while retaining completeness, can be obtained by initially considering models with only a few paths of interactions, and adding in additional paths when the model fails to account for the symptoms. He also introduces the notion of *adjacency*: components are adjacent to each other if they can interact with each other by some means.

While we have not focussed on the task of diagnosis, one can see that our simplicity ordering on models lends itself to the above diagnosis technique: diagnosis starts with the minimal causal model, with successively more complex models being used if a model is unable to account for the symptoms. The restrictions in Chapter 5 ensure that using more complex models will add new paths of causal interaction. In addition, the component interaction heuristic introduced in Chapter 8 is closely related to the notion of adjacency: adjacent components must have compatible models.

### 9.1.6 Reasoning about model accuracy

Accuracy is a very important characteristic of an adequate device model: a model must be sufficiently accurate to be useful. In this thesis we have not developed any sophisticated techniques for reasoning about model accuracy. In particular, a model is deemed to be accurate enough if it satisfies all the behavioral preconditions and coherence constraints, with different levels of accuracy corresponding to different settings of the thresholds. However, our system does not reason about the settings of the thresholds; the threshold values are part of the input. In this section we give a brief overview of some ongoing work by various authors on the topic of reasoning about model accuracy.

In [Nayak, 1991], we present a domain-independent method for validating approximate *equilibrium* models against more accurate models. The method makes predictions based on the approximate model, and estimates the error in these predictions, with respect to the more accurate model. We also derive conditions under which the estimated error is guaranteed to be an upper bound on the actual error.

Shirley and Falkenhainer develop a framework for reasoning about model accuracy, and show how approximate models involving differential equations

can be validated against a base model, using known accuracy requirements on certain parameters [Shirley and Falkenhainer, 1990]. Error estimation is done by computing a linear approximation of an error function, and numerically integrating it over the interval of interest. The validity of this method is based on the assumption that the linear terms in the error function dominate the higher order terms.

Falkenhainer extends the above techniques in two ways. In [Falkenhainer, 1992b], he shows how accuracy measures obtained from earlier problem solving episodes can be used to predict accuracy bounds for models in new settings. In [Falkenhainer, 1992a], he shows how idealizations can be constructed using two common idealization assumptions. He also develops an error estimation technique, based on sampling the error's behavior and fitting a polynomial to it, and uses it to define the applicability region of a model.

An alternate method for constructing idealizations and their applicability regions is presented in [Raiman and Williams, 1992]. This method proceeds by first finding ordinal relations between terms using MINIMA [Williams, 1991b], and then exaggerating these ordinal relations, e.g., replacing ">" by "≫." Using the resulting order of magnitude relations, the equations are simplified, and the applicability region of each simplified equation is determined, by ESTIMATES [Raiman, 1991].

Finally, Weld and Addanki introduce the *help/hinder* heuristic—a query-directed technique for the automatic generation of an approximate device model [Weld and Addanki, 1991]. The help/hinder heuristic generates an approximate model by systematically introducing overestimates and underestimates in component models so that the predictions of the approximate model are guaranteed to be sound.

### 9.1.7 Microscopic ontologies

While much of the research in qualitative reasoning about physical systems has focused on *macroscopic* theories of the domain, some researchers have proposed the use of both macroscopic and *microscopic* domain theories: Hayes gives the macroscopic *contained stuff* ontology and the microscopic *piece of stuff* ontology for reasoning about liquids [Hayes, 1985]; Collins and Forbus develop a specialization of the above "piece of stuff" ontology, called the *molecular collection* ontology, and present techniques for generating and reasoning with fluids as "pieces of stuff" [Collins and Forbus, 1987]; Rajamoney and Koo present a qualitative representation for microscopic theories and describe a method for obtaining the macroscopic behavior from such theories [Rajamoney and Koo, 1990].

However, most of the work in this area has not focused on the problem of selecting an appropriate ontology. A notable exception is [Liu and Farley, 1990], in which they present a query-driven method for selecting and shifting between macroscopic and microscopic domain theories. The selection and

shift of ontologies is driven by a set of *ontological choice rules*. However, the generality and scope of these rules is not clear.

While we have not actually developed any microscopic domain theories, the discussion in Section 5.2 applies in a straightforward manner: if the macroscopic and microscopic domain theories are mutually consistent, then the techniques developed in this thesis provide a general method for selecting the appropriate ontology. This seems likely because the macroscopic and microscopic are usually used for different purposes, e.g., in [Collins and Forbus, 1987], the macroscopic "contained stuff" ontology is used to establish global properties like temperature and pressure gradients, while the microscopic "molecular collection" ontology is used to determine how a molecular collection moves through the system. Furthermore, the "molecular collection" ontology is *parasitic* (i.e., dependent) upon the "contained stuff" ontology, and hence the two ontologies are mutually consistent.

## 9.2 Logical approaches

Logical approaches to dealing with multiple domain theories have been proposed by Hobbs and by McCarthy. Hobbs outlines a framework for a theory of granularity, which is a means of constructing simpler theories out of more complex ones using the notion of *indistinguishability* with respect to a set of relevant predicates [Hobbs, 1985]. He proposes the use of a set of *articulation* axioms to link the different granularities, and to allow shifts of perspective during problem-solving. McCarthy introduces the notion of *context*, which captures the implicit assumptions underlying any axiomatization [McCarthy, 1987]. He proposes that all axioms make assertions about some context, and a set of nonmonotonic rules allow inheritance to more general and more specific contexts.

However, both the above proposals lack detail. In his thesis [Guha, 1991], Guha works out some of the details of McCarthy's proposal and demonstrates the use of contexts in CYC.[1] He develops a syntax, semantics, and proof theory of a language for expressing the contextual dependence of axioms. He introduces *lifting* axioms, which allow a formula in one context to be converted into an equivalent formula in another context. He also introduces a set of default lifting axioms, which he says takes care of a majority of the lifting.

Guha's work is certainly more ambitious in scope than ours, since it attempts to deal with general first-order theories, rather than equation models of physical systems. However, when restricted to modeling physical systems, his approach seems very similar to Falkenhainer and Forbus's work on compositional modeling [Falkenhainer and Forbus, 1991]. In particular, modeling

[1] CYC is a large, multi-domain, common sense knowledge base, described in [Lenat and Guha, 1990].

is primarily driven by the terms mentioned in the query and the set of lifting axioms. Furthermore, it is not at all clear that his default lifting axioms will be of much help in selecting appropriate approximations. This is in contrast with our work, where the modeling is primarily driven by the expected behavior. Hence, the difference between our work and compositional modeling can be reiterated here (specifically the first two differences).

# 10. Conclusions

In this thesis we investigated the problem of automatically selecting adequate models for physical systems. We will now present a summary of the techniques developed in this thesis, reiterating the main contributions. We will then suggest directions for future work.

## 10.1 Summary and contributions

We formulated the problem of selecting adequate models as a search problem, requiring answers to the following three questions:

- What is a model, and what is the space of possible models?
- What is an adequate model?
- How do we search the space of possible models for adequate models?

We defined a model as a set of model fragments, where a model fragment is a set of independent algebraic, qualitative, and/or differential equations that partially describes some physical phenomena. The space of possible models was defined implicitly by the set of applicable model fragments: different subsets of this set of applicable model fragments correspond to different models. The set of applicable model fragments was defined by (a) the structure of the physical system, which specifies the system's components; and (b) a component library, which specifies the types of model fragments that can be used to model each type of component.

We gave a clear definition of model adequacy, which was tuned to the task of generating parsimonious causal explanations. An adequate model was defined as a consistent and complete model that could explain the phenomenon of interest. In addition, an adequate model was required to satisfy any domain-independent and domain-dependent constraints on the structure and the behavior of the physical system. Finally, an adequate model was required to be be as simple as possible, with model simplicity being based on the intuition that modeling fewer phenomena more approximately leads to simpler models.

We then developed a formal statement of the problem of finding adequate models, and showed that, in general, the problem is intractable (NP-hard).

We also identified three different sources of intractability: (a) deciding *what* phenomena to model, i.e., deciding which assumption classes to select; (b) deciding *how* to model selected phenomena, i.e., deciding which model fragment to use from each selected assumption class; and (c) having to satisfy all the domain-independent and domain-dependent constraints. We also showed that some related problems are also intractable, e.g., the problem of finding coherent models is intractable.

The intractability of the problem of finding adequate models means that, in general, we can't do much better than search the whole space of possible models. Unfortunately, even for simple devices, the space of possible models is extremely large, making any sort of brute force search completely impractical. To address this problem, we introduced a set of restrictions on the space of possible models, and used these restrictions to develop an efficient algorithm for finding adequate models. The most significant such restriction was that all the approximation relations between model fragments were required to be causal approximations. However, this does not appear to be a serious restriction since most of the commonly used approximations are causal approximations.

Our definition of model adequacy requires us to generate the behavior of a device. To this end, we developed a novel order of magnitude reasoning technique which strikes a balance between purely qualitative and purely quantitative methods. In this technique, the order of magnitude of a parameter is defined on a logarithmic scale, and a set of rules are used to propagate orders of magnitudes through equations. A novel feature of the set of propagation rules is that they effectively handle non-linear simultaneous equations, using linear programming in conjunction with backtracking. We showed that order of magnitude reasoning using this technique is intractable, and developed an approximate reasoning scheme that works well in practice.

Finally, we described an implemented model selection program based on the above techniques. This program includes a heuristic method, based on the component interaction heuristic, for finding an initial causal model. The model selection program was tested on a variety of electromechanical devices. These tests provided empirical evidence for the theoretical claims made in the rest of the thesis.

## 10.2 Future work

The work described in this thesis can be extended in a number of different ways. We now discuss four specific directions for future work.

**Expressivity of the expected behavior.** In this thesis we represented the expected behavior as a causal relation between parameters. While this representation has proved to be useful, it is clearly not very expressive. More expressive languages will allow us to represent a wider range of expected behaviors. For example, in addition to causal relations between parameters, we

may want to include information about the relative directions of change (increasing $T_t$ causes $\theta_p$ to decrease), we may want to include information about specific functional relationships between parameters ($T_t$ and $\theta_p$ are linearly related), or we may want languages for expressing the device's *function* (e.g., see the papers on functional reasoning in [Chandrasekaran, 1991]).

While developing more expressive languages is in itself not difficult, the real challenge is to develop more expressive *tractable* languages. This is important because a central goal of selecting adequate models is to aid effective problem solving. This goal is compromised if the model selection method resulting from using an expressive language for the expected behavior is itself intractable. Hence, an important direction of future research is the development of more expressive languages for expressing the expected behavior that still allow efficient model selection algorithms.

**Reasoning about model accuracy.** In comparing our work to the work on compositional modeling (see Section 9.1.1), we noted that expressing the expected behavior as a set of declarative constraints is difficult. A similar comment applies to our work with respect to reasoning about model accuracy.

Accuracy is a very important characteristic of adequate device models: a model must be sufficiently accurate to be useful. In our work, a model is deemed to be accurate enough if all the behavioral preconditions and coherence constraints are satisfied, with the level of accuracy being determined by the settings of the thresholds. However, we do no reasoning about the settings of these thresholds: they are part of the input. This places a heavy burden on the user: it is not easy to craft a set of behavioral constraints, with appropriately set thresholds, that ensure that adequate models are accurate enough, while also ensuring that models are not required to be too accurate.

A much better approach is to allow the user to specify the desired accuracy of the model much more easily, e.g., by specifying tolerances on certain parameters. We then need to develop techniques for finding models that guarantee that predictions will lie within the specified tolerances. In Section 9.1.6 we discussed some initial work along these lines, but much still needs to be done.

**Multiple operating regions.** Many devices go through multiple operating regions during the course of their normal operations. Different operating regions can have different characteristics, requiring the use of different models. In this thesis, multiple operating regions are handled by requiring a new set of inputs for each region, thereby allowing us to tailor a model for each operating region. However, this requires that the user be aware of each of the device's operating regions. While this may be reasonable in the context of design, e.g., see [Iwasaki and Chandrasekaran, 1992], it seems undesirable in most situations.

The above shortcoming can be addressed as follows. First, behavior generation must include some form of simulation, so that the different operating

regions can be discovered automatically. While both qualitative and numerical simulation can be used, an interesting alternative is to develop a generalization of our order of magnitude reasoning technique that allows simulation.

Second, we need to develop techniques for inferring the expected behavior of each operating region, given the overall expected behavior of the device. For example, the expected behavior of the ignition system in an automobile can be expressed as follows: "turning the ignition key causes the engine to start." The ignition system goes through a series of operating regions to achieve this expected behavior (see [Macaulay, 1988] for details). Rather than specifying the expected behavior of each operating region, it is desirable to have them automatically inferred. We believe that techniques for doing such inference will be tightly integrated with the simulation technique used to generate the operating regions.

**Other tasks.** The model selection techniques developed in this thesis have been focussed on the task of generating parsimonious causal explanations. This suggests a natural direction for future work—developing model selection techniques for other tasks. A particularly promising task appears to be diagnosis, where there is an emerging understanding of what it means for a model to be adequate for diagnosis [Davis, 1984; Hamscher, 1991]. Furthermore, as the discussion in Section 9.1.5 suggests, we believe that the techniques developed in this thesis will prove valuable in developing methods for selecting adequate models for diagnosis.

# A. Examples of causal approximations

In this appendix we present a list of commonly used approximations that can be expressed as causal approximations. Most of these approximations have been borrowed from the fitting approximations listed in [Weld, 1991], though most of the actual equations have been adapted from [Halliday and Resnick, 1978].

Each of the items in this list correspond to a single assumption class. We provide a brief description of the various ways of modeling each phenomena. The equations of these different model fragments are then presented in a tabular form, with a horizontal line separating the different model fragments. Model fragments lower in the table are approximations of model fragments higher in the table, while model fragments at the same level are not approximations of each other. It is easy to verify that all the approximations listed here are causal approximations.

1. **Translational inertia**
   Newton's second law of motion predicts that the acceleration, $a$, of a body of mass, $m$, is proportional to the net force, $F$, acting on the body. It is common to approximate this law by assuming that the mass, and hence the net force, is zero.

| | |
|---|---|
| Newton's second law | $\boldsymbol{F} = m\boldsymbol{a}$ |
| No translational inertia | $\boldsymbol{F} = 0$ |

2. **Rotational inertia**
   This is similar to translational inertia. Newton's second law of motion predicts that the angular acceleration, $\alpha$, of a body of moment of inertia, $I$, is proportional to the net torque, $\tau$, acting on the body. It is common to approximate this law by assuming that the moment of inertia, and hence the net torque, is zero.

| | |
|---|---|
| Newton's second law | $\boldsymbol{\tau} = I\boldsymbol{\alpha}$ |
| No rotational inertia | $\boldsymbol{\tau} = 0$ |

3. **Relativistic mass**
   Einstein's special theory of relativity predicts that the mass, $m$, of an object increases as its velocity, $v$, increases. The mass at zero velocity is called the rest mass, $m_0$. However, this effect is noticeable only at

velocities approaching the speed of light, $c$. At more ordinary velocities, it is common to assume that the mass is constant.

| | |
|---|---|
| Special theory of relativity | $m = \dfrac{m_0}{\sqrt{1-(v/c)^2}}$ |
| Non-relativistic mass | $exogenous(m)$ |

4. **Relativistic motion**
Let $S$ and $S'$ be observers such that $S'$ is moving at velocity $v$ with respect to $S$. Let $S$ and $S'$ observe the same event. Let $S$ record the time and position of the event as $t$ and $x$, and let $S'$ record the time and position of the event as $t'$ and $x'$. The relationship between $x$, $x'$, $t$, and $t'$ is given by the Lorentz transformation. However, at velocities much smaller than the speed of light, $c$, it is common to use the simpler Galilean transformations.

| | |
|---|---|
| Lorentz transformation | $x' = \dfrac{x - vt}{\sqrt{1-(v/c)^2}}$<br>$t' = \dfrac{t - (v/c^2)x}{\sqrt{1-(v/c)^2}}$ |
| Galilean transformation | $x' = x - vt$<br>$t' = t$ |

5. **Deformable bodies**
When elastic bodies are acted upon by a force, $F$, they deform by an amount, $x$. The deformation is proportional to the force ($k$ is the constant of proportionality), and the relationship between the two is given by Hooke's law. However, it is common to assume that bodies are rigid, so that there is no deformation caused by an applied force.

| | |
|---|---|
| Hooke's law | $F = -kx$ |
| Rigid bodies | $x = 0$ |

6. **Friction**
When two bodies move against each other a frictional force, $f$, impedes the motion. The frictional force is proportional to the force, $N$, acting normal to the direction of motion, and the constant proportionality is called the coefficient of friction, $\mu$. However, when motion involves smooth surface, it is common to disregard the frictional force.

| | |
|---|---|
| Motion with friction | $f = \mu N$ |
| Frictionless motion | $f = 0$ |

7. **Gravitational fields**
Newton's law of gravitation predicts that the acceleration due to gravity, $g$, at a distance $r$ from an object of mass $M$ is proportional to the mass and is inversely proportional to the square of the distance (the constant of proportionality is the Gravitational constant, $G$). When the variation

in $r$ is small compared the magnitude of $r$, it is common to assume that the acceleration due to gravity is essentially constant. This can be further approximated, when $r$ becomes sufficiently large, by assuming that the acceleration due to gravity is essentially zero.

| Newton's law of gravity | $g = GM/r^2$ |
|---|---|
| Constant gravity | $exogenous(g)$ |
| Zero gravity | $g = 0$ |

8. **Collisions**
Collisions between objects are typically inelastic. If an object approaches a stationary wall at velocity $v_i$, then the velocity after the collision $v_f$ is attenuated by the coefficient of restitution, $\alpha$. This is often approximated by assuming that the collision is elastic, so that the initial and final velocities are equal in magnitude.

| Inelastic collision | $v_f = -\alpha v_i$ |
|---|---|
| Elastic collision | $v_f = -v_i$ |

9. **Gas laws**
The ideal gas law provides a relationship between the pressure, $P$, the volume, $V$, and the temperature, $T$, of a mole of gas. A more accurate gas law is the Van der Waals equation of state, that accounts for the non-zero size of gas molecules, and that gas molecules repel each other at short distances. In these equations, $R$ is the universal gas constant, and $a$ and $b$ are experimental constants.

| Van der Waals gas | $(P + \frac{a}{V^2})(V - b) = RT$ |
|---|---|
| Ideal gas law | $PV = RT$ |

10. **Thermal conduction**
The rate of heat flow, $f$, across a thermal conductor is proportional to the difference in temperature at the two ends of the conductor ($T_1$ and $T_2$ are the two temperatures). The constant of proportionality is the thermal conductance, $\gamma$. There are two different ways of approximating this model. First, we can assume that the conductor is an ideal thermal insulator, so that there is no heat flow. Second, we can assume that the conductor is an ideal thermal conductor, so that there is never a difference between the two temperatures.

| Thermal conduction | $f = \gamma(T_2 - T_1)$ | | |
|---|---|---|---|
| Ideal thermal insulator | $f = 0$ | Ideal thermal conductor | $T_1 = T_2$ |

11. **Thermal conductance**
The thermal conductance, $\gamma$, of a thermal conductor is dependent on the length, $l$, the cross-sectional area, $A$, and the thermal conductivity, $k$, of the conductor. When the dependence of $\gamma$ on these factors is unnecessary, one can merely assume that it is constant.

| Dependent thermal conductance | $\gamma = kA/l$ |
|---|---|
| Constant thermal conductance | $exogenous(\gamma)$ |

12. **Electrical conduction**
The current flow, $i$, across an electrical conductor is proportional to the voltage drop, $V$, across the conductor. The constant of proportionality is the resistance, $R$, and the relationship is Ohm's law. There are two different ways of approximating this model. First, we can assume that the conductor is an ideal electrical insulator, so that there is no current flow. Second, we can assume that the conductor is an ideal electrical conductor, so that the voltage drop is always zero.

| Ohm's law | $\boldsymbol{V = iR}$ | | |
|---|---|---|---|
| Ideal electrical insulator | $\boldsymbol{i = 0}$ | Ideal electrical conductor | $\boldsymbol{V = 0}$ |

13. **Electrical resistance**
The electrical resistance, $R$, of an electrical conductor is dependent on the length, $l$, the cross-sectional area, $A$, and the resistivity, $\rho$, of the conductor. When the dependence of $R$ on these factors is unnecessary, one can merely assume that it is constant.

| Dependent resistance | $\boldsymbol{R = \rho l/A}$ |
|---|---|
| Constant resistance | $exogenous(\boldsymbol{R})$ |

14. **Resistivity**
The resistivity, $\rho$, of an electrical conductor is a function of the temperature, $T$, of the conductor. $\rho_0$ is the resistivity at temperature $T_0$, and $\alpha$ is the coefficient of resistivity. However, this dependence is often neglected, and the resistivity is assumed to be constant.

| Temperature dependent resistivity | $\boldsymbol{\rho = \rho_0(1 + \alpha(T - T_0))}$ |
|---|---|
| Constant resistivity | $exogenous(\boldsymbol{\rho})$ |

15. **Heat engine**
A heat engine can be thought of as a cyclic process that extracts heat from a high temperature source, converts part of this heat into work, and discharges the rest of the heat to a low temperature sink. The efficiency, $e$, of a heat engine is the fraction of extracted heat that is converted into work. Carnot showed that the efficiency of an ideal heat engine is a function of the source temperature, $T_1$, and sink temperature, $T_2$, and that the efficiency of a real heat engine is less than or equal to the ideal efficiency by an efficiency factor, $\gamma$.

| Real heat engine | $\boldsymbol{e = \gamma(1 - T_2/T_1)}$ |
|---|---|
| Ideal heat engine | $\boldsymbol{e = (1 - T_2/T_1)}$ |

16. **Laminar flow in horizontal pipes**
The rate, $V$, of laminar flow of a fluid in a pipe is a proportional to the difference between the pressure at one end of the pipe, $P_1$, and the pressure at the other end of the pipe, $P_2$. The pressure drop in the pipe is due to the viscous resistance, $R$, of the fluid. This model is often approximated to disregard the viscous resistance, so that there is no pressure drop across the pipe.

| | |
|---|---|
| Viscous flow | $\boldsymbol{P_1 - P_2 = RV}$ |
| Inviscid flow | $\boldsymbol{P_1 = P_2}$ |

17. **Thermal expansion**
When objects are heated, they expand. The amount of expansion, $\delta$, is a function of the object's temperature, $T$, and the coefficient of thermal expansion, $\alpha$. $\delta$ is assumed to be zero when the size of the object is $l_0$ at temperature $T_0$. This expansion is often quite small, and can be disregarded for many purposes.

| | |
|---|---|
| Thermal expansion | $\delta = \alpha l_0(T - T_0)$ |
| No thermal expansion | $\delta = 0$ |

18. **Exogenizing and equilibrating differential equations**
Chapter 6 shows that exogenizing and equilibrating differential equations can be considered to be causal approximations. For example, the rate of change of the temperature, $T$, of an object is a function of the net heat, $F$, flowing into the object and the object's heat capacity, $C$. This differential equation can be exogenized by assuming that the the temperature is constant. It can be equilibrated by assuming that the temperature quickly adjusts itself to ensure that the net heat flow is zero.

| | | | |
|---|---|---|---|
| Dynamic thermal model | | $\boldsymbol{\frac{dT}{dt} = CF}$ | |
| Constant temp. | *exogenous*$(\boldsymbol{T})$ | Equilibrium temp. | $\boldsymbol{F = 0}$ |

It is interesting to ask whether there are commonly used approximations that are not causal approximations. We have identified such a class of such approximations that do not exactly fit our definition of causal approximations, but are close. The problem is that the more approximate model fragments contain parameters not found in the more accurate model fragments. Here are some examples:

1. **Viscosity of gases**
The viscosity, $\mu$, of a gas is a function of its temperature, $T$, and mass, $m$, (equivalently, its molecular weight, $M$). There are at least two models of this dependence. An approximate model assumes that the gas molecules are hard balls of diameter $d$. A more accurate model models the gas molecule as a force field, and uses the Lennard Jones potential energy function. These models have been taken from [Welty *et al.*, 1984].

| | |
|---|---|
| Force field model | $\mu = 2.6693 \times 10^{-6} \dfrac{\sqrt{MT}}{\sigma^2 \omega_\mu}$ |
| Rigid sphere model | $\mu = \dfrac{2}{3\pi^{3/2}} \dfrac{\sqrt{m\kappa T}}{d^2}$ |

Note that the force field model does not contain parameters like $d$ that are found in the rigid sphere model.

2. **Linearizations**
   Complicated equations are often approximated by linearizing them. Such linearizations introduce additional parameters, such as the slope of the line. These parameters are clearly not part of the original equation.

While the above two approximations do not fit our definition of a causal approximation, one can see that they almost do. In particular, in the first case, if we are only interested in the dependence of $\mu$ on $T$, then the approximation behaves like a causal approximation. Similarly, in the second case, if we are not interested in the dependence of any parameter on the additional parameters like the slope, then the linearization behaves like a causal approximation.

Hence, we can generalize our definition of causal approximations by allowing more approximate model fragments to have parameters not in the more accurate model fragment, but add the restriction that such parameters be local to the more approximate model fragment. Furthermore, we must also require that we are not interested in the dependence of any parameter on such parameters. Both these restrictions seem reasonable in the above cases.

# B. Example devices

In this appendix we describe the electromechanical devices that our model selection program was tested on.

## B.1 Bimetallic strip temperature gauge

The bimetallic strip temperature gauge is shown in Figure B.1. It is based on a similar temperature gauge described in [Macaulay, 1988, page 290]. A thermistor is a semi-conductor device; a small increase in its temperature causes a large decrease in its resistance. A bimetallic strip consists of two strips made of different metals that are joined together. Temperature changes cause the two strips to expand by different amounts, causing the bimetallic strip to bend.

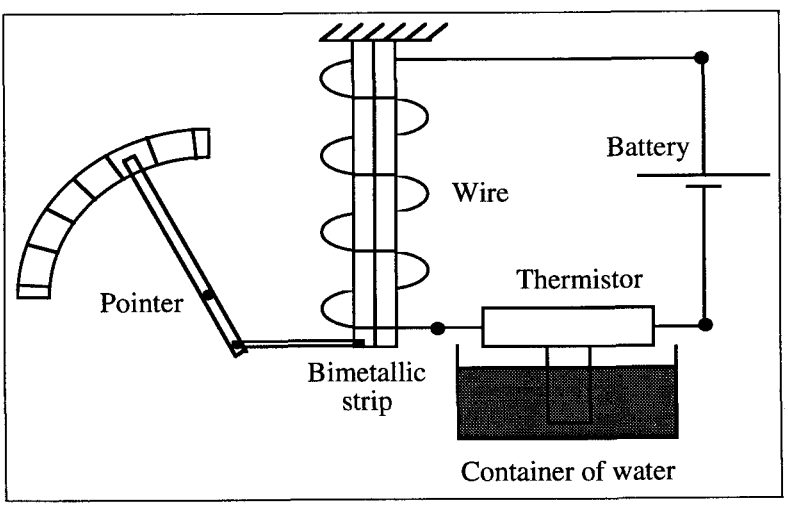



**Fig. B.1.** Bimetallic strip temperature gauge

It works as follows: the thermistor senses the water temperature. The thermistor's temperature determines its resistance, which determines the current

flowing in the circuit. This determines the amount of heat dissipated in the wire, which determines the temperature of the bimetallic strip. The temperature of the bimetallic strip determines its deflection, which determines the position of the pointer along the scale.

## B.2 Bimetallic strip thermostat

The bimetallic strip thermostat is shown in Figure B.2. It is based on thermostats described in [Macaulay, 1988, page 162]. Its operation is very straightforward: the bimetallic strip senses the temperature of its environment. The bimetallic strip's temperature determines its deflection. If the temperature is too high, the bimetallic strip bends enough to lose connection with the contact, which breaks the electrical circuit and turns the heater off. Otherwise, the bimetallic strip does not bend enough to lose its connection with the contact, so that the electrical circuit is completed, and the heater is turned on.

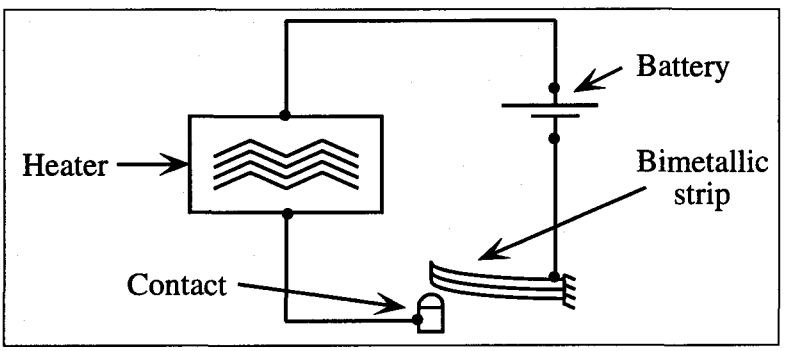



**Fig. B.2.** Bimetallic strip thermostat

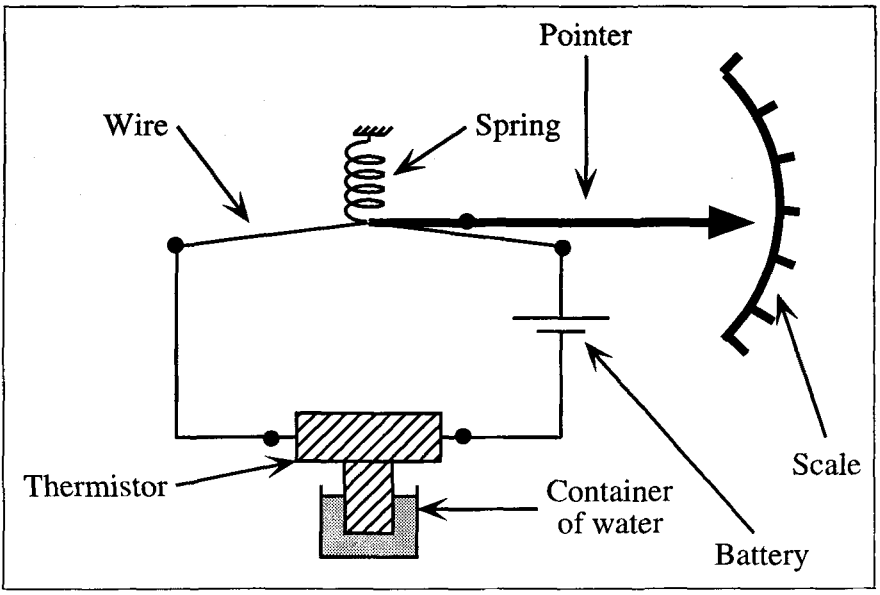



**Fig. B.3.** Flexible link temperature gauge

## B.3 Flexible link temperature gauge

The flexible link temperature gauge is shown in Figure B.3. It is adapted from a flexible link resistance box mechanism described in [Artobolevsky, 1980, page 451]. Its operation is based on the principle that the length of a wire is dependent on its temperature.

It works as follows: the thermistor senses the water temperature. The thermistor's temperature determines its resistance, which determines the current flowing in the circuit. The current in the circuit determines the amount of heat generated in the wire, which determines the wire's temperature. The wire's temperature determines the wire's length. Since the wire is fixed at its two ends, the length of the wire determines the deflection of the spring, and hence the position of the pointer along the scale.

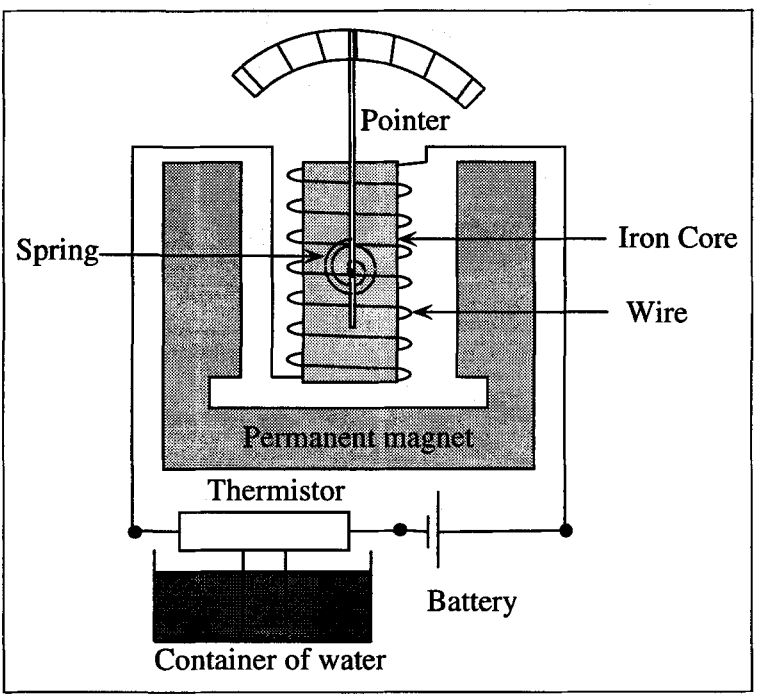



**Fig. B.4.** Galvanometer temperature gauge

## B.4 Galvanometer temperature gauge

The galvanometer temperature gauge is shown in Figure B.4. The galvanometer shown in this figure has been adapted from [Halliday and Resnick, 1978, page 726]. Its operation is based on the interaction between the field of a permanent magnet and the magnetic field generated by an electric current.

It works as follows: the thermistor senses the water temperature. The thermistor's temperature determines its resistance, which determines the current flowing in the circuit. The current flowing in the wire generates a magnetic field, which is magnified by the iron core. This magnetic field interacts with the magnetic field generated by the permanent magnet, producing a torque on the iron core. This causes the iron core, and hence the pointer, to deflect. The spring provides a restoring torque, so that the deflection of the pointer is proportional to the strength of the torque.

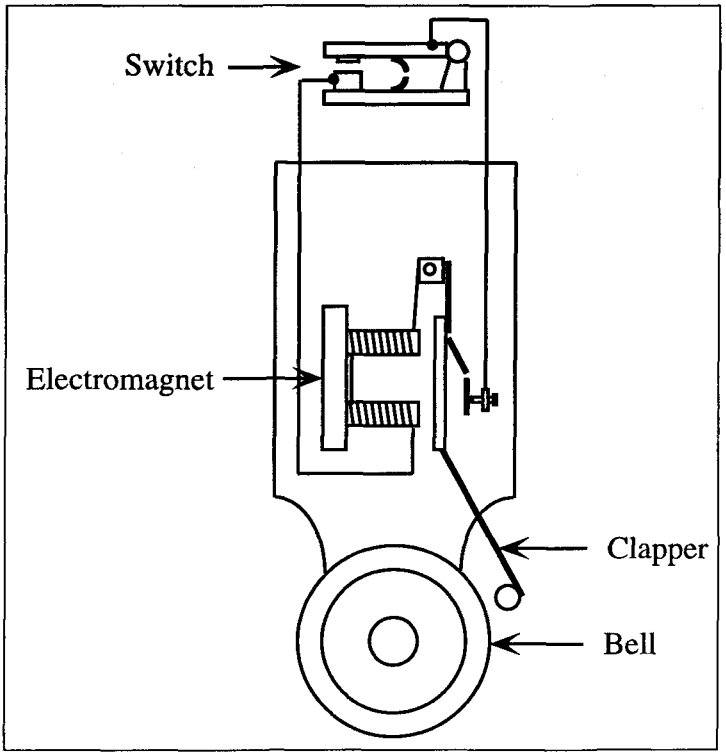



**Fig. B.5.** Electric bell

## B.5 Electric bell

The electric bell is shown in Figure B.5. It has been adapted from [Artobolevsky, 1980, page 129]. It works as follows: when the switch is pressed, the electric circuit is closed. This energizes the winding of the electromagnet, attracting the armature so that the clapper can strike the bell. This also breaks the circuit at the contact, and hence deactivates the electromagnet. The armature returns to its original position due to the flat spring, closing the circuit and repeating the above cycle.

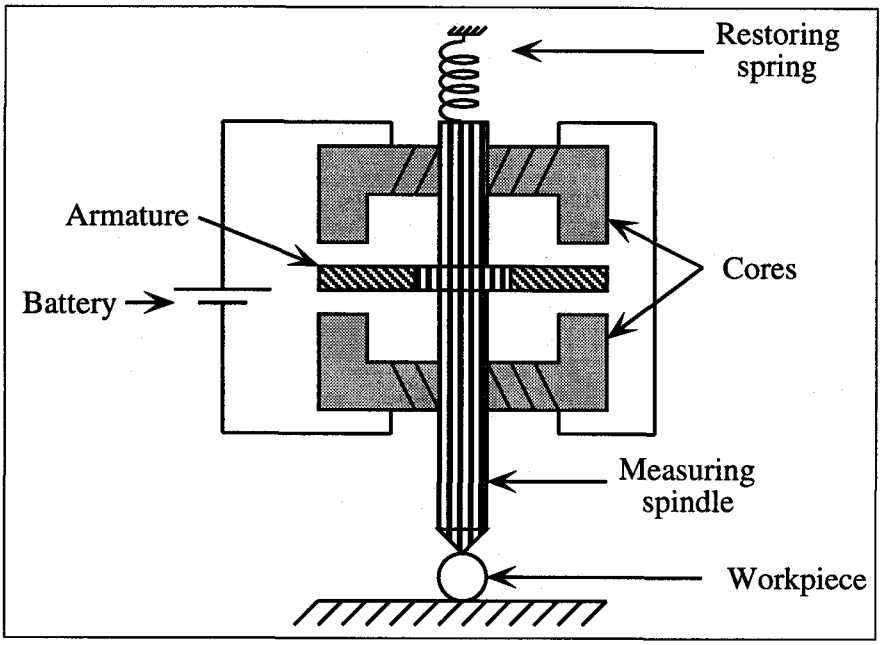



**Fig. B.6.** Magnetic sizing device

## B.6 Magnetic sizing device

The magnetic sizing device is shown in Figure B.6. It has been adapted from [Artobolevsky, 1980, page 57]. It is used to ensure that the size of the workpiece is within desired limits.

It works as follows: the position of the measuring spindle is determined by the size of the workpiece. This determines the position of the armature with respect to the two cores, and hence determines the width of the four air gaps between the ends of the armature and the ends of the cores. The current flowing in the circuit generates a magnetic flux in the cores and the air gaps. The strength of the magnetic flux is determined by the width of the air gaps. Hence, the strength of the magnetic flux can be used as a measure of the size of the workpiece.

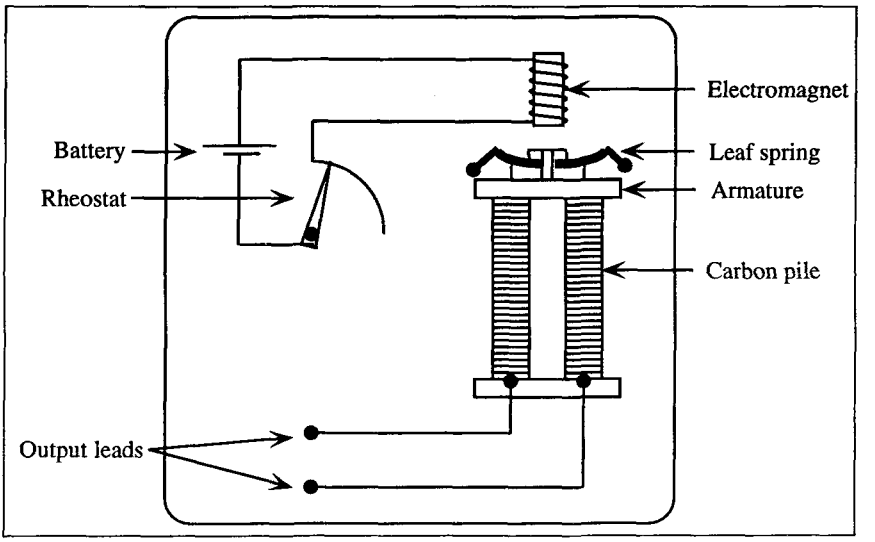



**Fig. B.7.** Carbon pile regulator

## B.7 Carbon pile regulator

The carbon pile regulator is shown in Figure B.7. This device is adapted from [Artobolevsky, 1980, page 108]. Its operation is based on the fact that the resistance of a carbon pile is dependent on the compressive force acting on it.

The carbon pile regulator works as follows: the position of the rheostat determines the resistance of the top circuit. This determines the current flowing in that circuit, which determines the strength of the magnetic field generated by the electromagnet. The electromagnet attracts the armature, with the leaf spring providing a restoring force. The force on the armature decreases the compressive force on the carbon pile, thereby changing its resistance. Hence, the resistance at the output leads is a function of the position of the rheostat.

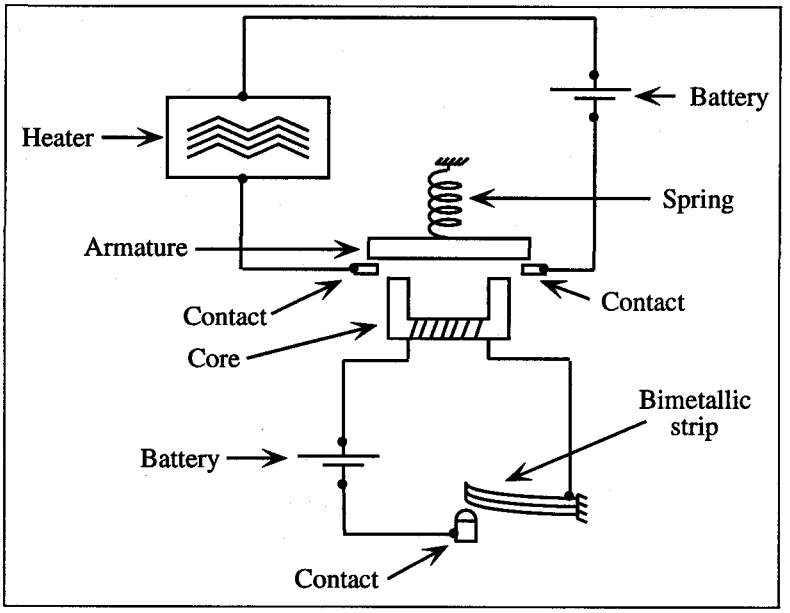



**Fig. B.8.** Electromagnetic relay thermostat

## B.8 Electromagnetic relay thermostat

The electromagnetic relay thermostat is shown in Figure B.8. This device is similar to the bimetallic strip thermostat. The primary difference is that, in this device, the bimetallic strip turns on an electromagnetic relay which turns on the heater, rather than the bimetallic strip turning on the relay directly.

The operation of the electromagnetic relay thermostat is similar to the bimetallic strip thermostat. When the environment becomes too cold, the bimetallic strip deflects less and hence closes the contact. The resulting current flow creates a magnetic field in the winding around the core, which attracts the armature. The armature moves downwards and closes the contacts, turning on the heater. When the room is too hot, the bimetallic strip breaks its connection with the contact, and current stops flowing in the lower circuit. This causes the armature to return to its original position due to the restoring spring, thereby turning the heater off.

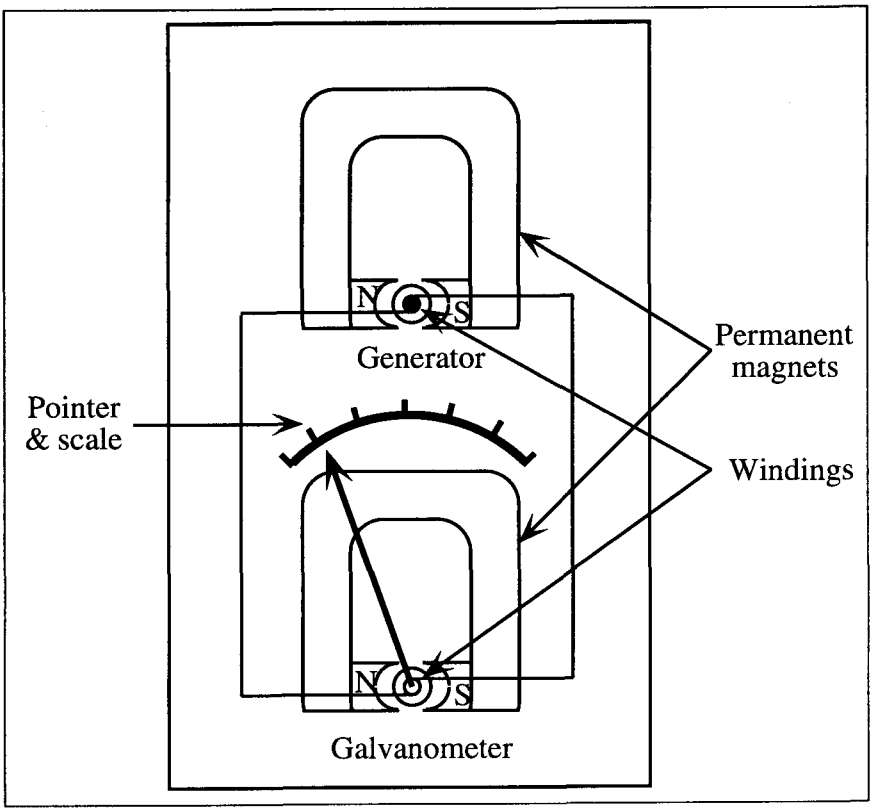



**Fig. B.9.** Tachometer

## B.9 Tachometer

The tachometer is shown in Figure B.9. This device has been adapted from [Artobolevsky, 1980, page 90]. The tachometer measures the angular velocity of the core of the windings at the top of the figure.

The tachometer consists of a generator and a galvanometer. The generator consists of the windings and permanent magnet at the top of the figure, while the galvanometer consists of the windings, permanent magnet, and pointer assembly at the bottom of the figure. The tachometer works as follows: the rotation of the windings of the generator within the magnetic field of the permanent magnet induces an electromotive force. This induced electromotive force drives current in the circuit, generating an electromagnetic field in the windings of the galvanometer. This electromagnetic field interacts with the magnetic field of the permanent magnet, causing the pointer to deflect along the scale (the restoring spring is not shown).

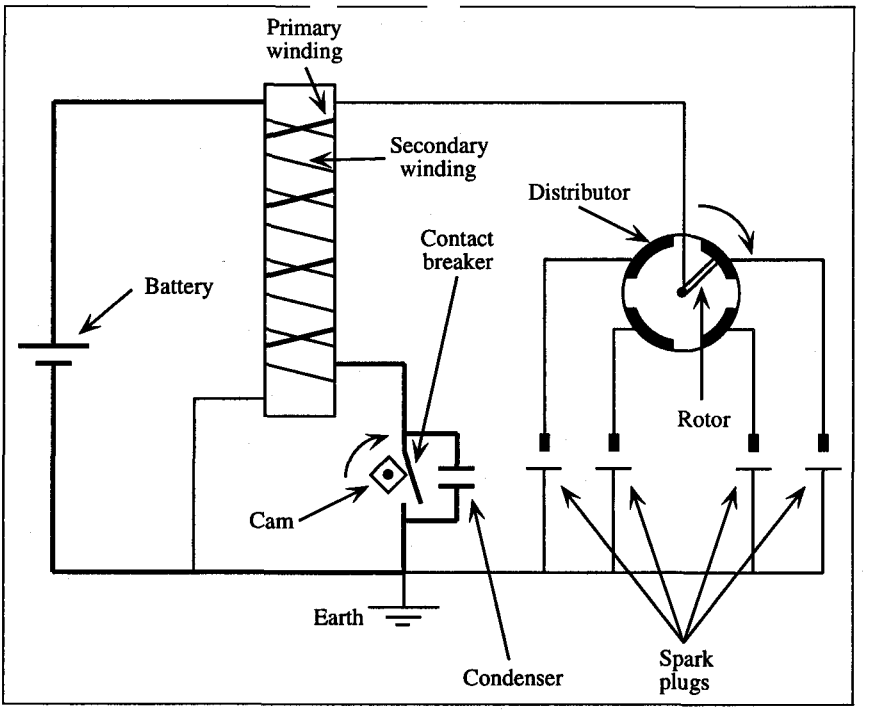



**Fig. B.10.** Car distributor system

## B.10 Car distributor system

A schematic of the car distributor system is shown in Figure B.10. This system has been adapted from [van Amerongen, 1967, pages 482–483]. The function of the distributor system in a car is to ensure that the spark plugs fire at appropriate times.

The distributor system works as follows: as the cam rotates, it opens the contact breaker. This causes the current in the primary windings to drop rapidly (the condenser prevents a spark from jumping across the contact breaker). The rapid change in current in the primary winding causes a large induced electromotive force in the secondary winding. At the same time, the distributor rotor connects the secondary winding to one of the spark plugs (the rightmost spark plug in the figure). The large induced electromotive force causes a spark to jump across the spark plug.

# C. Composable operators

In this appendix we describe the composable operators shown in Table 2.1. All the composable operators share two characteristics: (a) they are all converted into an equivalent set of algebraic equations; and (b) they all use a form of the closed world assumption. In particular, a set $C$ of compositional operators is converted into an equivalent set $E$ of algebraic equations, using the closed world assumption that the only elements in $C$ are the ones that are known to be in $C$.

## C.1 Influences

The $I+$ and $I-$ operators are the same as in [Forbus, 1984]. $I+(q_1, q_2)$ states that $q_2$ is a positive influence on $q_1$, while $I-(q_1, q_2)$ states that $q_2$ is a negative influence on $q_1$. Given a set of $I+$ and $I-$ statements, we can collect together all the positive and negative influences on each parameter. For each parameter we do the following: (a) form the term resulting from the difference of the sum of the all the positive influences and the sum of all the negative influences; and (b) make the derivative of the parameter be equal to this term. For example, from the set $\{I+(\boldsymbol{q_1}, \boldsymbol{q_2}), I-(\boldsymbol{q_1}, q_3), I+(\boldsymbol{q_1}, \boldsymbol{q_4})\}$ we would produce the equation:

$$\boldsymbol{dq_1/dt} = \boldsymbol{q_2} - q_3 + \boldsymbol{q_4}$$

Note how we controlled the parameters that could be determined by the resulting equation. However, the left hand side parameter $(dq_1/dt)$ can always be determined by the equation.

The *sum-term* operator is exactly like the $I+$ operator, except that step (b) above is modified by making the parameter itself equal to the constructed term. For example, from the set $\{\textit{sum-term}(\boldsymbol{q_1}, \boldsymbol{q_2}), \textit{sum-term}(\boldsymbol{q_1}, -\boldsymbol{q_3}), \textit{sum-term}(\boldsymbol{q_1}, q_4)\}$ we would produce the equation:

$$\boldsymbol{q_1} = \boldsymbol{q_2} - \boldsymbol{q_3} + q_4$$

## C.2 Kirchhoff's laws

The *sum-to-zero*, *same-value*, *same-reference*, and *same-circuit* operators are primarily used to implement Kirchhoff's laws, though they can be used for other purposes also. Kirchhoff's laws are necessary when modeling generalized flows in networks. We will discuss these laws in the context of electrical circuits, but the discussion is equally applicable to other types of circuits, including fluid, thermal, and magnetic circuits. We will also restrict our discussion to electrical components with two terminals; the generalization to components with three or more terminals is straightforward.

An electrical network is formed by connecting terminals of electrical components. Each component terminal has two important attributes: (a) the voltage of the terminal; and (b) the current flow into the component at that terminal. Hence, a network with $n$ nodes and $e$ edges has $2e$ currents and $2e$ voltages. Hence, $4e$ independent equations are needed to determine all the currents and voltages. Kirchhoff's laws and network theory tell us the source of these $4e$ independent equations:

1. Each component in the network has a component equation. For example, Ohm's law for a resistor is a component equation. Since each component is an edge in the network, there are $e$ independent component equations.
2. Kirchhoff's current law tells us that the net current flow into each component is zero. Hence, the sum of the two currents flowing into each component is zero. There are $e$ such independent equations.
3. Kirchhoff's current law tells us that the net current flowing into any node in the network is zero. Hence, there are $n$ equations stating Kirchhoff's current law for each of the $n$ nodes in the network. However, network theory tells us that for a connected network with $n$ nodes, only $(n-1)$ of these equations are independent equations (though any $(n-1)$ of these equations will do).
4. Kirchhoff's voltage law tells us that the voltages of connected terminals are equal. Hence, at a node where $k$ terminals are connected, there are $k-1$ independent equations stating the equality of the $k$ voltages. One can verify that there are a total of $2e-n$ such independent equations.
5. The terminal voltages are measured with respect to a reference voltage. Hence, for a connected network with $n$ nodes, any one voltage can be arbitrarily selected as the reference voltage, and this voltage can be set to zero. This provides one equation.

Adding up all the equations, we see that there are $4e$ independent equations that can be used to determine values for the $4e$ voltages and currents.

One can easily associate the $2e$ equations under points 1 and 2 with the corresponding components, and hence those equations are easy to generate.

Given a network, one can write a program to generate the remaining $2e$ independent equations. Indeed, we started by doing precisely that. However, the program turned out to be very difficult to understand, maintain,

and generalize. In particular, we needed the following two generalizations: (a) since our initial implementation dealt only with electrical circuits, we wanted to generalize it to other types of circuits; and (b) we wanted to handle disconnected networks. Disconnected networks need special handling. Each connected component needs its own reference voltage, and each connected component has one dependent Kirchhoff's current law equation.

To facilitate these generalizations, we discarded the above *procedural* method of generating the $2e$ equations, and instead developed a more *declarative* method. This method is based on the use of the *same-value*, *same-reference*, *sum-to-zero*, and *same-circuit* operators.

The *same-value* operator is a binary operator that states that its two arguments are equal. Given a set of *same-value* statements, it is easy to partition the set of parameters into equivalence classes such that two parameters are in the same equivalence class if and only if the set of *same-value* constraints require them to be equal. Then, for an equivalence class with $k$ parameters, one can then generate $(k-1)$ independent equality constraints that ensure that all the parameters in that assumption class are equal.

We use the *same-value* operator to generate the $(2e-n)$ equality constraints discussed in point 4 above. In particular, if $t_1$ and $t_2$ are any two connected electrical terminals, then we assert *same-value(voltage($t_1$), voltage($t_2$))*, i.e., we have the following rule:

$$\textit{connected-to}(t_1, t_2) \Rightarrow \textit{same-value}(\textit{voltage}(t_1), \textit{voltage}(t_2))$$

This allows us to generate the required equality constraints.

The *same-reference* operator is a binary operator that states that its two arguments share a common reference voltage. Given a set of *same-reference* statements, it is easy to partition the set of voltages into equivalence classes such that two voltages are in the same equivalence class if and only if they share the same voltage. It is then easy to pick an arbitrary member from each such equivalence class and set it to zero.

We use the *same-reference* operator to generate the equation discussed under point 5 above. We enforce this by ensuring that voltages of connected terminals have the same reference, and voltages of terminals belonging to the same component have the same reference. This is equivalent to the following rules:

$$\textit{connected-to}(t_1, t_2) \Rightarrow \textit{same-reference}(\textit{voltage}(t_1), \textit{voltage}(t_2))$$
$$\textit{same-component}(t_1, t_2) \Rightarrow \textit{same-reference}(\textit{voltage}(t_1), \textit{voltage}(t_2))$$

This ensures that every connected network has a single reference voltage.

The advantage of the declarative method can be clearly demonstrated with a slight variation of the second rule above. In particular, if $t_1$ and $t_2$ are terminals of an electrical switch that is *off*, then it is incorrect to say that their corresponding voltages necessarily have the same reference. Using our declarative method, it is easy to modify the above rule to exclude this case.

The *sum-to-zero* operator is a binary operator that identifies parameters that will be terms in a sum that will be zero. Like *same-value* and *same-reference*, *sum-to-zero* is an equivalence relation. Hence, given a set of *sum-to-zero* statements, we can partition the set of parameters into a set of equivalence classes. The *sum-to-zero* operator is interpreted as stating that the parameters in each equivalence class sum up to zero.

We use the *sum-to-zero* operator to generate the $n$ equations under point 3 above.[1] We do this by stating that the currents corresponding to connected terminals sum to zero. In particular, we have the following rule:

$$\textit{connected-to}(t_1, t_2) \Rightarrow \textit{sum-to-zero}(\textit{current}(t_1),\ \textit{current}(t_2))$$

This ensures that the currents flowing out of every node sum to zero, as required by Kirchhoff's current law.

Finally, *same-circuit* is a binary relation on currents that states that its two arguments are part of the same circuit. Given a set of *same-circuit* statements, we can partition the set of currents into a set of equivalence classes such that two currents are in the same equivalence class if and only if they are part of the same circuit. Currents are in the same circuit if (a) they belong to connected terminals; or (b) they belong to terminals of the same component. In particular, we have the following rules:

$$\textit{connected-to}(t_1, t_2) \Rightarrow \textit{same-reference}(\textit{current}(t_1),\ \textit{current}(t_2))$$
$$\textit{same-component}(t_1, t_2) \Rightarrow \textit{same-reference}(\textit{current}(t_1),\ \textit{current}(t_2))$$

The identification of currents that are in the same circuit allows us to generate exactly $(n-1)$ independent equations, as required by point 3 above. In particular, the *sum-to-zero* equivalence classes generated above, are further clustered into classes, such that two *sum-to-zero* equivalence classes are in the same class if and only if a parameter in one *sum-to-zero* equivalence class is in the same circuit as a parameter in the other *sum-to-zero* equivalence class. The *sum-to-zero* equivalence classes in the same class correspond to the $n$ equations of a connected network discussed in point 3 above. To get the $(n-1)$ independent equations, we merely discard any one of the *sum-to-zero* equivalence classes from each class, and use the remaining *sum-to-zero* equivalence classes as above.

## C.3 Other methods

It is worth noting that the *same-value* and *sum-to-zero* operators are necessary only because of our representation of the *connected-to* relation, i.e., because *connected-to* directly relates terminals of components. A different representation would make *same-value* and *sum-to-zero* unnecessary. In particular, we can (a) introduce new entities called *nodes*, one for each node in

[1] The fact that only $(n-1)$ of these equations will be independent is handled when we discuss *same-circuit*.

the network; and (b) require that terminals can only be connected to nodes, and not to other terminals. (This representation has been used by others, e.g., [Williams, 1989].)

Using this representation, we can associate a voltage with each node, and a net current flow into each node. We can easily make the voltages of all terminals connected to a node equal to the node voltage, and we can use the *sum-term* operator to state that the net current flow into a node is equal to the sum of the current in the terminals connected to that node. We can further state that the net current flow into a node is zero.

While the above representation removes the need for *same-value* and *sum-to-zero*, it does not remove the need for *same-reference* and *same-circuit*. These latter two operators are still necessary to ensure that exactly one reference voltage is selected for every connected network, and an independent set of Kirchhoff's current law equations are generated.

# D. Matchings in bipartite graphs

In this appendix we discuss an algorithm, based on network flow techniques, for finding maximum matchings in bipartite graphs. The relationship of this algorithm to network flow is discussed in textbooks such as [Cormen *et al.*, 1990; Even, 1979]. We present the algorithm here since some of the proofs in Chapter 6 depend on an understanding of it.

**Definition D.1 (Bipartite graph).** *A bipartite graph $G = (X, Y, E)$, where $X \cup Y$ is the set of nodes and $E$ is the set of edges, is a graph whose nodes can be partitioned into two sets, $X$ and $Y$, such that all edges in $E$ connect a node in $X$ to a node in $Y$.*

**Definition D.2 (Matching).** *A matching in a bipartite graph is a subset of the edges of the graph such that no two edges in the matching share a common node. A maximum matching in a bipartite graph is a matching of maximum cardinality.*

**Definition D.3 (Augmenting path).** *Let $G = (X, Y, E)$ be a bipartite graph, and let $U \subseteq E$ be a matching. An augmenting path in $G$, with respect to $U$, is a sequence of nodes such that:*

1. *no node is repeated in the sequence;*
2. *alternate nodes in the sequence are in $X$, with the first node being in $X$;*
3. *alternate nodes in the sequence are in $Y$, with the last node being in $Y$;*
4. *if $x \in X$ and $y \in Y$ are consecutive nodes in the sequence, with $x$ before $y$, then there is an edge $e \in E$ such that $e$ is incident on both $x$ and $y$, and $e$ is not in $U$, i.e., the augmenting path goes from nodes in $X$ to nodes in $Y$ via edges not in the matching;*
5. *if $y \in Y$ and $x \in X$ are consecutive nodes in the sequence, with $y$ before $x$, then there is an edge $e \in E$ such that $e$ is incident on both $x$ and $y$, and $e$ is in $U$, i.e., the augmenting path goes from nodes in $Y$ to nodes in $X$ via edges in the matching;*

*The edges of an augmenting path are the edges that connect consecutive nodes in the augmenting path.*

For example, Figure D.1a shows a bipartite graph. The bold edges with arrows at each end form a matching. One can easily verify that the sequence

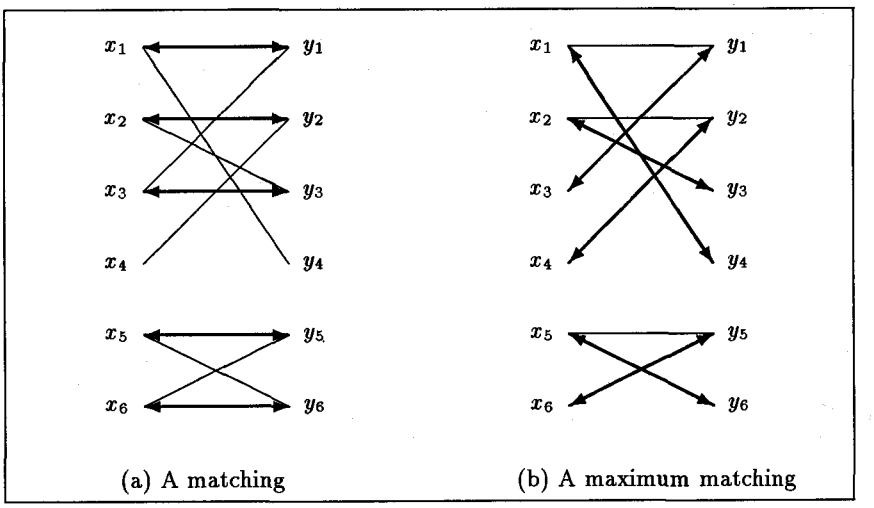



**Fig. D.1.** Matchings in a bipartite graph

$$(x_4, y_2, x_2, y_3, x_3, y_1, x_1, y_4) \tag{D.1}$$

is an augmenting path. Note that any augmenting path has one less edge in the matching than not in the matching. Hence, augmenting paths can be used to increase the cardinality of a matching by replacing the edges of the matching that are in the augmenting path by edges in the augmenting path that are not in the matching. Hence, we have the following:

**Lemma D.0.1 (Increasing a matching).** *Let $G = (X, Y, E)$ be a bipartite graph, and let $U \subseteq E$ be a matching. Let $P$ be an augmenting path in $G$, with respect to $U$, and let $E_P$ be the edges of $P$. Let $U' \subseteq E$ be the union of the set of edges in $U$ but not in $E_P$ and the set of edges in $E_P$ but not in $U$:*

$$U' = (U \setminus E_P) \cup (E_P \setminus U)$$

*Then $U'$ is a matching in $G$, and $|U'| > |U|$.*

For example, the augmenting path in Equation D.1 can be used to increase the matching shown in Figure D.1a, by replacing each bold edge in the path with a light edge, and each light edge in the path with a bold edge. The resulting matching, of a higher cardinality, is shown in Figure D.1b.

The above lemma shows how we can increase the cardinality of a matching $U$ by first identifying an augmenting path with respect to $U$. The next lemma tells us that if no such augmenting path exists, then the matching is of maximum cardinality, i.e., it is a maximum matching.

**Lemma D.0.2.** *Let $G = (X, Y, E)$ be a bipartite graph, and let $U \subseteq E$ be a matching in $G$ such that there is no augmenting path with respect to $U$. Then $U$ is a maximum matching.*

The above lemma is a special case of a similar result of network flow algorithms. The interested reader is referred to any textbook on network flow algorithms, e.g., [Cormen *et al.*, 1990; Even, 1979], for the details.

Lemmas D.0.1 and D.0.2 give us the following algorithm for finding a maximum matching in a bipartite graph $G = (X, Y, E)$:

```
function find-maximum-matching(G)
  Let U be any matching in G
  while there is an augmenting path P, wrt U
     Use P to increase the cardinality of U
  return U
end
```

**Fig. D.2.** Algorithm for finding a maximum matching

The initial matching $U$ can be any matching, including the empty set. The complexity of the above algorithm depends on the algorithm used to find augmenting paths. For practical purposes, we have found that a straightforward depth first search gives adequate results, though faster algorithms are possible, e.g., Dinic's network flow algorithm described in [Even, 1979]. If we use Dinic's algorithm for finding augmenting paths, the complexity of the above algorithm is $O(|V|^{1/2}|E|)$, where $V = X \cup Y$ is the set of nodes and $E$ is the set of edges in $G$ (see [Even, 1979] for the details).

# References


[Addanki *et al.*, 1991] Addanki, Sanjaya; Cremonini, Roberto; and Penberthy, J. Scott 1991. Graphs of models. *Artificial Intelligence* 51:145–177.

[Amarel, 1968] Amarel, Saul 1968. On representations of problems of reasoning about actions. *Machine Intelligence* 3:131–171. Also appears in Webber, Bonnie Lynn and Nilsson, Nils J., editors 1981, *Readings in Artificial Intelligence*. Morgan Kaufmann Publishers. 2–22.

[Artobolevsky, 1980] Artobolevsky, Ivan I. 1980. *Mechanisms in Modern Engineering Design*, volume 5. Mir Publishers, Moscow.

[Bennett, 1987] Bennett, Scott W. 1987. Approximation in mathematical domains. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Los Altos, CA. International Joint Conferences on Artificial Intelligence, Inc., Morgan Kaufmann Publishers, Inc. 239–241.

[Bobrow, 1984] Bobrow, D., editor 1984. *Qualitative Reasoning About Physical Systems*. North-Holland.

[Brown *et al.*, 1982] Brown, John Seely; Burton, R. R.; and de Kleer, Johan 1982. Pedagogical, natural language and knowledge engineering techniques in SOPHIE I, II and III. In Sleeman, Derek and Brown, John Seely, editors 1982, *Intelligent Tutoring Systems*. Academic Press, New York. 227–282.

[Chandrasekaran, 1991] Chandrasekaran, B., editor 1991. *IEEE Expert* 6(2). Special issue on functional reasoning.

[Christensen, 1990] Christensen, Jens 1990. A hierarchical planner that generates its own abstraction hierarchies. In *Proceedings of AAAI-90.* 1004–1009.

[Collins and Forbus, 1987] Collins, John W. and Forbus, Kenneth D. 1987. Reasoning about fluids via molecular collections. In *Proceedings of AAAI-87.* 590–595.

[Coren, 1989] Coren, Richard L. 1989. *Basic Engineering Electromagnetics*. Prentice Hall, Englewood Cliffs, NJ 07632.

[Cormen *et al.*, 1990] Cormen, Thomas H.; Leiserson, Charles E.; and Rivest, Ronald L. 1990. *Introduction to Algorithms*. The MIT Press/McGraw-Hill Book Company.

[Crawford *et al.*, 1990] Crawford, James; Farquhar, Adam; and Kuipers, Benjamin 1990. QPC: A compiler from physical models into qualitative differential equations. In *Proceedings of AAAI-90.* 365–372.

[Dahlquist *et al.*, 1974] Dahlquist, Germund; Björk, Åke; and Anderson, Ned 1974. *Numerical Methods*. Prentice-Hall.

[Davenport, 1970] Davenport, Wilbur B. Jr. 1970. *Probability and Random Processes*. McGraw-Hill Book Company.

[Davis, 1984] Davis, Randall 1984. Diagnostic reasoning based on structure and behavior. *Artificial Intelligence* 24:347–410.

[de Kleer and Brown, 1984] de Kleer, Johan and Brown, John Seely 1984. A qualitative physics based on confluences. *Artificial Intelligence* 24:7–83.

[de Kleer and Brown, 1986] de Kleer, Johan and Brown, John Seely 1986. Theories of causal ordering. *Artificial Intelligence* 29:33–61.

[de Kleer, 1977] de Kleer, Johan 1977. Multiple representations of knowledge in a mechanics problem-solver. In *Proceedings of the International Joint Conference on Artificial Intelligence.* 299–304.

[Ellman, 1990] Ellman, Thomas, editor 1990. *Working Notes of the Automatic Generation of Approximations and Abstractions Workshop.*

[Ellman, 1992] Ellman, Thomas, editor 1992. *Working Notes of the Workshop on Approximation and Abstraction of Computational Theories.*

[Even, 1979] Even, Shimon 1979. *Graph Algorithms.* Computer Science Press.

[Falkenhainer and Forbus, 1988] Falkenhainer, Brian and Forbus, Kenneth D. 1988. Setting up large-scale qualitative models. In *Proceedings of the Seventh National Conference on Artificial Intelligence.* 301–306.

[Falkenhainer and Forbus, 1991] Falkenhainer, Brian and Forbus, Kenneth D. 1991. Compositional modeling: Finding the right model for the job. *Artificial Intelligence* 51:95–143.

[Falkenhainer, 1992a] Falkenhainer, Brian 1992a. A look at idealization. In *Proceedings of the AAAI-92 Workshop on Approximation and Abstraction of Computational Theories.*

[Falkenhainer, 1992b] Falkenhainer, Brian 1992b. Modeling without amnesia: Making experience-sanctioned approximations. In *Proceedings of the Sixth International Workshop on Qualitative Reasoning.*

[Forbus and Stevens, 1981] Forbus, Kenneth D. and Stevens, A. 1981. Using qualitative simulation to generate explanations. In *Proceedings of the Third Annual Meeting of the Cognitive Science Society.* 219–221.

[Forbus, 1984] Forbus, Kenneth D. 1984. Qualitative process theory. *Artificial Intelligence* 24:85–168.

[Forbus, 1990] Forbus, Kenneth D. 1990. The qualitative process engine. In Weld, Daniel S. and de Kleer, Johan, editors 1990, *Readings in Qualitative Reasoning about Physical Systems.* Morgan Kaufmann. 220–235.

[Garey and Johnson, 1979] Garey, Michael R. and Johnson, David S. 1979. *Computers and Intractability.* W. H. Freeman and Company.

[Guha, 1991] Guha, Ramanathan V. 1991. *Contexts: A Formalization and Some Applications.* Ph.D. Dissertation, Stanford University, Department of Computer Science, Stanford, CA. Also Report No. STAN-CS-91-1399.

[Halliday and Resnick, 1978] Halliday, David and Resnick, Robert 1978. *Physics.* John Wiley and Sons, third edition.

[Hamscher, 1988] Hamscher, Walter C. 1988. Model-based troubleshooting of digital systems. Technical Report 1074, Artificial Intelligence Laboratory, MIT, Cambridge, MA.

[Hamscher, 1991] Hamscher, Walter C. 1991. Modeling digital circuits for troubleshooting. *Artificial Intelligence* 51:223–271.

[Hayes, 1979] Hayes, Patrick J. 1979. The logic of frames. In Metzing, D., editor 1979, *Frame Conceptions and Text Understanding.* Walter de Gruyter and Company. 46–61. Also appears in Webber, Bonnie Lynn and Nilsson, Nils J., editors 1981, *Readings in Artificial Intelligence.* Morgan Kaufmann Publishers. 451–458.

[Hayes, 1985] Hayes, Patrick J. 1985. Naive physics I: Ontology for liquids. In Hobbs, Jerry R. and Moore, Robert C., editors 1985, *Theories of the Commonsense World.* Ablex Publishing Corporation. 71–89.

[Hillier and Lieberman, 1980] Hillier, Fredrick S. and Lieberman, Gerald J. 1980. *Introduction to Operations Reesearch.* Holden-Day, Inc., third edition.

[Hobbs, 1985] Hobbs, Jerry R. 1985. Granularity. In *Proceedings of IJCAI-85.* 432–435.

[Iwasaki and Chandrasekaran, 1992] Iwasaki, Yumi and Chandrasekaran, B. 1992. Design verification through function- and behavior-oriented representation. In Gero, John S., editor 1992, *Artificial Intelligence in Design '92.* Kluwer Academic Publishers. 597–616.

[Iwasaki and Low, 1991] Iwasaki, Yumi and Low, Chee-Meng 1991. Model generation and simulation of device behavior with continuous and discrete changes. Technical Report KSL 91-69, Stanford University, Knowledge Systems Laboratory.

[Iwasaki and Simon, 1986a] Iwasaki, Yumi and Simon, Herbert 1986a. Theories of causal ordering: Reply to de Kleer and Brown. *Artificial Intelligence* 29:63–72.

[Iwasaki and Simon, 1986b] Iwasaki, Yumi and Simon, Herbert A. 1986b. Causality in device behavior. *Artificial Intelligence* 29:3–32.

[Iwasaki, 1988] Iwasaki, Yumi 1988. Causal ordering in a mixed structure. In *Proceedings of the Seventh National Conference on Artificial Intelligence.* 313–318.

[Karp, 1972] Karp, Richard M. 1972. Reducibility among combinatorial problems. In Miller, R. E. and Thatcher, J. W., editors 1972, *Complexity of Computer Computations.* Plenum Press, New York. 85–103.

[Khachian, 1979] Khachian, L. G. 1979. A polynomial algorithm in linear programming. *Dokl. Akad. Nauk. SSSR* 244:1093–1096 (in Russian). English translation in *Soviet Math. Dokl.* **20** (1979), 191–194.

[Knoblock, 1991] Knoblock, Craig A. 1991. *Automatically Generating Abstractions for Problem Solving.* Ph.D. Dissertation, Carnegie Mellon University, School of Computer Science. Technical Report CMU-CS-91-120.

[Korf, 1980] Korf, Richard 1980. Towards a model of representation changes. *Artificial Intelligence* 14(1):41–78.

[Kuipers, 1986] Kuipers, Benjamin 1986. Qualitative simulation. *Artificial Intelligence* 29:289–338.

[Kuipers, 1987] Kuipers, Benjamin 1987. Abstraction by time-scale in qualitative simulation. In *Proceedings of the Sixth National Conference on Artificial Intelligence.* 621–625.

[Lenat and Guha, 1990] Lenat, Douglas B. and Guha, R. V. 1990. *Building Large Knowledge-Based Systems.* Addison Wesley.

[Liu and Farley, 1990] Liu, Zheng-Yang and Farley, Arthur M. 1990. Shifting ontological perspective in reasoning about physical systems. In *Proceedings of the Eighth National Conference on Artificial Intelligence.* 395–400.

[Mavrovouniotis and Stephanopolous, 1987] Mavrovouniotis, M. and Stephanopolous, G. 1987. Reasoning with orders of magnitude and approximate relations. In *Proceedings of AAAI-87.*

[Macaulay, 1988] Macaulay, David 1988. *The Way Things Work.* Houghton Mifflin Company, Boston.

[Mackie, 1974] Mackie, John L. 1974. *The Cement of the Universe.* Oxford University Press.

[McAllester, 1980] McAllester, David 1980. An outlook on truth maintenance. Memo 551, MIT Artificial Intelligence Laboratory.

[McCarthy, 1987] McCarthy, John 1987. Generality in artificial intelligence. *Communications of the ACM* 30(12):1030–1035.

[Mittal and Falkenhainer, 1990] Mittal, Sanjay and Falkenhainer, Brian 1990. Dynamic constraint satisfaction. In *Proceedings Eighth National Conference on Artificial Intelligence.* 25–32.

[Moore, 1979] Moore, Ramon E. 1979. *Methods and Applications of Interval Analysis.* SIAM Studies in Applied Mathematics. SIAM, Philadelphia.

[Murthy, 1988] Murthy, Seshashayee S. 1988. Qualitative reasoning at multiple resolutions. In *Proceedings of AAAI-88.* 296–300.

[Nayak and Joskowicz, 1995] Nayak, P. Pandurang and Joskowicz, Leo 1995. Efficient compositional modeling for generating causal explanations. *Artificial Intelligence.* To appear.

[Nayak *et al.*, 1992] Nayak, P. Pandurang; Joskowicz, Leo; and Addanki, Sanjaya 1992. Automated model selection using context-dependent behaviors. In *Proceedings of AAAI-92.* 710–716.

[Nayak, 1989] Nayak, P. Pandurang 1989. An overview of causal reasoning. Technical Report KSL 89-27, Stanford University, Knowledge Systems Laboratory.

[Nayak, 1991] Nayak, P. Pandurang 1991. Validating approximate equilibrium models. In *Proceedings of the 1991 Model-Based Reasoning Workshop.*

[Nayak, 1992a] Nayak, P. Pandurang 1992a. Causal approximations. In *Proceedings of AAAI-92.* 703–709.

[Nayak, 1992b] Nayak, P. Pandurang 1992b. Order of magnitude reasoning using logarithms. In Nebel, B.; Rich, C.; and Swartout, W., editors 1992b, *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR92)*, San Mateo, CA. Morgan Kaufmann.

[Nayak, 1994] Nayak, P. Pandurang 1994. Causal approximations. *Artificial Intelligence* 70:277–334.

[Patil *et al.*, 1981] Patil, Ramesh S.; Szolovits, Peter; and Schwartz, William B. 1981. Causal understanding of patient illness in medical diagnosis. In *Proceedings of IJCAI-81.* 893–899.

[Pearl, 1988] Pearl, Judea 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, CA.

[Pople, 1982] Pople, Harry E. Jr. 1982. Heuristic methods for imposing structure on ill-structured problems: The structuring of medical diagnostics. In Szolovits, Peter, editor 1982, *Artificial Intelligence in Medicine.* Westview Press. 119–190.

[Press *et al.*, 1989] Press, William H.; Flannery, Brian P.; Teukolsky, Saul A.; and Vetterling, William T. 1989. *Numerical Recipes in Pascal: The Art of Scientific Computing.* Cambridge University Press.

[Raiman and Williams, 1992] Raiman, Olivier and Williams, Brian C. 1992. Caricatures: Generating models of dominant behavior. In *Proceedings of the AAAI-92 Workshop on Approximation and Abstraction of Computational Theories.*

[Raiman, 1986] Raiman, Olivier 1986. Order of magnitude reasoning. In *Proceedings of AAAI-86.* 100–104.

[Raiman, 1991] Raiman, Olivier 1991. Order of magnitude reasoning. *Artificial Intelligence* 51:11–38.

[Rajamoney and Koo, 1990] Rajamoney, Shankar A. and Koo, Sang Hoe 1990. Qualitative reasoning with microscopic theories. In *Proceedings of AAAI-90.* AAAI Press/The MIT Press. 401–406.

[Rieger and Grinberg, 1977] Rieger, Chuck and Grinberg, Milt 1977. The declarative representation and procedural simulation of causality in physical mechanisms. In *Proceedings of IJCAI-77.* 250–256.

[Rose and Kramer, 1991] Rose, Philippe and Kramer, Mark A. 1991. Qualitative analysis of causal feedback. In *Proceedings of the Ninth National Conference on Artificial Intelligence.* 817–823.

[Sacerdoti, 1974] Sacerdoti, Earl 1974. Planning in a hierarchy of abstraction spaces. *Artificial Intelligence* 5:115–135.

[Sacks, 1987] Sacks, Elisha 1987. Hierarchical reasoning about inequalities. In *Proceedings of the Sixth National Conference on Artificial Intelligence.* 649–654.

[Schaefer, 1978] Schaefer, T. J. 1978. The complexity of satisfiability problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing.* Association for Computing Machinery. 216–226.

[Serrano and Gossard, 1987] Serrano, D. and Gossard, David C. 1987. Constraint management in conceptual design. In Sriram, D. and Adey, R. A., editors 1987, *Knowledge Based Expert Systems in Engineering: Planning and Design.* Computational Mechanics Publications. 211–224.

[Shirley and Falkenhainer, 1990] Shirley, Mark and Falkenhainer, Brian 1990. Explicit reasoning about accuracy for approximating physical systems. In *Working Notes of the Automatic Generation of Approximations and Abstractions Workshop.* 153–162.

[Shoham, 1985] Shoham, Yoav 1985. Reasoning about causation in knowledge-based systems. In *Proceedings of the Second Conference on AI Applications.* Institute of Electrical and Electronic Engineers. 629–634.

[Shoham, 1988] Shoham, Yoav 1988. *Reasoning about Change: Time and Causation from the Standpoint of Artificial Intelligence.* The MIT Press, Cambridge, Massachusetts.

[Simmons, 1986] Simmons, Reid 1986. "Commonsense" arithmetic reasoning. In *Proceedings of AAAI-86.* 118–124.

[Subramanian and Genesereth, 1987] Subramanian, Devika and Genesereth, Michael R. 1987. The relevance of irrelevance. In *Proceedings of IJCAI-87.* 416–422.

[Sussman and Steele Jr., 1980] Sussman, Gerald J. and Steele Jr., Guy L. 1980. CONSTRAINTS: A language for expressing almost-hierarchical descriptions. *Artificial Intelligence* 14(1):1–39.

[Unruh and Rosenbloom, 1989] Unruh, Amy and Rosenbloom, Paul S. 1989. Abstraction in problem solving and learning. In *Proceedings of IJCAI-89.* 681–687.

[van Amerongen, 1967] van Amerongen, C. 1967. *The Way Things Work.* Simon and Schuster.

[Van Baalen and Davis, 1988] Van Baalen, Jeffrey and Davis, Randall 1988. Overview of an approach to representation design. In *Proceedings of AAAI-88.* 392–397.

[van Emden and Kowalski, 1976] van Emden, M. H. and Kowalski, R. A. 1976. The semantics of predicate logic as a programming language. *Journal of the Association for Computing Machinery* 23(4):733–742.

[Wallis and Shortliffe, 1982] Wallis, J. W. and Shortliffe, E. H. 1982. Explanatory power for medical expert systems: Studies in the representation of causal relationships for clinical consultations. *Methods Inform. Med.* 21:127–136.

[Weiss *et al.*, 1978] Weiss, Sholom M.; Kulikowski, Casimir A.; Amarel, Saul; and Safir, Aran 1978. A model-based method for computer-aided medical decision-making. *Artificial Intelligence* 11:145–172.

[Weld and Addanki, 1991] Weld, Daniel S. and Addanki, Sanjaya 1991. Query-directed approximation. In Faltings, Boi and Struss, Peter, editors 1991, *Recent Advances in Qualitative Physics.* MIT Press, Cambridge, MA.

[Weld and de Kleer, 1990] Weld, Daniel S. and de Kleer, Johan, editors 1990. *Readings in Qualitative Reasoning About Physical Systems.* Morgan Kaufmann Publishers, Inc., San Mateo, California.

[Weld, 1983] Weld, Daniel S. 1983. Explaining complex engineered devices. Technical Report TR-5511, BBN, Cambridge, MA.

[Weld, 1990] Weld, Daniel S. 1990. Approximation reformulations. In *Proceedings Eighth National Conference on Artificial Intelligence.* 407–412.

[Weld, 1991] Weld, Daniel S. 1991. Reasoning about Model Accuracy. Technical Report 91-05-02, University of Washington, Department of Computer Science and Engineering.

[Welty *et al.*, 1984] Welty, James R.; Wicks, Charles E.; and Wilson, Robert E. 1984. *Fundamentals of Momentum, Heat, and Mass Transfer.* John Wiley and Sons, third edition.

[Williams, 1984] Williams, Brian C. 1984. Qualitative analysis of MOS circuits. *Artificial Intelligence* 24:281–346.

[Williams, 1989] Williams, Brian C. 1989. *Invention from First Principles via Topologies of Interactions.* Ph.D. Dissertation, M.I.T.

[Williams, 1990] Williams, Brian C. 1990. Interaction-based invention: Designing novel devices from first principles. In *Proceedings Eighth National Conference on Artificial Intelligence.* 349–356.

[Williams, 1991a] Williams, Brian C. 1991a. Critical abstraction: Generating simplest models for causal explanation. In *Proceedings of the Fifth International Workshop on Qualitative Reasoning about Physical Systems.*

[Williams, 1991b] Williams, Brian C. 1991b. A theory of interactions: unifying qualitative and quantitative algebraic reasoning. *Artificial Intelligence* 51:39–94.

# Lecture Notes in Artificial Intelligence (LNAI)

Vol. 837: S. Wess, K.-D. Althoff, M. M. Richter (Eds.), Topics in Case-Based Reasoning. Proceedings, 1993. IX, 471 pages. 1994.

Vol. 838: C. MacNish, D. Pearce, L. M. Pereira (Eds.), Logics in Artificial Intelligence. Proceedings, 1994. IX, 413 pages. 1994.

Vol. 847: A. Ralescu (Ed.) Fuzzy Logic in Artificial Intelligence. Proceedings, 1993. VII, 128 pages. 1994.

Vol: 861: B. Nebel, L. Dreschler-Fischer (Eds.), KI-94: Advances in Artificial Intelligence. Proceedings, 1994. IX, 401 pages. 1994.

Vol. 862: R. C. Carrasco, J. Oncina (Eds.), Grammatical Inference and Applications. Proceedings, 1994. VIII, 290 pages. 1994.

Vol 867: L. Steels, G. Schreiber, W. Van de Velde (Eds.), A Future for Knowledge Acquisition. Proceedings, 1994. XII, 414 pages. 1994.

Vol. 869: Z. W. Raś, M. Zemankova (Eds.), Methodologies for Intelligent Systems. Proceedings, 1994. X, 613 pages. 1994.

Vol. 872: S Arikawa, K. P. Jantke (Eds.), Algorithmic Learning Theory. Proceedings, 1994. XIV, 575 pages. 1994.

Vol. 878: T. Ishida, Parallel, Distributed and Multiagent Production Systems. XVII, 166 pages. 1994.

Vol. 886: M. M. Veloso, Planning and Learning by Analogical Reasoning. XIII, 181 pages. 1994.

Vol. 890: M. J. Wooldridge, N. R. Jennings (Eds.), Intelligent Agents. Proceedings, 1994. VIII, 407 pages. 1995.

Vol. 897: M. Fisher, R. Owens (Eds.), Executable Modal and Temporal Logics. Proceedings, 1993. VII, 180 pages. 1995.

Vol. 898: P. Steffens (Ed.), Machine Translation and the Lexicon. Proceedings, 1993. X, 251 pages. 1995.

Vol. 904: P. Vitányi (Ed.), Computational Learning Theory. EuroCOLT'95. Proceedings, 1995. XVII, 415 pages. 1995.

Vol. 912: N. Lavrăç S. Wrobel (Eds.), Machine Learning: ECML – 95. Proceedings, 1995. XI, 370 pages. 1995.

Vol. 918: P. Baumgartner, R. Hähnle, J. Posegga (Eds.), Theorem Proving with Analytic Tableaux and Related Methods. Proceedings, 1995. X, 352 pages. 1995.

Vol. 927: J. Dix, L. Moniz Pereira, T.C. Przymusinski (Eds.), Non-Monotonic Extensions of Logic Programming. Proceedings, 1994. IX, 229 pages. 1995.

Vol. 928: V.W. Marek, A. Nerode, M. Truszczynski (Eds.), Logic Programming and Nonmonotonic Reasoning. Proceedings, 1995. VIII, 417 pages. 1995.

Vol. 929: F. Morán, A. Moreno, J.J. Merelo, P.Chacón (Eds.), Advances in Artificial Life. Proceedings, 1995. XIII, 960 pages. 1995.

Vol. 934: P. Barahona, M. Stefanelli, J. Wyatt (Eds.), Artificial Intelligence in Medicine. Proceedings, 1995. XI, 449 pages. 1995.

Vol. 941: M. Cadoli, Tractable Reasoning in Artificial Intelligence. XVII, 247 pages. 1995.

Vol. 946: C. Froidevaux, J. Kohlas (Eds.), Symbolic Quantitative and Approaches to Reasoning under Uncertainty. Proceedings, 1995. X, 430 pages. 1995.

Vol. 954: G. Ellis, R. Levinson, W. Rich. J.F. Sowa (Eds.), Conceptual Structures: Applications, Implementation and Theory. Proceedings, 1995. IX, 353 pages. 1995.

Vol. 956: X. Yao (Ed.), Progress in Evolutionary Computation. Proceedings, 1993, 1994. VIII, 314 pages. 1995.

Vol. 957: C. Castelfranchi, J.-P. Müller (Eds.), From Reaction to Cognition. Proceedings, 1993. VI, 252 pages. 1995.

Vol. 961: K.P. Jantke. S. Lange (Eds.), Algorithmic Learning for Knowledge-Based Systems. X, 511 pages. 1995.

Vol. 981: I. Wachsmuth, C.-R. Rollinger, W. Brauer (Eds.), KI-95: Advances in Artificial Intelligence. Proceedings, 1995. XII, 269 pages. 1995.

Vol. 984: J.-M. Haton, M. Keane, M. Manago (Eds.), Advances in Case-Based Reasoning. Proceedings, 1994. VIII, 307 pages. 1995.

Vol. 990: C. Pinto-Ferreira, N.J. Mamede (Eds.), Progress in Artificial Intelligence. Proceedings, 1995. XIV, 487 pages. 1995.

Vol. 991: J. Wainer, A. Carvalho (Eds.), Advances in Artificial Intelligence. Proceedings, 1995. XII, 342 pages. 1995.

Vol. 992: M. Gori, G. Soda (Eds.), Topics in Artificial Intelligence. Proceedings, 1995. XII, 451 pages. 1995.

Vol. 997: K. P. Jantke, T. Shinohara, T. Zeugmann (Eds.), Algorithmic Learning Theory. Proceedings, 1995. XV, 319 pages. 1995.

Vol. 1003: P. Pandurang Nayak, Automated Modeling of Physical Systems. XXI, 232 pages. 1995.

Vol. 1010: M. Veloso, A. Aamodt (Eds.), Case-Based Reasoning Research and Development. Proceedings, 1995. X, 576 pages. 1995.

Vol. 1011: T. Furuhashi (Ed.), Advances in Fuzzy Logic, Neural Networks and Genetic Algorithms. Proceedings, 1994. VIII, 223 pages. 1995.

Vol. 1020: I. D. Watson (Ed.), Progress in Case-Based Reasoning. Proceedings, 1995. VIII, 209 pages. 1995.

# Lecture Notes in Computer Science

Vol. 988: A.U. Frank, W. Kuhn (Eds.), Spatial Information Theory. Proceedings, 1995. XIII, 571 pages. 1995.

Vol. 989: W. Schäfer, P. Botella (Eds.), Software Engineering — ESEC '95. Proceedings, 1995. XII, 519 pages. 1995.

Vol. 990: C. Pinto-Ferreira, N.J. Mamede (Eds.), Progress in Artificial Intelligence. Proceedings, 1995. XIV, 487 pages. 1995. (Subseries LNAI).

Vol. 991: J. Wainer, A. Carvalho (Eds.), Advances in Artificial Intelligence. Proceedings, 1995. XII, 342 pages. 1995. (Subseries LNAI).

Vol. 992: M. Gori, G. Soda (Eds.), Topics in Artificial Intelligence. Proceedings, 1995. XII, 451 pages. 1995. (Subseries LNAI).

Vol. 993: T.C. Fogarty (Ed.), Evolutionary Computing. Proceedings, 1995. VIII, 264 pages. 1995.

Vol. 994: M. Hebert, J. Ponce, T. Boult, A. Gross (Eds.), Object Representation in Computer Vision. Proceedings, 1994. VIII, 359 pages. 1995.

Vol. 995: S.M. Müller, W.J. Paul, The Complexity of Simple Computer Architectures. XII, 270 pages. 1995.

Vol. 996: P. Dybjer, B. Nordström, J. Smith (Eds.), Types for Proofs and Programs. Proceedings, 1994. X, 202 pages. 1995.

Vol. 997: K.P. Jantke, T. Shinohara, T. Zeugmann (Eds.), Algorithmic Learning Theory. Proceedings, 1995. XV, 319 pages. 1995.

Vol. 998: A. Clarke, M. Campolargo, N. Karatzas (Eds.), Bringing Telecommunication Services to the People – IS&N '95. Proceedings, 1995. XII, 510 pages. 1995.

Vol. 999: P. Antsaklis, W. Kohn, A. Nerode, S. Sastry (Eds.), Hybrid Systems II. VIII, 569 pages. 1995.

Vol. 1000: J. van Leeuwen (Ed.), Computer Science Today. XIV, 643 pages. 1995.

Vol. 1002: J.J. Kistler, Disconnected Operation in a Distributed File System. XIX, 249 pages. 1995.

Vol. 1003: P. Pandurang Nayak, Automated Modeling of Physical Systems. XXI, 232 pages. 1995. (Subseries LNAI).

Vol. 1004: J. Staples, P. Eades, N. Katoh, A. Moffat (Eds.), Algorithms and Computation. Proceedings, 1995. XV, 440 pages. 1995.

Vol. 1005: J. Estublier (Ed.), Software Configuration Management. Proceedings, 1995. IX, 311 pages. 1995.

Vol. 1006: S. Bhalla (Ed.), Information Systems and Data Management. Proceedings, 1995. IX, 321 pages. 1995.

Vol. 1007: A. Bosselaers, B. Preneel (Eds.), Integrity Primitives for Secure Information Systems. VII, 239 pages. 1995.

Vol. 1008: B. Preneel (Ed.), Fast Software Encryption. Proceedings, 1994. VIII, 367 pages. 1995.

Vol. 1009: M. Broy, S. Jähnichen (Eds.), KORSO: Methods, Languages, and Tools for the Construction of Correct Software. X, 449 pages. 1995. Vol.

Vol. 1010: M. Veloso, A. Aamodt (Eds.), Case-Based Reasoning Research and Development. Proceedings, 1995. X, 576 pages. 1995. (Subseries LNAI).

Vol. 1011: T. Furuhashi (Ed.), Advances in Fuzzy Logic, Neural Networks and Genetic Algorithms. Proceedings, 1994. VIII, 223 pages. 1995. (Subseries LNAI).

Vol. 1012: M. Bartošek, J. Staudek, J. Wiedermann (Eds.), SOFSEM '95: Theory and Practice of Informatics. Proceedings, 1995. XI, 499 pages. 1995.

Vol. 1013: T.W. Ling, A.O. Mendelzon, L. Vieille (Eds.), Deductive and Object-Oriented Databases. Proceedings, 1995. XIV, 557 pages. 1995.

Vol. 1014: A.P. del Pobil, M.A. Serna, Spatial Representation and Motion Planning. XII, 242 pages. 1995.

Vol. 1015: B. Blumenthal, J. Gornostaev, C. Unger (Eds.), Human-Computer Interaction. Proceedings, 1995. VIII, 203 pages. 1995.

Vol. 1016: R. Cipolla, Active Visual Inference of Surface Shape. XII, 194 pages. 1995.

Vol. 1017: M. Nagl (Ed.), Graph-Theoretic Concepts in Computer Science. Proceedings, 1995. XI, 406 pages. 1995.

Vol. 1018: T.D.C. Little, R. Gusella (Eds.), Network and Operating Systems Support for Digital Audio and Video. Proceedings, 1995. XI, 357 pages. 1995.

Vol. 1019: E. Brinksma, W.R. Cleaveland, K.G. Larsen, T. Margaria, B. Steffen (Eds.), Tools and Algorithms for the Construction and Analysis of Systems. Proceedings, 1995. VII, 291 pages. 1995.

Vol. 1020: I.D. Watson (Ed.), Progress in Case-Based Reasoning. Proceedings, 1995. VIII, 209 pages. 1995. (Subseries LNAI).

Vol. 1021: M.P. Papazoglou (Ed.), OOER '95: Object-Oriented and Entity-Relationship Modelling. Proceedings, 1995. XVII, 451 pages. 1995.

Vol. 1022: P.H. Hartel, R. Plasmeijer (Eds.), Functional Programming Languages in Education. Proceedings, 1995. X, 309 pages. 1995.

Vol. 1023: K. Kanchanasut, J-J. Lévy (Eds.), Algorithms, Concurrency and Knowlwdge. Proceedings, 1995. X, 410 pages. 1995.