Lecture Notes in Computer Science 5055

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Khalid Al-Begain Armin Heindl
Miklós Telek (Eds.)

Analytical and Stochastic Modeling Techniques and Applications

15th International Conference, ASMTA 2008
Nicosia, Cyprus, June 4-6, 2008
Proceedings

Springer

Volume Editors

Khalid Al-Begain
Integrated Communications Research Centre
Faculty of Advanced Technology
University of Glamorgan
Wales, UK
E-mail: kbegain@glam.ac.uk

Armin Heindl
Computer Networks and Communication Systems
University Erlangen-Nürnberg
Erlangen, Germany
E-mail: armin.heindl@informatik.uni-erlangen.de

Miklós Telek
Department of Telecommunications
Budapest University of Technology and Economics
Budapest, Hungary
E-mail: telek@hit.bme.hu

Library of Congress Control Number: 2008928016

CR Subject Classification (1998): D.2.4, C.2, F.3, D.2.8, D.4, C.4, K.6.2

LNCS Sublibrary: SL 2 – Programming and Software Engineering

ISSN 0302-9743
ISBN-10 3-540-68980-X Springer Berlin Heidelberg New York
ISBN-13 978-3-540-68980-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12279739 06/3180 5 4 3 2 1 0

Preface

It is our great pleasure to present the proceedings of the 15th International Conference on Analytical and Stochastic Modelling Techniques and Applications (ASMTA 2008) that took place on the beautiful island of Cyprus for the first time.

The conference has become an important annual event in the fields of analytical modelling and performance evaluation in Europe and internationally. Nevertheless, efforts have been made year after year to raise the standard and the quality of the programme. This year the proceedings are published as part of Springer's prestigious *Lecture Notes in Computer Science* (LNCS) series. This is another sign of the growing confidence in the quality standards and procedures followed in the reviewing process and the programme compilation.

The conference was honored to have a distinguished keynote speaker in the person of Raymond Marie from IRISA/INRIA, France, who is a prominent figure of the analytical modelling community in Europe and worldwide. The conference had a high-quality programme with an acceptance ratio of 40%. The programme comprised 22 high-quality papers organized into 7 sessions. Almost every paper was peer reviewed by three reviewers from the International Programme Committee. The reviewers were truly wonderful this year, as well, and in most cases the reviews provided valuable comments that contributed to increasing the quality of the final versions of the papers. In many cases, discussion panels were also organized when the reviews were not decisive.

We would therefore like to give a special thanks to all the members of the International Programme Committee for the excellent work in the reviewing process and the subsequent discussion panels during the selection process.

Keeping the traditions, ASMTA was co-located with the European Conference on Modelling and Simulation, the official conference of the European Council on Modelling and Simulation, and also the International Conference on High Performance Simulation. This gave participants of ASMTA a unique opportunity to interact with colleagues from these very relevant and complementary areas. They also enjoyed the keynote talks delivered by prominent figures in those areas.

The local organizers made every effort to make the conference a memorable event. For that we give them our sincere thanks and appreciation.

April 2008

Khalid Al-Begain
Armin Heindl
Miklós Telek

Organization

Workshop Chair

Khalid Al-Begain, University of Glamorgan, UK

Program Chairs

Armin Heindl, University of Erlangen-Nürnberg, Germany
Miklós Telek, Technical University of Budapest, Hungary

Program Commitee

Vladimir Anisimov, GlaxoSmithKline, UK
Imad Antonios, Southern Connecticut State University, USA
Amine Berqia, University of Algarve, Portugal
Gnther Bolch, University of Erlangen-Nuremberg, Germany
Alexander Dudin, Belarusian State University, Belarus
Antonis Economou, University of Athens, Greece
Dieter Fiems, University of Ghent, Belgium
Jean-Michel Fourneau, University of Versailles, France
Rossano Gaeta, University of Turin, Italy
Reinhard German, University of Erlangen-Nuremberg, Germany
Marco Gribaudo, University of Turin, Italy
Guenter Haring, University of Vienna, Austria
Gabor Horvath, Budapest University of Technology, Hungary
Helen Karatza, Aristotle University of Thessaloniki, Greece
Remco Litjens, TNO Inf. & Comm. Technology, Netherlands
Maria Lopez-Herrero, Complutense University of Madrid, Spain
Don McNickle, University of Canterbury, New Zealand
Bruno Mueller-Clostermann, University of Duisburg-Essen, Germany
Mohamed Ould-Khaoua, University of Glasgow, UK
Krzysztof Pawlikowski, University of Canterbury, New Zealand
Marie-Ange Remiche, Free University of Brussels, Belgium
Evgenia Smirni, College of William and Mary, USA
Bruno Sericola, IRISA/INRIA Rennes, France
Janos Sztrik, Lajos Kossuth University, Hungary
Baris Tan, Koc University, Turkey
Nigel Thomas, University of Newcastle, UK
Petr Tuma, Charles University of Prague, Czech Republic
Dietmar Tutsch, TU Berlin, Germany

Kurt Tutschku, University of Wuerzburg, Germany
Sabine Wittevrongel, University of Ghent, Belgium
Katinka Wolter, Humboldt University of Berlin, Germany

External Referees

Freimut Brenner, University of Duisburg-Essen, Germany
Susanna Donatelli, University of Turin, Italy
Tobias Hossfeld, University of Wuerzburg, Germany
Daniele Manini, University of Turin, Italy
Ningfang Mi, College of William and Mary, VA, USA
Katy Paroux, University of Franche-Comté, France
Andreas Pillekeit, University of Duisburg-Essen, Germany
Philipp Reinecke, Humboldt University Berlin, Germany
Daniel Schlosser, University of Wuerzburg, Germany
Matteo Sereno, University of Turin, Italy
Eddy Zhang, College of William and Mary, VA, USA
Thomas Zinner, University of Wuerzburg, Germany

Table of Contents

Traffic Modeling

Queueing Systems

Analytical Methods and Applications

Distributions in Stochastic Modeling

Queueing Networks

Simulation and Model Checking

Wireless Networks

Markovian Characterisation of H.264/SVC Scalable Video

Dieter Fiems, Veronique Inghelbrecht, Bart Steyaert, and Herwig Bruneel

SMACS Research Group
Department of Telecommunications and Information Processing, Ghent University
St-Pietersniewstraat 41, 9000 Gent, Belgium
{df,vi,bs,hb}@telin.UGent.be

Abstract. In this paper, a multivariate Markovian traffic model is proposed to characterise H.264/SVC scalable video traces. Parametrisation by a genetic algorithm results in models with a limited state space which accurately capture both the temporal and the inter-layer correlation of the traces. A simulation study further shows that the model is capable of predicting performance of video streaming in various networking scenarios.

1 Introduction

A video stream is called scalable if parts of the stream (layers) can be removed such that the resulting substream forms another valid video stream for some decoder [1]. The benefits of efficient scalable video coding for video transmission over packet-switched networks include, amongst others, support for heterogeneous clients in multicast transmission scenarios and the capability of graceful degradation of the video quality if required by the network conditions. Since a single stream offers the same content in different formats, heterogeneous clients may decode multicast video streams in their preferred format: preferred spatial and temporal resolution and preferred fidelity. Moreover, by service differentiation the network can ensure delivery of some substreams of the video stream while discarding other substreams during network congestion. From the vantage point of the receivers, the perceived video quality then degrades gracefully during congestion by reducing the spatial or temporal resolution or the fidelity.

This contribution addresses the statistical characterisation of H.264/SVC scalable video streams. Sufficiently detailed statistical traffic models are important for network design and performance evaluation [2]. A good statistical model is able to predict performance of the real video source in various networking scenarios. It should lead to comparable performance predictions when used instead of the actual video traces. Furthermore, if the model is to be used as input for various queueing analyses, the mathematical tractability of the associated queueing models is an additional requirement. Obviously, there is a trade-off between the mathematical tractability and the complexity and accuracy of the video models.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 1–15, 2008.

Characterisation of variable bit rate video has been an active research area for almost 20 years. See amongst others the survey by Izquierdo and Reeves [5] on statistical traffic models for MPEG-1 encoded video. Proposed statistical models include Markovian processes [6,7,8,9,10], autoregressive processes [11,12], spatial renewal processes [13] and self-similar processes [14]. Although video coding standards such as MPEG-2, H.263 and MPEG-4 Visual already support various scalability modes, they have been rarely used in practice due to decreased coding efficiency and increased coding complexity [1]. These issues however have recently been addressed in the H.264/SVC standard.

Hence, few statistical traffic models focus on the scalability of the video sources. Existing models describe two layers at most; a base layer and a temporal enhancement layer. However, traffic models for H.264/SVC video need to take into account the correlation between the base and enhancement layers not only in the temporal, but also in the spatial and quality scalable dimensions. Capturing the multi-layered structure of H.264/SVC is the subject of this paper.

In this contribution, we propose a multivariate Markovian model — in particular, a multi-class discrete batch Markovian arrival process (DBMAP) — to characterise H.264/SVC scalable video. This model can capture the temporal correlation as well as the correlation between the different layers of the H.264/SVC source. Moreover, the associated queueing behaviour can be investigated efficiently by means of matrix analytic methods [3, 4].

We use a genetic algorithm to parametrise the Markovian model for a given trace. The genetic approach for video characterisation has recently been proposed by Kempken and Luther [15] in the context of characterising single layer H.264 video sources at the group of pictures (GOP) level. These authors show that a genetic approach yields Markovian models with small state spaces that can capture the time correlation of the trace accurately. We here extend the approach of [15] to Markovian characterisation of multi-layered video at the sub-GOP level. In particular, the different types of slices are grouped into a number of traffic classes and the H.264/SVC source is investigated at the traffic class level. A key notion of the present approach is the matrix-valued covariance function which is used to assess the fitness (see further) of the Markov model. As opposed to the correlation function used in [15], the covariance function depends on the second order moments in each of the Markov states. Hence, inter-layer correlation is taken into account while assessing the fitness of the model.

The remainder of this paper is organised as follows. In the next section, the H.264/SVC traces and their statistical properties are introduced. We then focus on the properties of the multivariate Markovian process in section 3. The genetic algorithm for video characterisation is presented in section 4 and numerically evaluated in section 5. Finally conclusions are drawn in section 6.

2 H.264/SVC Traces

The H.264/SVC traces are encoded with the SVC reference software version JSVM 8.9. We characterise two traces: a trace of the movie “Brotherhood of

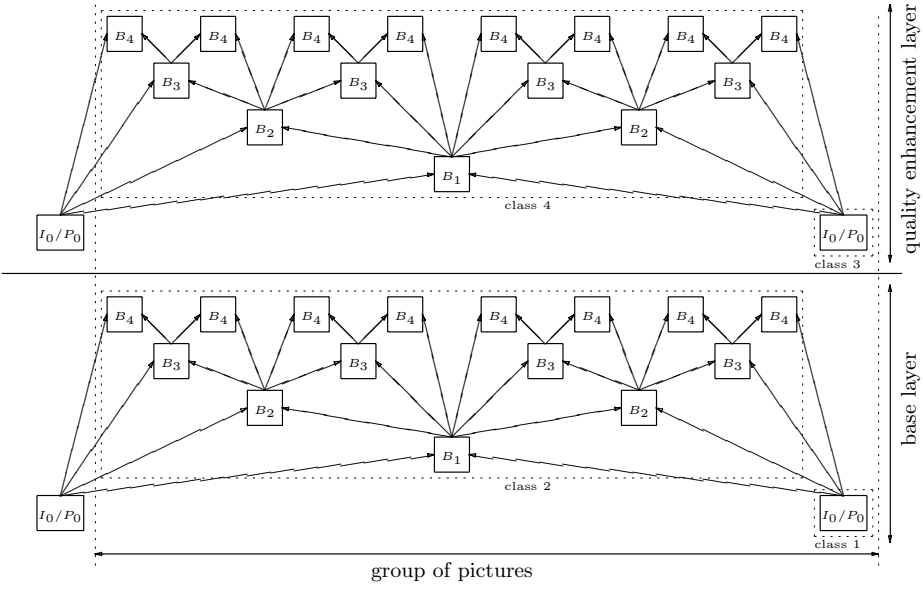


Fig. 1. Group of picture structure of the H.264 traces under consideration

the Wolf" consisting of 13447 GOPs and a trace of the movie "Pirates of the Caribbean" consisting of 12770 GOPs. Both traces are encoded into a base layer and a Medium Grain Scalability (MGS) quality enhancement layer. Further, within each of these layers there are 5 temporal layers as depicted in Figure 1. Notice that I, P and B slices refer to intra-picture predictive coded, inter-picture predictive coded and bi-directional predictive coded slices respectively. In total, there are 2×16 slices per GOP. The frame rate is equal to 25 frames per second — hence a GOP corresponds to $0.64s$ — and the spatial dimensions are equal to 720×304 pixels for the base layer as well as for the enhancement layer. Encoder quality parameters (QP, MeQP1 to MeQP5 and MeQPLP) are set to 31 and 25 for the base layer and the enhancement layer respectively. Further information on H.264/SVC encoding can be retrieved from [1].

As already mentioned, we here investigate the H.264/SVC sources at the traffic class level. We define the following 4 traffic classes:

- class 1: I and P slices of the base layer;
- class 2: B slices of the base layer;
- class 3: I and P slices of the quality enhancement layer;
- class 4: B slices of the quality enhancement layer.

The correspondence between these traffic classes and the different slices of the GOP is also illustrated in Figure 1.

Although we focus on the traffic classes defined above in the remainder of this contribution, our approach also works for different sets of traffic classes. From

the vantage point of characterising the video source at the traffic class level, a trace is a series of row vectors $X = \{X_k, k = 0, \ldots, N-1\}$ where X_k is a row vector of size C whose jth element $X_{k,j}$ equals the number of bytes of class j that are generated during GOP k. Here, N denotes the number of GOPs in the trace and C denotes the number of traffic classes. Given the trace X, we now define some statistics.

The sample mean (row) vector and the $C \times C$ sample covariance matrix of the trace are defined as,

$$\bar{X} = \frac{1}{N} \sum_{n=0}^{N-1} X_n \,, \quad \mathrm{cov}(X) = \frac{1}{N} \sum_{n=0}^{N-1} X_n' X_n - \bar{X}' \bar{X} \,, \tag{1}$$

respectively. Here the vector Y' is the transposed of the vector Y. Notice that we use the biased estimator of the covariance matrix. This will hardly influence the results since the numbers of GOPs N in the traces under consideration are sufficiently large.

With the trace X we associate the sum traces $S^{(k)}(X) = \{S_n^{(k)}(X), n = 0, \ldots, N-1\}$, with,

$$S_n^{(k)}(X) = \sum_{i=n}^{(n+k) \bmod N} X_i \,. \tag{2}$$

for $k = 0, 1, 2, \ldots$. The (matrix valued) sample covariance function then maps k onto the sample covariance matrix of $S^{(k)}(X)$,

$$\mathrm{cf}(k) = \frac{1}{N} \sum_{n=0}^{N-1} S_n^{(k)}(X)' S_n^{(k)}(X) - (k+1)^2 \bar{X}' \bar{X} \,, \tag{3}$$

for $k = 0, 1, 2, \ldots$.

Of particular interest for the statistical characterisation of the video source at the traffic class level are the scalar valued covariance functions of the aggregated traffic of a subset of the traffic classes. Let $\mathcal{C}$ be any subset of the set of traffic classes $\{1, 2, \ldots, C\}$ and let $X_{\mathcal{C}}$ denote the series

$$X_{\mathcal{C}} = \left\{ \sum_{j \in \mathcal{C}} X_{k,j}, k = 0, 1, \ldots N-1 \right\} . \tag{4}$$

The corresponding scalar valued covariance function $\mathrm{cf}_{\mathcal{C}}(k)$ then equals,

$$\mathrm{cf}_{\mathcal{C}}(k) = \epsilon_{\mathcal{C}} \, \mathrm{cf}(k) \epsilon_{\mathcal{C}}' \,, \tag{5}$$

with $\epsilon_{\mathcal{C}}$ a row vector of size C whose ith element equals 1 if i is an element of $\mathcal{C}$ and 0 if this is not the case.

3 The Multi-Class DBMAP

Let $q = \{q_i, i = 0, 1, \ldots\}$ denote an irreducible Markov chain defined on a finite state space $\mathcal{K} = \{1, \ldots, K\}$. Such a Markov chain is completely characterised by the transition probabilities $\gamma_{i,j}$, $i, j \in \mathcal{K}$. Further, let $\Gamma = [\gamma_{i,j}]_{i,j\in\mathcal{K}}$ denote the $K \times K$ transition matrix of the Markov chain and let $\pi = [\pi_1, \ldots, \pi_K]$ denote the row vector of its steady state probabilities. The vector π is the unique normalised solution of $\pi = \pi\Gamma$. The Markov chain q is referred to as the modulating Markov chain of the DBMAP.

A multi-class discrete-time batch Markovian arrival process $x = \{x_k, k = 0, 1, 2, \ldots\}$ is a discrete time stochastic process which takes values in $\mathbb{R}^C$ such that the multivariate distribution of x_k only depends on the states q_k and q_{k+1} of the modulating Markov chain (for all k). Hence, this process is completely characterised by the transition matrix Γ of the modulating Markov chain and the doubly indexed set of multivariate distribution functions $\{\Phi_{i,j}(y), i, j \in \mathcal{K}, y \in \mathbb{R}^C\}$. $\Phi_{i,j}(y) = \Pr[x_{k,1} < y_1, \ldots x_{k,C} < y_C | q_k = i, q_{k+1} = j]$ is the multivariate distribution of x_k given $q_k = i$ and $q_{k+1} = j$. Here $x_{k,j}$ and y_j are the jth entries of the vectors x_k and y respectively. Notice that in literature DBMAPs only take values in $\mathbb{N}^C$. The process described here is somewhat more general but we will also refer to it as a DBMAP. However, one may discretise the traffic model for performance evaluation purposes. The resulting process is then a DBMAP as described in literature.

In the remainder, we will make the additional assumptions that (i) the process x_k at time k only depends on q_k and (ii) that for a given state q_k, the random vector x_k is multivariate normally distributed. Hence, the process (x_k, q_k) is completely characterised by the transition matrix Γ and the following mean vectors and covariance matrices in the different states $i \in \mathcal{K}$,

$$\mu_i = \mathrm{E}[x_k | q_k = i]\,, \quad \Omega_i = \mathrm{E}[x_k' x_k | q_k = i] - \mu_i' \mu_i\,. \tag{6}$$

Notice that the normal distribution takes negative values with a positive probability. This probability will however turn out to be negligibly small for the DBMAPs that characterise the video traces.

We now retrieve some characteristics of the process x in terms of the characteristics Γ, π, μ_i and Ω_i of the multi-class DBMAP. The mean vector μ and covariance matrix Ω are given by,

$$\mu = \sum_{i\in\mathcal{K}} \pi_i \mu_i\,, \quad \Omega = \sum_{i=1}^{N} \pi_i(\Omega_i + \mu_i'\mu_i) - \mu'\mu\,. \tag{7}$$

Further, let $s^{(k)} = \{s_n^{(k)}, n = 0, 1, 2, \ldots\}$ denote the sum process corresponding to x. Here $s_n^{(k)}$ is defined as,

$$s_n^{(k)} = \sum_{i=n}^{n+k} x_i\,. \tag{8}$$

The matrix valued covariance function $\Theta(k)$ of the DBMAP then maps k on the covariance matrix of the process $s^{(k)}$,

$$\Theta(k) = \mathrm{E}[(s_n(k) - \mathrm{E}[s_n(k)])'(s_n(k) - \mathrm{E}[s_n(k)])] \,. \tag{9}$$

Since we have $s_n(k) = s_n(k-1) + x_{n+k}$ and by conditioning on the states of the modulating Markov chain, one shows that the covariance function can be obtained recursively by means of the following expressions,

$$\Theta(k) = \Theta(k-1) + \Psi(k) + \Psi(k)' \,, \tag{10}$$

with,

$$\begin{aligned} \Theta(0) &= \Omega \,, \\ \Psi(k) &= \sum_{i,j \in \mathcal{K}} \psi_i(k-1)' \mu_j \gamma_{i,j} \,, \\ \psi_l(0) &= \mu_l \pi_l \,, \\ \psi_l(k) &= \mu_l + \sum_{j \in \mathcal{K}} \psi_j(k-1) \gamma_{j,l} \,, \end{aligned} \tag{11}$$

for $k = 1, 2, \ldots$ and for $l \in \mathcal{K}$.

As for the traces, the scalar valued covariance functions of the aggregated traffic of subsets of the traffic classes can be expressed in terms of the covariance function $\Theta(k)$. For any subset $\mathcal{C}$ of the traffic classes, let $x_{\mathcal{C}}$ denote following scalar valued DBMAP,

$$x_{\mathcal{C}} = \left\{ \sum_{j \in \mathcal{C}} x_{k,j}, k = 0, 1, \ldots N-1 \right\} \,. \tag{12}$$

The scalar valued covariance function of $x_{\mathcal{C}}$ then equals,

$$\Theta_{\mathcal{C}}(k) = \epsilon_{\mathcal{C}} \Theta(k) \epsilon_{\mathcal{C}}' \,. \tag{13}$$

Recall that $\epsilon_{\mathcal{C}}$ denotes a row vector of size C whose ith element equals 1 if i is an element of $\mathcal{C}$ and 0 if this is not the case.

4 Genetic Characterisation

The genetic approach under consideration exploits the fact that Markovian characterisation is almost trivial if not only the trace, but also the states of the modulating Markov chain are known. That is, given the trace X and a series of states $Q = \{Q_k, k = 0, \ldots, N-1\}$ associated to the consecutive GOPs, one easily constructs a multi-class DBMAP by counting state transitions and by calculating conditional moments. We find the following DBMAP characteristics,

$$
\begin{aligned}
N_i &= \sum_{k=0}^{N-1} 1(Q_k = i)\,, \\
\gamma_{i,j} &= \frac{1}{N_i} \sum_{k=0}^{N-1} 1(Q_k = i, Q_{(k+1) \bmod N} = j)\,, \\
\mu_i &= \frac{1}{N_i} \sum_{k=0}^{N-1} X_k 1(Q_k = i)\,, \\
\Omega_i &= \frac{1}{N_i} \sum_{k=0}^{N-1} X_k' X_k 1(Q_k = i) - \mu_i' \mu_i\,. \qquad (14)
\end{aligned}
$$

In the former expressions $1(\cdot)$ denotes the standard indicator function.

Hence, finding a multi-class DBMAP corresponds to finding a series Q such that the statistical properties of the derived DBMAP closely match the statistical properties of the trace. It is easily shown that the mean vector and the covariance matrix of the DBMAP with parameters as given in equation (14) match the sample mean and sample covariance matrix of the trace. To capture temporal correlation, the authors of [15] focus on matching the autocorrelation function of the DBMAP with the autocorrelation function of the trace. However, the autocorrelation of a DBMAP only depends on the covariance matrix Ω of the DBMAP and not on the covariance matrices Ω_i in the different states of the DBMAP. Since it is our objective to accurately capture the correlation between the different traffic classes, we here focus on matching the sample covariance function of the trace with the covariance function of the DBMAP. In accordance with equations (10) and (11), the covariance function does depend on the matrices Ω_i.

Given the length of the trace N and the number of states of the Markov chain K, there are K^N possible series Q of state assignments. For the traces under consideration and for a Markov state space of size $K = 3$ this means that there are more than 10^{6000} possible series Q of state assignments. The size of the set of possible solutions clearly makes it infeasible to find the best solution by testing all possible solutions. We therefore use a genetic algorithm to search the space of all possible state assignments more efficiently.

A genetic algorithm is a search and optimisation procedure, inspired by Darwin's theory of evolution (see e.g. [16] for a survey on genetic algorithms). A genetic algorithm transforms a set or generation of solutions into a new generation of solutions using the Darwinian principle of reproduction and survival of the fittest. That is, given a generation of solutions, a new generation of solutions is created by means of mutation and crossover of solutions of the given generation (reproduction) and only the best solutions are retained (survival of the fittest).

In the present context, a solution is a series Q of state assignments to the consecutive GOPs. Its fitness measures the accuracy of the match of the covariance

function of the DBMAP with the sample covariance function of the trace. In particular, the fitness of a solution Q is defined as follows,

$$\frac{1}{\text{fitness}(Q)} = \frac{1}{L}\sum_{k=1}^{L} |w(\text{cf}(k) - \Theta(k))w'| \,. \tag{15}$$

Here $\text{cf}(k)$ is the sample covariance function of the trace as given in equation (3), L is the maximal lag for which the covariance function of the DBMAP and the sample covariance function of the trace are compared and $\Theta(k)$ is the covariance function of the DBMAP. The latter can be obtained by means of equations (10) and (11) in terms of the vectors π and μ_i $(i = 1, \ldots, K)$ and the matrices Γ and Ω_i $(i = 1, \ldots, K)$ that characterise the multi-class DBMAP. For a given solution Q, these vectors and matrices are given in equation (14). Finally, w is a weight row vector. Notice that a high fitness indicates an accurate match of covariance and sample covariance functions.

The row vector w in equation (15) allows one to assign different weights to the various elements of the covariance matrix. In particular, one can assign weights to the variance functions of aggregated traffic from subsets of the traffic classes as follows. For all i, let α_i denote the weight assigned to the variance function of the aggregated traffic class $\mathcal{C}_i$ ($\mathcal{C}_i$ is a subset of the set of traffic classes). In view of property (5) of sample covariance functions and in view of property (13) of the covariance function of a DBMAP, we obtain the following weight vector,

$$w = \sum_i \sqrt{\alpha_i}\, \epsilon_{\mathcal{C}_i} \,. \tag{16}$$

Recall that $\epsilon_{\mathcal{C}_i}$ is a row vector of size C whose ith element equals 1 if i is an element of $\mathcal{C}$ and 0 if this is not the case.

Given the fitness function, the execution of the genetic algorithm operating on the generations of solutions of state assignments can now be summarised as follows.

First, a random generation of solutions is created. Each solution is created by randomly assigning states to the consecutive GOPs. Then, the following substeps are performed until a sufficiently fit solution is found.

1. Calculate the fitness of each solution in the current generation. The fitness of the generation is defined as the fitness of the best solution of the generation.
2. Create a new generation by means of the following operations.
   (a) Mutation: randomly choose a solution of the current generation with probability inversely proportional to the fitness of the solution. Then either (i) change a continuous block of state assignments to a randomly chosen state or (ii) reverse a continuous block of state assignments. Add this new (mutated) solution to the new generation.
   (b) Crossover: randomly choose two solutions of the current generation with probability inversely proportional to the fitness of these solutions. Interchange the state assignments of a randomly chosen sub-part of the state assignments. Add these new solutions to the new generation.

3. Copy the fittest solution of the current generation to the new generation.

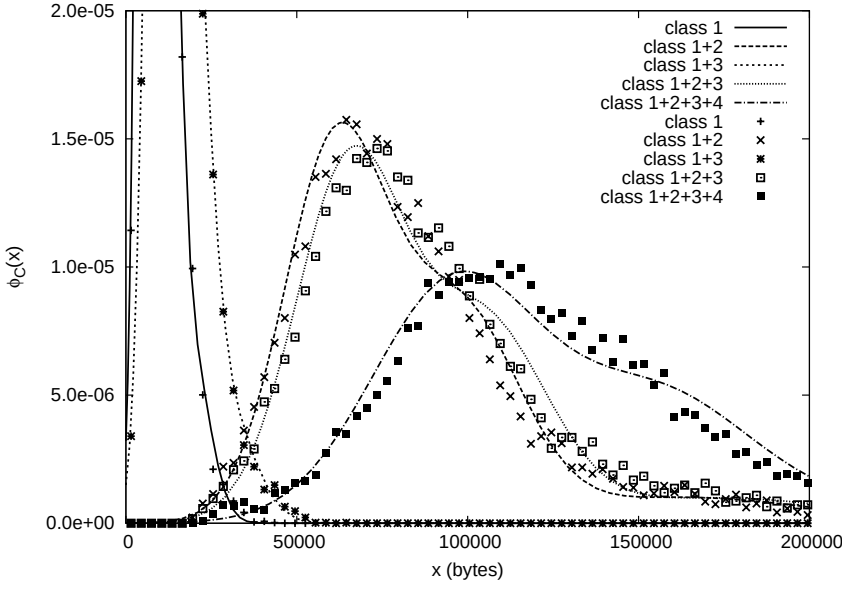
Notice that step 3 implies that the fitness of consecutive generations never decreases.

5 Numerical Results

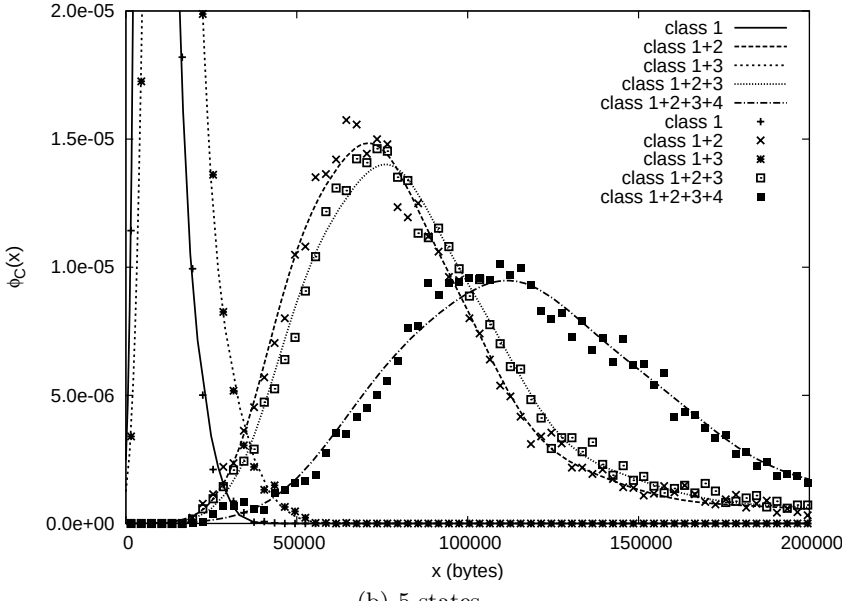
We now evaluate the genetic video characterisation approach by means of a numerical example. We here focus on the characterisation of the trace of the movie "Brotherhood of the Wolf". Similar results were obtained for the trace of the movie "Pirates of the Caribbean". The size of the state space of the generated DBMAP is set to $K = 3$ or $K = 5$ and the fitness of a solution is based on the (sample) covariance functions of lag 1 up to lag $L = 10$. The weight vector w equals $[5, 3, 3, 1]$. This choice is motivated by assigning the same weight to all decodable substreams; class 1, class 1 and 2, class 1 and 3, class 1 to 3 and class 1 to 4 constitute decodable substreams.

In Figures 2 and 3 the statistical characteristics of the multi-class DBMAP are compared with the corresponding characteristics of the trace. For all plots, the lines correspond to DBMAP values whereas points correspond to trace values and different values of the state space size are considered as indicated. Figures 2 and 3 show that the distributions and the variance functions of subclasses of the multi-class DBMAP closely match the corresponding trace histograms and sample variance functions respectively. From Figure 2 it is seen that the match between distribution functions and histograms does not improve by increasing the number of states of the DBMAP. This is not unexpected since the genetic algorithm does not take into account how well histogram and distribution match. In fact, it is somewhat surprising that there is such a close match in the first place. In contrast, increasing the size of the state space improves the accuracy of the match of the (sample) variance functions as can be seen in Figure 3.

We now evaluate the Markov model in a networking scenario. For this, we consider a discrete-time finite capacity buffer with partial buffer sharing (PBS) through which the traffic stemming from 4 video sources is routed. The data from the video sources is divided into fixed length packets of 1500 bytes of which the buffer can store up to 200. The bandwidth of the output line of the buffer equals 8 Mbit and the partial buffer sharing thresholds are set to 90%, 80% and 70%. That is, class 2, 3 and 4 packets are accepted as long as the buffer occupancy is less than 90%, 80% and 70% respectively. In Figure 4 the probability mass function (pmf) of the queue content of the different traffic classes is depicted. Both the results from trace based simulations (thick lines) and model based simulations (thin lines) are depicted. The match is already reasonable for the 3 state DBMAP but further improves by adding states. However, one can see that even for the 5 state DBMAP, there is no accurate match for the tails of the distributions of the queue content. Nevertheless, this is crucial since the tail distribution relates to the packet loss ratio which is an important performance measure. That is, a close match is necessary if the model is to be used to assess the packet loss ratio accurately.

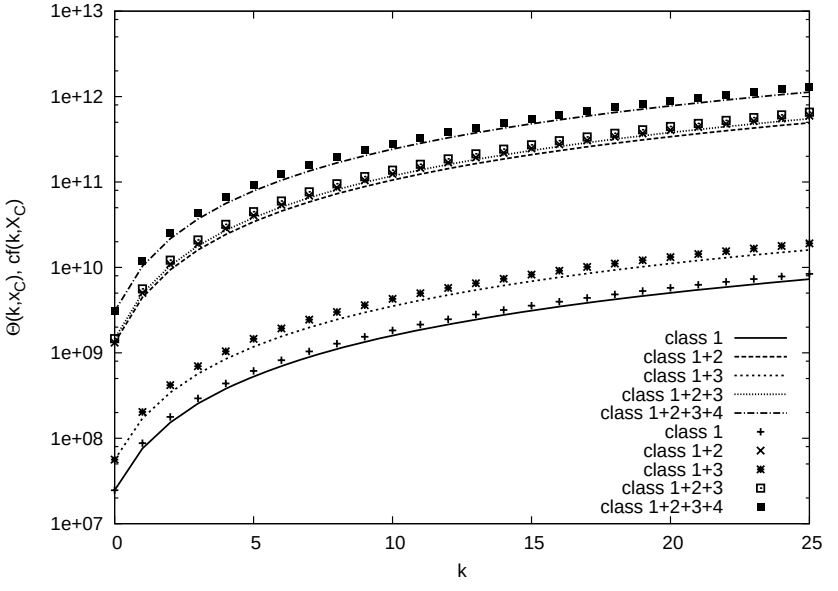


(a) 3 states

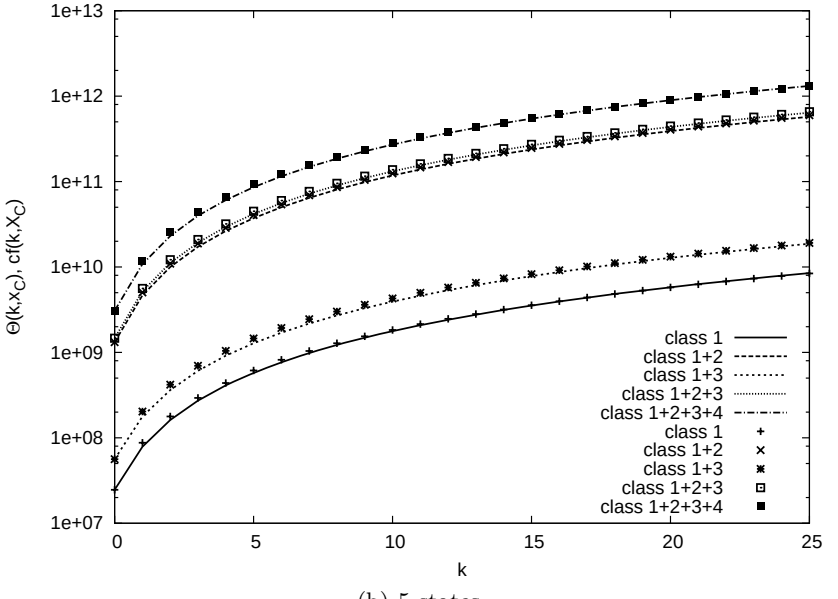


(b) 5 states

Fig. 2. Histogram of various subtraces and the corresponding density functions of the trace "Brotherhood of the Wolf" for different DBMAP state sizes

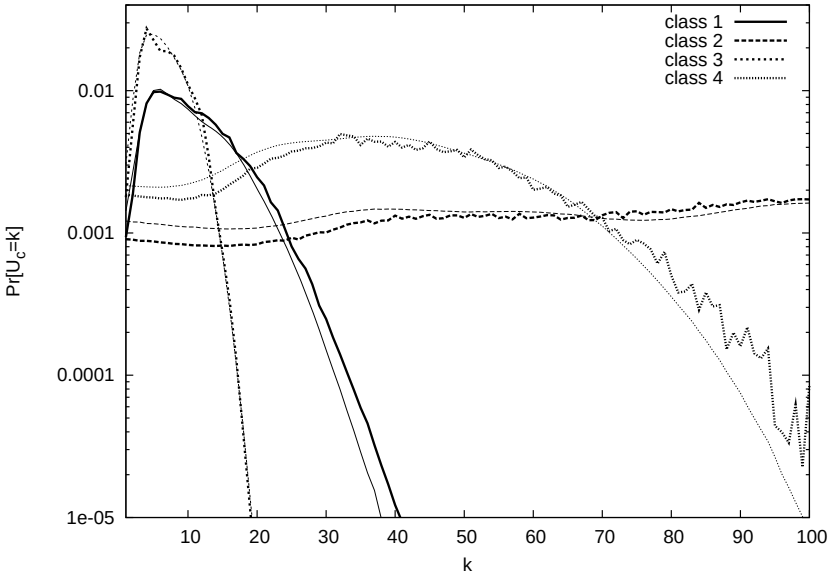


(a) 3 states

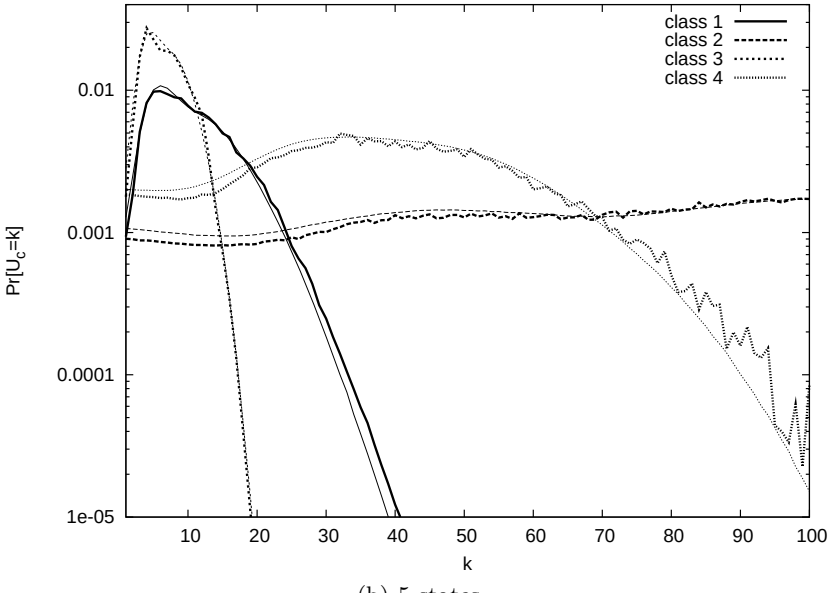


(b) 5 states

Fig. 3. Sample variance functions of various subtraces of the trace "Brotherhood of the Wolf" and the corresponding variance functions for different DBMAP state sizes

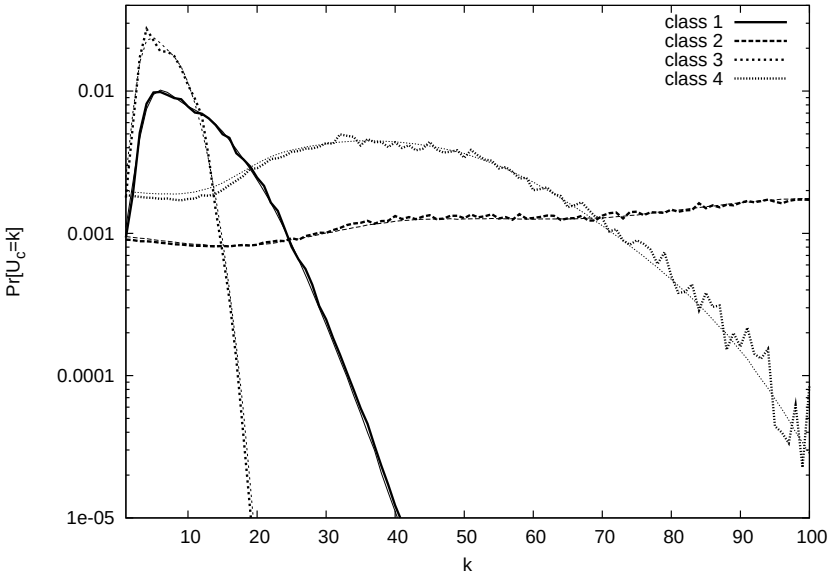


(a) 3 states

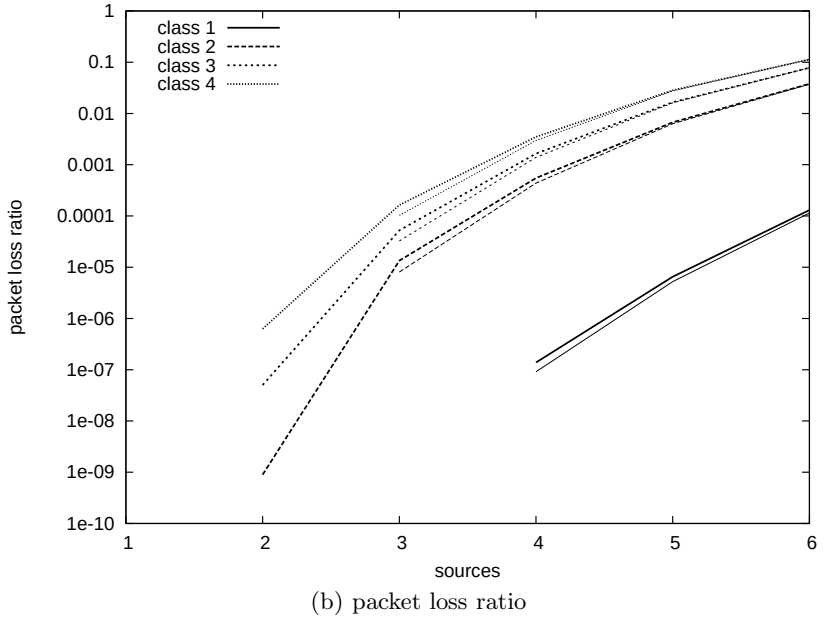


(b) 5 states

Fig. 4. Probability mass function of the queue content of the different classes for a PBS queue with an 8 Mbit output line and aggregated traffic from 4 times "Brotherhood of the Wolf"



(a) pmf of the queue content



(b) packet loss ratio

Fig. 5. Probability mass function of the queue content of the different classes for a PBS queue with an 10 Mbit output line and aggregated traffic from 4 times "Brotherhood of the Wolf" (a) and packet loss ratio versus the number of video sources for a PBS queue with a 12 Mbit output line (b)

The former observation suggests that more Markov states are required to accurately assess the packet loss by means of the Markov model. Hence, we investigate performance of the Markov model for DBMAPs with $K = 10$ states: the pmf of the queue content of the different traffic classes is depicted in Figure 5(a) for the network scenario of Figure 4. Now, an accurate match of the tail distributions is obtained. Finally, the packet loss ratio is depicted versus the number of video sources that are routed through a PBS buffer with a 12 Mbit output line in Figure 5(b). Buffer size and thresholds are the same as in Figure 4. Clearly, a good match is obtained between model based and trace based simulations which shows that the DBMAP can be used to accurately assess the packet loss ratio.

6 Conclusions

In this contribution we proposed the multi-class DBMAP for the characterisation of H.264/SVC scalable video at the sub-GOP level. We showed that — by means of a genetic algorithm — multi-class DBMAPs with a limited state space can be found that accurately capture the characteristics of the video traces. Finally, a simulation study demonstrated that the Markov model can be used to accurately assess the performance of video streaming in networking scenarios.

Acknowledgement

This work has been carried out in the framework of the Q-MATCH project sponsored by the Flemish Institute for the Promotion of Scientific and Technological Research in the Industry (IWT). The first author is a Postdoctoral Fellow with the Research Foundation - Flanders (F.W.O.-Vlaanderen), Belgium.

References

1. Schwartz, H., Marpe, D., Wiegand, T.: Overview of the scalable video coding extension of the H.264/AVC standard. 264/AVC standard. IEEE Transactions on Circuits and Systems for Video Technology 17(9), 1103–1120 (2007)
2. Macfadyen, N.W.: Traffic characterisation and modeling. BT Technology Journal 20(3), 14–30 (2002)
3. Latouche, G., Ramaswami, V.: Introduction to matrix analytic methods in stochastic modeling. Series on statistics and applied probability. ASA-SIAM (1999)
4. Blondia, C., Casals, O.: Statistical multiplexing of VBR sources: A matrix-analytic approach. Performance Evaluation 16, 5–20 (1992)
5. Izquierdo, M.R., Reeves, D.S.: A survey of statistical source models for variable-bit-rate compressed video. Multimedia Systems 7(3), 199–213 (1999)
6. Conti, M., Gregori, E., Larsson, A.: Study of the impact of MPEG-1 correlations on video-sources statistical multiplexing. IEEE Journal of Selected Areas in Communications 14(7), 1455–1471 (1996)
7. Conti, M., Gregori, E.: Modeling MPEG scalable sources. Multimedia Tools And Applications 13(2), 127–145 (2001)

8. Klemm, A., Lindemann, C., Lohmann, M.: Modeling IP traffic using the batch markovian arrival process. Performance Evaluation 54(2), 149–173 (2003)
9. Moltchanov, D., Koucheryavy, Y., Harju, J.: The model of single smoothed MPEG traffic source based on the D-BMAP arrival process with limited state space. In: Proceedings of ICACT 2003, Phoenix Park, South Korea, pp. 57–63 (2003)
10. Zhao, J., Li, B., Ahmad, I.: Traffic model for layered video: an approach on Markovian arrival process. In: Packet Video 2003, Nantes, France (2003)
11. Won, Y., Ahn, S.: GOP ARIMA: Modeling the nonstationarity of VBR processes. Multimedia Systems 10(5), 359–378 (2005)
12. Lazaris, A., Koutsakis, P., Paterakis, M.: On Modeling Video Traffic from Multiplexed MPEG-4 Videoconference Streams. In: Koucheryavy, Y., Harju, J., Iversen, V.B. (eds.) NEW2AN 2006. LNCS, vol. 4003, pp. 46–57. Springer, Heidelberg (2006)
13. Liew, C.H., Kodikara, C.K., Kondoz, A.M.: MPEG-encoded variable bit-rate video traffic modelling. IEE Proceedings-Communications, 152(5), 749–756 (2005)
14. Ansari, N., Liu, H., Shi, Y.Q., Zhao, H.: On modeling MPEG video traffics. IEEE Transactions On Broadcasting 48(4), 337–347 (2002)
15. Kempken, S., Luther, W.: Modeling of H.264 high definition video traffic using discrete-time semi-Markov processes. In: Mason, L.G., Drwiega, T., Yan, J. (eds.) ITC 2007. LNCS, vol. 4516. Springer, Heidelberg (2007)
16. Koza, J.: Survey of genetic algorithms and genetic programming. In: Proceedings of Wescon 1995, San Francisco, CA (1995)

Using the Whittle Estimator for the Selection of an Autocorrelation Function Family

M.E. Sousa-Vieira and A. Suárez-González

Department of Telematics Engineering, University of Vigo, Spain
{estela,asuarez}@det.uvigo.es

Abstract. With the increasing popularity of multimedia applications, video data represents a large portion of the traffic in modern networks. Consequently, adequate models of video traffic, characterized by a high burstiness and a strong positive correlation, are very important for the performance evaluation of network architectures and protocols. This paper presents a method that uses the Whittle estimator to choose, between several models for VBR video traffic based on the M/G/∞ process, the one that gives rise to a better adjustment of the spectral density, and therefore of the correlation structure, of the traffic to model.

Keywords: Video traffic; Autocorrelation; M/G/∞ process; Whittle estimator.

1 Introduction

Several traffic measurement results have convincingly shown the existence of persistent correlations in several kinds of traffic [23,20,3,30,4,7,40,25,18]. These experimental findings stimulated the opening of a new branch in the stochastic modeling of traffic, since the impact of the correlation on the performance metrics may be drastic [21,28,22,10]. The use of classes of stochastic processes for network modeling purposes, that can display forms of correlation as diverse as possible by making use of few parameters (parsimonious modeling) is essential.

VBR video traffic, characterized by a high burstiness and a strong positive correlation, represents an important part of the load in modern networks. So, adequate models for this type of traffic are needed for network design and performance evaluation.

Several studies have been conducted in modeling VBR video traffic, based on different stochastic methods [13,16,19,24,26,27,12,33]. In this paper we focus on the M/G/∞-type processes [5,9]. In essence, the M/G/∞ process is a stationary version of the occupancy process of an M/G/∞ queueing model. In addition to its theoretical simplicity, it can be used to model different traffic sources, because it is flexible enough to exhibit both Short-Range Dependence (SRD) and Long-Range Dependence (LRD). Moreover, queueing analytical studies are sometimes feasible [8,38,32,29], but when they are not, it has important advantages in simulation studies [19,31], such as the possibility of on-line generation and the lower computational cost (exact methods for the generation of a trace

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 16–30, 2008.

of length n require only $\mathcal{O}(n)$ computations, compared to at least $\mathcal{O}(n \log n)$ for other processes able to exhibit LRD such as Fractional AutoRegressive Integrated Moving Average (F-ARIMA) or Fractional Gaussian Noise (FGN) [25]). In [34] we presented a method to improve the efficiency of the generator when the distribution of the service time of the M/G/∞ system has subexponetial decay.

In order to apply a model to the synthetic generation of traces with a correlation structure similar to that of real sequences, a fundamental problem is the estimation of the parameters of the model. Between the methods proposed in the literature [39,1,37], those based on Maximum Likelihood Estimators (MLE), as the Whittle estimator, are especially interesting because they permit to fit the whole spectral density and to obtain confidence intervals of the estimated parameters.

This paper presents a method based on the prediction error of the Whittle estimator to choose, between several models for compressed VBR video traffic based on the M/G/∞ process, the one that gives rise to a better adjustment of the spectral density, and therefore of the correlation structure, of the traffic to model.

The remainder of the paper is organized as follows. We begin reviewing the main concepts related to SRD and LRD in Section 2 and those related to the Whittle estimator in Section 3. The M/G/∞ process and the different models based on it that we use in this study are described in Section 4. In Section 5, we show how the method that we propose allows to select the model that gives rise to the best adjustment of the correlation structure of VBR compressed video sequences. Finally, concluding remarks and guidelines for further work are given in Section 6.

2 SRD and LRD

It is said that a process exhibits SRD when its autocorrelation function is summable (or, equivalently, its spectral density is bounded at the origin) i.e., $\sum_{k=0}^{\infty} r[k] < \infty$, like in those processes whose autocorrelation function decays exponentially:

$$\exists \delta \in (0,1) \left| \lim_{k \to \infty} \frac{r[k]}{\delta^k} = \mathsf{c}_\mathrm{r} \in (0, \infty) \right. \quad .$$

Conversely, it is said that a process exhibits LRD when its autocorrelation function is not summable (its spectral density has a singularity at the origin), i.e., $\sum_{k=1}^{\infty} r[k] = \infty$, like in those processes whose autocorrelation function decays hyperbolically:

$$\exists \alpha \in (0,1) \left| \lim_{k \to \infty} \frac{r[k]}{k^{-\alpha}} = \mathsf{c}_\mathrm{r} \in (0, \infty) \right. \quad .$$

Mathematically, the physical phenomenon of LRD is closely related to the concept of self-similarity [6].

3 Whittle Estimator

Let $f_\theta(\lambda)$ be the spectral density function of a zero-mean Gaussian stochastic process, $X = \{X_n; n = 1, 2, \dots\}$ and let $I_{X^{\mathsf{N}}}(\lambda) = \frac{1}{2\pi\mathsf{N}} \left| \sum_{i=0}^{\mathsf{N}-1} X_{i+1} \mathrm{e}^{-\jmath\lambda i} \right|^2$ be the periodogram from a sample of size N of X. $\theta = \{\theta_k; k = 1, \dots, \mathsf{M}\}$ is the vector of parameters to be estimated.

The approximate Whittle MLE is the vector $\widehat{\theta} = \left\{\widehat{\theta_k}; k = 1, \dots, \mathsf{M}\right\}$ that minimizes, for a given sample x^{N} of size N of X, the statistical:

$$Q_{X^{\mathsf{N}}}(\theta) \triangleq \frac{1}{2\pi} \left[\int_{-\pi}^{\pi} \frac{I_{X^{\mathsf{N}}}(\lambda)}{f_\theta(\lambda)} \mathrm{d}\lambda + \int_{-\pi}^{\pi} \log f_\theta(\lambda) \, \mathrm{d}\lambda \right] \ , \tag{1}$$

i.e., $\widehat{\theta}$ will be the minimizer:

$$Q_{X^{\mathsf{N}}}\left(\widehat{\theta}\right) = \min_{\theta} \left(Q_{X^{\mathsf{N}}}(\theta) \, | X^{\mathsf{N}} = x^{\mathsf{N}} \right) \ .$$

Moreover, if θ^{o} is the real value of θ:

$$\Pr\left[\left| \hat{\theta} - \theta^{\mathrm{o}} \right| < \epsilon \right] \xrightarrow[\mathsf{N}\to\infty]{} 1 \quad \forall \epsilon > 0 \ ,$$

then $\sqrt{\mathsf{N}}(\hat{\theta} - \theta^{\mathrm{o}})$ converges in distribution to ζ, as $\mathsf{N} \to \infty$, where ζ is a zero-mean Gaussian vector with matrix of covariances $\mathrm{C}(\theta^{\mathrm{o}}) = 2\,\mathrm{D}^{-1}(\theta^{\mathrm{o}})$, being:

$$\mathrm{D}_{ij}(\theta^{\mathrm{o}}) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{\partial}{\partial\theta_i} \log f_\theta(\lambda) \frac{\partial}{\partial\theta_j} \log f_\theta(\lambda) \, \mathrm{d}\lambda \Big|_{\theta=\theta^{\mathrm{o}}} \ . \tag{2}$$

So, confidence intervals of the estimated values can be obtained.

In [14] the author proposes to approximate (1) by replacing the integration by a Riemann sum. So, the discrete Whittle estimator is defined as the vector $\widehat{\theta} = \left\{\widehat{\theta_k}; k = 1, \dots, \mathsf{M}\right\}$ that minimizes, for a given sample of size N of X, the function:

$$\widetilde{Q}_{X^{\mathsf{N}}}(\theta) = \frac{4\pi}{\mathsf{N}} \left[\sum_{k=1}^{\mathsf{N}^*} \frac{I_{X^{\mathsf{N}}}(\lambda_k)}{f_\theta(\lambda_k)} + \sum_{k=1}^{\mathsf{N}^*} \log f_\theta(\lambda_k) \right] \ , \tag{3}$$

with $\lambda_k = \frac{2\pi k}{\mathsf{N}}; k = 1, 2, \dots, \mathsf{N}^*$ and $\mathsf{N}^* = \left\lfloor \frac{\mathsf{N}-1}{2} \right\rfloor$.

A simplification of (1) and (3) can be achieved by choosing a special scale parameter θ_1, such that:

$$f_\theta(\lambda) = \theta_1 f_{\theta^*}(\lambda) = \theta_1 f_\eta^*(\lambda) \ ,$$

and:

$$\int_{-\pi}^{\pi} \log f_{\theta^*}(\lambda) \, \mathrm{d}\lambda = \int_{-\pi}^{\pi} \log f_\eta^*(\lambda) \, \mathrm{d}\lambda = 0 \ ,$$

where $\eta = \{\theta_i; i = 1, \dots, \mathsf{M}\}$ and $\theta^* = (1, \eta)$.

Thus:

$$\theta_1 = \exp\left(\frac{1}{2\pi}\int_{-\pi}^{\pi} \log f_\theta(\lambda)\,\mathrm{d}\lambda\right) = \frac{\sigma_\epsilon^2}{2\pi} \ ,$$

where σ_ϵ^2 is the optimal one-step-ahead prediction error, that is equal to the variance of the innovations of the AR(∞) representation of the process [2]:

$$X_i = \sum_{j=1}^{\infty} \beta_j X_{i-j} + \epsilon_i \ .$$

Equation (1) therefore simplifies to:

$$Q_{X^{\mathbb{N}}}(\theta^*) = Q^*_{X^{\mathbb{N}}}(\eta) = \int_{-\pi}^{\pi} \frac{I_{X^{\mathbb{N}}}(\lambda)}{f_{\theta^*}(\lambda)}\mathrm{d}\lambda = \int_{-\pi}^{\pi} \frac{I_{X^{\mathbb{N}}}(\lambda)}{f^*_{\eta}(\lambda)}\mathrm{d}\lambda \ ,$$

and the discrete version:

$$\widetilde{Q}_{X^{\mathbb{N}}}(\theta^*) = \widetilde{Q}^*_{X^{\mathbb{N}}}(\eta) = \frac{4\pi}{\mathsf{N}} \sum_{k=1}^{\mathsf{N}^*} \frac{I_{X^{\mathbb{N}}}(\lambda_k)}{f^*_{\eta}(\lambda_k)} \ .$$

Additionally [2]:

$$\widehat{\sigma_\epsilon^2} = \widetilde{Q}^*_{X^{\mathbb{N}}}(\widehat{\eta}) \ . \tag{4}$$

We propose to use $\widehat{\sigma_\epsilon^2}$ as a measure of the suitability of a model, since smaller values of $\widehat{\sigma_\epsilon^2}$ (or $\widetilde{Q}^*_{X^{\mathbb{N}}}(\widehat{\eta})$) mean better adjustment to the actual correlation of the sample.

4 M/G/∞ Process

The M/G/∞ process [5] is a stationary version of the occupancy process of an M/G/∞ queueing system at any instant t, $\{X(t); t \in \Re^+\}$. We are interested on its discrete-time version, that is, $X \triangleq \left\{X_n \triangleq X(n); n = 1, 2, \ldots\right\}$.

Let λ be the arrival rate to the system, and denote by S the service time distribution. If the initial number of users is a Poisson random variable of mean value $\lambda \mathsf{E}[S]$, and their service times are mutually independent and have the same distribution as the residual life of S, $\widehat{S}$:

$$\Pr\left[\widehat{S} = k\right] = \frac{\Pr[S >= k]}{\mathsf{E}[S]} \ ,$$

then the stochastic process X is strict-sense stationary, ergodic, and enjoys equivalent properties to those of the original continuous-time M/G/∞ process.

In particular it has Poisson marginal distribution with mean value:

$$\mathsf{E}[X] = \lambda \mathsf{E}[S] \ ,$$

and the autocorrelation function is:

$$r[k] = \Pr\left[\widehat{S} > k\right] \quad \forall k \ .$$

So, the autocorrelation structure of X is completely determined by the distribution of S. In particular, the process exhibits LRD when S has infinite variance, as it happens in heavy-tailed distributions.

On the other hand, in [19] the authors show that an $\Re^+$-valued sequence $r[k]$ can be the autocorrelation function of the stationary M/G/∞ process, with integrable S, if and only if it is decreasing and integer-convex, with $r[0] = 1 > r[1]$ and $\lim_{k\to\infty} r[k] = 0$, in which case the probability mass function of S is given by:

$$\Pr[S = k] = \frac{r[k-1] - 2r[k] + r[k+1]}{1 - r[1]} \quad \forall k > 0 \ . \tag{5}$$

Its mean value is:

$$\mathsf{E}[S] = \frac{1}{1 - r[1]} \ .$$

Regarding the application of the Whittle estimator taking the M/G/∞ process as the underlying one, in [35] we checked that the estimator converges quickly as the mean increases (a high mean value of a Poisson random variable means resemblance to a Gaussian density), yielding unbiased point estimates of the unknown parameters.

4.1 M/G/∞-Based Models

We consider different models for traffic correlations based on the M/G/∞ process, able to exhibit SRD or LRD in a parsimonious way.

All the distributions for the service time of the resulting M/G/∞ systems have subexponential decay. Therefore, it is possible to use the method that we proposed in [34] in order to improve the efficiency of the on-line generators of synthetic traces.

First Model (SRD-1). This model adjusts the autocorrelation function of the M/G/∞ process with the function proposed in [19] in order to model the correlation structure of some VBR video sequences:

$$r[k] = r_\beta[k] \stackrel{\Delta}{=} \mathsf{e}^{-\beta\sqrt{k}} \quad \beta > 0 \ .$$

The spectral density can be approximated by:

$$f_X(\lambda) \sim f_\beta(\lambda) \stackrel{\Delta}{=} \mathsf{Var}[X] \left\{ \frac{1}{2\pi} + 2 \sum_{k=1}^{\mathsf{F}} r_\beta[k] \cos(\lambda k) \right\} \ ,$$

if F is chosen so that the error is negligible. The efficiency can be improved by using rational interpolation.

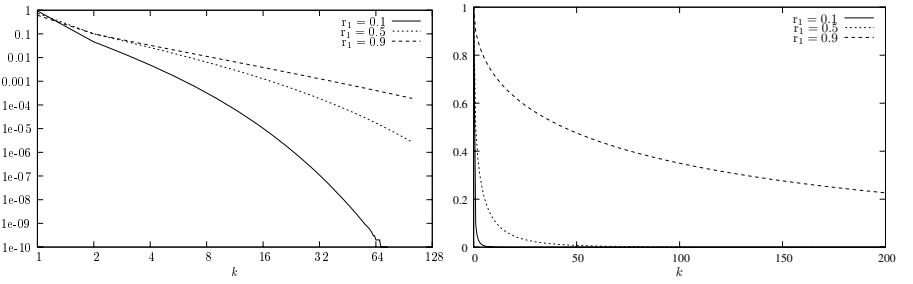


Fig. 1. Probability mass function of K (left) and autocorrelation of the M/K/∞ process (right)

It is easy to show:

$$\sum_{k=0}^{\infty} r_\beta[k] \leq 1 + \frac{1}{\beta^2} \ ,$$

hence this correlation structure yields a SRD process.

Applying (5) one obtains the probability mass function of the service time in a M/G/∞ system generating an occupancy process with such correlation structure. The resulting function is:

$$\Pr[S = k] = \frac{\mathsf{e}^{-\beta\sqrt{k-1}} - 2\mathsf{e}^{-\beta\sqrt{k}} + \mathsf{e}^{-\beta\sqrt{k+1}}}{1 - \mathsf{e}^{-\beta}} \quad \forall k > 0 \ .$$

The mean value is:

$$\mathsf{E}[S] = \frac{1}{1 - \mathsf{e}^{-\beta}} \ .$$

We denote this random variable by K.

The distribution function of the residual life is:

$$\Pr\left[\widehat{S} \leq k\right] = 1 - \mathsf{e}^{-\beta\sqrt{k}} \quad \forall k > 0 \ .$$

In Fig. 1 (left) we show the probability mass function of the service time for several values of the parameter β ($\mathrm{r}_1 = r_\beta[1] = \mathsf{e}^{-\beta}$). And in Fig. 1 (right) we show the autocorrelation function of the resulting M/K/∞ process.

Second Model (LRD-1). This model fixes the autocorrelation function of the resulting M/G/∞ process within the class of FGN processes:

$$r[k] = r_{\mathsf{H}}[k] \stackrel{\Delta}{=} \frac{1}{2}\left[(k+1)^{2\mathsf{H}} - 2k^{2\mathsf{H}} + (k-1)^{2\mathsf{H}}\right] \ ,$$

with H the Hurst parameter [17].

The spectral density:

$$f_X(\lambda) = f_{\mathsf{H}}(\lambda) \stackrel{\Delta}{=} \mathsf{c}_f \left|\mathsf{e}^{\jmath\lambda} - 1\right|^2 \sum_{i=-\infty}^{+\infty} |2\pi i + \lambda|^{-2\mathsf{H}-1} \forall \lambda \in [-\pi, \pi] \ ,$$

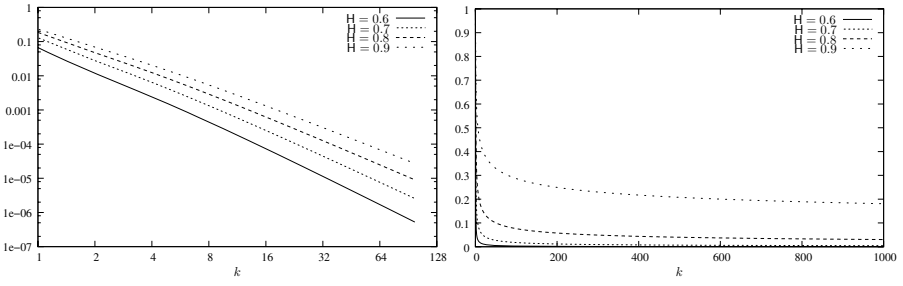


Fig. 2. Probability mass function of F (left) and autocorrelation function of the M/F/∞ process (right)

can be computed efficiently with Euler's formula [15].

For $0.5 < \mathsf{H} < 1$ the process is LRD.

From (5) the probability mass function of the service time in a M/G/∞ system generating an occupancy process with such correlation function is:

$$\Pr[S = k] = \frac{0.5(k-2)^{2\mathsf{H}} - 2(k-1)^{2\mathsf{H}} + 3k^{2\mathsf{H}} - 2(k+1)^{2\mathsf{H}} + 0.5(k+2)^{2\mathsf{H}}}{2 - 2^{2\mathsf{H}-1}} \quad \forall k > 0 \ .$$

The mean value is:

$$\mathsf{E}[S] = \frac{1}{2 - 2^{2\mathsf{H}-1}} \ .$$

We denote this random variable by F.

The distribution function of the residual life is:

$$\Pr\left[\widehat{S} \leq k\right] = 1 - 0.5\left[(k+1)^{2\mathsf{H}} - 2k^{2\mathsf{H}} + (k-1)^{2\mathsf{H}}\right] \quad \forall k > 0 \ .$$

In Fig. 2 (left) we show the probability mass function of the service time for several values of the parameter H. And in Fig. 2 (right) we show the autocorrelation function of the resulting M/F/∞ process.

Third Model (SRD-2). This model adjusts the autocorrelation function with the subexponential function:

$$r[k] = r_{\{\delta,\gamma\}} = \delta^{k^\gamma} \quad 0 < \delta < 1 \quad 0 < \gamma < 1 \ .$$

The spectral density can be approximated by:

$$f_X(\lambda) \sim f_{\{\delta,\gamma\}}(\lambda) \triangleq \mathsf{Var}[X]\left\{\frac{1}{2\pi} + 2\sum_{k=1}^{\mathsf{F}} r_{\{\delta,\gamma\}}[k]\cos(\lambda k)\right\} \ ,$$

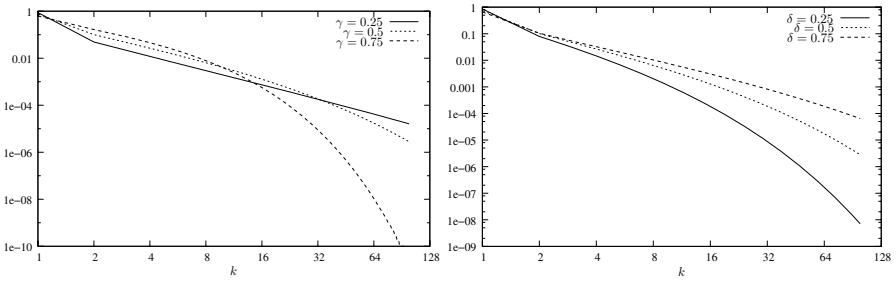


Fig. 3. Probability mass function of D for $\delta = 0.5$ (left) and $\gamma = 0.5$ (right)

with F chosen to make the error negligible. As before, rational interpolation can be used for better efficiency. Note that the first model is a special case of this with $\delta = \mathsf{e}^{-\beta}$ and $\gamma = 0.5$.

Since:

$$\sum_{k=0}^{\infty} r_{\{\delta,\gamma\}}[k] \sim \left(\gamma^{-1}\right)! \left(-\ln(\delta)\right)^{\gamma^{-1}} \quad ,$$

this correlation structure is that of a SRD process.

Once again, using (5) to obtain the distribution function of the equivalent M/G/∞ system, the result is:

$$\Pr[S = k] = \frac{\delta^{(k-1)^\gamma} - 2\delta^{k^\gamma} + \delta^{(k+1)^\gamma}}{1-\delta} \quad \forall k > 0 \ .$$

The mean value is:

$$\mathsf{E}[S] = \frac{1}{1-\delta} \ .$$

We denote this random variable by D.

The distribution function of the residual life is:

$$\Pr\left[\widehat{S} \leq k\right] = 1 - \delta^{k^\gamma} \quad \forall k > 0.$$

In Fig. 3 we show the probability mass function of the service time for several values of the parameters δ and γ.

And in Fig. 4 we show the autocorrelation function of the resulting M/D/∞ process.

Fourth Model (LRD-2). Now, the model picks the S distribution proposed in [36] as the service time of the M/G/∞ queueing system.

Its main characteristic is that of being a heavy-tailed distribution with two parameters, m and α, a feature that allows to model simultaneously the short-term correlation behavior (by means for example of the one-lag autocorrelation coefficient $r[1]$) and the long-term correlation behavior (by means of the Hurst parameter) of the M/G/∞ process.

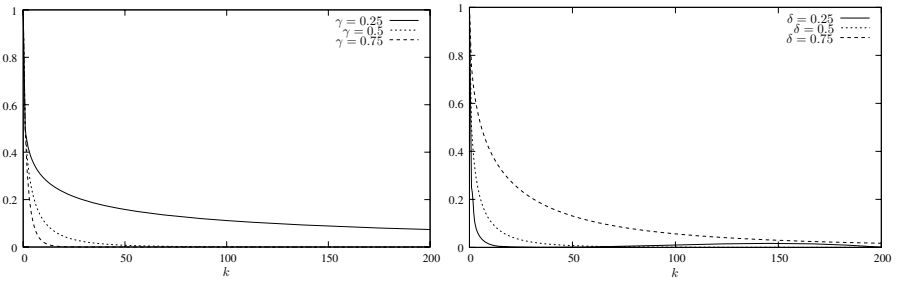


Fig. 4. Autocorrelation function of the M/D/∞ process for $\delta = 0.5$ (left) and $\gamma = 0.5$ (right)

Depending on the value of the parameter m, the mass probability function is, for $\mathsf{m} \leq 1$:

$$\Pr[S = k] = \begin{cases} 1 + \dfrac{\mathsf{m}^\alpha}{\alpha\mathsf{m} - \mathsf{m}^\alpha}\left[(k+1)^{1-\alpha} - k^{1-\alpha}\right] & k = 1 \\ \dfrac{\mathsf{m}^\alpha}{\alpha\mathsf{m} - \mathsf{m}^\alpha}\left[(k+1)^{1-\alpha} - 2k^{1-\alpha} + (k-1)^{1-\alpha}\right] & \forall k > 1 \ , \end{cases}$$

and for $\mathsf{m} > 1$:

$$\Pr[S = k] = \begin{cases} 1 + k - \mathsf{m} + \dfrac{\mathsf{m}^\alpha}{\alpha - 1}\left[(k+1)^{1-\alpha} - \mathsf{m}^{1-\alpha}\right] & k = \lfloor \mathsf{m} \rfloor \\ 1 + \mathsf{m} - k + \dfrac{\mathsf{m}^\alpha}{\alpha - 1}\left[(k+1)^{1-\alpha} - 2k^{1-\alpha} + \mathsf{m}^{1-\alpha}\right] & k = \lceil \mathsf{m} \rceil \\ \dfrac{\mathsf{m}^\alpha}{\alpha - 1}\left[(k+1)^{1-\alpha} - 2k^{1-\alpha} + (k-1)^{1-\alpha}\right] & \forall k > \lceil \mathsf{m} \rceil \ . \end{cases}$$

The mean value is:

$$\mathsf{E}[S] = \begin{cases} \dfrac{\alpha\mathsf{m}}{\alpha\mathsf{m} - \mathsf{m}^\alpha} & \forall \mathsf{m} \in (0, 1] \\ \dfrac{\alpha\mathsf{m}}{\alpha - 1} & \forall \mathsf{m} \geq 1 \ . \end{cases}$$

The distribution function of the residual life is:

$$\Pr\left[\widehat{S} \leq k\right] = \begin{cases} \dfrac{\alpha - 1}{\alpha\mathsf{m}} k & \forall k \leq \mathsf{m} \\ 1 - \dfrac{1}{\alpha}\left(\dfrac{\mathsf{m}}{k}\right)^{\alpha - 1} & \forall k \geq \mathsf{m} \ . \end{cases}$$

The autocorrelation function is:

$$r[k] = r_{\{\mathsf{m},\alpha\}}[k] \stackrel{\Delta}{=} \begin{cases} 1 - \dfrac{\alpha - 1}{\mathsf{m}\alpha} k & \forall k \in (0, \mathsf{m}] \\ \dfrac{1}{\alpha}\left(\dfrac{\mathsf{m}}{k}\right)^{\alpha - 1} & \forall k \geq \mathsf{m} \ . \end{cases}$$

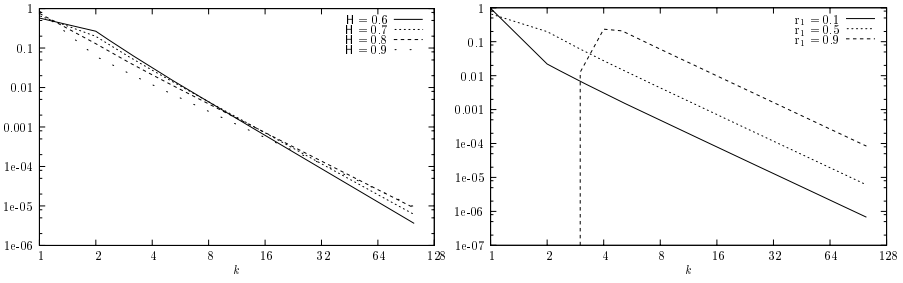


Fig. 5. Probability mass function of S for $r_1 = 0.5$ (left) and $\mathsf{H} = 0.7$ (right)

with:

$$\alpha = 3 - 2\mathsf{H} \ ,$$

$$\mathsf{m} = \begin{cases} (\alpha r[1])^{\frac{1}{\alpha-1}} & \forall r[1] \in \left(0, \frac{1}{\alpha}\right] \\ \dfrac{\alpha - 1}{\alpha - \alpha r[1]} & \forall r[1] \in \left[\frac{1}{\alpha}, 1\right) \end{cases} \ .$$

Finally, the spectral density is given by [35]:

$$f_X(\lambda) = f_{\{\mathsf{m},\alpha\}}(\lambda) \stackrel{\Delta}{=} \mathsf{Var}[X] \left\{ \frac{\mathsf{m}^{\alpha-1}}{\alpha\cos(\lambda) - \alpha} f_{\mathsf{h}}(\lambda) + \frac{1}{2\pi} \right.$$
$$\left. + \frac{1}{\pi} \sum_{k=1}^{\lfloor \mathsf{m} \rfloor} \cos(\lambda k) \left[\frac{\alpha(m-k)+k}{\mathsf{m}\alpha} - \frac{1}{\alpha} \left(\frac{\mathsf{m}}{k}\right)^{\alpha-1} \right] \right\} \ ,$$

where f_{h} is the spectral density of a FGN process with $\mathsf{h} = \frac{(1-\alpha)}{2}$.

If $\alpha \in (1, 2)$, then $\mathsf{H} \in (0.5, 1)$, and $\sum_{k=0}^{\infty} r_{\{\mathsf{m},\alpha\}}[k] = \infty$. Hence, in this case, this correlation structure gives rise to an LRD process.

In Fig. 5 we show the probability mass function of the service time for several values of the parameters H and $r_1 = r_{\{\mathsf{m},\alpha\}}[1]$.

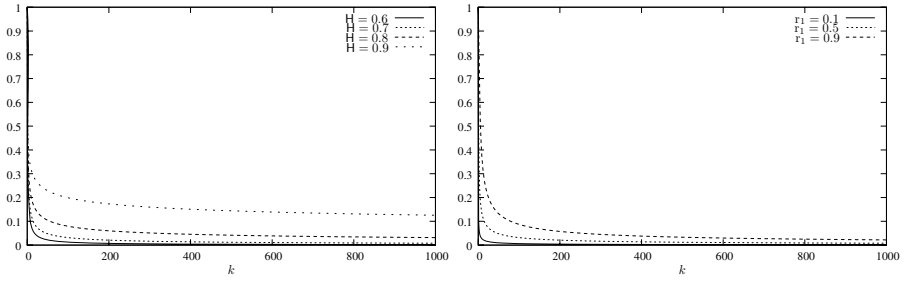


Fig. 6. Autocorrelation function of the M/S/∞ process for $r_1 = 0.5$ (left) and $\mathsf{H} = 0.7$ (right)

And in Fig. 6 we show the autocorrelation function of the resulting M/S/∞ process.

5 Selection of the Best Model

In this section we explain, by means of two examples, how to use the Whittle estimator to build the model that fits best the autocorrelation function measured from empirical VBR traces.

We consider the following empirical traces of the Group of Pictures (GoP) sizes of MPEG encoded videos:

- T-1: *"IceAge"*; length $\mathsf{N} = 9000$.
- T-2: *"The Lord of the Rings"*; length $\mathsf{N} = 77400$.

T-1 is available in [11] and we generated T-2 from the three parts of the trilogy "The Lord of the Rings" in order to obtain a sequence of high length.

As the marginal distribution in both cases is approximately Lognormal, we apply a change of distribution to make the marginal statistics of the empirical traces approximately Gaussian, and fit the mean and variance to a large enough value.

After the change of distribution the estimations of the parameters of each model, computed via the Whittle estimator, are as follows:

- T-1:
  - Model SRD-1: $\widehat{\beta} = 0.442$.
  - Model LRD-1: $\widehat{\mathsf{H}} = 0.879$.
  - Model SRD-2: $\widehat{\delta} = 0.643$ and $\widehat{\gamma} = 0.483$.
  - Model LRD-2: $\widehat{\mathsf{m}} = 0.586$ and $\widehat{\alpha} = 1.253$.
- T-2:
  - Model SRD-1: $\widehat{\beta} = 0.677$.
  - Model LRD-1: $\widehat{\mathsf{H}} = 0.741$.
  - Model SRD-2: $\widehat{\delta} = 0.412$ and $\widehat{\gamma} = 0.223$.
  - Model LRD-2: $\widehat{\mathsf{m}} = 0.013$ and $\widehat{\alpha} = 1.097$.

The variance (or confidence intervals) of the estimations can be computed from (2).

In Table 1 we show the estimations of the prediction error, $\widehat{\sigma_\epsilon^2}$.

The results show that increasing the number of parametres leads to smaller prediction errors (better fit of the spectral density) within the SRD or LRD family of models. It is striking that the LRD-1 model fits the sequence T-1 better than the SRD-2 one, even though it uses one parameter less.

In Fig. 7 we represent the adjustment of the correlation structure of each trace with that of a synthetic trace of each model, after the inverse change of distribution.

Table 1. Estimations of the prediction error with each model

Model	T-1	T-2
SRD-1	5004.351	1253745.941
LRD-1	4983.254	1208692.789
SRD-2	5003.814	1204010.758
LRD-2	4979.428	1188819.675

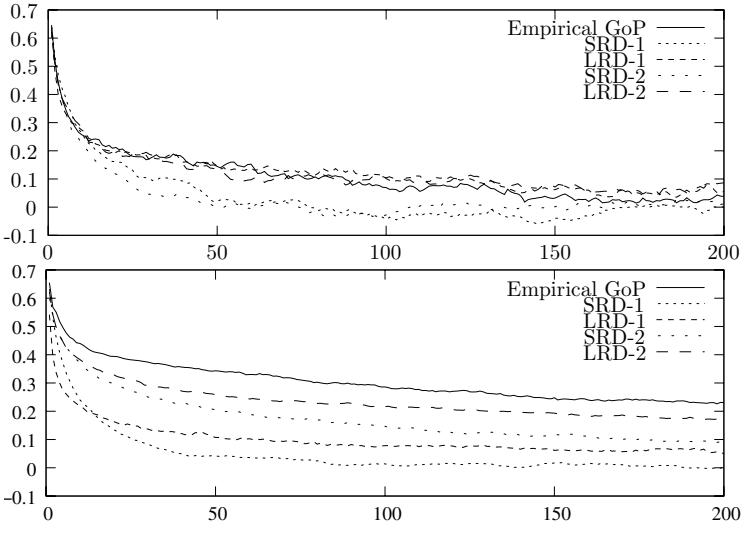


Fig. 7. Adjustment of the autocorrelation function with each model. T-1 (top) and T-2 (bottom).

5.1 Hypothesis Test over the Spectral Density

In this section we propose an hypothesis test over the spectral density, based on the prediction error of the Whittle estimator, to decide between two models (for example LRD-1 and LRD-2).

We consider the null hypothesis:

$$H_0 : f\left(\lambda\right) = f_{\mathsf{H}}\left(\lambda\right) \quad ,$$

and the alternative one:

$$H_1 : f\left(\lambda\right) = f_{\{\mathsf{m},\alpha\}}\left(\lambda\right) \quad ,$$

We define the test statistic as the difference between the estimation of the prediction error when the null hypothesis is supposed and the estimation of the prediction error when the alternative hypothesis is supposed divided by the sample variance:

$$E \triangleq \frac{\widehat{\sigma_\epsilon^2}(H_0) - \widehat{\sigma_\epsilon^2}(H_1)}{\widehat{\sigma}^2} \quad .$$

We generate a batch of 500 synthetic traces of different sizes ($\mathsf{N} = 8192$, $\mathsf{N} = 32768$ and $\mathsf{N} = 131072$) of the M/F/∞ process for each one of three values of the Hurst parameter:

$$\mathsf{H} \in \{0.6, 0.75, 0.9\}$$

and we calculate the value of the statistical.

The critical region, W, for a significance level of 0.05 for each value of H is:

- $\mathsf{N} = 8192$
  - $W_{\mathsf{H}=0.6}$=$(5.081 \cdot 10^{-3},\infty)$
  - $W_{\mathsf{H}=0.75}$=$(4.521 \cdot 10^{-4},\infty)$
  - $W_{\mathsf{H}=0.9}$=$(1.702 \cdot 10^{-4},\infty)$
- $\mathsf{N} = 32768$
  - $W_{\mathsf{H}=0.6}$=$(1.521 \cdot 10^{-4},\infty)$
  - $W_{\mathsf{H}=0.75}$=$(3.432 \cdot 10^{-5},\infty)$
  - $W_{\mathsf{H}=0.9}$=$(9.985 \cdot 10^{-6},\infty)$
- $\mathsf{N} = 131072$
  - $W_{\mathsf{H}=0.6}$=$(-1.123 \cdot 10^{-5},\infty)$
  - $W_{\mathsf{H}=0.75}$=$(-8.543 \cdot 10^{-5},\infty)$
  - $W_{\mathsf{H}=0.9}$=$(-5.743 \cdot 10^{-5},\infty)$

For each one of the empirical traces we estimate the Hurst parameter assuming the null hypothesis and interpolate the border of the critical region in order to check if the null hypothesis is rejected:

- T-1: $\widehat{E} = 4.182 \cdot 10^{-4}$ and the interpolation of the critical region is:

  $$W_{\widehat{\mathsf{H}}=0.879} = (-5.103 \cdot 10^{-5}, \infty)$$

  so the null hypothesis is rejected.
- T-2: $\widehat{E} = 1.072 \cdot 10^{-2}$ and the interpolation of the critical region is:

  $$W_{\widehat{\mathsf{H}}=0.741} = (-8.364 \cdot 10^{-5}, \infty)$$

  so the null hypothesis is rejected.

6 Conclusions and Further Work

In this paper, we have proposed a method to choose, between several models for VBR video traffic based on the M/G/∞ process, the one that gives rise to a better adjustment of the spectral density, and therefore of the correlation structure, of the traffic to model.

All the models enjoy several interesting features: the possibility of on-line generation, a highly efficient simulation model of the resulting M/G/∞ models and the possibility to capture the correlation structure in a parsimonious way.

We have applied the Whittle estimator in order to estimate the parameters of each model and proposed to use the estimation of the prediction error as a measure of the suitability of each model.

Now, we are working on extending the hypothesis test to other combinations of models.

References

1. Abry, P., Veitch, D.: Wavelet analysis of Long-Range Dependent traffic. IEEE Transactions on Information Theory 44(1), 2–15 (1998)
2. Beran, J.: Statistics for Long-Memory Processes. Chapman and Hall, Boca Raton (1994)
3. Beran, J., Shreman, R., Taqqu, M.S., Willinger, W.: Long-Range Dependence in Variable-Bit-Rate video traffic. IEEE Transactions on Communications 43(2/4), 1566–1579 (1995)
4. Conti, M., Gregori, E., Larsson, A.: Study of the impact of MPEG-1 correlations on video sources statistical multiplexing. IEEE Journal on Selected Areas in Communications 14(7), 1455–1471 (1996)
5. Cox, D.R., Isham, V.: Point Processes. Chapman and Hall, Boca Raton (1980)
6. Cox, D.R.: Long-Range Dependence: A review. Statistics: An Appraisal, pp. 55–74. Lowa State University Press (1984)
7. Crovella, M.E., Bestavros, A.: Self-similarity in World Wide Web traffic: Evidence and possible causes. IEEE/ACM Transactions on Networking 5(6), 835–846 (1997)
8. Duffield, N.: Queueing at large resources driven by long-tailed M/G/∞ processes. Queueing Systems 28(1/3), 245–266 (1987)
9. Eliazar, I.: The M/G/∞ system revisited: Finiteness, summability, Long-Range Dependence and reverse engineering. Queueing Systems 55(1), 71–82 (2007)
10. Erramilli, A., Narayan, O., Willinger, W.: Experimental queueing analysis with Long-Range Dependent packet traffic. IEEE/ACM Transactions on Networking 4(2), 209–223 (1996)
11. Fitzek, F.H.P., Reisslein, M.: MPEG-4 and H. 263 video traces for network performance evaluation. IEEE Network 15(6), 40–54 (2001)
12. Frey, M., Nguyen-Quang, S.: A Gamma-based framework for modeling Variable-Bit-Rate MPEG video sources: The GOP GBAR model. IEEE/ACM Transactions on Networking 8(6), 710–719 (2000)
13. Garrett, M.W., Willinger, W.: Analysis, modeling and generation of self-similar VBR video traffic. In: Proc. ACM SIGCOMM 1994, London, UK, pp. 269–280 (1994)
14. Graf, H.: Long-range correlations and estimation of the self-similarity parameter. Ph.D. dissertation, ETH, Zurich, Switzerland (1983)
15. Graham, R.L., Knuth, D.E., Patashnik, O.: Concrete Mathematics. Addison-Wesley, Reading (1994)
16. Huang, C., Devetsikiotis, M., Lambadaris, I., Kaye, A.R.: Modeling and simulation of self-similar Variable-Bit-Rate compressed video: A unified approach. In: Proc. ACM SIGCOMM 1995, Cambridge, UK, pp. 114–125 (1995)
17. Hurst, H.E.: Long-term storage capacity of reservoirs. Transactions of the American Society of Civil Engineers 116, 770–799 (1951)
18. Jiang, M., Nikolic, M., Hardy, S., Trajkovic, L.: Impact of self-similarity on wireless data network performance. In: Proc. IEEE ICC 2001, Helsinki, Finland, pp. 477–481 (2001)
19. Krunz, M., Makowski, A.: Modeling video traffic using M/G/∞ input processes: A compromise between markovian and LRD models. IEEE Journal on Selected Areas in Communications 16(5), 733–748 (1998)
20. Leland, W.E., Taqqu, M.S., Willinger, W., Wilson, D.V.: On the self-similar nature of Ethernet traffic (extended version). IEEE/ACM Transactions on Networking 2(1), 1–15 (1994)

21. Li, S.Q., Hwang, C.L.: Queue response to input correlation functions: Discrete spectral analysis. IEEE/ACM Transactions on Networking 1(5), 317–329 (1993)
22. Likhanov, N., Tsybakov, B., Georganas, N.D.: Analysis of an ATM buffer with self-similar ("fractal") input traffic. In: Proc. IEEE INFOCOM 1995, Boston, MA, EEUU, pp. 985–992 (1995)
23. Livny, M., Melamed, B., Tsiolis, A.K.: The impact of autocorrelation on queueing systems. Management Science 39(3), 322–339 (1993)
24. Lombardo, A., Morabito, G., Schembra, G.: An accurate and treatable Markov model of MPEG video traffic. In: Proc. IEEE INFOCOM 1998 San Francisco, CA, EEUU, pp. 217–224 (1998)
25. López, J.C., López, C., Suárez, A., Fernández, M., Rodríguez, R.F.: On the use of self-similar processes in network simulation. ACM Transactions on Modeling and Computer Simulation 10(2), 125–151 (2000)
26. Ma, S., Ji, C.: Modeling video traffic using wavelets. IEEE Communications Letters 2(4), 100–103 (1998)
27. Melamed, B., Pendarakis, D.E.: Modeling full-length VBR video using Markov-renewal-modulated TES models. IEEE Journal on Selected Areas in Communications 16(5), 600–611 (1998)
28. Norros, I.: A storage model with self-similar input. Queueing Systems 16, 387–396 (1994)
29. Parulekar, M.: Buffer engineering for M/G/∞ input processes. Ph.D. Thesis, University of Maryland, College Park, MD, EEUU (2001)
30. Paxson, V., Floyd, S.: Wide-area traffic: The failure of Poisson modeling. IEEE/ACM Transactions on Networking 3(3), 226–244 (1995)
31. Poon, W., Lo, K.: A refined version of M/G/∞ processes for modeling VBR video traffic. Computer Communications 24(11), 1105–1114 (2001)
32. Resnick, S., Rootzen, H.: Self-similar communication models and very heavy tails. Annals of Applied Probability 10(3), 753–778 (2000)
33. Sarkar, U.K., Ramakrishnan, S., Sarkar, D.: Modeling full-length video using Markov-modulated Gamma-based framework. IEEE/ACM Transactions on Networking 11(4), 638–649 (2003)
34. Sousa, M.E., Suárez, A., Fernández, M., López, C., Rodríguez, R.F.: A highly efficient M/G/∞ generator of self-similar traces. In: Proc. 2006 Winter Simulation Conference, Monterey, CA, EEUU, pp. 2146–2153 (2006)
35. Sousa, M.E., Suárez, A., López, J.C., López, C., Fernández, M., Rodríguez, R.F.: Application of the Whittle estimator to the modeling of traffic based on the M/G/∞ process. IEEE Communications Letters 11(10), 817–819 (2007)
36. Suárez, A., López, J.C., López, C., Fernández, M., Rodríguez, R.F., Sousa, M.E.: A new heavy-tailed discrete distribution for LRD M/G/∞ sample generation. Performance Evaluation 47(2/3), 197–219 (2002)
37. Taqqu, M.S., Teverovsky, V.: On estimating the intensity of Long-Range Dependence in finite and infinite variance time series. In: A Practical Guide to Heavy Tails, pp. 177–218. Birkhauser, Basel (1998)
38. Tsoukatos, K.P., Makowski, A.M.: Heavy traffic analysis for a multiplexer driven by M/G/∞ input processes. In: Proc. 15th International Teletraffic Congress, Washington, DC, EEUU, pp. 497–506 (1997)
39. Whittle, P.: Estimation and information in stationary time series. Arkiv Matematick 2(23), 423–434 (1953)
40. Willinger, W., Taqqu, M.S., Sherman, R., Wilson, D.V.: Self-similarity through high variability: Statistical analysis of Ethernet LAN traffic at the source level. IEEE/ACM Transactions on Networking 5(1), 71–86 (1997)

Using Load Transformations to Predict the Impact of Packet Fragmentation and Losses on Markovian Arrival Processes

Stephan Heckmüller and Bernd E. Wolfinger

Dept. of Computer Science, TKRN, University of Hamburg
Vogt-Kölln-Str. 30, D-22527 Hamburg, Germany
{heckmueller,wolfinger}@informatik.uni-hamburg.de

Abstract. Load transformation can be used to derive valid load models for secondary loads in computer networks, as they can be observed at lower layer interfaces in protocol stacks. In this paper we apply the technique of load transformation onto load represented by BMAP (Batch Markov Arrival Process) models and choose as concrete transformations the following two practically relevant classes of load transforming processes in packet-switched networks: "Fragmentation" and "Losses of Packets". Comparing the analytical characteristics of load after transformation with corresponding load predictions based on load transformation by simulation, we are able to show that realistic modeling is feasible for both classes of transformations considered. We also illustrate the usefulness of the analytical models elaborated by deriving a detailed prediction of secondary load characteristics which, in turn, can be used to specifically optimize performance measures (such as network throughput).

1 Introduction

The strongly increasing relevance of applications with real-time requirements in current computer networks (e.g. voice communications via mobile networks, VoIP & IPTV in the Internet) increases the demand for quality-of-service (QoS) and performance evaluation studies. It is well-known that valid load models are indispensable prerequisites in order to obtain realistic QoS and performance predictions based on simulation or analytical models.

Load in a computer network can be defined as a sequence of requests (cf. Section 2 for details) which is offered by a service user to a service provider at a network internal interface. Evidently, load can be observed and modeled at various interfaces (e.g. video stream to be transmitted, observed at an application-oriented interface, or packets to be transferred, observed at a TCP/IP interface). In order to derive load models for a given interface IF one can directly observe/measure real load at IF and then use these measurements for some *direct load modeling*. As an important alternative to direct modeling also an *indirect load modeling* by means of load transformation [WZHB02] is possible: Indirect load modeling assumes that a load model is needed for an interface IF_S (so-called *secondary load*

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 31–46, 2008.

(SL) interface) for which direct load modeling is very difficult or infeasible, e.g. because load measurements for IF_S are hard or impossible to perform. The approach of load modeling based on load transformation assumes that a valid load model for a "higher" layer interface IF_P (so-called *primary load (PL) interface*) is available – higher in terms of the protocol stack, i.e. an interface closer to the end-user or the application. Moreover, it is assumed in indirect load modeling that the functionality of the load transformation process is well-known, which transforms the requests of PL (e.g. by manipulating the requests according to the protocols) and which creates a corresponding sequence of requests of SL observable at IF_S (i.e. the secondary load which is induced by PL).

The concept of load transformation has been successfully applied to solve a number of problems: In [WZHB02] the distribution of packet lengths for fragmented loads was derived. The authors applied these results to $MPEG$ video streams. In [HW07a] analytical transformations based on $BMAPs$ were given for fragmentation and sliding window procedures. Additionally, we proposed analytical transformations based on $BMAPs$ with which we are able to incorporate the influence of loss processes occuring at wireless links [HW07b]. Besides their direct use for load modeling these results can be used to compute an optimal maximum fragment length with respect to throughput maximization.

Moreover, the concept of load transformation is closely related to that of departure processes, where the transformed load is described by means of process specifications. In [ZHS05] the authors give approximative descriptions of the departure process from a $BMAP/MAP/1$-Queue. The authors assume infinite capacity of the queue and give approximations, whose complexity depends on the lag up to which the correlation of the queue length is captured.

Batch Markovian Arrival Processes (BMAP) were first introduced with alternative notation in [Neu79]. The author uses the term *versatile Markovian Point Process*. BMAPs in the formulation used in this paper were introduced in [Luc91]. Since then considerable effort has been made to investigate *Batch Markovian Arrival Processes*. Recent works include for example the blocking probability of a $BMAP/G/1$ Queueing System [CW06], the departure process from a $BMAP/G/1$ Queue [FC01] or the workload distribution for systems with threshold [LB06]. Here the server is assumed to stay idle until workload, i.e. the number of requests waiting for service, exceeds a certain threshold. Other works extend the $BMAP/G/1$ Queueing System: In [Hof01] the $BMAP/G/1$ Queueing System is extended in order to allow the arrival stream to depend on the system's state, i.e. on the number of customers in the system.

In [KLL03] BMAPs are used to model network traffic at IP-Level. The authors give procedures to match model parameters with measured data. Several modeling techniques for video traffic using $BMAP$s were proposed. In [HSB04] video traffic is modeled by discrete $BMAP$s in order to investigate the queueing behaviour. The authors use the results for buffer dimensioning.

This article is structured as follows: In section 2 we present the concept of load transformation and illustrate how this concept can be used to describe alteration of load (based on BMAPs) in computer networks. We recapitulate our results

concerning fragmentation processes and present new results on the distribution of fragmented packet lengths in section 2.2. In section 3 we propose a model for load transformations as induced by packet losses in wireless networks. We show that the resulting secondary load has the desired characteristics, e.g. in terms of timing and burstiness, and can be used for parameter tuning and for performance evaluation as well. Finally we give conclusions in section 4.

2 Load Transformations

Within networks of service stations the load offered to a single station is in general influenced by the service offered by other stations. This is especially true for modern computer networks, which are organized in layers and interconnected systems. Here, each preceding station influences the load offered to its successor and by that transforms, in our terminology, *primary load* into *secondary load*. (The resulting sequence of transformations is illustrated exemplarily in figure 1 for the transmission over a wireless link.)

In order to describe these transformations in a formally rigorous manner, we define load as shown in Definition 1 [Wol99].

Definition 1. Load $L = L(E, S, IF, T)$ *is defined as the sequence of requests, which are provided to the service system* S *during interval* T *by its environment* E*. Requests are passed via interface* IF*, which separates the service system from its environment.* ◊

The load L can be described by a sequence of arriving requests r_i arriving during time interval T. For a well defined load L we define the arrival process as tuples

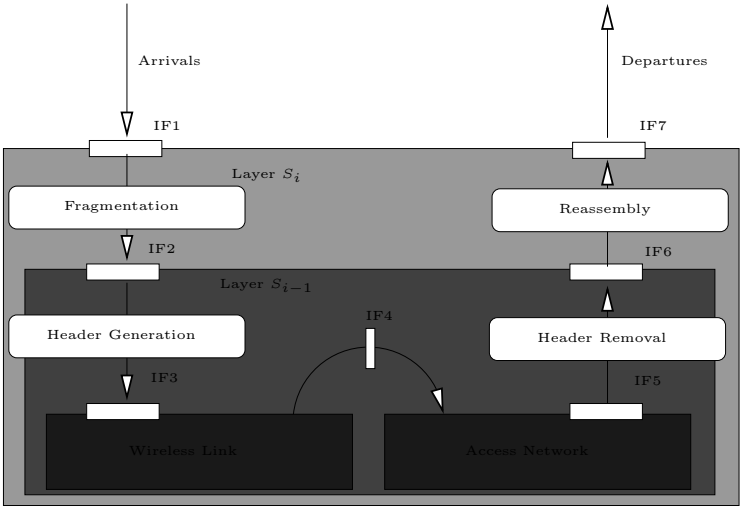


Fig. 1. Packet Transmission as a Sequence of Transformations

of request r_i and corresponding arrival time t_i

$$\{(r_i, t_i) | r_i \in \mathcal{R}_i, t_1 \leq t_2 \leq ... \leq t_N, t_{1,...,N} \in T\} \tag{1}$$

Single requests r_i could for instance represent data transmission requests or connection setup requests. The concept presented here is not limited to the modeling of computer networks. Considering database systems we could for instance model transactions as requests. Moreover to specify requests r_i we additionally introduce *request types* (e.g. packet transmission request) and *request attributes* (e.g. packet length or destination address). Based on Definition 1 we define four types of transformations

1. We define request transformation as a function T_R, which maps the sequence of primary load requests $R^p = (r_1^p, ..., r_N^p)$ to the sequence of secondary load requests $R^s = (r_1^s, ..., r_K^s)$ for a given transformation.
$$T_R : R^p \rightarrow R^s$$
2. Transformation of timing is defined as a function mapping interarrival times of primary load $T^p = (t_1^p, ..., t_N^p)$ to interarrival times of secondary load $T^s = (t_1^s, ..., t_K^s)$
$$T_T : T^p \rightarrow T^s$$
3. If the transformation of request attributes cannot be achieved independently of timing, we additionally use $T'_R : R^p \times T^p \rightarrow R^s$
4. Likewise, if the transformation of timing cannot be achieved independently of request attributes, we use $T'_T : R^p \times T^p \rightarrow T^s$

Note that due to the generality of Definition 1, we are able to take into account the state of the service station, which is of great importance for load-dependent transformations. In addition, we are also able to model request generation respectively request elimination, because K does not necessarily have to equal N. Moreover, the elements of R^p and R^s don't need to be of the same type.

The concept of load transformation used in this paper can be used to gain deeper understanding of the properties of the resulting secondary load and its relationship to primary load in cases where secondary load is directly measurable. Moreover in many cases measurement of secondary load is hard or impossible to perform. In these cases load transformation provides a tool to obtain models for load as seen at the corresponding interfaces.

2.1 Load Transformations for Markovian Arrival Processes

It is of course possible to conduct load transformations directly on concrete arrival sequences by means of a simulative approach. This proceeding bears the disadvantage that it operates only on realizations of the underlying stochastic process. In order to circumvent this loss of generality we choose a different approach: The transformation of process specifications. Because of its advantageous

properties concerning analytical tractability, its thorough treatment in the literature and its applicability to a wide range of application scenarios we choose the *Batch Markovian Arrival Process* as the process class to be used (see Def. 2) [Luc93].

Definition 2. *Let the* Batch Markovian Arrival Process *(BMAP) be defined as a markovian arrival process with infinitesimal generator matrix:*

$$\mathbf{Q} = \begin{pmatrix} D_0 & D_1 & D_2 & D_3 & \dots \\ 0 & D_0 & D_1 & D_2 & \dots \\ 0 & 0 & D_0 & D_1 & \dots \\ \vdots & & \vdots & & \ddots \end{pmatrix}$$

With $D_{0,\dots,\infty}$ *being* $(m \times m)$ *matrices defined by:*

$$(D_0)_{ii} = -\lambda_i, \quad 1 \leq i \leq m$$

$$(D_0)_{ij} = \lambda_i p_i(0,j), \quad 1 \leq i,j \leq m \wedge i \neq j$$

$$(D_k)_{ij} = \lambda_i p_i(k,j), \quad 1 \leq i,j \leq m \wedge k > 0$$

$$\sum_{\substack{j=1 \\ j \neq i}}^{m} p_i(0,j) + \sum_{k=1}^{\infty} \sum_{j=1}^{m} p_i(k,j) = 1, \quad 1 \leq i \leq m$$

◊

Here $p_i(k,j)$ denotes the probability that – given the process is in state i – a transition to state j occurs with a batch arrival of size k. Thus, we are able to model arbitrary discrete distributions with possibly infinite support. The overall rate with which the process leaves state i is given by λ_i. As can be seen in Def. 2 BMAPs provide means for the unifying description of interarrival times and packet lengths which is especially of value if interarrival times and packet lengths are correlated.

As we want our transformation procedures to be close with respect to BMAPs in following we restrict ourselves to transformations mapping BMAPs to BMAPs:

$$T_{BMAP} : \mathcal{BMAP}^p \rightarrow \mathcal{BMAP}^s \tag{2}$$

In [HW07a] we already demonstrated that the transformation induced by fragmentation and sliding window procedures can be modeled with very good accuracy by $BMAP$ transformations. In this article we further investigate the transformation for modeling fragmentation (cf. Prop. 1).

Proposition 1. *Let the primary load be defined by* $BMAP$ B*. The secondary load after fragmentation with maximum packet length* M *is modeled as* $BMAP_F^S$ *with the set of* generating states

$$G = \{g_i, 0 < i \leq m\},$$

the set of waiting states

$$W = \{w_{i,j}, \forall i, j. \exists k. p_i(j,k) > 0\}$$

and matrices $D_{0,\ldots,\infty}$

$$(D_k)_{w_{i,j},g_j} = \begin{cases} \frac{\lambda_j}{1-\lambda_j \cdot E[T_i^F]}, & k = 0 \\ 0, & k \neq 0 \end{cases}$$

$$(D_k)_{w_{i,j},w_{i,j}} = \begin{cases} -\frac{\lambda_j}{1-\lambda_j \cdot E[T_i^F]}, & k = 0 \\ 0, & k \neq 0 \end{cases}$$

$$(D_k)_{g_i,g_i} = \begin{cases} \lambda_f \cdot \left(1 - \frac{1}{E[L_i^F]+1}\right), & k = M \\ 0, & k \neq M \wedge k \neq 0 \\ -\lambda_f, & k = 0 \end{cases}$$

$$(D_k)_{g_i,w_{i,j}} = \begin{cases} \lambda_f \cdot p_i^{TF}(k,j) \cdot \frac{1}{E[L_i^F]+1}, & k \leq M \\ 0, & k > M \end{cases}$$

$\Diamond$

Here λ_f denotes the rate with which fragments are generated and $p_i^{TF}(k,j)$ denotes the probability that a transition to state j occurs with an arrival of a fragment of non-maximum length k – given that the process was in state i. Each state of $BMAP$ B results in two or more states of the transformed process. While states $w_{i,j}$ represent the actual transition rates of the $BMAP$ describing primary load, states g_i model the generation of fragments. Because fragmentation is not modeled as a timeless process we have to increase the rate with which states $w_{i,j}$ are left. This is accomplished by reducing the mean sojourn time by mean fragmentation time $E[T_i^F]$. Transition rates are chosen as shown in Proposition 1 to match the original fragment distribution in its mean. The resulting number of fragments per packet is geometrically distributed with parameter $p = \frac{1}{E[L_i^F]+1}$. (Here, L_i^F denotes the random variable specifying the number of fragments generated in state i.) As this is clearly not a sufficiently valid assumption for general packet length distributions we also proposed approximative models for long-tailed or normal distributions of packet lengths [HW07a].

2.2 The Distribution of Fragmented Packet Lengths

In [HW07a] we discussed the influence of fragmentation on packet length distribution. Here for further usage we introduce a new representation of the distribution of the size of fragments of non-maximum length – which is the last one induced by the corresponding packet – as illustrated in formula (3).

$$f_f(x) = \sqcap\left(\frac{x - \frac{1}{2}M}{M}\right)(f(x) * \text{Ш}_M(x)) \tag{3}$$

Here $\sqcap(x)$ denotes the rectangular function and $\text{Ш}(x)$ is the dirac comb:

$$\text{Ш}_M(x) = \sum_{k=-\infty}^{\infty} \delta(x - kM).$$

The correctness of this formula can be seen by noting that $f(x) * \delta(x - kM) = f(x - kM)$. This means that the convolution of $f(x)$ with the dirac comb under consideration leads to $\sum_{k=-\infty}^{\infty} f(x - kM)$. Each summand represents a replication of $f(x)$ shifted by kM and we obtain the probability distribution of fragments of non maximum length by restricting the support of the distribution to the interval $[0, M]$ which is done by multiplying the function by the rectangular function.

This representation has two implications:

- The Fourier transform of formula (3) is given by

$$F_f(f) = (Msinc(fM)e^{-i2\pi\frac{1}{2}Mf}) * (F(f)\frac{1}{M}\text{Ш}_{\frac{1}{M}}(f)) \tag{4}$$

  which is the formula used to represent an ideal sampler in time domain, whose bandwidth is subsequently limited. (Here *sinc* denotes the cardinal sin function.) This shows that calculating the distribution of fragments of non-maximum length is the same as sampling only with time resp. frequency domain interchanged. As a direct consequence of the sampling theorem it may be impossible to reconstruct the original packet length distribution.
- By means of formula (3) we are able to formally prove that the size of fragments of non-maximum length tends to uniformity with increasing variance. This fact has often been reported from experiments but, to the best of our knowledge, has not been proven yet. We give proofs for several distribution types in the following. These results allow the simplification of models for fragmented loads, especially for the transmission over channels with high error rates, for which lower values of the maximum fragment length are advantageous in order to ensure a higher throughput.

We first state the general condition under which the distribution of fragmented packet lengths equals the uniform distribution. We then specialize this result to the normal resp. negative exponential distribution.

Theorem 1. *The distribution of the length of fragments of non-maximum length is uniformly distributed if* $F_f(0) = 1$ *and* $\forall x \neq 0.F_f(x) = 0$.

Proof. The Fourier transform of the fragment length distribution $f_f(x)$ is given by formula (4). Given that $F_f(0) = 1$ and $\forall x \neq 0.F_f(x) = 0$ it reduces to:

$$\begin{aligned} F_f(f) &= (Msinc(fM)e^{-i2\pi\frac{1}{2}Mf}) * (\frac{1}{M}\delta(f)) \\ &= \frac{1}{M}(Msinc(fM)e^{-i2\pi\frac{1}{2}Mf}) \end{aligned}$$

$$\mathfrak{F}^{-1}(\frac{1}{M}(Msinc(fM)e^{-i2\pi\frac{1}{2}Mf})) = \frac{1}{M} \sqcap \left(\frac{x-\frac{1}{2}M}{M}\right)$$

completes the proof, where $\mathfrak{F}^{-1}$ denotes the inverse fourier transformation. □

Based on theorem 1 we now show that the fragment distribution as induced by the negative exponential distribution tends to the uniform distribution with increasing variance. Here λ denotes the rate parameter of this distribution.

Theorem 2. *The distribution of the length of fragments of non-maximum length for negative exponential packet length distribution tends to the uniform distribution as* $\lambda \to 0$.

Proof. The Fourier transform of the negative exponential distribution is given by:

$$F_e(f) = \frac{\lambda}{\lambda + i2\pi f}$$

Indepently of λ we have $F_e(0) = 1$ and if $\lambda \to 0$ we see by separating the real and imaginary part of $F_e(f)$

$$F_e(f) = \frac{\lambda^2}{\lambda^2 + 4\pi^2 f^2} - \frac{i2\pi\lambda f}{\lambda^2 + 4\pi^2 f^2}$$

that both summands tend increasingly faster to 0 as λ approaches 0 so that the function vanishes except for $F_e(0)$. Using theorem 1 completes the proof. □

By using theorem 2 it is also possible to obtain the same result for the hyperexponential resp. the hypoexponential distribution. The density of the former is given by a weighted sum of neg. exponential densities for which the Fourier transform is given by a weighted sum of the Fourier transform of the neg. exponential distribution so that the argument follows the same line. It is worth mentioning that this distribution can be used for the approximation of heavy-tailed distribution [FW97], so that the results obtained here account for heavy-tailed distributions at least in an approximative way. For the hypoexponential distribution – which includes the Erlang distribution – the Fourier transform is given by a product of the Fourier transform of neg. exponential densities. The uniformity follows from the fact that the product meets the assumption of theorem 1 if all factors do.

Because of its importance for aggregated loads and for video traffic modeling we additionally show that the normal distribution meets the assumptions given in theorem 1. As the normal distribution has negative support it can be used as a distribution for packet lengths only if the probability of negative values is negligible. Nonetheless in the following we show that the Fourier transform of the normal distribution $\mathcal{N}(\mu, \sigma)$ meets the conditions assumed in theorem 1 for all μ if σ tends to ∞. This includes parametrisations of the distribution which are typical in modeling of real load.

Theorem 3. *The distribution of the length of fragments of non-maximum length for normal distributed packet length distribution tends to the uniform distribution as* $\sigma \to \infty$.

Proof. The Fourier transform of the density of the normal distribution is given by:

$$F_n(f) = e^{-i2\pi f\mu} e^{-2\pi^2 f^2 \sigma^2}$$

Independently of μ and σ we have $F_n(0) = 1$ and taking the limit $\sigma \to \infty$ we see that the second factor of $F_n(f)$ tends to 0 increasingly faster with increasing σ so that the function vanishes except for $F_n(0)$. Using theorem 1 completes the proof. □

Moreover, for concrete densities $f(x)$ we compute the mean deviation of the resulting fragment length distribution as given in equation 5.

$$\begin{aligned} d_u &= \frac{1}{M}\int_{-\infty}^{\infty} |\frac{1}{M} \sqcap \left(\frac{x-\frac{1}{2}M}{M}\right) - \sqcap \left(\frac{x-\frac{1}{2}M}{M}\right) (f(x) * \text{Ш}_M(x))| dx \\ &= \frac{1}{M}\int_{-\infty}^{\infty} |\sqcap \left(\frac{x-\frac{1}{2}M}{M}\right) (\frac{1}{M} - f(x) * \text{Ш}_M(x))| dx \\ &= \frac{1}{M}\int_{0}^{M} |\frac{1}{M} - \sum_{k=-\infty}^{\infty} f(x-kM)| dx \end{aligned} \tag{5}$$

Using eq. (5) it can be seen that the limits given in theorems 2 resp. 3 are not of mere theoretical value: In the right half of figure 2 we plotted d_u for three Erlang distributions[1] with $k = 1, 5, 10$. (The Erlang distribution equals the negative exponential distribution for $k = 1$.) The deviation from the uniform distribution decreases quite fast for all three distributions with increasing standard deviation.

In order to demonstrate that fragment distributions resulting from heavy-tailed distributed packet lengths tend to the uniform distribution as well we additionally present results for the log-normal distribution[2] in the right half of figure 2. Here too, the deviation from the uniform distribution decreases fast with increasing standard deviation.

3 Load Transformations for Lossy Markovian Arrival Processes

With the growing popularity of wireless networks the importance of packet losses due to channel errors is increasing. So in order to predict secondary load in wireless networks it is highly desirable to elaborate models for load transformations taking into account packet losses. For that reason we propose a transformation procedure based on BMAPs modeling the influence of both, state dependent and constant loss rates (cf. Prop. 2). In case of state dependent loss rates, we assume that the underlying process is markovian. (This assumption is widely used in the literature, e.g. [ZRM95, ZK99].) The proposed procedure mainly applies to

[1] The pdf of the Erlang distribution is given by $f(x) = \frac{\lambda^k x^{k-1} e^{-\lambda x}}{(k-1)!}$.

[2] The pdf of the log-normal distribution is given by $f(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{\left(-\frac{[\ln(x)-\mu]^2}{2\sigma^2}\right)}$.

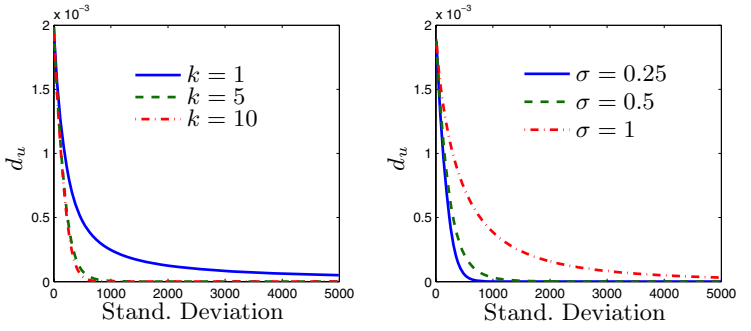


Fig. 2. Deviation from the uniform distribution on $[0, M]$ of the fragment distribution for Erlang distributed packet lengths (left) resp. log-normal distributed packet lengths (right).

scenarios with *Forward Error Correction* and can be used for load modeling and parameter optimization.

Proposition 2. *Let the instantaneous loss rate at time instant t be driven by the Markov process $X = (X(t) : \ t \geq 0)$ with generator matrix Q_L. The loss probability for a packet of length j in state i is given by $r_{i,j}$. The lossy BMAP B_L^S, which results from a primary load model B^P, has the state set $Z = \{Z_B \times Z_X\}$ and is defined by the matrices*

$$(D_0^L) = D_0^P \oplus Q_L + \sum_{i=1}^{\infty} E_i \otimes D_i^P$$

$$(D_i^L) = (I - E_i) \otimes D_i^P, \quad i \geq 1$$

Moreover, E_i is given by

$$E_i = diag(r_{1,i}, ..., r_{m_L,i}) = \begin{pmatrix} r_{1,i} & 0 & \cdots & 0 \\ 0 & r_{2,i} & \cdots & 0 \\ \vdots & \cdots & \ddots & \vdots \\ 0 & \cdots & 0 & r_{m_L,i} \end{pmatrix}.$$

◊

The operator $\otimes$ is the Kronecker product of two matrices, $\oplus$ is the Kronecker sum (cf. e.g. [Lau05]), m_L is the number of states of $X(t)$ and I is an identity matrix. The resulting BMAP B_L^S retains the properties of both markov processes, that is the rates between state sets corresponding to specific states of the loss process (arrival process) equal the original rates (see [HW07b] for a detailed discussion).

For validation purposes we compared the arrival process induced by a simulatively transformed BMAP to the corresponding BMAP, which was analytically transformed. It is worth mentioning that both arrival processes were transformed three times in series: That is fragmentation, header generation and transmission over the lossy link. Thus we compare both models at the interface from physical to data link layer at the receiving station.

For the experiments shown here we consider two *Gilbert-Elliot-Models* as loss processes.[3] A *Gilbert-Elliot-Model* consists of two states with state 1 having a lower error rate than state 2. We used the two generator matrices Q_1 resp. Q_2 as shown in formula (6) and the error rates $r_1 = 0$ and $r_2 = 10^{-3}$ for both processes.

$$Q_1 = \begin{pmatrix} -8 & 8 \\ 2 & -2 \end{pmatrix} \quad Q_1 = \begin{pmatrix} -2 & 2 \\ 8 & -8 \end{pmatrix} \tag{6}$$

In the following we illustrate the influence of the proposed loss transformation on a BMAP-model of an MPEG stream representing a recorded soccer game. The untransformed BMAP consists of three states, each representing one frame type used in MPEG streams (i.e. *I*-, *P*- and *B*-Frames). As can be seen in the upper half of figure 3 the proposed transformation procedure is able to capture both, the interarrival times and the fragment length with good accuracy. We used Q_2 as the generator matrix of the loss process and the MPEG model mentioned above as the arrival process. In case of the packet lengths no differences are observable. The deviations which occur locally in case of the distribution of the interarrival times stem from the constant transmission delay for fragments of maximum length at $t = 0.12ms$ which can only be approximated in the markovian setting used here. It has to be stressed that the logarithmic scale chosen her overemphasizes these deviations as they occur at small time scales. Outside of this region the curves are hardly distinguishable.

As a large maximum batch size of the transformed model may hinder the subsequent usage of analytic methods we could choose to reduce the transformed BMAP-Model to a MAP-Model without batch arrivals. This is especially advantageous in the case of modeling data transmissions over links with high loss rates as considered here: With typically lower maximum fragment length the error induced by neglecting the real fragment length is lower than it would be for high maximum fragment lengths or even unfragmented packets. Furthermore if the length of the packets is large compared to the fragment length we can legitimately assume that the fragments of non-maximum length are uniformly distributed which simplifies the formula used for reducing the BMAP model. The conversion procedure for this case is shown in eq. (7). We chose to model a fragment of non-maximum length as an arrival with probability 0.5 in order to match the mean rate in case these fragments are uniformly distributed.

$$D_0^{MAP} = D_0^{BMAP} + \frac{1}{2}\sum_{i=1}^{M-1} D_i^{BMAP}, \quad D_1^{MAP} = D_M^{BMAP} + \frac{1}{2}\sum_{i=1}^{M-1} D_i^{BMAP} \tag{7}$$

As is illustrated in the lower half of figure 3 we are able to capture the interarrival times and the mean arrival rate of the simulatively transformed $BMAP$ model with good accuracy even though we neglected the batch sizes. For this experiment we used Q_1 and Q_2 as generator matrices for the loss process and

[3] The parametrisations used here are of exemplary nature. Parametrisations for real links can be found e.g. in [ZRM95].

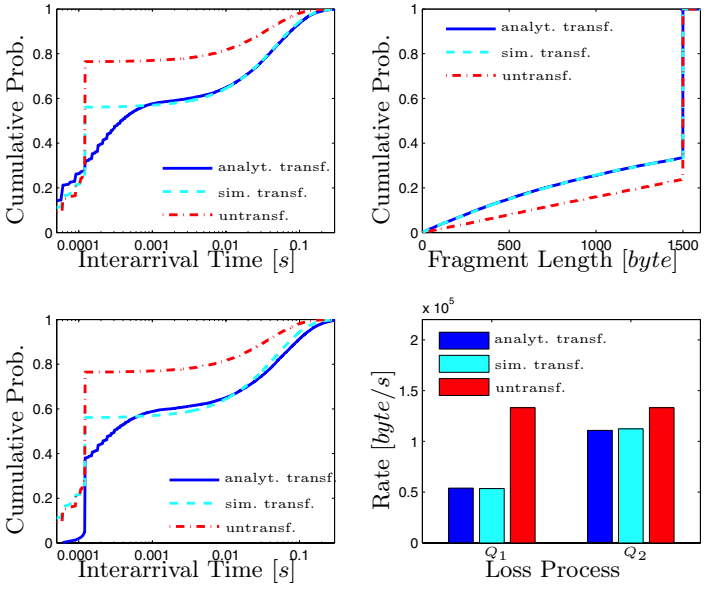


Fig. 3. Evaluation of the empirical distribution of the fragment lengths and the interarrival times for the transformed $BMAP$ (upper half) resp. MAP (lower half)

applied the loss transformation to the MAP resulting from application of formula (7) to the $BMAP$ model of the MPEG stream after fragmentation. In the lower left plot the empirical distribution of the interarrival times in case of Q_2 is shown. As can be seen higher deviations of the curves occur for small interarrival times. This results from the disregard of the batch sizes of the fragments of non-maximum lengths which experience a lower transmission delay. For fragments of maximum length the transmission delay of $0.12ms$ bounds the interarrival time. Nonetheless the MAP approximation is able to capture the behaviour of the simulatively transformed arrival process with good accuracy, which is quite remarkable given that the batch size is neglected. In the right lower plot we additionally illustrate that the mean arrival rate is captured with high accuracy for both loss processes. We do not show the empirical distribution because the packet lengths resulting from the MAP are constant.

Mean Arrival Rate

In the proposed transformation procedure packet arrivals are modeled as timeless events. Therefore, the probability of an arrival of a corrupted packet solely depends on the stationary state probability of the loss process π^L. The mean rate of the lossy $BMAP$ thus can be computed as given by formula (8). Here e denotes a column vector $[1\ 1\ ...\ 1]'$ of correctly chosen length. Furthermore, π is the stationary state probability of the $BMAP$ defining primary load and

P^i denotes the matrix with the state-dependent transmission probabilities of a packet of length i on the main diagonal.

$$R = \sum_{i=1}^{\infty} i \left((\pi \cdot D_i \cdot e) \cdot (\pi^L \cdot P^i \cdot e) \right) \tag{8}$$

If the arrivals are fragmented prior to the transmission over the lossy link, the mean rate can be computed based on the following considerations: The fragmented arrival process, which is subject to packet losses, consists of a sequence of arrivals of maximum length, interrupted by an arrival of non-maximum length. This corresponds to a fragmentation of packet at primary load level in $n-1$ fragments of maximum length M and one single fragment of length 0 to $M-1$.[4] The fragments of maximum length experience a transmission probability P^M and the arrival of a packet at primary load level induces $E[L^F]$ such fragments in mean. It follows:

$$\begin{aligned} R = & \sum_{i=1}^{M-1} i \left((\pi \cdot D_i^{TF} \cdot e) \cdot (\pi^L \cdot P^i \cdot e) \right) + \\ & M \cdot \left((\pi \cdot \underline{\lambda}) \cdot (\pi^L \cdot P^M \cdot e) \right) \cdot E[L^F] \end{aligned} \tag{9}$$

Here, $\underline{\lambda}$ denotes the row vector consisting of the rates out of the states $1, ..., n$ and D_i^{TF} the rate matrix of fragments of length $i < M$ (see [HW07a]). For the realistic evaluation of the mean arrival rate we also consider the additional overhead each fragment induces. This leads to a tradeoff between lower error probabilities for shorter packets and an increasing overhead. Therefore, we enhance formula (9) by $\frac{i}{i+h}$, which leads to:

$$\begin{aligned} R' = & \sum_{i=1}^{M-1} i \left((\pi \cdot D_i^{TF} \cdot e) \cdot (\pi^L \cdot P^i \cdot e) \right) \cdot \frac{i}{i+h} + \\ & M \cdot \left((\pi \cdot \underline{\lambda}) \cdot (\pi^L \cdot P^M \cdot e) \right) \cdot E[L^F] \cdot \frac{M}{M+h} \end{aligned} \tag{10}$$

By using formula (10) we now can optimize the maximum fragment size for a given combination of arrival and loss process. It is worth mentioning that the fragment size only has to be computed when the characteristics of the arrival resp. the loss process change. Furthermore, the major influence of the mean packet size at primary load level can be recognized by inspecting formula (10). The distribution of the packet length only influences the sum in this formula, whose impact on the result decreases with increasing mean packet length.

This assumption is confirmed by the three cases illustrated in figure 4. All three primary load models generate 138000 arrivals per second. The arrival processes can be described as follows: BMAP B_1 generates packets of uniformly distributed length at a rate of 1000 arrivals per second. BMAP B_2 is the model

[4] The fragment length 0 of the $n-th$ packet models the case, where the packet length is a multiple of the fragment length.

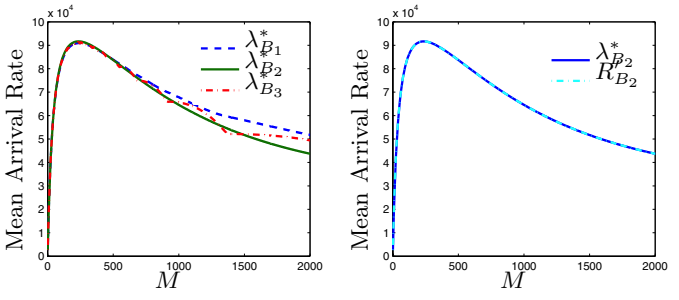


Fig. 4. Mean arrival rate λ^* of the transformed BMAPs $B_{1,2,3}$ and the directly computed rate R'_{B_2}

for an $MPEG$ stream used above. The model has 3 states and packet lengths are distributed according to the empirical distribution derived from the trace data. BMAP B_3 generates batch arrivals of constant length at a rate of 1000. The mean arrival rate depending on the maximum fragment length was computed by means of formula (11). Analogous to formula (10) the formula for the mean arrival rate of a $BMAP$ (see [Luc93]) was enhanced by the factor $\frac{i}{i+h}$ in order to consider the additionally introduced overhead. In the left half of figure 4 it can be observed that, despite the different properties of the three arrival processes, the maximum according to formula (11) lies in the same region for all processes. This emphasises the major influence of the mean packet length, which was identical for all of the three processes. In the right half of the figure we additionally plotted the curves according to formula (10) resp. (11) for validation purposes. No differences are visible.

$$\lambda^* = \pi \left(\sum_{i=1}^{\infty} \frac{i}{i+h} i D_i \right) e \tag{11}$$

Moreover, based on the considerations presented in section 2.2 in many cases a uniform distribution may serve as a good approximation for lengths of those fragments which have non-maximum size. Here, the distribution of packet lengths can be neglected and the preceeding equation reduces to:

$$R' \approx \frac{\pi \cdot \underline{\lambda}}{M} \sum_{i=1}^{M-1} \frac{i^2}{i+h} \left((\pi^L \cdot P^i \cdot e) \right) +$$
$$\frac{M^2}{M+h} \cdot \left((\pi \cdot \underline{\lambda}) \cdot (\pi^L \cdot P^M \cdot e) \right) \cdot E[L^F] \tag{12}$$

The assumption of uniformity is especially legitimate in the case of wireless links with high loss rates. In these scenarios the maximum fragment length which maximizes the throughput is typically much smaller than in conventional networks. By inspecting the sum in equation (12) it can be seen that the sin-

gle summands do not depend on M anymore. Hence, besides the reduction of matrix multiplications, by re-using the computed values the number of operations necessary only depends on the maximum value of M used and not on the number of distinct values chosen for M. This means that we are able to compute the throughput for a high number of parametrisations without significantly increasing the number of operations necessary.

4 Conclusions

In this paper the concept of load transformation has been discussed. This concept has been applied to primary loads the behaviour of which is reflected by BMAPs. As concrete load transformations, on the one hand, we assumed packet fragmentation (as it occurs in most of current networks) the effect of which could be modeled with high accuracy. Moreover we showed that distribution of fragments of non-maximum lengths tends to the uniform distribution which allows us to simplify the proposed modeling approach for load transformations.

On the other hand, we were also able to elaborate realistic analytical models taking into account the impact of packet losses in wireless networks on sequences of packets to be transmitted. Finally we illustrated how to use the analytical description of secondary load in order to compute efficiently the throughput at lower layer interfaces. This result can be directly used for the optimization of network parameters.

We will continue to extend the work presented here. Besides the development of further models for load transformation and the validation with measured data we are also continuing to use the transformed models for the investigation of network load characteristics as illustrated in this article. Moreover we hope that the concept of load transformation presents a valuable means for performance evaluation.

References

[CW06] Chydzinski, A., Winiarczyk, R.: Blocking Probability in a BMAP Queue. In: ISCC 2006: Proc. 11th IEEE Symp. on Computers and Communications, Washington, DC, USA, pp. 547–553. IEEE Computer Society, Los Alamitos (2006)

[FC01] Ferng, H.-W., Chang, J.-F.: Departure Processes of BMAP/G/1 Queues, Queueing Syst. Theory Appl. 39(2-3), 109–135 (2001)

[FW97] Feldmann, A., Whitt, W.: Fitting Mixtures of Exponentials to Long-Tail Distributions to Analyze Network Performance Models. INFOCOM (3), 1096–1104 (1997)

[Hof01] Hofmann, J.: The BMAP/G/1 Queue with Level-Dependent Arrivals - An Overview. Telecommunication Systems 16(3-4), 347–359 (2001)

[HSB04] Hofkens, T., Spaey, K., Blondia, C.: Transient Analysis of the D-BMAP/G/1 Queue with an Application to the Dimensioning of a Playout Buffer for VBR Video. Networking, 1338–1343 (2004)

[HW07a] Heckmüller, S., Wolfinger, B.E.: Load Transformations for Markovian Arrival Processes. In: ASMTA 2007, pp. 35–43 (June 2007)

[HW07b] Heckmüller, S., Wolfinger, B.E.: Modellierung verlustinduzierender Lasttransformationen für markovsche Ankunftsprozesse. In: MMBNet 2007 Workshop (in German) (2007)

[KLL03] Klemm, A., Lindemann, C., Lohmann, M.: Modeling IP traffic using the batch Markovian arrival process. Perform. Eval. 54(2), 149–173 (2003)

[Lau05] Laub, A.J.: Matrix Analysis for Scientists and Engineers. SIAM (2005)

[LB06] Lee, H.W., Baek, J.W.: Threshold Workload Control in the BMAP/G/1 Queue. In: International Conference on the Quantitative Evaluation of Systems (QEST), pp. 353–364. IEEE, Los Alamitos (2006)

[Luc91] Lucantoni, D.M.: New results for the single server queue with a batch Markovian arrival process. Stoch. Mod. 7, 1–46 (1991)

[Luc93] Lucantoni, D.M.: The BMAP/G/1 queue: a tutorial, Models and Techniques for Performance Evaluation of Computer and Communication Systems. Doniatiello, L., Nelson, R. (eds.), pp. 330–358. Springer, New York (1993)

[Neu79] Neuts, M.F.: A versatile Markovian point process. J. Appl. Prob. 16, 764–779 (1979)

[Wol99] Wolfinger, B.E.: Characterization of Mixed Traffic Load in Service-Integrated Networks. Systems Science Journal 25(2), 65–86 (1999)

[WZHB02] Wolfinger, B.E., Zaddach, M., Heidtmann, K.D., Bai, G.: Analytical modeling of primary and secondary load as induced by video applications using UDP/IP. Computer Communications 25(11-12), 1094–1102 (2002)

[ZHS05] Zhang, Q., Heindl, A., Smirni, E.: Models of the departure process of a BMAP/MAP/1 queue. SIGMETRICS Perform. Eval. Rev. 33(2), 18–20 (2005)

[ZK99] Zhang, Q., Kassam, S.A.: Finite-State Markov Model for Rayleigh Fading Channels. IEEE/ACM Trans. Communic. 47(11), 1688–1692 (1999)

[ZRM95] Zorzi, M., Rao, R., Milstein, L.: On the accuracy of a first-order Markov Model for data transmission on fading channels. In: Proc. IEEE ICUPC 1995, November 1995, pp. 211–215 (1995)

Modeling Web Server Traffic with Session-Based Arrival Streams

Laurence Hoflack, Stijn De Vuyst, Sabine Wittevrongel, and Herwig Bruneel

Ghent University, TELIN Department, SMACS Research Group, Sint-Pietersnieuwstraat 41, B-9000 Gent, Belgium
{lhoflack,sdv,sw,hb}@telin.UGent.be

Abstract. In this paper, we consider a discrete-time queueing model with infinite waiting room and one single output channel. Users from an infinite user population can start and end sessions during which they are active and send packets to a buffer system. Each active user generates a random but strictly positive number of packets per time slot. The queueing model is applied to study the traffic of a web server, where each session corresponds to one file download. This approach has the advantage of leading to analytical expressions for the main performance measures (buffer occupancy, delay, ...) of the output buffer of the web server. Specifically, in this paper, a closed-form expression for the mean session delay is obtained. We then apply the model to a web server, based on a trace of actual web traffic. Using the model, we can rapidly study the influence of the different system parameters and compare various alternatives.

Keywords: Discrete-time queueing model, session-based arrivals, performance evaluation, packet-based networks, web servers.

1 Introduction

In many communication systems buffers are used for the temporary storage of information packets. A deep understanding of the behavior of these buffers is therefore important to study the performance of the whole system. The performance measures of a packet buffer, such as buffer occupancy, delay, ... strongly depend on the nature of the packet arrival process. Session-based arrival streams are a new approach for modeling the traffic streams that arise in modern telecommunication networks. We consider an infinite user population where each user is capable of starting and ending sessions. During a session a user is active and sends information through the communication system.

In this paper, we consider a session-based arrival process where each active user generates a random but strictly positive number of packets per slot. Hence, each user generates a non-interrupted data flow until the user's session ends, where the number of generated packets varies from slot to slot, but is never equal to zero. The model under study can thus be seen as an extension of the train

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 47–60, 2008.

arrival process (see e.g. [3,4,6,8,9,10,15,16,14]), where messages (the equivalent of what we consider sessions) arrive in the buffer at the rate of one packet per slot. Also related is the correlated train arrival process (see e.g. [12,17]), where a finite number of users generate one packet per slot during active periods and no packets during passive periods.

Although in our model the generated traffic does not exhibit interruptions, it has its practical use. Consider for instance a file server that serves a number of users (e.g. a video on demand server). If we define the download of a file by one user as one session, our model delivers a good description of the outgoing data buffer behavior. As another example, consider a user hitting the 'send' button after writing an email. The transmission of this email to the central mail server can be considered as a session.

In a previous paper [11], we have investigated the buffer occupancy and the packet delay in a discrete-time buffer system with session-based arrival streams. The aim of the present paper, however, is to derive analytical results for the mean session delay, which poses a significant additional effort. The delay of a session is regarded at the level of the buffer. It is the time from the moment that the first packet of the session arrives in the buffer, until the last packet of the session has left the buffer. The mean session delay is derived under the assumption of a first-come-first-served (FCFS) queueing discipline for packets. The model is also applied to a web server, based on a trace of actual web traffic.

The outline of the paper is as follows. In the next section, we describe the model under study. The mathematical model and the system equations are given in Sect. 3. Some results of [11], which are necessary for the analysis of the present paper, are summarized in Sect. 4. The analysis of the mean session delay is presented in Sect. 5. The model is applied to an actual web server in Sect. 6. Conclusions are given in Sect. 7.

2 Model Description

In this paper, we consider a discrete-time queueing system with one single output line and an infinite storage capacity. As usual for discrete-time models, the time axis is divided in fixed-length time intervals, referred to as slots, and packet transmissions can start and end at slot boundaries only [5,13]. Packets entering the queueing system cannot leave the buffer at the end of their arrival slot: their system time must include at least one whole time slot. An infinite user population starts and ends sessions, during which information is sent over the communication system. When a user starts a session, the user generates a variable but strictly positive number of packets per slot. The session ends when the user has no more data left to send. We define the length of a session as the time between the start and end of the session, i.e., the number of slots during which packets from the session arrive to the buffer.

Note that we do not put a limit on the number of users simultaneously generating packets. The numbers of new sessions starting in successive slots are assumed to be independent and identically distributed (i.i.d.) random variables.

In normal conditions, internet users act independently from each other and thus this seems a realistic assumption. The session lengths are assumed to be geometrically distributed.

The transmission time of a packet is also geometrically distributed and independent of the transmission times of other packets. Note that the assumption of geometric transmission times enables us to study buffer systems for packets of variable size. Alternatively, the model can also be used to describe a system with constant packet transmission times of one slot and an unreliable output line subject to random failures that occur independently from slot to slot. Indeed, the latter results in a geometric distribution for the effective transmission times required for the successful transmission of a packet.

Finally, the queueing discipline is FCFS for packets. This means that packets are transmitted in the order they were stored in the buffer, whereby packets that arrive during the same slot are assumed to be stored (and hence transmitted) in random order.

3 Mathematical Model and System Equations

Let us define the random variable a_k as the number of active sessions during slot k (k is an integer ≥ 1). Also, let s_k denote the number of new sessions generated during slot k and let u_k be the buffer occupancy (i.e., the total number of packets in the buffer) after slot k. The variable p_k^i indicates the number of packets generated during slot k by session i and m_k is the total number of packets generated during slot k.

As the s_k's are i.i.d. variables, the distribution of s_k is independent of the slot index k. The probability generating function (pgf) of the number of new sessions per slot is given by

$$S_k(z) \triangleq E[z^{s_k}] = \sum_{n=1}^{\infty} \text{Prob}[s_k = n]\, z^n = S(z)\,, \tag{1}$$

where $E[.]$ denotes the expected value operator and where z is a complex variable lying in the region of convergence of the power series in (1).

The pgf of the number of packets generated per slot per session is given by

$$P_k^i(z) \triangleq E\left[z^{p_k^i}\right] = P(z)\,, \tag{2}$$

where $P(0) = 0$, since at least one packet is generated per session per slot.

The session lengths are geometrically distributed with parameter α and thus described by the following probability mass function and pgf:

$$\ell(n) \triangleq \text{Prob[session length is } n \text{ slots}] = (1-\alpha)\alpha^{n-1}\,, \quad n \geq 1\,, \tag{3}$$

$$L(z) = \frac{(1-\alpha)z}{1-\alpha z}\,. \tag{4}$$

This means that the mean session length is given by $L'(1) = 1/(1-\alpha)$.

Analogously, the transmission time of a packet is also geometrically distributed with parameter $1-\sigma$ and the mean transmission time equals $1/\sigma$. Owing to the memoryless property of this geometric distribution, whenever there is a packet under transmission during a slot, this transmission will end at the end of this slot with probability σ and the transmission will continue with probability $1-\sigma$, independent of the length of the elapsed part of the transmission time.

Furthermore, we define a variable c_k^i that is one if the ith session that is active in slot $k-1$, still continues in slot k, and zero otherwise. As the session length is geometrically distributed, the c_k^i's are i.i.d. Bernoulli random variables with the following probability mass function:

$$\text{Prob}\big[c_k^i = 0\big] = 1-\alpha\,, \qquad \text{Prob}\big[c_k^i = 1\big] = \alpha\,. \tag{5}$$

In view of the above model description and definitions, the evolution in time of the number of active sessions, the total number of packets generated per slot and the buffer occupancy is then expressed by the following system equations (see Fig. 1):

$$a_k = s_k + \sum_{i=1}^{a_{k-1}} c_k^i\,, \tag{6}$$

$$m_k = \sum_{i=1}^{a_k} p_k^i\,, \tag{7}$$

$$u_k = m_k + (u_{k-1} - r_k)^+\,, \tag{8}$$

where $(.)^+ = \max(.,0)$, and where the r_k's are i.i.d. Bernoulli variables with probability mass function

$$\text{Prob}[r_k = 0] = 1-\sigma\,, \qquad \text{Prob}[r_k = 1] = \sigma\,. \tag{9}$$

Clearly, the interpretation of (8) and (9) is that if the system is not empty, a packet leaves with probability σ, which is indeed the case if the transmission times have a geometric distribution.

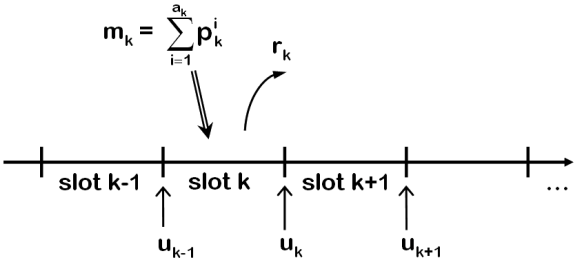


Fig. 1. Definitions of the random variables m_k, u_k and r_k

4 Preliminary Results

In [11], we have analyzed the arrival process and the buffer occupancy. One of the results is an implicit expression for the steady-state joint pgf $Q(x,z)$ of the number of active sessions and the buffer occupancy:

$$\begin{aligned} Q(x,z) &\triangleq \lim_{k\to\infty} E[x^{a_k} z^{u_k}] \\ &= \frac{S(xP(z))}{z} \left\{ \Phi(z) Q(C(xP(z)),z) + \sigma(z-1)p_0 \right\} , \end{aligned} \tag{10}$$

where $C(z) \triangleq 1-\alpha+\alpha z$ is the pgf of the variable c_k^i and where $\Phi(z)$ is equal to $\sigma+(1-\sigma)z$. The quantity p_0 denotes the steady-state probability of having an empty buffer at the start of an arbitrary slot and is given by

$$p_0 = 1 - \frac{P'(1)S'(1)}{\sigma(1-\alpha)} . \tag{11}$$

From the functional equation (10), the mean number of active sessions $A'(1)$ during a slot can be derived:

$$A'(1) = \frac{S'(1)}{1-\alpha} . \tag{12}$$

In [11], the derivation of the steady-state pgf $U(z)$ of the buffer occupancy is explained. Here we only give the expression for the mean buffer occupancy $U'(1)$:

$$\begin{aligned} U'(1) = \frac{1}{2(1-\alpha)^2\sigma p_0} &\big\{ 2P'(1)S'(1)[1-\alpha+P'(1)(\alpha-S'(1))] \\ &+ S''(1)P'(1)^2 + S'(1)P''(1)(1-\alpha) \big\} - \frac{\alpha P'(1)S'(1)}{(1-\alpha)^2} . \end{aligned} \tag{13}$$

5 Analysis of the Session Delay

In this section, we derive an expression for the mean session delay, under the assumption of a FCFS queueing discipline for packets. It is useful to study the delay of a whole session, especially in the case where we consider a non-interrupted data flow to be one session. We define the session delay d_s as the time period between the end of the slot in which the first packet of a session arrives in the buffer and the end of the last transmission slot of the last packet of the session. So, in the case of a file transfer, the session delay represents the time it takes to download the whole file. Determining the pgf of the session delay is complicated, and we therefore only derive the mean value. The mean session delay is given by

$$E[d_s] = \sum_{n=1}^{\infty} E\big[d_{s|n}\big] \, \text{Prob[session lasts } n \text{ slots]} , \tag{14}$$

where $d_{s|n}$ denotes the session delay in the case that the session lasts n slots. The probability that a session lasts n slots is given by the probability mass function of the session length $\ell(n)$ (see (3)). We need to make a distinction between $d_{s|1}$ and $d_{s|n}$ for $n > 1$.

5.1 Delay of a Session with Length 1

In view of the FCFS queueing rule for packets, the delay of a session with length 1 is given by the time needed to transmit all the packets present in the buffer after the session's arrival slot except the ones that arrived after the last packet of the session. If we define q_s as the total number of packets that arrive in the same slot as the session, but after the last packet of the session, and u_s as the total buffer occupancy after the session's arrival slot, we get the following expression for the mean session delay:

$$E\big[d_{s|1}\big] = (E[u_s] - E[q_s])E[t] \ , \tag{15}$$

where $E[t]$ is the mean packet transmission time, which is given by $1/\sigma$.

Note that u_s is the buffer occupancy after the arrival slot of a tagged newly started session. Let s_s and a_s denote the total number of new sessions and the number of active sessions respectively, during the arrival slot of the tagged session. In order to derive the joint pgf of s_s, a_s and u_s for *a slot in the steady state where a new session starts*, we first need to define $H(x, y, z)$ as the joint pgf of the random variables s_k, a_k and u_k for *a random slot k in the steady state*. Applying the system equations of Sect. 3, we obtain for the steady-state pgf $H(x, y, z)$:

$$\begin{aligned} H(x,y,z) &\triangleq \lim_{k\to\infty} E[x^{s_k} y^{a_k} z^{u_k}] \\ &= S(xyP(z)) \lim_{k\to\infty} E\Big[C(yP(z))^{a_{k-1}} z^{(u_{k-1}-r_k)^+}\Big] \\ &= \frac{S(xyP(z))}{z} \left\{\Phi(z) Q(C(yP(z)), z) + \sigma(z-1)p_0\right\} \ , \end{aligned} \tag{16}$$

where we have also used the property that $H(1, y, z) = Q(y, z)$.

The joint pgf $G(x, y, z)$ of the random variables s_s, a_s and u_s for *a slot in the steady state where a tagged new session starts*, is then given by (which can be shown using similar methods as in [3,4])

$$\begin{aligned} G(x,y,z) &\triangleq E[x^{s_s} y^{a_s} z^{u_s}] \\ &= \frac{x}{S'(1)} \frac{\partial}{\partial x} H(x,y,z) \\ &= \frac{x\, y\, P(z)\, S'(xyP(z))\, Q(y,z)}{S'(1)\, S(yP(z))} \ . \end{aligned} \tag{17}$$

This expression enables us to calculate the mean buffer occupancy after the arrival slot of a tagged new session and the mean number of active sessions during such a slot:

$$E[u_s] = \frac{\partial}{\partial z} G(1,1,1) = U'(1) + P'(1)(1 - S'(1)) + \frac{S''(1)P'(1)}{S'(1)}\,, \tag{18}$$

$$E[a_s] = \frac{\partial}{\partial y} G(1,1,1) = A'(1) + (1 - S'(1)) + \frac{S''(1)}{S'(1)}. \tag{19}$$

The only unknown quantity left to derive in (15) is $E[q_s]$. If we define m_s as the total number of packet arrivals during the tagged slot and m_t as the number of packets sent by the tagged session, q_s is only dependent on m_s and m_t, due to the random order of the packet arrivals in a certain slot. Specifically, it can be shown that

$$\text{Prob}[q_s = \ell | m_s = m, m_t = m^*] = \frac{\binom{m-\ell-1}{m^*-1}}{\binom{m}{m^*}}\,, \quad 0 \le \ell \le m - m^*\,. \tag{20}$$

If we further define $p(j,\ell,m,m^*)$ as $\text{Prob}[a_s = j, q_s = \ell, m_s = m, m_t = m^*]$ and $q(j,m,m^*)$ as $\text{Prob}[a_s = j, m_s = m, m_t = m^*]$, $E[q_s]$ is given by

$$\begin{aligned} E[q_s] &= \sum_{j=1}^{\infty} \sum_{m=j}^{\infty} \sum_{m^*=1}^{m-j+1} \sum_{\ell=0}^{m-m^*} \ell\, p(j,\ell,m,m^*) \\ &= \sum_{j=1}^{\infty} \sum_{m=j}^{\infty} \sum_{m^*=1}^{m-j+1} \frac{m-m^*}{m^*+1}\, q(j,m,m^*) \\ &= \sum_{j=1}^{\infty} E\left[\frac{m_s - m_t}{m_t+1} \Big| a_s = j\right] \text{Prob}[a_s = j]\,, \end{aligned} \tag{21}$$

where we have used the known property that

$$\sum_{t=k}^{n} \binom{t}{k} = \binom{n+1}{k+1}. \tag{22}$$

To determine the conditional expected value in (21), we need the joint pgf of the random variables m_s and m_t conditioned on a_s. We define this pgf as $\Omega_j(x,y)$, and we find

$$\begin{aligned} \Omega_j(x,y) &\triangleq E[x^{m_t} y^{m_s} | a_s = j] \\ &= E\left[x^{m_t} y^{m_t + \sum_{i=1}^{j-1} p_i} \Big| a_s = j\right] \\ &= P(xy) P(y)^{j-1}\,, \end{aligned} \tag{23}$$

because the number of packets p_i generated per session per slot is independent of the number of sessions and the number of packets generated in another session.

Using $\Omega_j(x,y)$, we then obtain

$$\begin{aligned}
&E\left[\frac{m_s - m_t}{m_t+1}|a_s = j\right] \\
&= \left(\frac{\partial}{\partial y}E\left[y^{m_s}\int_0^1 x^{m_t}dx|a_s=j\right]\right)\Bigg|_{y=1} - \left(1 - E\left[\int_0^1 x^{m_t}dx|a_s=j\right]\right) \\
&= \left(\frac{\partial}{\partial y}\int_0^1 \Omega_j(x,y)dx\right)\Bigg|_{y=1} - 1 + \int_0^1 \Omega_j(x,1)dx \\
&= (j-1)P'(1)\int_0^1 P(x)dx\,. \qquad (24)
\end{aligned}$$

Finally, combining (15), (18)–(21) and (24), we get for the mean session delay of a session with length one:

$$\begin{aligned}
E\big[d_{s|1}\big] = \frac{1}{\sigma}\Bigg\{&U'(1) + P'(1)(1-S'(1)) + \frac{S''(1)P'(1)}{S'(1)} \\
&- \left[\frac{\alpha S'(1)^2 + S''(1)(1-\alpha)}{(1-\alpha)S'(1)}\right]P'(1)\int_0^1 P(x)dx\Bigg\}. \qquad (25)
\end{aligned}$$

5.2 Delay of a Session with Length Larger than 1

The session delay of a session that lasts n slots ($n > 1$) is given by the total remaining transmission time needed to send the packets present in the buffer after the first slot, the new packets arriving in the following $n-2$ slots, and the packets arriving no later than the last packet of the session in the last slot (including the last packet). So if we tag a new session and the slot it started in as slot J, we define $u_{(J+i)}$ as the buffer occupancy after the $(i+1)$th slot of the session ($0 \le i \le n-1$), $m_{(J+i)}$ as the total number of packets arriving in the $(i+1)$th slot and $\bar{q}$ as the number of packets that do not arrive after the last packet of the session in the last slot. We can then write the mean session delay for a session with length n as

$$E\big[d_{s|n}\big] = \left(E\big[u_{(J)}\big] + \sum_{i=1}^{n-2} E\big[m_{(J+i)}\big] + E[\bar{q}]\right)E[t]\,. \qquad (26)$$

The first term $E\big[u_{(J)}\big]$ is equal to $E[u_s]$ defined in the previous section, and therefore given by (18).

The terms $E\big[m_{(J+i)}\big]$ are equal to $P'(1)E\big[a_{(J+i)}\big]$, where $a_{(J+i)}$ is defined as the number of active sessions in slot $J+i$. The following holds ($1 \le i \le n-1$):

$$a_{(J+i)} = s_{(J+i)} + \sum_{k=1}^{a_{(J+i-1)}-1} c^k_{(J+i)} + 1\,, \qquad (27)$$

as the number of active sessions is equal to the sum of the new sessions and the ones that continue, and where the term 1 represents the tagged session that continues with certainty. By taking the mean of this expression, we find

$$\begin{aligned} E\big[a_{(J+i)}\big] - 1 &= S'(1) + (E\big[a_{(J+i-1)}\big] - 1)\alpha \\ &= \sum_{k=0}^{i-1} \alpha^k S'(1) + \alpha^i (E\big[a_{(J)}\big] - 1)\,. \end{aligned} \tag{28}$$

The term $E\big[a_{(J)}\big]$ in (28) is equal to $E[a_s]$ defined in the previous section and given by (19). Substituting this, we get for $E\big[a_{(J+i)}\big]$:

$$E\big[a_{(J+i)}\big] = 1 + \frac{S'(1)}{1-\alpha} + \alpha^i \left(\frac{S''(1)}{S'(1)} - S'(1) \right), \quad 1 \le i \le n-1\,. \tag{29}$$

To derive the mean of $\bar{q}$, we define m_t as the number of packets that the tagged session generates in its last slot. Due to the random order of the packet arrivals in a certain slot, $\bar{q}$ (the number of packets arriving in the last slot of the session no later than the last packet of the session) is only dependent on m_t and the total number of packet arrivals during the last slot $m_{(J+n-1)}$. Specifically, it can be shown that

$$\mathrm{Prob}\big[\bar{q} = \ell | m_{(J+n-1)} = m, m_t = m^*\big] = \frac{\binom{\ell-1}{m^*-1}}{\binom{m}{m^*}}, \quad 0 \le \ell \le m - m^*\,. \tag{30}$$

In an analogous way as in the previous section, we get

$$E[\bar{q}] = \sum_{j=1}^{\infty} E\left[\frac{m_t(m_{(J+n-1)}+1)}{m_t+1} \big| a_{(J+n-1)} = j \right] \mathrm{Prob}\big[a_{(J+n-1)} = j\big]\,. \tag{31}$$

Because the random variables $m_{(J+n-1)}$ and m_s (defined in the previous section) are identically distributed, we can use the result in (24) to obtain

$$E\left[\frac{m_t(m_{(J+n-1)}+1)}{m_t+1} \big| a_{(J+n-1)} = j \right] = jP'(1) - (j-1)P'(1) \int_0^1 P(x)dx\,. \tag{32}$$

Combining (18), (19), (26) and (29)–(32), we obtain for $E\big[d_{s|n}\big]$:

$$\begin{aligned} E\big[d_{s|n}\big] = \frac{1}{\sigma} \Bigg\{ & U'(1) + P'(1)(1 - S'(1)) + \frac{S''(1)P'(1)}{S'(1)} \\ & + (n-1)P'(1)\left(1 + \frac{S'(1)}{1-\alpha}\right) + P'(1)\frac{\alpha - \alpha^n}{1-\alpha}\left(\frac{S''(1)}{S'(1)} - S'(1)\right) \\ & - \left[\frac{S'(1)}{1-\alpha} + \alpha^{n-1}\left(\frac{S''(1)}{S'(1)} - S'(1)\right)\right] P'(1) \int_0^1 P(x)dx \Bigg\}\,. \end{aligned} \tag{33}$$

5.3 Mean Session Delay

With the results we have obtained, we can derive an explicit expression for the mean session delay:

$$\begin{aligned} E[d_s] &= \sum_{n=1}^{\infty} E[d_{s|n}](1-\alpha)\alpha^{n-1} \\ &= \frac{1}{\sigma}\Bigg\{U'(1) + \frac{P'(1)}{1-\alpha} + \frac{P'(1)S''(1)}{S'(1)(1-\alpha^2)} + \frac{P'(1)S'(1)(\alpha^2+2\alpha-1)}{(1-\alpha^2)(1-\alpha)} \\ &\quad - \left[\frac{2\alpha S'(1)}{1-\alpha^2} + \frac{S''(1)}{S'(1)(1+\alpha)}\right] P'(1)\int_0^1 P(x)dx\Bigg\}. \end{aligned} \tag{34}$$

6 Applying the Model to a Web Server

We consider a web server: this is a computer system that accepts requests from users for a certain web page or embedded file (e.g. a picture or a movie), and that responds by sending the requested file to the user. A web server generally contains the web pages of many websites. We apply our model on the situation depicted in Fig. 2: the web server is connected to the internet through a gateway. This gateway contains a data buffer for outgoing data (from the server to the internet). If we consider the transmission of one file to a client as one session, it is obvious that the buffer for outgoing data will experience session-like data traffic. We also suppose that the data transfer from the server to the gateway is uninterrupted. The model proposed in this paper can therefore be applied. Fixed-length packets requiring one slot of transmission time are considered. The gateway has an unreliable output line subject to random failures, occurring independently from slot to slot.

We need to assign realistic values to the parameters that are involved in the model. First of all, we look at the pgf $P(z)$ of the number of packets generated per session per slot. We assume that the output line of the gateway (connected to the internet) has a typical bandwidth of 100 Mbit/s. The pgf $P(z)$ is dependent on the characteristics of the web server itself: we assume that the web server has a maximum transfer rate of 1000 Mbit/s (as is the case for e.g. the HP ProLiant DL585 Rack Storage Server). Therefore, the pgf $P(z)$ can be written as $\sum_{i=1}^{I} \text{Prob}[p=i]\, z^i$, where the value of I is 10, as the transfer rate is at most 10 times higher than the bandwidth of the output channel. We need to choose the

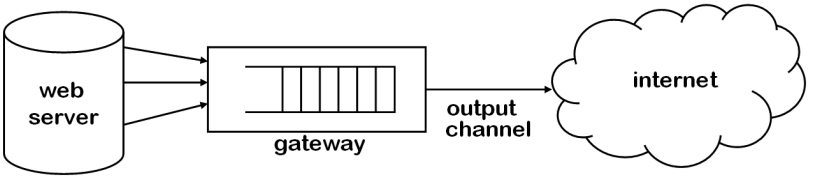


Fig. 2. A web server connected to the internet through a gateway

probabilities Prob$[p = i]$ in a realistic manner. For a certain number of packets p per slot, the transfer rate is equal to p times 100 Mbit/s. The assignment of the probabilities to these values is done by dividing the values of p into three classes. When p = 1-3, the transfer rate per session equals 100-300 Mbit/s, so this corresponds with a low transfer rate per session. Because this low transfer rate occurs the least frequently, we assign a weight factor equal to 1 to these p-values. Analogously, the second class contains p = 4-6 (medium transfer rate) and has a weight factor 3, because a medium transfer rate will occur the most frequently. The third class contains p = 7-10 (high to maximum transfer rate) and has a weight factor 2. The pgf $P(z)$ then looks as follows:

$$P(z) = \frac{1}{20}\left\{\left(\sum_{p=1}^{3} z^p\right) + 3\left(\sum_{p=4}^{6} z^p\right) + 2\left(\sum_{p=7}^{10} z^p\right)\right\}, \tag{35}$$

and $P'(1) = 5.95$ packets are generated per session per slot. This means that we assume that the mean transfer rate per file from the server equals 5.95 times 100 Mbit/s (so 595 Mbit/s).

Until now, we have not defined the length of a slot (in seconds). If we assume that the parameter σ models the availability of the output channel (see further), the slot length t_{slot} is the time needed to send one packet if the channel is available. We define the size of one packet as 100 bytes. Because the bandwidth of the output channel is 100 Mbit/s, the slot length is given by 100 bytes/(100 Mbit/s). So, the slot length t_{slot} is equal to 8 μs.

At http://ita.ee.lbl.gov/html/contrib/EPA-HTTP.html, a trace of real web traffic can be found. A trace is a log file that contains all the file requests of clients during a whole day. We removed the invalid and empty requests from it, and extracted the time stamp and the byte size of all the file requests. Each file request then corresponds to a session. If we look at the byte sizes of the files occurring in the trace, it is possible to approximate the distribution of the file size by a geometric distribution, for the smallest byte sizes (a size smaller than 6000 bytes). For higher file sizes, the geometric distribution is a coarser approximation: the distribution of the actual file sizes is heavy-tailed [7]. The files on a web server are typically either small or either very large [2], hence this result. We neglect this heavy tail and as a result we obtain that the mean file size (in the trace) is equal to 1862 bytes. To derive a value for the mean session length T_s (and hence the parameter α), we need this mean file size. We know that the pgf of the total number of packets generated by a session is given by

$$P_{tot}(z) = E[z^{p_{tot}}] = E\left[z^{\sum_{k=1}^{\ell_i} p_k^i}\right] = L(P(z)), \tag{36}$$

where $L(z)$ is given by (4) and $P(z)$ by (35). We match the first moment of this distribution to the mean file size to obtain an equation for the mean session length $T_s = L'(1)$:

$$E[p_{tot}] = L'(1)P'(1) = \frac{E[\text{file size in trace}]}{100 \text{ bytes}}. \tag{37}$$

Table 1. Performance measures

	$\sigma = 0.95$	$\sigma = 0.5$	$\sigma = 0.2$
System load ρ	0.157	0.298	0.745
Mean number of active sessions $A'(1)$	0.025	0.025	0.025
Mean buffer occupancy $U'(1)$	2.756	6.696	47.936
Mean session delay $E[d_s]$	22.723 slots $= 181.78$ μs	51.053 slots $= 408.42$ μs	333.84 slots $= 2670.7$ μs

Solving this equation gives the following result:

$$T_s = 3.13 \text{ slots}\,, \tag{38}$$

and hence

$$\alpha = 0.6805\,. \tag{39}$$

Left to determine is the pgf $S(z)$ of the number of new sessions per slot. Because the slot length is small, we assume that at most one new session begins per slot, so $S(z) = 1-\beta+\beta z$. The parameter β is the probability that one new session starts. We look at a web server that serves a lot of customers during a rush period. If we assume that on average 1 session is started per millisecond, β is equal to 0.008.

The remaining parameter is σ. We assume that σ is a measure for the availability of the output channel and we look at the performance measures for different

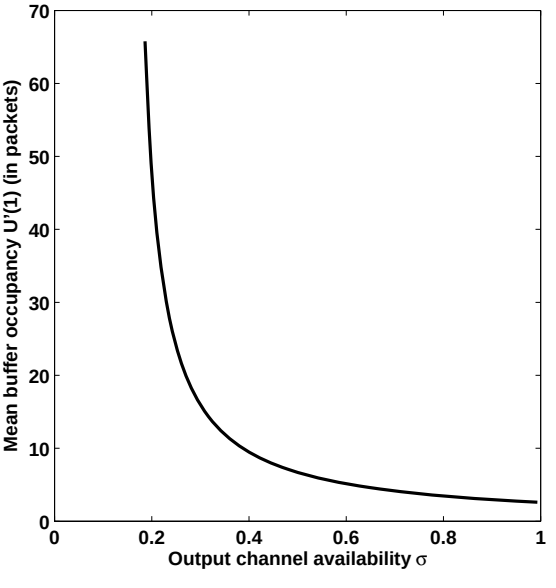


Fig. 3. Mean buffer occupancy (in packets) versus the output channel availability σ

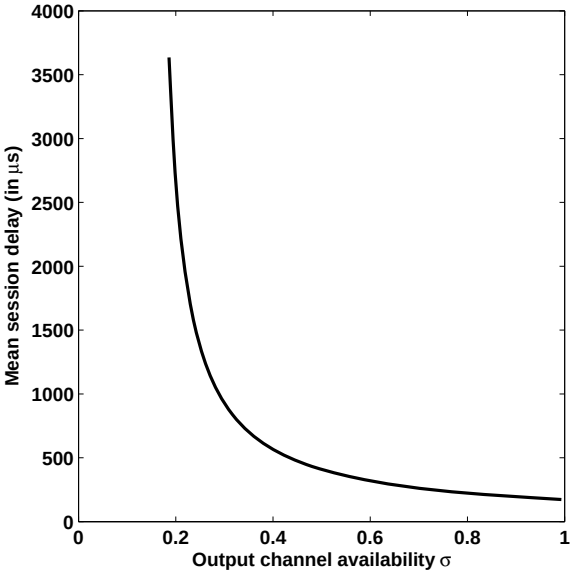


Fig. 4. Mean session delay (in μs) versus the output channel availability σ

values of σ. Figure 3 shows the mean buffer occupancy versus σ, whereas Fig. 4 shows the mean session delay versus σ. Numerical results are also given in Table 1 for three values of σ: $\sigma = 0.95$, $\sigma = 0.5$ (channel available half of the time), and $\sigma = 0.2$ (unreliable channel).

Note that a lower value of σ corresponds with a higher load: the load for $\sigma = 0.2$ is almost five times higher than the load for $\sigma = 0.95$. The results indicate that the mean buffer occupancy and the mean session delay increase with a factor higher than 5. For $\sigma = 0.2$, the system becomes slightly overloaded: the mean session delay has a dimension of milliseconds, while the actual mean transmission time is only $1/\sigma = 40$ μs. The mean number of active sessions is (logically) the same for all values of σ, as it does not depend on σ.

As this example shows, our model gives us the opportunity of quickly comparing different situations corresponding with different values of the system parameters, for dimensioning purposes.

7 Conclusions

In this paper, we have extended the analysis of the queueing model studied in [11], i.e., a discrete-time buffer system with session-based arrival streams. By means of a generating-functions approach, we have derived an explicit expression for the mean session delay.

We also have applied the model to a web server, based on a trace of actual web traffic. It is possible to fit the data to the model, by introducing some restrictions. The most important restriction is the assumption that the session

length is geometric. Allowing non-geometric session lengths can therefore lead to more accurate results. This will be the subject of future research.

Acknowledgment. The first author gratefully acknowledges the financial support from the Research Foundation - Flanders (FWO-Vlaanderen).

References

1. Allman, M., Paxson, V., Stevens, W.: TCP Congestion Control. RFC 2581, Proposed Standard (1999)
2. Arlitt, M., Williamson, C.: Internet Web Servers: Workload Characterization and Performance Implications. IEEE/ACM Transactions on Networking 5, 631–645 (1997)
3. Bruneel, H.: Packet Delay and Queue Length for Statistical Multiplexers with Low-Speed Access Lines. Computer Networks and ISDN Systems 25, 1267–1277 (1993)
4. Bruneel, H.: Calculation of Message Delays and Message Waiting Times in Switching Elements with Slow Access Lines. IEEE Transactions on Communications 42, 255–259 (1994)
5. Bruneel, H., Kim, B.G.: Discrete-Time Models for Communication Systems Including ATM. Kluwer Academic Publishers, Boston (1993)
6. Choi, B.D., Choi, D.I., Lee, Y., Sung, D.K.: Priority Queueing System with Fixed-Length Packet-Train Arrivals. IEE Proceedings-Communications 145, 331–336 (1998)
7. Crovella, M.E., Bestavros, A.: Self-similarity in World Wide Web Traffic: Evidence and Possible Causes. IEEE/ACM Transactions on Networking 5, 835–846 (1997)
8. Daigle, J.: Message Delays at Packet-Switching Nodes Serving Multiple Classes. IEEE Transactions on Communications 38, 447–455 (1990)
9. De Vuyst, S., Wittevrongel, S., Bruneel, H.: Statistical Multiplexing of Correlated Variable-Length Packet Trains: an Analytic Performance Study. Journal of the Operational Research Society 52, 318–327 (2001)
10. De Vuyst, S., Wittevrongel, S., Bruneel, H.: Mean Value and Tail Distribution of the Message Delay in Statistical Multiplexers with Correlated Train Arrivals. Performance Evaluation 48, 103–129 (2002)
11. Hoflack, L., De Vuyst, S., Wittevrongel, S., Bruneel, H.: System Content and Packet Delay in Discrete-Time Queues with Session-Based Arrivals. In: 5th International Conference on Information Technology, ITNG 2008. Las Vegas (2008)
12. Kamoun, F.: Performance Analysis of a Discrete-Time Queuing System with a Correlated Train Arrival Process. Performance Evaluation 63, 315–340 (2006)
13. Takagi, H.: Queueing Analysis – A Foundation of Performance Evaluation. In: Discrete-Time Systems, vol. 3. North-Holland, Amsterdam (1993)
14. Walraevens, J., Wittevrongel, S., Bruneel, H.: A Discrete-Time Priority Queue with Train Arrivals. Stochastic Models 23, 489–512 (2007)
15. Wittevrongel, S.: Discrete-Time Buffers with Variable-Length Train Arrivals. Electronics Letters 34, 1719–1721 (1998)
16. Wittevrongel, S., Bruneel, H.: Correlation Effects in ATM Queues due to Data Format Conversions. Performance Evaluation 32, 35–56 (1998)
17. Xiong, Y., Bruneel, H.: Buffer Behaviour of Statistical Multiplexers with Correlated Train Arrivals. International Journal of Electronics and Communications 51, 178–186 (1997)

Mixed Finite-/Infinite-Capacity Priority Queue with Interclass Correlation

Thomas Demoor*, Joris Walraevens, Dieter Fiems,
and Herwig Bruneel

SMACS Research Group,
Department of Telecommunications and Information Processing,
Ghent University,
St.-Pietersnieuwstraat 41,
B-9000 Gent, Belgium
Tel.: +3292648902
{thdemoor,jw,df,hb}@telin.ugent.be

Abstract. We consider a discrete-time queueing system with two priority classes and absolute priority scheduling. In our model, we capture potential correlation between the arrivals of the two priority classes. For practical use, it is required that the high-priority queue is of (relatively) small size and we hence use a model with finite high-priority queue capacity. We obtain expressions for the probability mass functions of the steady-state system content and delay of the high-priority class as well as for the probability generating functions and moments of the steady-state system content and delay of the low-priority class. The results are compared to those of a similar system, but with an infinite capacity for high priority packets, and it is shown that the latter can be inaccurate. We also investigate the effect of correlation between the arrivals of both priority classes on the performance of the system.

Keywords: Queueing Systems and Network Models, Performance Modelling.

1 Introduction

The huge difference between the Quality of Service (QoS) demands for real-time traffic flows and best-effort traffic flows emburdens packet-based telecommunication networks, such as the Internet. Real-time traffic, such as Voice over IP, can often endure some packet loss but requires low delays and/or low delay jitter. Best-effort traffic benefits from low packet loss, hence avoiding retransmissions, but has less stringent delay characteristics. Therefore the packets are distributed into classes according to their QoS requirements. In the nodes (routers, ...) of the network, packets typically have to wait before being transmitted to the next node and thus constitute a queueing process. The order in which packets are transmitted is based on class-dependent priority rules. This approach to QoS

* Corresponding author.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 61–74, 2008.

differentiation is applied in the DiffServ architecture for Internet Protocol (IP), successfully implemented in corporate networks and debated on as one of the possible approaches to QoS in the future Internet [1].

In this paper, we study a queueing system with a single server supplying two queues, one per priority class, and an Absolute Priority scheduling algorithm, in order to minimize the delay of the high-priority (real-time) packets. The low-priority packets (class 2) are only served if there are no high-priority packets (class 1) in the system. This is the most drastic scheduling method, minimizing class-1 delay at the cost of class-2 performance, but is easy to implement. Analytic studies of queueing systems generally assume infinite queue capacity facilitating the mathematical analysis of the system. In the setting under consideration however, the class-1 packets are delay-sensitive so we require the class-1 queue capacity to be as small as possible while still meeting the required packet loss constraints for this traffic. Note that class-1 traffic that does not arrive at its destination in time is of no use and can be considered lost. We therefore consider a system with finite class-1 queue capacity. On the other hand, the loss-sensitivity of the class-2 packets results in a class-2 queue capacity as large as practically feasible, justifying the assumption of an infinite class-2 queue capacity. Notice that the number of arrivals of both classes can be correlated. A single user often generates packets of both classes simultaneously or no packets at all, yielding positive correlation. On the other hand, negative correlation arises when the number of sources that generate (class-1 or class-2) packets is limited as will be further investigated in the study of an output-queueing switch in the applications.

In the literature, priority queues have been discussed with various arrival and service processes, such as in the contributions [2,3,4,5,6,7]. The presented paper complements [2] where both queues are presumed to be of infinite capacity. As the class-1 queue capacity must be small in order to obtain a low class-1 delay the assumption that this queue has infinite capacity can lead to inaccurate results. Assessing the impact of the finite class-1 queue capacity on the performance measures is the purpose of the present contribution. Finite queue capacity is considered in [5] as well but only the packet loss is investigated profoundly and the delay is not analysed.

This paper is constituted as follows. First the model under consideration will be thoroughly described. In section 3 we investigate the system content for both classes and the class-1 packet loss ratio. Section 4 handles the delay of both classes. Afterwards, the results are applied in some numerical examples. We finally formulate our conclusions.

2 Model

We consider a discrete-time single-server priority queueing system with 2 classes, finite class-1 queue capacity N and an infinite class-2 queue. Class-1 packets are served with absolute priority over class-2 packets and within a class the queueing discipline is First-Come-First-Served (FCFS). The Tail Drop queue management

algorithm is used for the class-1 queue, hence the system accepts packets until the corresponding queue is entirely filled and packets that arrive at a full queue are dropped by the system. Time is divided into fixed-length slots corresponding to the transmission time of a packet. A packet can only enter the server at the beginning of a slot, even if it arrives in an empty system, and its service takes until the end of that slot (deterministic). The system can contain up to $N + 1$ class-1 packets simultaneously in a slot, N in the queue and 1 in the server. Consequently, there are at most N class-1 packets in the system at the beginning of a slot. Also note that a class-1 packet thus resides in the system for at most N slots, which bounds its delay.

We assume that for both classes the number of arrivals in consecutive slots form a sequence of independent and identically distributed (i.i.d.) random variables. We define $a_{i,k}$ as the number of class-i ($i = 1, 2$) packet arrivals during slot k. The arrivals of both classes are characterized by the joint probability mass function (pmf)

$$a(m, n) = \text{Prob}[a_{1,k} = m, a_{2,k} = n] \ , \tag{1}$$

and joint probability generating function (pgf)

$$A(z_1, z_2) = \text{E}[z_1^{a_{1,k}} z_2^{a_{2,k}}] \ . \tag{2}$$

Note that the arrival process allows correlation between both classes. Let the mean number of class-i arrivals per slot (class-i arrival load) be

$$\lambda_i = \text{E}[a_{i,k}] = \left. \frac{\partial A(z_1, z_2)}{\partial z_i} \right|_{z_1=1, z_2=1} , \quad (i = 1, 2) \ . \tag{3}$$

The total arrival load equals $\lambda_T = \lambda_1 + \lambda_2$.

We also define the pgf of the class-2 arrivals in a slot with i (i or more) class-1 arrivals as $A_i(z)$ ($A_i^*(z)$), yielding

$$\begin{aligned} A_i(z) &= \text{E}[z^{a_{2,k}} \mathbf{1}_{a_{1,k}=i}] \ , \\ A_i^*(z) &= \sum_{l=i}^{\infty} A_l(z) \ . \end{aligned} \tag{4}$$

Note that the indicator function $\mathbf{1}_{x=i}$ is 1 if $x = i$ and equals 0 otherwise.

The aim is to express the system content and delay of both classes in terms of the arrival process. The system content at the beginning of a slot is the number of packets contained by the system, thus by the queue or by the server, before packets arrive in the considered slot. The delay of a packet is the number of slots between its arrival slot and the slot after its departure. The class-1 packet loss ratio, this is the fraction of packets that arrive at the system but are not accepted into the system because the class-1 queue is entirely filled, is to be obtained as well.

3 System Content

Let the class-i system content at the beginning of slot k be denoted by $u_{i,k}$. The corresponding joint pgf is referred to as

$$U_k(z_1, z_2) = \mathrm{E}[z_1^{u_{1,k}} z_2^{u_{2,k}}] \ . \tag{5}$$

The (partial) pgf of the class-2 system content in a slot with class-1 system content equal to i is defined as

$$U_{i,k}(z) = \mathrm{E}[z^{u_{2,k}} \mathbf{1}_{u_{1,k}=i}] \ . \tag{6}$$

Note that

$$U_k(z_1, z_2) = \sum_{i=0}^{N} U_{i,k}(z_2) z_1^i \ . \tag{7}$$

Relating the system contents at the beginning of slots k and $k+1$ yields

$$\begin{aligned} u_{1,k+1} &= (u_{1,k} - 1)^+ + a^e_{1,k}, \\ u_{2,k+1} &= \begin{cases} (u_{2,k} - 1)^+ + a_{2,k}, & \text{if } u_{1,k} = 0, \\ u_{2,k} + a_{2,k}, & \text{if } u_{1,k} > 0, \end{cases} \end{aligned} \tag{8}$$

where $(x)^+$ denotes the maximum of x and 0. Due to the finite class-1 capacity, we only take the effectively admitted class-1 arrivals into account. The number of effective class-1 arrivals in slot k, denoted by $a^e_{1,k}$, is clearly influenced by the class-1 system content in slot k. This can be expressed as

$$a^e_{1,k} = \min(a_{1,k}, N - (u_{1,k} - 1)^+) \ . \tag{9}$$

Standard z-transform techniques enable the expression of (8) in terms of pgfs. We establish the system of equations

$$\begin{aligned} U_{i,k+1}(z) =& \frac{1}{z} U_{0,k}(z) A_i(z) + \frac{z-1}{z} U_{0,k}(0) A_i(z) \\ &+ \left(\sum_{j=1}^{i+1} U_{j,k}(z) A_{i-j+1}(z) \right), \quad i = 0 \ldots N-1 \ , \\ U_{N,k+1}(z) =& \frac{1}{z} U_{0,k}(z) A^*_N(z) + \frac{z-1}{z} U_{0,k}(0) A^*_N(z) \\ &+ \left(\sum_{j=1}^{N} U_{j,k}(z) A^*_{N-j+1}(z) \right) \ . \end{aligned} \tag{10}$$

The impact of the finite class-1 queue capacity is apparent when the queue is entirely filled due to i extra effective arrivals. These effective arrivals can correspond with i arrivals, or with $i+1$ arrivals of which one is dropped because the queue is full, or with $i+2$ arrivals of which two are dropped, ... This leads to the appearance of the pgfs $A^*_i(z)$ in the last equation of (10).

Under the assumption that the system reaches steady state, on which we will elaborate at the end of this section, let us define

$$\begin{aligned} U_i(z) &= \lim_{k\to\infty} U_{i,k}(z) = \lim_{k\to\infty} U_{i,k+1}(z), \quad i = 0 \dots N \ , \\ U(z_1, z_2) &= \lim_{k\to\infty} U_k(z_1, z_2) = \sum_{i=0}^{N} U_i(z_2) z_1^i \ . \end{aligned} \tag{11}$$

In steady-state the system of equations (10) becomes

$$\begin{aligned} U_i(z) =& \frac{1}{z} U_0(z) A_i(z) + \frac{z-1}{z} U_0(0) A_i(z) \\ &+ \left(\sum_{j=1}^{i+1} U_j(z) A_{i-j+1}(z) \right), \quad i = 0 \dots N-1 \ , \\ U_N(z) =& \frac{1}{z} U_0(z) A_N^*(z) + \frac{z-1}{z} U_0(0) A_N^*(z) \\ &+ \left(\sum_{j=1}^{N} U_j(z) A_{N-j+1}^*(z) \right) . \end{aligned} \tag{12}$$

We now define the $(N+1) \times (N+1)$ matrix

$$\mathbf{X}(z) = \begin{pmatrix} A_0(z) & A_1(z) & A_2(z) & \cdots & A_{N-1}(z) & A_N^*(z) \\ A_0(z)z & A_1(z)z & A_2(z)z & \cdots & A_{N-1}(z)z & A_N^*(z)z \\ 0 & A_0(z)z & A_1(z)z & \cdots & A_{N-2}(z)z & A_{N-1}^*(z)z \\ 0 & 0 & A_0(z)z & \cdots & A_{N-3}(z)z & A_{N-2}^*(z)z \\ \vdots & \vdots & \ddots & \cdots & \vdots & \vdots \\ 0 & \cdots & \cdots & \cdots & A_0(z)z & A_1^*(z)z \end{pmatrix} , \tag{13}$$

and the row vectors of $N+1$ elements

$$\mathbf{Y}(z) = \begin{pmatrix} A_0(z) \\ A_1(z) \\ \vdots \\ A_{N-1}(z) \\ A_N^*(z) \end{pmatrix}^T , \quad \mathbf{U}(z) = \begin{pmatrix} U_0(z) \\ U_1(z) \\ \vdots \\ U_{N-1}(z) \\ U_N(z) \end{pmatrix}^T . \tag{14}$$

In view of these definitions, the system of equations (12) is equivalent with

$$\mathbf{U}(z) \Big(z\mathbf{I}_{N+1} - \mathbf{X}(z) \Big) = (z-1) U_0(0) \mathbf{Y}(z) \ . \tag{15}$$

Here $\mathbf{I}_n$ denotes the $n \times n$ identity matrix. We have expressed $\mathbf{U}(z)$ in terms of known quantities and the unknown constant $U_0(0)$. For $z = 1$ this yields

$$\mathbf{U}(1) \Big(\mathbf{I}_{N+1} - \mathbf{X}(1) \Big) = \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \ . \tag{16}$$

As $\mathbf{X}(1)$ is a right stochastic matrix we find

$$\text{Rank}\Big(\mathbf{I}_{N+1} - \mathbf{X}(1)\Big) = N \ . \tag{17}$$

We thus require an additional relation in order to determine the $N+1$ unknowns in the vector $\mathbf{U}(1)$. From (6) it is clear that $U_i(1) = \text{Prob}[u_1 = i]$. As the class-1 system content is normalised over the $N+1$ possible states we establish

$$\sum_{i=0}^{N} U_i(1) = 1 \ . \tag{18}$$

By replacing a relation in equation (16) by the normalisation condition we obtain the pmf of the class-1 system content as

$$\mathbf{U}(1) = \begin{pmatrix} 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \Big(\big[\mathbf{I}_{N+1} - \mathbf{X}(1) \big| \mathbf{1}_{N+1}\big] \Big)^{-1} \ . \tag{19}$$

Note that $\mathbf{1}_{N+1}$ is the column vector consisting of $N+1$ ones and that $[\mathbf{A}|\mathbf{B}]$ equals the matrix $\mathbf{A}$ with the last column replaced by $\mathbf{B}$.

We now determine the unknown constant $U_0(0)$, the probability that the system is empty. In steady state, the average number of packets accepted by a system equals the average number of packets leaving that system. Class-1 traffic is not affected by class-2 traffic and consequently class-1 can be seen as an independent system. The mean number of class-1 packets accepted by the system during a slot is denoted by λ_1^e. A class-1 packet leaves the system when the class-1 system content is larger than 0. This leads to $\lambda_1^e = 1 - U_0(1)$. The same reasoning for the system containing both queues yields $\lambda_1^e + \lambda_2 = 1 - U_0(0)$. Bringing these two equations together provides

$$U_0(0) = U_0(1) - \lambda_2 \ . \tag{20}$$

The pgf of the class-2 system content can now be found from (15). The moment-generating property of pgfs enables determination of the moments of the system content. Application of matrix properties significantly expedites the computation of these moments by expressing them in terms of the derivates of the pgfs of the arrival process.

From the class-1 system content we easily obtain the class-1 packet loss ratio plr_1. This is the fraction of class-1 packets that arrive at the system but are dropped. We have

$$plr_1 = \frac{\lambda_1 - \lambda_1^e}{\lambda_1} = 1 - \frac{1 - U_0(1)}{\lambda_1} \ . \tag{21}$$

As the class-1 queue has finite capacity and excess packets are thus dropped the class-1 system is always stable. For the entire system to reach steady state it is imperative that the average number of class-2 packets that is served exceeds the average number of class-2 arrivals, or that $\lambda_2 < 1 - \lambda_1^e$. Notice that requiring that $U_0(0) > 0$ is an equivalent stability constraint.

4 Packet Delay

We tag an arbitrary class-i packet. Let the delay of the packet be denoted by d_i. The arrival slot of the packet is assumed to be slot k. As stated earlier, the class-1 packets are not affected by class-2 traffic. Consequently, the delay of a class-1 packet can easily be obtained from the system content using the distributional form of Little's Theorem [8]. For the pmf of the class-1 delay this leads to

$$d_1(n) = \frac{U_n(1)}{1 - U_0(1)}, \quad n = 1 \dots N \ . \tag{22}$$

For a class-2 packet the analysis is more elaborate. Some preliminary work is performed before we tackle the delay. We first determine the (remaining) class-1 busy period. Next, the extended service completion time of a class-2 packet is defined. We finally establish the number of class-i packets in the system at the end of slot k to be served before the tagged class-2 packet.

The remaining class-1 busy period in slot k, $r_{1,k}$, corresponds with the number of slots until the next slot with class-1 system content equal to 0. Recall that class-1 traffic is unaffected by class-2 traffic. Relating $r_{1,k}$ and $r_{1,k+1}$ and letting k go to infinity results in a system of equations for $R_{1,n}(z)$, the conditional pgf of the remaining class-1 busy period in steady state, at the beginning of a slot during a busy period, if the class-1 system content at the beginning of that slot equals n. We obtain

$$\begin{aligned} R_{1,n}(z) =& z \sum_{m=0}^{N-n} R_{1,n-1+m}(z) A_m(1) \\ &+ z R_{1,N}(z) A^*_{N-n+1}(1), \quad n = 1 \dots N \ . \end{aligned} \tag{23}$$

Note that $R_{1,0}(z) = 1$ as the class-1 busy period ends when the class-1 queue is empty. Again the pgfs $A^*_i(z)$ appear when the class-1 queue is completely filled. Let us define the $N \times N$ matrix

$$\mathbf{M} = \begin{pmatrix} A_1(1) & A_0(1) & 0 & 0 & \cdots & 0 \\ A_2(1) & A_1(1) & A_0(1) & 0 & \cdots & 0 \\ A_3(1) & A_2(1) & A_1(1) & A_0(1) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A_{N-1}(1) & A_{N-2}(1) & A_{N-3}(1) & A_{N-4}(1) & \cdots & A_0(1) \\ A^*_N(1) & A^*_{N-1}(1) & A^*_{N-2}(1) & A^*_{N-3}(1) & \cdots & A^*_1(1) \end{pmatrix} , \tag{24}$$

and the row vectors of N elements

$$\mathbf{L} = \begin{pmatrix} A_0(1) \\ 0 \\ \vdots \\ 0 \end{pmatrix}^T , \quad \hat{\mathbf{R}}(z) = \begin{pmatrix} R_{1,1}(z) \\ R_{1,2}(z) \\ \vdots \\ R_{1,N}(z) \end{pmatrix}^T . \tag{25}$$

In matrix notation the system of equations (23) leads to

$$\hat{\mathbf{R}}(z) = z\mathbf{L}\Big(\mathbf{I}_N - z\mathbf{M}\Big)^{-1} . \tag{26}$$

Note that the relation between $R_{1,0}(z)$ and $R_{1,1}(z)$ is expressed in $\mathbf{L}$. We have determined $\hat{\mathbf{R}}(z)$ and we now extend this vector with $R_{1,0}(z) = 1$ resulting in the row vector of $N + 1$ elements

$$\mathbf{R}(z) = \begin{pmatrix} R_{1,0}(z) \\ R_{1,1}(z) \\ \vdots \\ R_{1,N}(z) \end{pmatrix}^T = \begin{pmatrix} 1 \\ \hat{\mathbf{R}}(z) \end{pmatrix}^T . \tag{27}$$

Notice that a class-1 busy period is simply the remaining class-1 busy period in a random slot preceded by a slot with an empty class-1 system content at the beginning of the slot and a number of arrivals larger than 0. Thus we obtain the pgf of the steady-state class-1 busy period as

$$B_1(z) = \frac{\sum_{m=1}^{N-1} R_{1,m}(z)A_m(1) + R_{1,N}(z)A_N^*(1)}{1 - A_0(1)} . \tag{28}$$

The extended service completion time of a class-2 packet, denoted by t_2, starts at the slot where the packet starts service and lasts until the next slot wherein a class-2 packet can be serviced [10]. If no class-1 packets arrive during the service-slot of the packet, the server can handle another class-2 packet in the next slot. If there are class-1 arrivals, we have to wait for a class-1 busy period after the service-slot until the service of another class-2 packet can start. We can thus express the pgf of the extended service completion time in steady state as

$$T_2(z) = A_0(1)z + \big(1 - A_0(1)\big)B_1(z)z . \tag{29}$$

The number of class-i packets in the system at the end of slot k that have to be served before the tagged class-2 packet is denoted by $v_{i,k}$. Let $u_{i,k}^*$ denote the number of class-i packets that remain in the system during slot k. This equals the class-i system content at the beginning of slot k diminished by 1 if a class-i packet is in service during slot k. As all class-1 packets that arrive during slot k are to be served before the tagged packet it is clear that $v_{1,k} = u_{1,k}^* + a_{1,k}^e$. The class-2 packets that arrive during slot k but after the tagged packet are not to be served before it. Consequently $v_{2,k} = u_{2,k}^* + \hat{a}_{2,k}$ where $\hat{a}_{2,k}$ denotes the number of class-2 arrivals during slot k to be served before the tagged packet. We will now determine some corresponding pgfs.

Foremost we define the steady-state pgfs

$$U_i^*(z) = \lim_{k\to\infty} \mathrm{E}[z^{u_{2,k}^*}\mathbf{1}_{u_{1,k}^*=i}], \quad i = 0 \ldots N-1 . \tag{30}$$

Standard z-transform techniques lead to

$$\begin{aligned} U_0^*(z) =& U_0(0)\frac{z-1}{z} + \frac{U_0(z)}{z} + U_1(z) \ , \\ U_i^*(z) =& U_{i+1}(z), \quad i = 1 \ldots N-1 \ . \end{aligned} \tag{31}$$

The corresponding column vector of N elements is denoted by

$$\mathbf{U}^*(z) = \begin{pmatrix} U_0^*(z) \\ U_1^*(z) \\ \vdots \\ U_{N-1}^*(z) \end{pmatrix} \ . \tag{32}$$

Determination of the number of class-2 arrivals before the tagged packet is a bit more involved. If the arrivals of both classes are correlated it is clear that $a_{1,k}$ and $\hat{a}_{2,k}$ are correlated as well. We again define steady-state pgfs

$$\begin{aligned} \hat{A}_i(z) =& \lim_{k\to\infty} \mathrm{E}[z^{\hat{a}_{2,k}} \mathbf{1}_{a_{1,k}=i}] \ , \\ \hat{A}_i^*(z) =& \sum_{l=i}^{\infty} \hat{A}_l(z) \ . \end{aligned} \tag{33}$$

Taking into account that the tagged class-2 packet is more likely to arrive in a slot with more arrivals [11] the pmf of the number of class-1 and class-2 arrivals in the arrival slot of a tagged class-2 packet is given by

$$\tilde{a}(m,n) = \frac{n a(m,n)}{\lambda_2} \ . \tag{34}$$

The pmf of the number of class-2 arrivals before the tagged packet in a slot with m class-1 arrivals is given by

$$\hat{a}(m,n) = \sum_{l=n+1}^{\infty} \frac{\tilde{a}(m,l)}{l} = \sum_{l=n+1}^{\infty} \frac{a(m,l)}{\lambda_2} \ . \tag{35}$$

Now it is straightforward that

$$\hat{A}_i(z) = \sum_{n=0}^{\infty} \hat{a}(m,n) z^n = \frac{A_i(z) - A_i(1)}{\lambda_2 (z-1)} \ . \tag{36}$$

Analogously we find that

$$\hat{A}_i^*(z) = \frac{A_i^*(z) - A_i^*(1)}{\lambda_2 (z-1)} \ . \tag{37}$$

Let us then define the $(N+1) \times N$ matrix

$$\hat{\mathbf{A}}(z) = \begin{pmatrix} \hat{A}_0(z) & 0 & 0 & \cdots & 0 & 0 \\ \hat{A}_1(z) & \hat{A}_0(z) & 0 & \cdots & 0 & 0 \\ \hat{A}_2(z) & \hat{A}_1(z) & \hat{A}_0(z) & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \hat{A}_{N-2}(z) & \hat{A}_{N-3}(z) & \hat{A}_{N-4}(z) & \cdots & \hat{A}_0(z) & 0 \\ \hat{A}_{N-1}(z) & \hat{A}_{N-2}(z) & \hat{A}_{N-3}(z) & \cdots & \hat{A}_1(z) & \hat{A}_0(z) \\ \hat{A}^*_N(z) & \hat{A}^*_{N-1}(z) & \hat{A}^*_{N-2}(z) & \cdots & \hat{A}^*_2(z) & \hat{A}^*_1(z) \end{pmatrix} . \tag{38}$$

We can now finally describe the class-2 delay. The number of slots a class-2 packet spends in the system equals

$$d_2 = r_{1,k+1} + \sum_{i=1}^{v_{2,k}} t_2 + 1 \ . \tag{39}$$

Keeping in mind that $r_{1,k+1}$ is completely defined by $v_{1,k}$ and that the $u^*_{i,k}$ are independent of the $a_{i,k}$ we find that

$$\begin{aligned} D_2(z) =& \mathrm{E}[z^{d_2}] = \sum_{i=0}^{N} \mathrm{E}[z^{d_2} \mathbf{1}_{v_{1,k}=i}] \\ &= \sum_{i=0}^{N-1} z R_{1,i}(z) \sum_{j=0}^{i} \hat{A}_{i-j}(T_2(z)) U^*_j(T_2(z)) \\ &\quad + z R_{1,N}(z) \sum_{j=0}^{N-1} \hat{A}^*_{N-j}(T_2(z)) U^*_j(T_2(z)) \ . \end{aligned} \tag{40}$$

This can be equivalently expressed as

$$D_2(z) = z \mathbf{R}(z) \hat{\mathbf{A}}(T(z)) \mathbf{U}^*(T(z)) \ . \tag{41}$$

By taking proper derivatives, moments of the class-2 delay can be calculated.

5 Applications

With the formulas at hand we study an output-queueing switch with S inlets and S outlets and two types of traffic as in [2]. On each inlet a batch arrives according to a Bernouilli process with parameter μ_T. A batch contains b (fixed) packets of class 1 with probability μ_1/μ_T or b packets of class 2 with probability μ_2/μ_T (with $\mu_1 + \mu_2 = \mu_T$). The incoming packets are then routed uniformely to the outlets where they arrive at a queueing system as described in this paper.

Therefore all the outlets can be considered identical and analysis of one of them is sufficient. The arrival process at the queueing system can consequently be described by the pmf

$$a(bn, bm) = \frac{S!\left(\frac{\mu_1}{S}\right)^n \left(\frac{\mu_2}{S}\right)^m \left(1 - \frac{\mu_T}{S}\right)^{S-n-m}}{n!m!(S-n-m)!}, \quad n+m \leq S \ , \tag{42}$$

and by $a(p, q) = 0$, for other values of p and q. Obviously the number of arrivals of class-1 and class-2 are negatively correlated. For instance in a slot with x class-1 arrivals there can be no more than $Sb - x$ class-2 arrivals. For increasing values of S the correlation increases and for S going to infinity the numbers of arrivals of both types become uncorrelated.

We now study an 8×8 output-queueing switch. Assume $b = 4$ and $\mu_1 = \mu_2 = 0.1$ yielding $\lambda_1 = \lambda_2 = 0.4$. On average the system thus receives the same amount of packets of both classes. In Fig. 1 the mean and the standard deviation (jitter) of the delay of both classes are plotted versus the class-1 queue capacity N. We clearly see the effect of the priority scheduling. The low mean and standard deviation for the class-1 delay give us the performance required for real-time traffic at the cost of the class-2 performance measures. The values increase for increasing N, as the number of dropped class-1 packets decreases. For larger N the values clearly converge to the values corresponding with the infinite system [2], represented by the dashed lines. However, the convergence is rather slow, especially for the class-2 delay.

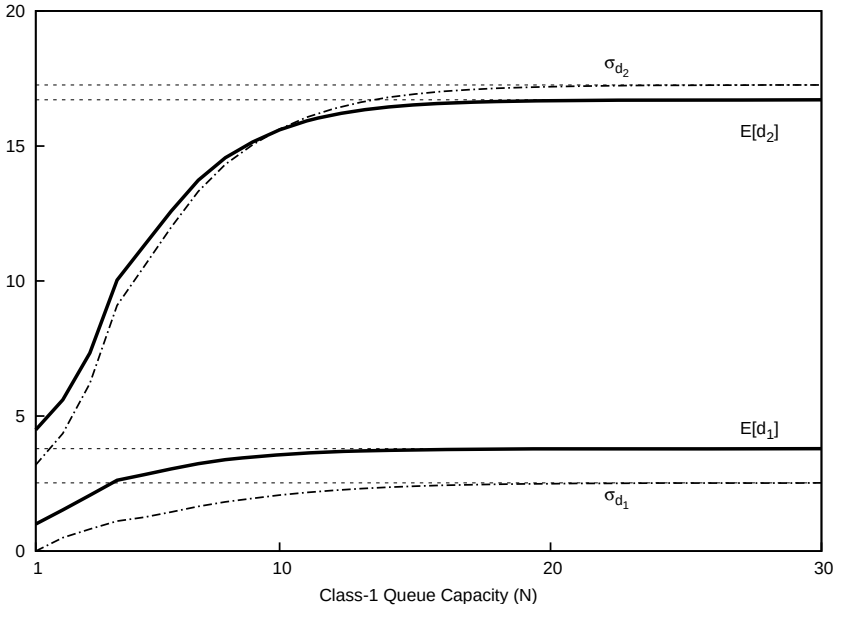


Fig. 1. Delays versus class-1 queue capacity

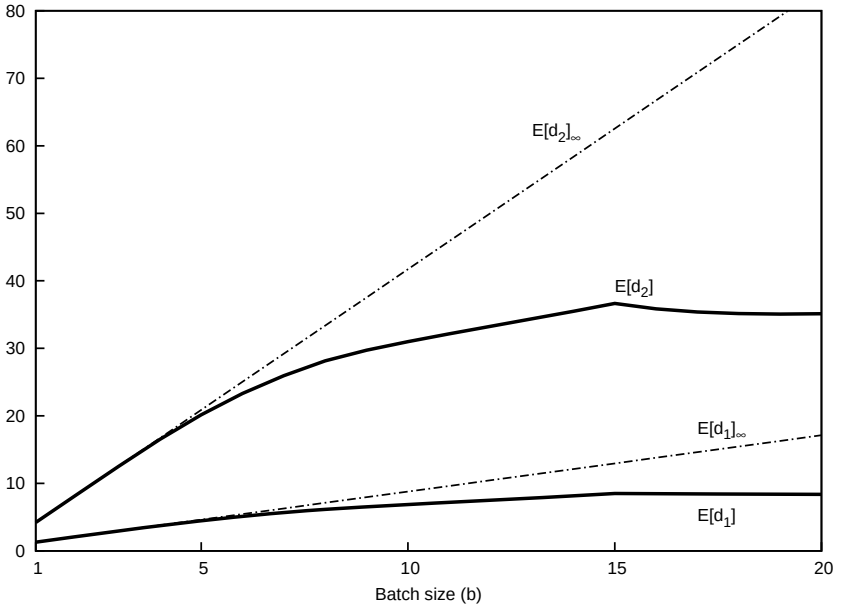


Fig. 2. Mean delays versus batch size

Now assume $N = 15$ and $\lambda_1 = \lambda_2 = 0.4$. We increase the batch size b while adjusting the μ_i accordingly in order to keep the λ_i constant. For increasing b the system thus receives the same amount of packets but the variance of the number of arrivals increases. In Fig. 2, we depict the mean delay of both classes versus the batch size b (as well as the mean delays of the infinite system). We clearly see that the delay increases and that the infinite system leads to inaccurate results when the variance in the arrival process increases. Since in practice arrival processes with high variance are very common, this proves that the infinite model can be imprecise. The decrease of the mean delays for $b > 15$ can be attributed to a high loss rate since for $b > 15$ the batch size exceeds the class-1 queue capacity.

In the $S \times S$ output-queueing switch only a moderate amount of (negative) correlation is present. In order to study the correlation between both classes profoundly, we end this section with the results for a very simple arrival process. A batch of class i arrives according to a Bernoulli distribution with parameter μ_i. A batch contains b (fixed) packets and thus $\lambda_i = b\mu_i$. The joint pmf is given by

$$\begin{aligned} a(0,0) =& 1 - \mu_1 - \mu_2 + c, \\ a(b,0) =& \mu_1 - c, \\ a(0,b) =& \mu_2 - c, \\ a(b,b) =& c. \end{aligned} \tag{43}$$

Notice that this arrival process allows the arrival of a batch of each class in a slot. The concurrence of the arrivals of both classes is controlled by the parameter c.

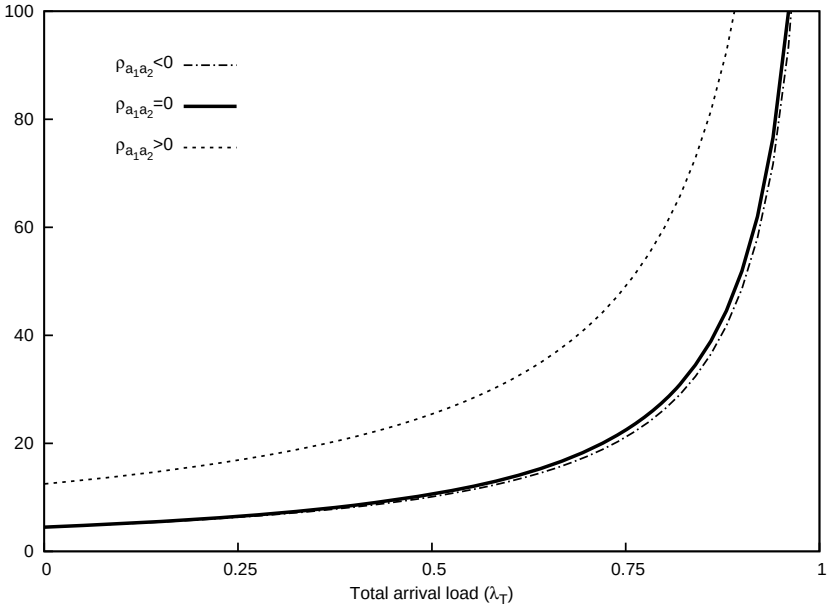


Fig. 3. Mean class-2 delay versus total load for various values of $\rho_{a_1 a_2}$

The correlation factor is given by

$$\rho_{a_1 a_2} = \frac{c - \mu_1 \mu_2}{\sqrt{\mu_1 \mu_2 (1 - \mu_1)(1 - \mu_2)}}. \tag{44}$$

By varying the value of c, while keeping the μ_i constant, we can alter the correlation between both classes. For $c = 0$ there are no slots in which a batch of each class arrives and thus the correlation is minimal ($\rho_{a_1 a_2} < 0$). For $c = \mu_1 \mu_2$ there is no correlation ($\rho_{a_1 a_2} = 0$), while for $c = \min(\mu_1, \mu_2)$ a batch of the class with the lowest arrival rate always arrives in a slot wherein a batch of the other class arrives, yielding (maximum) positive correlation ($\rho_{a_1 a_2} > 0$).

In Fig. 3 we depict the mean class-2 delay versus the total arrival load (λ_T) for the three values for c mentioned above. Assume $N = 15$, $b = 8$ and $\lambda_1 = \lambda_2$. The increase in mean delay between the uncorrelated case and the positively correlated case is remarkable, especially for higher values of λ_T. This follows from the fact that positive correlation between the arrivals of both classes increases the probability that a class-2 packet arrives in the same slot as a class-1 batch, its delay then more frequently includes service of an entire class-1 batch. For negative correlation the inverse effect is established but its influence is less noticeable in this example.

6 Conclusion

We have determined the probability mass functions of the high-priority (class-1) system content and delay and the probability generating functions of the

low-priority (class-2) system content and delay in a two-class priority queue with finite capacity for the high-priority packets. The class-1 packet loss ratio was also obtained. From these formulas it was shown that the infinite class-1 queue capacity approximation, that is frequently used, can yield inaccurate results. The presented model takes the exact class-1 queue capacity into account allowing the determination of precise values for the performance measures even when the class-1 queue capacity is small. In practice one needs to compromise between delay and allowed packet loss in order to determine a suitable class-1 queue capacity N. Once N is chosen the performance measures of both classes can be obtained as in this paper. It is also apparent that correlation between the arrivals of the different classes can have a huge impact on the performance measures and thus should not be considered negligible.

Acknowledgement. The second and third authors are Postdoctoral Fellows with the Fund for Scientific Research, Flanders (F.W.O.-Vlaanderen), Belgium.

References

1. Carpenter, B.E., Nichols, K.: Differentiated services in the Internet. Proceedings of the IEEE 90(9), 1479–1494 (2002)
2. Walraevens, J., Steyaert, B., Bruneel, H.: Performance analysis of a single-server ATM queue with a priority scheduling. Computers & Operations Research 30(12), 1807–1829 (2003)
3. Takine, T., Sengupta, B., Hasegawa, T.: An analysis of a discrete-time queue for broadband ISDN with priorities among traffic classes. IEEE Transactions on Communications 42(2-4), 1837–1845 (1994)
4. Takine, T.: A nonpreemptive priority MAP/G/1 queue with two classes of customers. Journal of Operations Research Japan 39(2), 266–290 (1996)
5. Van Velthoven, J., Van Houdt, B., Blondia, C.: The impact of buffer finiteness on the loss rate in a priority queueing system. In: Horváth, A., Telek, M. (eds.) EPEW 2006. LNCS, vol. 4054, pp. 211–225. Springer, Heidelberg (2006)
6. Mehmet Ali, M., Song, X.: A performance analysis of a discrete-time priority queueing system with correlated arrivals. Performance Evaluation 57(3), 307–339 (2004)
7. Sidi, M., Segall, A.: Structured priority queueing systems with applications to packet-radio networks. Performance Evaluation 3(4), 265–275 (1983)
8. Vinck, B., Bruneel, H.: Delay analysis for single server queues. Electronics Letters 32(9), 802–803 (1996)
9. Fiems, D., Steyaert, B., Bruneel, H.: Discrete-time queues with generally distributed service times and renewal-type server interruptions. Performance Evaluation 55(3-4), 277–298 (2004)
10. Fiems, D.: Analysis of discrete-time queueing systems with vacations. PhD thesis. Ghent University (2003)
11. Bruneel, H., Kim, B.G.: Discrete-time models for communication systems including ATM. Kluwer Academic Publishers, Dordrecht (2004)

Performance Evaluation of a Gradual Differentiation Scheme for Telecommunication Networks

Tom Maertens, Joris Walraevens, and Herwig Bruneel

Ghent University,
Department of Telecommunications and Information Processing,
SMACS Research Group,
Sint-Pietersnieuwstraat 41,
B-9000 Ghent, Belgium
{tmaerten,jw,hb}@telin.UGent.be

Abstract. Supporting different services with different Quality of Service (QoS) requirements is not an easy task in modern telecommunication networks. An efficient priority scheduling discipline is thus of great importance. The static, Head-Of-Line (HOL) discipline achieves differentiation between different types of traffic, but may have a too severe impact on the performance of low-priority traffic. In this paper, we propose a priority discipline with priority jumps: packets of the low-priority level can jump to the high-priority level in the course of time. We use probability generating functions to study the system analytically. Some interesting mathematical challenges thereby arise. The impact of priority jumps on the performance of a telecommunication system is finally evaluated by some numerical examples.

1 Introduction

Modern integrated telecommunication systems are designed to offer a wide variety of services, such as telephony, data transfer, and (interactive) voice and video. Different services however have extremely diverse *Quality-of-Service* (QoS) requirements. Real-time services, like video conferencing or internet telephony, do not tolerate *delay* but can sustain some *loss*, while non-real-time services, like sending data files, allow for some delay, but are quite vulnerable to loss. In this paper, we focus on delay as QoS measure. Regarding their different delay requirements, we then categorize real-time traffic as *delay-sensitive*, and non-real-time traffic as *delay-tolerant*.

To support different types of traffic in modern telecommunication systems, many scheduling disciplines have been proposed over the years. Most of them are based on *priority queueing*: different priority levels for different types of traffic. In the *static*, Head-Of-Line (HOL) priority discipline, delay-sensitive traffic is *always* given the *high-priority* level, and packets of the *low-priority* level are *only* transmitted when there are no high-priority packets in the system. The

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 75–87, 2008.

static priority discipline provides low delays for the delay-sensitive traffic, but when the network is highly loaded and a large portion of the traffic consists of delay-sensitive traffic, it can cause excessive delays for the delay-tolerant traffic (see e.g., [2,9]). Although this type of traffic tolerates a certain amount of delay, extreme values have to be avoided as much as possible. The Transmission Control Protocol (TCP) e.g., could consider a delay-tolerant packet with a too big delay as lost, and would consequently decrease its transmission rate. This would decrease the throughput – which is particularly detrimental to data services – but is unnecessary since the packet is not lost. The delay differentation between both types of traffic may thus be too drastic in some cases.

Dynamic priority disciplines aim for a more *gradual* delay differentiation. Both types of traffic can for example be transmitted in a *weighted* order (see e.g., [5, 8]). In this case, the priority levels do not change, but low-priority packets are with a certain regularity scheduled for transmission before high-priority packets. Another way to reduce performance degradation for the delay-tolerant traffic, is to dynamically *vary* the priority levels with time (see e.g., [3, 4]). For instance, the priority levels of the packets can change according to the system content. A third class of dynamic priority disciplines are discplines with *priority jumps* (see e.g., [6,7]): packets of the low-priority level may in the course of time jump to the high-priority level.

In this paper, we evaluate the performance of a queueing system that adopts a priority discipline with priority jumps. We opt for a straightforward model, so that we can analytically study the effect of priority jumps, and the influence of the system parameters on the performance of the system. The introduction of a jumping parameter β makes the model also very efficient. Indeed, the value of β can be chosen in such a way that the delay-tolerant traffic stays within its delay requirements: e.g., the more stringent the delay requirement, the larger the value of β. The delay differentiation between the different types of traffic can thus be efficiently controlled by the parameter β.

The outline of the paper is as follows. In Section 2, we describe the mathematical model. In Sections 3 and 4, we derive the steady-state system content and study the delays of both types of packets, respectively. Numerical examples are presented in Section 5. Finally, we formulate some conclusions in Section 6.

2 Mathematical Model

We consider a *discrete-time* (i.e., time is *slotted*) queueing system with *two queues* of *infinite capacity* and *one transmission channel*. Two types of packets arrive at the system: type-1 packets, representing delay-sensitive traffic, and type-2 packets, which are delay-tolerant. The numbers of arrivals of both types of packets during slot k are denoted by $a_{1,k}$ and $a_{2,k}$ respectively. We assume that the $a_{1,k}$s and $a_{2,k}$s are independent and identically distributed (i. i. d.) from slot-to-slot. Within one slot however, $a_{1,k}$ and $a_{2,k}$ can be correlated. Their joint distribution is given by the pgf $A(z_1, z_2) \triangleq \mathrm{E}\left[z_1^{a_{1,k}} z_2^{a_{2,k}}\right]$. The marginal pgfs of the numbers of type-1 and type-2 arrivals per slot are then determined by

$A_1(z) \triangleq A(z,1)$ and $A_2(z) \triangleq A(1,z)$ respectively. We denote the total number of arrivals during slot k by $a_{T,k} \triangleq a_{1,k} + a_{2,k}$. Its pgf is given by $A_T(z) \triangleq A(z,z)$. The corresponding arrival rates, i.e., the mean number of arrivals per slot, are indicated by $\lambda_j \triangleq A_j'(1)$ $(j = 1, 2)$ and $\lambda_T \triangleq A_T'(1) = \lambda_1 + \lambda_2$.

Following their delay requirements, arriving type-1 packets enter the *high-priority* queue, while type-2 packets are originally stored in the *low-priority* queue. Within both queues, packets are stored according to a First-In-First-Out (FIFO) order. Packets of the low-priority queue can in the course of time jump to the high-priority queue. The packets of the two queues are transmitted according to a priority rule: when there are packets present in the high-priority queue at the beginning of a slot, they have transmission priority. Only when the high-priority queue is empty at the beginning of a slot, a packet of the low-priority queue can be transmitted. The transmission times of all the packets are deterministically equal to one slot.

Finally, we introduce the following jumping mechanism: at the end of each slot in which a packet of the high-priority queue is transmitted, the whole content of the low-priority queue jumps (or, is swapped) to the high-priority queue with probability β. So, if the high-priority queue is empty at the beginning of a slot, no jump occurs at the end of the slot. Since a possible jump occurs at the end of a slot, type-2 packets that jump in a slot are queued after type-1 packets that arrive during the same slot.

3 System Content

The *system content* is defined as the number of packets present in the system, spread over the two queues. In the assumption that the packet in transmission (if one) is part of the queue that is "served" in that slot, we denote the contents of the high- and low-priority queue at the beginning of slot k as $u_{H,k}$ and $u_{L,k}$ respectively. The system content at the beginning of slot k can then be described by the pair $(u_{H,k}, u_{L,k})$. The total system content, i.e., the total number of packets in the system, is given by $u_{T,k} \triangleq u_{H,k} + u_{L,k}$. In this section, we derive an expression for the joint pgf of the contents of both queues at the beginning of a random slot in the *steady state*. Following *system equations* are established:

– if $u_{H,k} = 0$:

$$u_{H,k+1} = a_{1,k}, \tag{1}$$

$$u_{L,k+1} = [u_{L,k} - 1]^+ + a_{2,k}, \tag{2}$$

– if $u_{H,k} > 0$:
  - with probability β:

$$u_{H,k+1} = u_{H,k} - 1 + a_{1,k} + u_{L,k} + a_{2,k}, \tag{3}$$

$$u_{L,k+1} = 0, \tag{4}$$

- with probability $1-\beta$:

$$u_{H,k+1} = u_{H,k} - 1 + a_{1,k}, \tag{5}$$
$$u_{L,k+1} = u_{L,k} + a_{2,k}, \tag{6}$$

where $[\cdots]^+$ denotes the maximum of the argument and zero. When the high-priority queue is empty at the beginning of slot k, a packet of the low-priority queue (if any) is transmitted during slot k. The arriving packets are queued according to their priority level and no packets jump from the low-priority queue to the high-priority queue (see Eqs. (1) and (2)). When the high-priority queue is non-empty at the beginning of slot k, a packet of the high-priority queue is transmitted during slot k. In this case, the packets of the low-priority queue plus the arriving packets of type 2 jump at the end of slot k with a probability β to the high-priority queue where they are queued behind the packets of type-1 that arrive during slot k (see Eqs. (3) and (4)). Note that when $\beta = 0$, we get the static priority discipline, which has been fully analysed in [9]. The introduction of pgfs in the system equations further yields the following relationship between $U_{k+1}(z_1, z_2)$ and $U_k(z_1, z_2)$:

$$\begin{aligned} U_{k+1}(z_1, z_2) \triangleq & \mathrm{E}\left[z_1^{u_{H,k+1}} z_2^{u_{L,k+1}}\right] \\ = & A(z_1, z_2)\frac{(z_2-1)U_k(0,0) + U_k(0, z_2)}{z_2} \\ & + \beta A_T(z_1)\frac{U_k(z_1, z_1) - U_k(0, z_1)}{z_1} \\ & + (1-\beta)A(z_1, z_2)\frac{U_k(z_1, z_2) - U_k(0, z_2)}{z_1}. \end{aligned} \tag{7}$$

This can be arranged as

$$\begin{aligned} U_{k+1}(z_1, z_2) = & A(z_1, z_2)\frac{\left\{\begin{array}{l} z_1(z_2-1)U_k(0,0) + (1-\beta)z_2 U_k(z_1, z_2) \\ +(z_1 - (1-\beta)z_2)U_k(0, z_2) \end{array}\right\}}{z_1 z_2} \\ & + \beta A_T(z_1)\frac{U_k(z_1, z_1) - U_k(0, z_1)}{z_1}. \end{aligned} \tag{8}$$

When we let $k \to \infty$ to reach the steady state and isolate $U(z_1, z_2)$, we find

$$U(z_1, z_2) = \frac{\left\{\begin{array}{l} z_1(z_2-1)A(z_1, z_2)U(0,0) + (z_1 - (1-\beta)z_2)A(z_1, z_2)U(0, z_2) \\ +\beta z_2 A_T(z_1)[U(z_1, z_1) - U(0, z_1)] \end{array}\right\}}{z_2(z_1 - (1-\beta)A(z_1, z_2))}. \tag{9}$$

Three quantities are needed to further determine $U(z_1, z_2)$: the constant $U(0,0)$ and the functions $U(0, z)$ and $U(z, z)$. First, we compute $U(z, z)$. This is the pgf of the total system content. By substituting z_1 and z_2 by z in (9), we find

$$U(z, z) = U(0,0)\frac{A_T(z)(z-1)}{z - A_T(z)}. \tag{10}$$

This expression is identical to the pgf of the system content of a queue with a FIFO scheduling discipline and with one type of arrivals (determined by $A_T(z)$). This is expected, because for the total system content, it does not matter in which order the packets are being served. Secondly, the constant $U(0,0)$ can be derived by applying the normalization condition $U(1,1) = 1$. We obtain the probability of having an empty system: $U(0,0) = 1 - \lambda_T$.

Thirdly, we determine the *boundary* function $U(0,z)$, which is always the hardest task in this type of two-dimensional queueing systems. By applying Rouché's theorem, it can be shown that for a given value of z_2 in the unit circle ($|z_2| < 1$), the equation $z_1 - (1-\beta)A(z_1, z_2) = 0$ has one solution in the unit circle for z_1 ($|z_1| < 1$). This solution is denoted by $Y(z_2)$, and is implicitly defined by $(1-\beta)A(Y(z_2), z_2)$. Since $z_1 = Y(z_2)$ is a zero of the denominator of the right-hand side of (9), and since $U(z_1, z_2)$ is a pgf and thus remains finite in the unit circle, $Y(z_2)$ must also be a zero of its numerator. Substituting z_1 by $Y(z_2)$ and z_2 by z in the numerator of the right-hand side of (9) yields

$$U(0,z) = \frac{\left\{ \begin{array}{l} (1-\lambda_T)Y(z)A(Y(z),z)(z-1) \\ +\beta z A_T(Y(z))[U(Y(z),Y(z)) - U(0,Y(z))] \end{array} \right\}}{Y(z)(z - A(Y(z),z))}, \tag{11}$$

where we have used the definition of $Y(z)$. By using expression (10) to calculate $U(Y(z),Y(z))$, we can arrange this as

$$U(0,z) = a(z) + b(z)U(0,Y(z)) \tag{12}$$

with

$$\begin{aligned} a(z) =& \beta(1-\lambda_T)\frac{zA_T(Y(z))^2(Y(z)-1)}{Y(z)(Y(z)-A_T(Y(z)))(z-A(Y(z),z))} \\ &+ (1-\lambda_T)\frac{(z-1)A(Y(z),z)}{z-A(Y(z),z)}, \\ b(z) =& \beta\frac{zA_T(Y(z))}{Y(z)(A(Y(z),z)-z)}. \end{aligned} \tag{13}$$

Expression (12) thus yields a relation between $U(0,z)$ and $U(0,Y(z))$. We will show how this relation can be used for an iterative procedure to calculate $U(0,z)$, for all z ($|z| < 1$). We therefore recursively define $Y_i(z)$ as $Y(Y_{i-1}(z))$ ($i \geq 1$), with $Y_0(z) = z$. Based on [1], it can be shown that $Y_i(z) \to C$, for $i \to \infty$ and z any complex number inside the unit disk, and where $C \triangleq (1-\beta)A_T(C)$. By successively applying Eq. (12), using the definition of $Y_i(z)$ and the fact that $\lim_{i\to\infty} Y_i(z) = C$, we obtain

$$\begin{aligned} U(0,z) =& a(z) + b(z)U(0,Y_1(z)) \\ =& a(z) + b(z)a(Y_1(z)) + b(z)b(Y_1(z))U(0,Y_2(z)) \\ =& \cdots \\ =& \sum_{k=0}^{\infty} a(Y_k(z)) \prod_{l=0}^{k-1} b(Y_l(z)) + U(0,C)\prod_{l=0}^{\infty} b(Y_l(z)) \end{aligned} \tag{14}$$

To calculate the constant $U(0, C)$, we use Rouché's theorem on Eq. (11). In particular, we can prove that the equation $z - A(Y(z), z) = 0$ has one solution in the unit circle ($|z| < 1$). This solution, denoted by $F \triangleq A(Y(F), F)$, is thus a zero of the denominator. However, since $U(0, z)$ is analytic for $|z| < 1$, the numerator of (11) must also vanish for $z = F$. This yields

$$U(0, Y(F)) = (1 - \lambda_T) \left\{ \frac{Y(F)A(Y(F), F)(F - 1)}{\beta F A_T(Y(F))} + \frac{A_T(Y(F))(Y(F) - 1)}{Y(F) - A_T(Y(F))} \right\}, \tag{15}$$

where we have used expression (10). Substitution of z by $Y(F)$ in (14) further gives us a second expression for $U(0, Y(F))$. By then combining both expressions, we can finally determine $U(0, C)$.

All unknown quantities in (9) are now calculated, and we can thus derive a semi-analytic expression for the joint pgf $U(z_1, z_2)$. If we then substitute z_1 and z_2 by the appropriate values, we can also obtain the marginal pgfs $U_H(z) \triangleq U(z, 1)$ and $U_L(z) = U(1, z)$ of the contents of the high- and low-priority queue respectively. From these marginal pgfs, expressions for the moments can be derived by invoking the moment generating property of pgfs. They are however omitted because of their size.

4 Packet Delay

The *packet delay* is defined as the total amount of time that a packet spends in the system, i.e., the number of slots between the end of the packet's arrival slot and the end of its departure slot. Assuming that the system is in the steady state, we denote the delay of a type-j packet as d_j. In this section, we derive the pgfs of d_1 and d_2.

4.1 Type-1 Packet Delay

Let us first consider a random but "tagged" type-1 packet that arrives at the system. We mark the arrival slot of the packet as slot I. Since a possible jump of the content of the low-priority queue to the high-priority queue takes place at the end of a slot, the type-1 packets that arrive during slot I are stored in front of the type-2 packets that possibly jump in slot I. As a consequence, the delay of the tagged type-1 packet only depends on the content of the high-priority queue at the beginning of slot I ($u_{H,I}$). If $f_{1,I}$ represents the number of type-1 packets that arrive during slot I, but that have to be transmitted before the tagged packet, we can write

$$d_1 = [u_{H,I} - 1]^+ + f_{1,I} + 1. \tag{16}$$

Due to the i.i.d. arrivals from slot-to-slot, $u_{H,I}$ and the content of the high-priority queue at the beginning of an arbitrary slot have the same distribution. For the same reason, $u_{H,I}$ and $f_{1,I}$ are mutually independent. The pgf

$D_1(z)$ of the delay of a random type-1 packet can thus easily be expressed in terms of $U_H(z)$, which can be obtained from $U(z_1, z_2)$, and $A_1(z)$ (see e.g., [9] for a similar procedure):

$$D_1(z) = \frac{A_1(z) - 1}{\lambda_1(z-1)} \{U_H(z) + (z-1)U_H(0)\}. \tag{17}$$

4.2 Type-2 Packet Delay

Secondly, because of the priority scheduling, it is not straightforward to determine an expression for the pgf $D_2(z)$ of the delay of a random type-2 packet (see also e.g., [9]). Moreover, we have to take into account the possibility that type-2 packets jump to the high-priority queue during their waiting time. Let us tag an arbitrary type-2 packet that enters the system, and again denote its arrival slot by slot I.

The packets that are in the system at the end of slot I and that have to be transmitted before the tagged packet, are referred to as the *primary packets*. The tagged type-2 packet can only be transmitted when all primary packets and all type-1 packets that arrive while the tagged packet is waiting in the low-priority queue, are transmitted. Indeed, new type-1 packets can arrive while a primary packet is transmitted. When the tagged packet is still in the low-priority queue then, these type-1 packets get priority and are scheduled for transmission before the tagged packet. We say that the primary packet adds a so-called *sub-busy period* to the delay of the tagged packet. A sub-busy period *initiated by a packet* starts at the beginning of the slot in which the packet is transmitted, and ends at the beginning of the slot where – for the first time – the number of packets that have to be transmitted before the tagged packet, is one less than at the beginning of the sub-busy period.

When the tagged packet arrives, three possible situations may occur: no packet is in transmission, a packet of the low-priority queue is in transmission, or a packet of the high-priority queue is in transmission. Following equations for d_2 can be derived for the three cases:

- no packet is in transmission during slot I ($u_{H,I} = u_{L,I} = 0$)

$$d_2 = \sum_{m=1}^{f_{1,I}} v_m + \sum_{m=1}^{f_{2,I}} w_m + 1, \tag{18}$$

- a packet of the low-priority queue is in transmission during slot I ($u_{H,I} = 0, u_{L,I} > 0$)

$$d_2 = \sum_{m=1}^{f_{1,I}} v_m + \sum_{m=1}^{u_{L,I}-1+f_{2,I}} w_m + 1, \tag{19}$$

– a packet of the high-priority queue is in transmission during slot I ($u_{H,I} > 0$)

$$d_2 = \sum_{m=1}^{u_{H,I}-1+f_{1,I}} v_m + \sum_{m=1}^{u_{L,I}+f_{2,I}} w_m + 1., \tag{20}$$

where $u_{H,I}$ and $u_{L,I}$ give the contents of the high- and low-priority queue at the beginning of slot I, where $f_{1,I}$ and $f_{2,I}$ represent the type-1 and type-2 packets that arrive during slot I and that have to be transmitted before the tagged packet, and where v_m and w_m denote the lengths of the m-th sub-busy periods initiated by a packet residing in the high- and low-priority queue in slot I, or by a type-1 and a type-2 packet arriving during slot I. The introduction of pgfs in these equations yields

$$\begin{aligned}
D_2(z) \triangleq & \mathrm{E}\left[z^{d_2}\right] \\
= & \mathrm{E}\left[z^{d_2}\{u_{H,I} = u_{L,I} = 0\}\right] \\
& + \mathrm{E}\left[z^{d_2}\{u_{H,I} = 0, u_{L,I} > 0\}\right] \\
& + \mathrm{E}\left[z^{d_2}\{u_{H,I} > 0\}\right] \\
= & z\mathrm{E}\left[z^{\sum_{m=1}^{f_{1,I}} v_m + \sum_{m=1}^{f_{2,I}} w_m}\{u_{H,I} = u_{L,I} = 0\}\right] \\
& + z\mathrm{E}\left[z^{\sum_{m=1}^{f_{1,I}} v_m + \sum_{m=1}^{u_{L,I}-1+f_{2,I}} w_m}\{u_{H,I} = 0, u_{L,I} > 0\}\right] \\
& + z\mathrm{E}\left[z^{\sum_{m=1}^{u_{H,I}-1+f_{1,I}} v_m + \sum_{m=1}^{u_{L,I}+f_{2,I}} w_m}\{u_{H,I} > 0\}\right],
\end{aligned} \tag{21}$$

with $\mathrm{E}[X\{Y\}] \triangleq \mathrm{E}[X|Y]\,\mathrm{Prob}[Y]$. By conditioning on if and when the content of the low-priority queue jumps to the high-priority queue, we can furthermore consider three cases for a sub-busy period: the tagged packet is still in the low-priority queue at the beginning of the sub-busy period and no jump occurs during the sub-busy period, the tagged packet is still in the low-priority queue at the beginning of the sub-busy period and a jump occurs during the sub-busy period, or there was already a jump before the sub-busy period and the tagged packet is thus already in the high-priority queue. In the last case, new arriving type-1 packets are queued behind the tagged packet, and it takes one slot to decrease the number of packets in front of it by one ($v_m = w_m = 1$). For the other cases, we define the partial generating functions of v_m as

$$\begin{aligned}
V_1(z) &\triangleq \mathrm{E}\left[z^{v_m}\{\text{tagged packet does not jump during } v_m\}\right], \\
V_2(z) &\triangleq \mathrm{E}\left[z^{v_m}\{\text{tagged packet jumps during } v_m\}\right],
\end{aligned}$$

and of w_m as

$$\begin{aligned}
W_1(z) &\triangleq \mathrm{E}\left[z^{w_m}\{\text{tagged packet does not jump during } w_m\}\right], \\
W_2(z) &\triangleq \mathrm{E}\left[z^{w_m}\{\text{tagged packet jumps during } w_m\}\right].
\end{aligned}$$

We now express the terms in (21) as a function of these partial pgfs. The third term for instance can be written as

$$\begin{aligned}
&\text{Prob}\left[u_{H,I} > 0\right] z \times \\
&\quad \left\{\beta \text{E}\left[z^{u_{H,I}-1+f_{1,I}+u_{L,I}+f_{2,I}}\right]\right. \\
&\quad +(1-\beta)V_2(z)\text{E}\left[\left(\sum_{i=1}^{u_{H,I}-1+f_{1,I}} V_1(z)^{i-1} z^{u_{H,I}-1+f_{1,I}-i}\right) z^{u_{L,I}+f_{2,I}}\right] \\
&\quad +(1-\beta)W_2(z)\text{E}\left[V_1(z)^{u_{H,I}-1+f_{1,I}} \sum_{i=1}^{u_{L,I}+f_{2,I}} W_1(z)^{i-1} z^{u_{L,I}+f_{2,I}-i}\right] \\
&\quad \left. +\,(1-\beta)\text{E}\left[V_1(z)^{u_{H,I}-1+f_{1,I}} W_1(z)^{u_{L,I}+f_{2,I}}\right]\right\}. \qquad (22)
\end{aligned}$$

This can be explained as follows. When there was a jump at the end of slot I (with probability β), the tagged packet is thus already in the high-priority queue at the beginning of its delay. It then has to wait there until all primary packets, one per slot, are transmitted (first term). When there was no jump at the end of slot I (with probability $1-\beta$), three things can happen with the tagged packet during its waiting time: it jumps during one of the sub-busy periods initiated by the primary packets of the high-priority queue (second term), it jumps during one of the sub-busy periods initiated by the primary packets of the low-priority queue (third term), or it does not jump at all (fourth term). Using the definitions of the partial pgfs leads to (22). Similar expressions can be found for the first and the second term of (21).

Finally, we determine the partial generating functions of v_m and w_m. During the first slot of all sub-busy periods, when the initiating packet is transmitted, type-1 packets arrive at the system. These type-1 packets start sub-busy periods of their own, and these new sub-busy periods are part of the initial sub-busy period. The length of the initial sub-busy period thus equals one (the first slot) plus the sum of the lengths of the sub-busy periods initiated by the type-1 packets that arrive during the first slot, and defined as $v^1_{m,i}$. For a sub-busy period initiated by a packet of the high-priority queue, this yields

$$v_m = 1 + \sum_{i=1}^{a_1} v^1_{m,i}, \qquad (23)$$

with a_1 the number of type-1 arrivals during the first slot of the sub-busy period. For $V_1(z)$, we then obtain

$$V_1(z) = (1-\beta)zA_1(V_1(z)). \qquad (24)$$

Indeed, in this case, there are no jumps during the complete v_m. There is thus no jump at the end of the first slot (with probability $1-\beta$), and there are no jumps during the various $v^1_{m,i}$s. Consequently, these $v^1_{m,i}$s are stochastically indistinguishable from v_m, and they all have the same generating function $V_1(z)$. The

$v^1_{m,i}$s are moreover mutually independent, since they depend on the number of type-1 arrivals during different slots, and we thus find (24).

For the calculation of $V_2(z)$, we also start with Eq. (23), but we now consider an occurrence of a jump during v_m. By z-transforming Eq. (23), we get

$$V_2(z) = (1-\beta)zV_2(z)\mathrm{E}\left[\sum_{i=1}^{a_1} V_1(z)^{i-1}z^{a_1-i}\right] + \beta z A_1(z). \tag{25}$$

Either there is no jump at the end of the first slot of v_m (with probability $1-\beta$) and there thus occurs a jump during at least one of the $v^1_{m,i}$s, or there is a jump at the end of the first slot of the sub-busy period (with probability β) and from that moment on, the tagged packet is in the high-priority queue causing all $v^1_{m,i}$s to be equal to one. Computing the sum in (25) further leads to

$$V_2(z) = (1-\beta)zV_2(z)\frac{A_1(z) - A_1(V_1(z))}{z - V_1(z)} + \beta z A_1(z). \tag{26}$$

If we use expression (24) in (26) and isolate $V_2(z)$, we obtain

$$V_2(z) = \beta\frac{A_1(z)(z - V_1(z))}{1-(1-\beta)A_1(z)}. \tag{27}$$

When the high-priority queue is empty at the beginning of a slot, a packet of the low-priority queue is transmitted in the slot. This type-2 packet also starts a sub-busy period with new sub-busy periods, initiated by the type-1 packets arriving during its first slot, being part of it. However, at the end of the first slot, the content of the low-priority queue cannot jump to the high-priority queue, because of the studied jumping policy (see Eqs. (1) and (2)). For $W_1(z)$, we thus find

$$W_1(z) = zA_1(V_1(z)). \tag{28}$$

We follow a similar reasoning for $W_2(z)$. The high-priority queue is empty at the beginning of the first slot of w_m, and no jump occurs at the end of this slot. There is thus an occurrence of a jump during one of the $v^1_{m,i}$s. We get

$$W_2(z) = zV_2(z)\frac{A_1(z) - A_1(V_1(z))}{z - V_1(z)}. \tag{29}$$

Substituting (27) in Eq. (29), yields

$$W_2(z) = \beta\frac{zA_1(z)(A_1(z) - A_1(V_1(z)))}{1-(1-\beta)A_1(z)}. \tag{30}$$

Hence, $V_2(z)$, $W_1(z)$ and $W_2(z)$ can all be expressed as a function of $V_1(z)$, which in turn is implicitly defined by (24). All functions needed to determine $D_2(z)$ are then derived. Taking into account that $u_{H,I}$ and $u_{L,I}$ in slot I have

the same distribution as u_H and u_L in an arbitrary slot, due to the uncorrelated nature of the arrival process from slot-to-slot, and using the expression of $U(z_1, z_2) = \mathrm{E}\left[z_1^{u_H} z_2^{u_L}\right]$, we can further compute (22) and (21). This finally returns an expression for the pgf $D_2(z)$ of the delay of a random type-2 packet.

By taking the first derivatives of $D_1(z)$ and $D_2(z)$ for $z = 1$, we can get semi-explicit expressions for the mean delays $\mathrm{E}[d_1]$ and $\mathrm{E}[d_2]$ of a random type-1 and a random type-2 packet respectively. By taking higher order derivatives for $z = 1$, expressions for higher moments can also be obtained. The expressions for the pgfs and the moments of the packet delays are omitted because they take up too much space. The mean values are however illustrated in figures in the next section.

5 Numerical Example

In the previous section, we have described the procedures to obtain expressions for the mean packet delays of both types of traffic. In this section, we present some numerical examples. Specifically, we illustrate the effect of priority jumps on the mean packet delays, in comparison with the static, HOL priority discipline and the FIFO discipline. We consider the following arrival process:

$$A(z_1, z_2) = \left(1 - \frac{\lambda_1}{16}(1 - z_1) - \frac{\lambda_2}{16}(1 - z_2)\right)^{16}, \tag{31}$$

with λ_1 and λ_2 the arrival rates of type-1 (delay-sensitive) and type-2 (delay-tolerant) traffic respectively. This is the arrival process to a queue in an 16x16 output-queueing switch with Bernoulli arrivals at its inlets, and with uniform routing. We also define α as the fraction of delay-sensitive traffic in the overall traffic mix (i.e., $\alpha = \lambda_1/\lambda_T$, with $\lambda_T = \lambda_1 + \lambda_2$).

Fig. 1 shows the mean packet delays for both types of traffic as functions of α, when $\beta = 0.1$ and 0.4 respectively and when the total arrival rate λ_T equals 0.9. As one expects, the priority discipline with priority jumps has a lower $\mathrm{E}[d_2]$

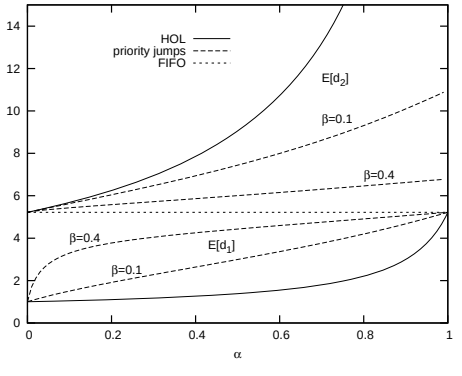


Fig. 1. Mean packet delays versus α when $\lambda_T = 0.9$

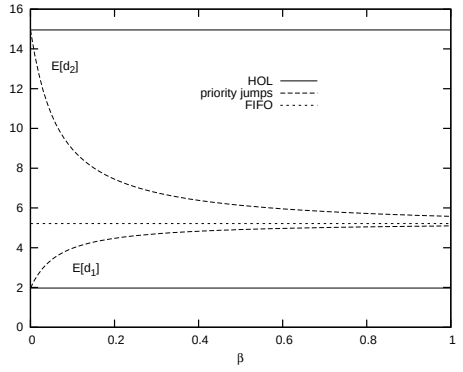


Fig. 2. Mean packet delays versus β when $\alpha = 0.75$ and $\lambda_T = 0.9$

and a higher $\mathrm{E}[d_1]$ than the static HOL priority discipline (for $0 < \alpha < 1$). We further see that when α increases, $\mathrm{E}[d_1]$ and $\mathrm{E}[d_2]$ increase. Indeed, more type-1 packets enter the system when α increases, and more type-2 packets thus suffer from larger delays. Increasing α also means that the probability of having an empty high-priority queue decreases and that more occasions arise for the content of the low-priority queue to jump. A higher increase of $\mathrm{E}[d_1]$ and a repressed increase of $\mathrm{E}[d_2]$ compared to the static priority discipline is the logic consequence.

When α is high, i.e., when a large portion of the traffic consists of delay-sensitive traffic, this then results in the intended fact that excessive delays for the delay-tolerant traffic can be avoided. E.g., when $\beta = 0.1$ and $\alpha = 0.9$, $\mathrm{E}[d_2]$ decreases from about 25.2 for the static priority discipline to 10.1 for the priority discipline with priority jumps. A higher value of β can further considerably decrease $\mathrm{E}[d_2]$ here (i.e., from 10.1 when $\beta = 0.1$ to 6.6 when $\beta = 0.4$), while the difference for $\mathrm{E}[d_1]$ is limited (i.e., 4.6 when $\beta = 0.1$ compared to 5.0 when $\beta = 0.4$).

In Fig. 2, we illustrate the mean packet delays for both types of traffic as functions of β, when $\alpha = 0.75$ and $\lambda_T = 0.9$. When $\beta = 0$, the analysed scheduling discipline with priority jumps equals a static priority scheme, as already mentioned in Section 3. The larger β, and thus the more jumps, the lower the negative effect from the priority scheduling on $\mathrm{E}[d_2]$. The price to pay is a higher $\mathrm{E}[d_1]$. The parameter β can be chosen according to the delay requirements of both types of traffic. A low β e.g., will highly favour the delay-sensitive traffic, while chosing β higher will achieve a limited delay differentiation between both types of traffic. This is practible if there is little difference in their delay requirements.

6 Conclusions

In this paper, we have evaluated the impact of priority jumps on the performance of a two-class priority queueing system. We have used probability

generating functions to analyse the system content and the packet delays. Some mathematical challenges, like the determination of a boundary function and the study of the low-priority packet delay, are thereby efficiently overcome. Probability generating functions are furthermore useful for the calculation of important performance measures, such as the mean values of the packet delays. The merit of the paper also lies in the model that is analysed. Indeed, we have chosen a straighforward and efficient model that perfectly meets the purpose of dynamic priorities: a gradual delay differentiation between different types of traffic. The impact of dynamic priorities on the system performance can be controlled by a single parameter.

Acknowledgment

The second author is a Postdoctoral Fellow with the Fund for Scientific Research, Flanders (F.W.O.-Vlaanderen), Belgium.

References

1. Abate, J., Whitt, W.: Solving probability transform functional equations for numerical inversion. Operations Research Letters 12, 275–281 (1992)
2. Bae, J.J., Suda, T.: Survey of traffic control schemes and protocols in ATM networks. IEEE/ACM Transactions on Networking 2(5), 508–519 (1994)
3. Choi, D.I., Choi, B.D., Sung, D.K.: Performance analysis of priority leaky bucket scheme with queue-length-threshold scheduling policy. IEEE Proceedings-Communications 145(6), 395–401 (1998)
4. Ganos, P.A., Koukias, M.N., Kokkinakis, G.K.: ATM switch with multimedia traffic priority control. European Transactions on Telecommunications 7(6), 527–540 (1996)
5. Kuurne, A., Miettinen, A.P.: Weighted round robin scheduling strategies in (e)gprs radio interface. In: Proceedings of the Vehicular Technology Conference (VTC2006-Fall), vol. 5, pp. 3155–3159 (2004)
6. Lim, Y., Kobza, J.E.: Analysis of a delay-dependent priority discipline in an integrated multiclass traffic fast packet switch. IEEE Transactions on Communications 38(5), 659–685 (1990)
7. Maertens, T., Walraevens, J., Bruneel, H.: On priority queues with priority jumps. Performance Evaluation 63(12), 1235–1252 (2006)
8. Parekh, A.K., Gallager, R.G.: A generalized processor sharing approach to flow control in integrated services network: the single node case. IEEE/ACM Transactions on Networking 1(3), 344–357 (1993)
9. Walraevens, J., Steyaert, B., Bruneel, H.: Performance analysis of a single-server ATM queue with a priority scheduling. Computers and Operations Research 30(12), 1807–1829 (2003)

An Analytical Study of the Resource Diffusion in Non-homogeneous P2P Networks

D. Manini and M. Gribaudo

Dipartimento di Informatica
Università di Torino
Corso Svizzera 185, 10149 Torino, Italy

Abstract. One of the most popular P2P application is file sharing. Its purpose is to spread out different contents, ranging from multimedia to data and software. In order to understand the dynamics of such application, it is important to study the movement of shared resources across the network. We consider a non-homogeneous distribution of the resources, by grouping peers into Autonomous Systems. We propose a probabilistic model that takes into account the peers behavior, and we exploit it to study the resource diffusion and the impact of freeloaders.

1 Introduction

The data flow originated nowadays by peer-to-peer (P2P) applications characterizes most of the Internet traffic[1]. P2P systems can be used in different contexts, such as file sharing (Gnutella, KaZaa, eDonkey, BitTorrent to name a few), telephony applications (Skype [2]), and content delivery infrastructures (see [5]). One of the most popular P2P application is file sharing. The service provided by this application is to spread out contents such as multimedia and software. The size of resources ranges from several kilobytes up to some gigabytes. In order to understand the dynamics of such application, it is interesting to study how shared resources move across the network.

In this paper we study the diffusion of a resource in a non-homogeneous environment. Peers are grouped into Autonomous Systems (AS), each one having its own parameters in terms of resource availability and demand. We propose a probabilistic model that describes the diffusion of a resource taking into account (non-homogenous) resource popularity and peers behavior. This work is the first step towards a project that aims at studying the impact of P2P file sharing applications on the overall network traffic, by considering the geographic location of peers and the communication costs. The main advantage of the proposed probabilistic model is that it exploits properties of the z-transform to consider very large models with low computational cost.

The reminder of the paper is as follow. After a brief discussion on some related work (Section 2), in Section 3 we present our probabilistic model, and in Section 4 we exploit it to perform some study on the system.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 88–100, 2008.

2 Related Works

There are several previous works that issues modeling P2P systems, and that are related with our proposal. The work presented in [7] shows a fluid model for the BitTorrent P2P application, and it is able to study steady-state performance measures, such as the number of peers that have a resource and remains in the system to allow its diffusion. Instead in our work we consider the transient behavior, by using an embedded process where time is not considered explicitly.

The simulation techniques proposed in [8] investigate the diffusion of a file in a e-Donkey system, as a function of several parameters such as sharing probability and requests arrival rate. The analytical model developed in [9] is based on biological epidemics. In particular, it is used to predict the diffusion of single files in a P2P network, whereas we focus on the diffusion of a single resource among different A.S.

The probabilistic model presented in [10] is inspired by the study of file swarming in BitTorrent like systems. The measurement-based technique utilized in [11] provides static, topological, and dynamic analysis of the P2P Gnutella environment. The dynamic analysis allows to study the variations in terms of popularity of individual files, and in terms of the number of available files at individual peers. We also compute the resource diffusion, but we focus on the traffic among A.S.

One of the models studied in [6], considers how a document is spread to the requesting peers into a bit-torrent environment. The model proposed in [13] describes P2P dynamics through a set of second order fluid equations, that allows to derive results related to the resource distribution among peers. The models developed in [3,4] are aimed at studying the transfer time distribution of a resource: their goal is to characterize the time required to diffuse a resource from a set of peers holding it. In this paper we do not focus on the transfer time of a single resource, but we consider the whole traffic produced by all the transfers.

3 P2P Resource Diffusion

In this section we describe the probabilistic model. We first define the peer-to-peer scenario (Section 3.1), then we provide the analytical representation of peers (Section 3.2). In Section 3.3 we study the evolution of the system in order to characterize the resource diffusion.

3.1 Network Scenario

We consider that peers are distributed across N different ASs. We assume that the total number of peers in the system is a discrete random variable that follows a given probability distribution $p(m)$. We express this distribution with its z-transform $\mathcal{G}(z)$:

$$\mathcal{G}(z) = \sum_{m=0}^{\infty} p(m) z^m \tag{1}$$

We only take into account peers that can participate in the diffusion, i.e., peers that either hold or request the resource. We denote by s_i the probability that a peer is in the i-th AS.

In each AS peers are divided into three different classes: a) peers holding the resource and available for sharing it, b) peers requiring the resource, and c) peers holding the resource, but not sharing it, i.e. freeloaders [12]. The class of each peer is determined randomly, according to a given initial probability: α_i for class a), β_i for class b) and γ_i for class c) (with $\alpha_i+\beta_i+\gamma_i = 1$). Not that γ_i has no impact on the system behavior, but it is of interest since it allows the investigation of the number of freeloaders. The number of peers of the three classes are respectively denoted by n_i, p_i and q_i. We denote by ξ_i the probability that a peer that gets the resource decides to not share it. Peers requiring the resource can either get it from peers lying in the the same AS or from peers belonging to others ASs. All model notations are summarized in Table 1.

Table 1. Model Notations

Notation	Description
N	Number of Autonomous Systems (AS)
s_i	Probability that a peer belongs to the i-th AS
$\mathcal{G}(z)$	Z-transform of the distribution of the number of peers in the network
α_i	Probability that a peer in the i-th AS holds the resource
β_i	Probability that a peer in the i-th AS wants the resource at the
γ_i	Probability that a peer in the i-th AS holds the resource but do not share it
ξ_i	Probability that a peer in the i-th AS does not share the resource after getting it
n_i	Number of peers holding the resource in the i-th AS
p_i	Number of peers requiring the resource in the i-th AS
q_i	Number of peers holding holding but not sharing in the i-th AS
u_i	Z-transform of the number of peers holding the resource in the i-th AS
v_i	Z-transform of the number of peers requiring the resource in the i-th AS
w_i	Z-transform of the number of peers holding but not sharing in the i-th AS

3.2 The Model

We represent the P2P system by introducing the distribution of the number of peers and its corresponding z-transform.

We call $\mathit{\Pi}(n_1...n_N, p_1...p_N, q_1...q_N)$ the joint distribution of the number of peers in each class, for each of the N ASs. The z-transform $g(\cdot)$ of this distribution can be computed from parameters $\mathcal{G}(z)$, s_i, α_i, β_i and γ_i as:

$$g(u_1...u_N, v_1...v_N, w_1...w_N) = \mathcal{G}\Big(\sum_{i=1}^{N} s_i\Big[\alpha_i u_i + \beta_i v_i + \gamma_i w_i\Big]\Big). \tag{2}$$

An intuitive interpretation of Equation (2), is that each peer randomly chooses both its AS and its class with probability $s_i\alpha_i$, $s_i\beta_i$ or $s_i\gamma_i$, which corresponds to $z = \sum_{i=1}^{N} s_i\Big[\alpha_i u_i + \beta_i v_i + \gamma_i w_i\Big]$.

The marginal distribution corresponding to the i-th AS is defined as:

$$\mathit{\Pi}_i(n_i, p_i, q_i) = \sum_{\substack{n_1 \cdots n_{i-1},\, n_{i+1} \cdots n_N,\\ p_1 \cdots p_{i-1},\, p_{i+1} \cdots p_N,\\ q_1 \cdots q_{i-1},\, q_{i+1} \cdots q_N}} \mathit{\Pi}(n_1...n_N, p_1...p_N, q_1...q_N) \tag{3}$$

We call $g_i(u_i, v_i, w_i)$ the z-transform of $\Pi_i(n_i, p_i, q_i)$. Using the properties of the z-transform and (2), we have that:

$$\begin{aligned} g_i(u_i, v_i, w_i) &= g(1...1, u_i, 1...1, 1...1, v_i, 1...1, 1...1, w_i, 1...1) = \\ &= \mathcal{G}\Big(s_i\Big[\alpha_i u_i + \beta_i v_i + \gamma_i w_i\Big] + 1 - s_i\Big) \end{aligned} \tag{4}$$

where we set to 1 all the transformed variables except the ones corresponding to the i-th AS.

We can compute the probability that a peer in the i-th AS holds the resource, given that the AS is not empty (by empty we mean that there are no peers that can participate in the resource diffusion as mentioned in Section 3.1) as:

$$\bar{\alpha_i} = \sum_{n_i+p_i+q_i \neq 0} \frac{n_i}{n_i + p_i + q_i} \Pi_i(n_i, p_i, q_i). \tag{5}$$

It can be shown that $\bar{\alpha_i}$ can be computed in the following way:

$$\bar{\alpha_i} = \int_0^1 \Big[\frac{\partial g_i(u_i, v_i, w_i)}{\partial u_i}\Big]_{u_i=y, v_i=y, w_i=y} dy. \tag{6}$$

If we calculate (6) with the definition (4) we get:

$$\bar{\alpha_i} = \alpha_i(1 - g_i(0, 0, 0)) \tag{7}$$

Hence we can write:

$$\alpha_i = \frac{\bar{\alpha_i}}{(1 - g_i(0, 0, 0))} = \frac{\bar{\alpha_i}}{(1 - \mathcal{G}(1 - s_i))} \tag{8}$$

The term $1 - \mathcal{G}(1 - s_i)$ corresponds to the probability that the i-th AS is not empty. The same computation can be made for β and γ:

$$\beta_i = \frac{\bar{\beta_i}}{(1 - g_i(0, 0, 0))},\ \bar{\beta_i} = \int_0^1 \Big[\frac{\partial g_i(u_i, v_i, w_i)}{\partial v_i}\Big]_{\substack{u_i = y \\ v_i = y \\ w_i = y}} dy \tag{9}$$

$$\gamma_i = \frac{\bar{\gamma_i}}{(1 - g_i(0, 0, 0))},\ \bar{\gamma_i} = \int_0^1 \Big[\frac{\partial g_i(u_i, v_i, w_i)}{\partial w_i}\Big]_{\substack{u_i = y \\ v_i = y \\ w_i = y}} dy \tag{10}$$

3.3 System Dynamics

We now study the evolution of parameters α_i, β_i and γ_i, and show how they characterize the resource diffusion in the system. In particular, we model the resource diffusion among the ASs with an *embedded time* process: time is not considered explicitly, instead is modeled by a discrete variable m that increases of one unit whenever a resource transfer is completed. With this assumption we compute α_i^m, β_i^m and γ_i^m, that correspond to the values of parameters α_i, β_i and γ_i at time m.

Parameters α and β vary only due to a resource transfer. For sake of simplicity, in this paper we neglect peers that give up requesting the resource and peers that quit sharing

it. However these assumptions could be easily removed by adding new parameters and difference equations on α and β. We expect that as time tends to the infinity, every requests will be satisfied, that is (for any i-th AS):

$$\lim_{m\to\infty} \alpha_i^m = \alpha_i^0 + (1-\xi_i)\beta_i^0 \tag{11}$$

$$\lim_{m\to\infty} \beta_i^m = 0 \tag{12}$$

$$\lim_{m\to\infty} \gamma_i^m = \gamma_i^0 + \xi_i\beta_i^0 \tag{13}$$

where α_i^0, β_i^0 and γ_i^0 represent the initial system parameters.We are interested in studying the evolution of α_i^m, β_i^m and γ_i^m until all transfers are completed. Note that changes in these parameters affect the joint distribution of the number of peers per class, i.e. $\mathit{\Pi}^m(n_1..n_N, p_1..p_N, q_1..q_N)$.

We can define $\bar{\alpha}_i^{m+1}$ as function of the system parameters at time m:

$$\begin{aligned}\bar{\alpha}_i^{m+1} = &\sum_{\substack{(\sum_k p_k = 0 \vee \\ \sum_k n_k = 0) \\ n_i+p_i+q_i \neq 0}} \frac{n_i}{n_i+p_i+q_i}\mathit{\Pi}^m(n_1..n_N, p_1..p_N, q_1..q_N) + \\ &+ \sum_{\substack{(\sum_k p_k \neq 0 \wedge \\ \sum_k n_k \neq 0) \\ n_i+p_i+q_i \neq 0}} \left[\frac{n_i+1}{n_i+p_i+q_i}\frac{p_i}{\sum_k p_k}(1-\xi_i) + \right. \\ &\left. + \frac{n_i}{n_i+p_i+q_i}\left(1-\frac{p_i}{\sum_k p_k}(1-\xi_i)\right)\right] \cdot \\ &\cdot\ \mathit{\Pi}^m(n_1..n_N, p_1..p_N, q_1..q_N)\end{aligned}$$

The first addendum on the r.h.s. accounts for the case in which no transfer occurs since either all requests in the system have been satisfied and there are no more peers requiring the resource ($\sum_k p_k = 0$), or there are no resources in the system ($\sum_k n_k = 0$). The second addendum on the r.h.s. considers the case where a resource is actually transferred. If the destination of the transfer is the i-th AS and the considered peer is not a freeloader (with probability $\frac{p_i}{\sum_k p_k}(1-\xi_i)$), n_i is increased by one (first term in square brackets), otherwise n_i remains constant (second term in square brackets).
By developing (14) we obtain:

$$\begin{aligned}\bar{\alpha}_i^{m+1} = &\sum_{n_i+p_i+q_i\neq 0} \frac{n_i}{n_i+p_i+q_i}\mathit{\Pi}^m(n_1..n_N, p_1..p_N, q_1..q_N) + \\ &+ \sum_{\substack{(\sum_k p_k \neq 0 \wedge \\ \sum_k n_k \neq 0) \\ n_i+p_i+q_i \neq 0}} \frac{1}{n_i+p_i+q_i}\frac{p_i}{\sum_k p_k}\cdot \\ &\cdot\ \mathit{\Pi}^m(n_1..n_N, p_1..p_N, q_1..q_N)(1-\xi_i)\end{aligned} \tag{14}$$

From (5) we can write:

$$\bar{\alpha}_i^{m+1} = \bar{\alpha}_i^m + \mathit{\Delta}_i^m(1-\xi_i) \tag{15}$$

where we define

$$\begin{aligned}\Delta_i^m &= \sum_{\substack{(\sum_k p_k \neq 0 \wedge \\ \sum_k n_k \neq 0) \\ n_i + p_i + q_i \neq 0}} \frac{1}{n_i + p_i + q_i} \frac{p_i}{\sum_k p_k} \Pi^m(n_1..n_N, p_1..p_N, q_1..q_N) \\ &= \sum_{\substack{\sum_k p_k \neq 0 \\ n_i + p_i + q_i \neq 0}} \frac{1}{n_i + p_i + q_i} \frac{p_i}{\sum_k p_k} \Pi^m(n_1..n_N, p_1..p_N, q_1..q_N) \\ &- \sum_{\substack{\sum_k p_k \neq 0 \\ n_i + p_i + q_i \neq 0}} \frac{1}{p_i + q_i} \frac{p_i}{\sum_k p_k} \Pi^m(0..0, p_1..p_N, q_1..q_N). \end{aligned} \tag{16}$$

$\Delta_i^m(1-\xi_i)$ represents the variation of $\bar{\alpha}_i$ at time m. In the same way the evolution of $\bar{\beta}_i$ and $\bar{\gamma}_i$ can be calculated as:

$$\bar{\beta}_i^{m+1} = \bar{\beta}_i^m - \Delta_i^m \tag{17}$$

$$\bar{\gamma}_i^{m+1} = \bar{\gamma}_i^m + \Delta_i^m \xi_i \tag{18}$$

Δ_i^m can be computed exploiting the z-transform representation of the number of peers in the system. We define $\bar{p}_i = \sum_{k \neq i} p_k$ and we denote with $\bar{v}_i$ its corresponding transformed variable. It can be shown that:

$$\Delta_i^m = \int_0^1 \left[\int_0^1 \left[\frac{\partial}{\partial v_i} \hat{g}_i(u_i, v_i, w_i, \bar{v}_i) \right]_{\substack{v_i = xy \\ \bar{v}_i = x}} dx \right]_{\substack{u_i = y \\ w_i = y}} dy. \tag{19}$$

where

$$\begin{aligned} &\hat{g}_i(u_i, v_i, w_i, \bar{v}_i) = g(1...1, u_i, 1...1, \bar{v}_i...\bar{v}_i, v_i, \bar{v}_i...\bar{v}_i, 1...1, w_i, 1...1) - \\ &\qquad g(0...0, \bar{v}_i...\bar{v}_i, v_i, \bar{v}_i...\bar{v}_i, 1...1, w_i, 1...1) = \\ &= \mathcal{G}\Big(1 + s_i\Big[\alpha_i(u_i - 1) + \beta_i(v_i - \bar{v}_i) + \gamma_i(w_i - 1)\Big] + B(\bar{v}_i - 1)\Big) - \\ &\quad \mathcal{G}\Big(1 - A + s_i\Big[\beta_i(v_i - \bar{v}_i) + \gamma_i(w_i - 1)\Big] + B(\bar{v}_i - 1)\Big) \end{aligned} \tag{20}$$

and $A = \sum_k s_k \alpha_k$ and $B = \sum_k s_k \beta_k$.

By calculating (19) with (20) we obtain:

$$\begin{aligned} \Delta_i^m = \int_0^1 \frac{s_i \beta_i}{s_i \beta_i (y-1) + B} \Big[&\mathcal{G}\left(1 + s_i(y-1)\right) - \\ &\mathcal{G}\left(1 - B + s_i(1-\beta_i)(y-1)\right) - \\ &\mathcal{G}\left(1 - A + s_i(1-\alpha_i)(y-1)\right) + \\ &\mathcal{G}\left(1 - A - B + s_i\gamma_i(y-1)\right) \Big] dy. \end{aligned} \tag{21}$$

Finally, from (15), (17), (18) and (21) we are able to compute $\alpha_i^m, \beta_i^m, \gamma_i^m$ for any step m and for each AS i. Given the initial parameters α_i^0, β_i^0 and γ_i^0 we have, by applying (8), (9) and (10), that:

$$\alpha_i^m = \frac{\bar{\alpha}_i^m}{(1 - \mathcal{G}(1 - s_i))} \tag{22}$$

$$\beta_i^m = \frac{\bar{\beta}_i^m}{(1 - \mathcal{G}(1 - s_i))} \tag{23}$$

$$\gamma_i^m = 1 - (\bar{\alpha}_i^m + \bar{\beta}_i^m) \tag{24}$$

4 Experimental Results

In Section 4.1 we validate via simulation the probabilistic model described previously. In Section 4.2 we point out the relevance of the probability distribution that describes peers participating at the resource diffusion. Finally in Section 4.3 we exploit the model to perform some study on the system.

4.1 Validation

We built a simulator that performs one step of the system evolution in order to validate the computation of Δ_i^m. The simulator first creates an instance of a P2P system conforming to the given parameters: it computes, for each AS i, the number of peers having resource (n_i), waiting for it (p_i), and not willing to share it (q_i). It then randomly selects the AS that will receive the resource according to its number of pending requests p_i. Finally it computes the variation of the model parameters α_i, β_i and γ_i caused by the transfer. The process is repeated for a large number of trials.

We considered a scenario with two ASs whose parameters are reported in Table 2:

Table 2. Validation parameters (two ASs)

AS	s_i	α_i^m	β_i^m	γ_i^m	ξ_i
AS_1	0.7	0.25	0.7	0.05	0.0
AS_2	0.3	0.4	0.4	0.2	0.0

We observed the values of Δ_1^{m+1} as function of the mean number of peers. We validate the model for different distributions of the number of peers: geometric, negative binomial [14], and Poisson distributions. The comparison of model and simulation results is reported in Fig. 1. The model results were obtained in few seconds, whereas the simulator required several minutes for $mean(N) > 1000$.

Furthermore, we run the analytical model in order to check that the limiting behavior of Equations (11), (12), and (13) is verified. We considered a topology with three AS's whose initial parameters are reported in Table 3:

Table 3. Validation parameters (three ASs)

AS	s_i	α_i^0	β_i^0	γ_i^0	ξ_i
AS_1	0.5	0.25	0.6	0.15	0.15
AS_2	0.34	0.5	0.4	0.1	0.05
AS_3	0.16	0.7	0.2	0.1	0.2

For this experiment we set the distribution of the number of peers to a geometric distribution with parameter $\rho = 0.001$, i.e. the mean number of peers in the system is $1/\rho = 1000$. We found that for each AS Equations (11), (12), and (13) are verified.The same test has been successfully performed in every experiments. We could not compare directly our results with the ones presented in the literature (such as in [9]), since we do not have an explicit representation of time.

4.2 Settings

The model we propose is mainly based on the representation of the number of peers participating in the resource diffusion. In particular, we use a probability distribution to express the number of peers in the whole system. In this section we study the impact of the distribution of the number of peers on the resource diffusion. We run the model on a three ASs scenario, varying only the distribution of the number of peers. In particular we considered: geometric, negative binomial with k stages and binomial distribution to model different peers behavior. The geometric and the negative binomial distributions are important because of their coefficient of variation. They are identical when the number of stages k is equal to 1, while when $k \to \infty$ the negative binomial

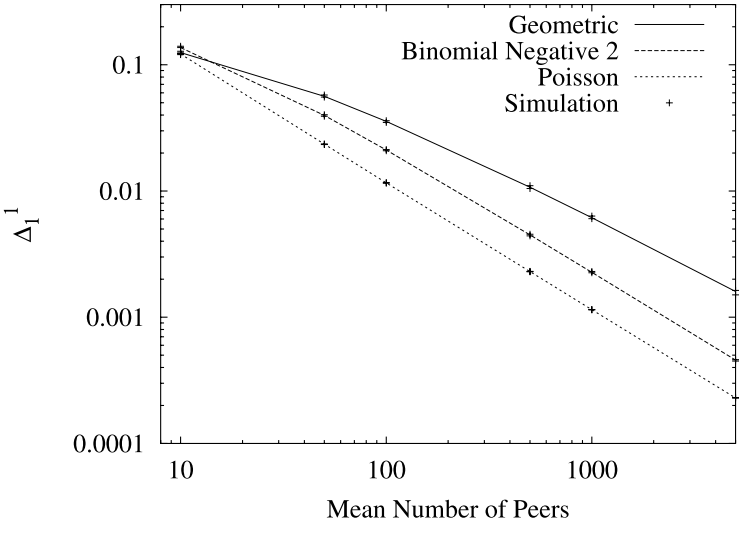


Fig. 1. Comparison of simulation and model results

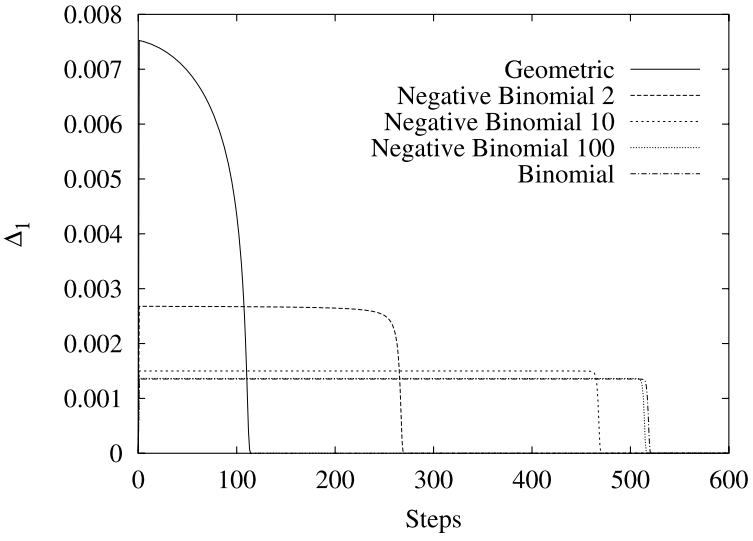


Fig. 2. Resource diffusion as function of the probability distribution of the number of peers (plot of Δ_1)

distribution tends to become deterministic. In particular, with $k = 1, 2, 10, 100$ we have that $var(N) = \frac{1-k\rho}{k\rho^2}$ is equal to $999000, 499000, 99000, 9000$ respectively.

The binomial distribution is important from an applicative point of view. In particular, it can be used to model a system in which there is a large number of peers, where only a portion of them are active. We tried a binomial distributions with parameters $p = 1/1000$, $k = 10^6$, $mean(N) = kp = 1000$, $var(N) = kp(1-p) = 1000$, modelling the case where there are 10^6 peers in the system but only 1000 of them active simultaneously. In all cases the mean value was set to 1000.

The experiments pointed out that output parameters are strongly affected by the coefficient of variation c_v of the distribution. In Fig. 2 we reported the evolution of Δ_1. It can be noted that when the c_v is high, such as in the geometric case (with parameters $\rho = 1/1000$, $mean(N) = 1/\rho = 1000$, $var(N) = \frac{1-\rho}{\rho^2} \simeq 10^6$) the evolution of Δ_1 starts with a higher value and it decreases rapidly. An higher variance implies a larger probability of having a small number of peers in the system. When the number of peers is small, a resource transfer causes a large change in α, β and γ, that is a large Δ. When the c_v decreases, such as in negative binomial distribution with many stages, Δ_1 tends to become constant until all the requests are served, time at which it falls to zero. Furthermore, Δ_1 become smaller (slowing the resource diffusion process) as the variance decreases. The diffusion of the resource results even slower in the binomial case.

4.3 Results

We exploit the proposed model to study the resource diffusion in a non-homogeneous environment. In Experiment 1 we consider a topology with two ASs to analyze the resource diffusion in one AS as function of the resource popularity in the other. We

Table 4. Parameters Experiment 1

AS	s	α^0	β^0	γ^0	ξ
AS_1	0.5	0.2	0.7	0.1	0.0
AS_2	0.5	[0.15..0.75]	[0.75..0.15]	0.1	0.0

suppose that AS_1 has few resources and a high number of requests. The parameters settings are reported in Table 4: we vary the resource availability of AS_2 by keeping constant $\gamma_2 = 0.1$ and tuning α_2 and β_2 (their sum is constant and equal to $\alpha_2 + \beta_2 = 0.9$). The number of peers in the system is geometrically distributed with parameter $\rho = 0.001$, i.e. the mean number of peers is $1/\rho = 1000$. Results reported in Fig. 3 show how the resource diffusion in AS_1 is much faster when the resource popularity in AS_2 is high. Δ_1 represents the rate at which the number of peers requesting the resource decreases at each step, and it can be considered as a measure of the "speed" of the resource diffusion. Δ_1 has a higher initial value and then it rapidly falls to zero when all requests are satisfied.

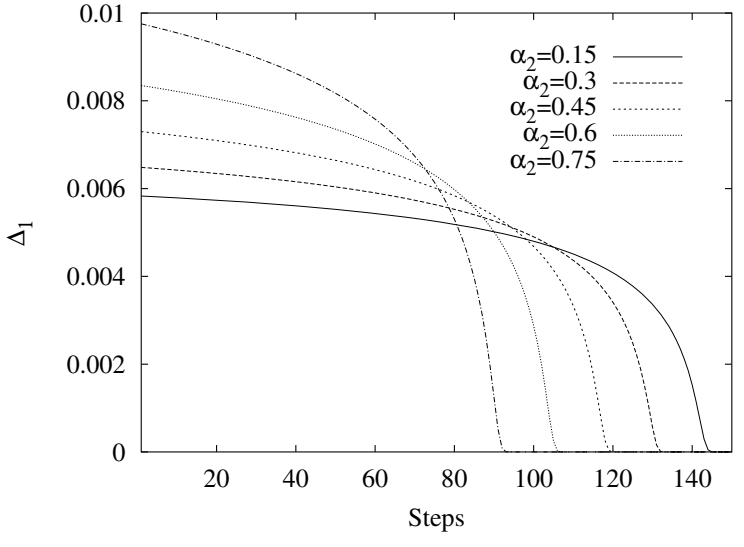


Fig. 3. Resource diffusion in AS_1 as function of the resource popularity in AS_2 (two AS's topology)

In Experiment 2 we investigate the resource diffusion in a three ASs topology. The parameters settings are reported in Table 5 ("Case 1 With freeloaders"). The number of peers in the system is geometrically distributed with parameter $\rho = 0.0001$, i.e. the mean number of peers is $1/\rho = 10000$. The evolution of Δ for each AS is reported in Fig. 4. A rather intuitive result shows that Δ_1 is higher due to both the larger number of peers ($s_1 = 0.5$), and the larger initial number of requests ($\beta_1^0 = 0.6$). It is interesting to note

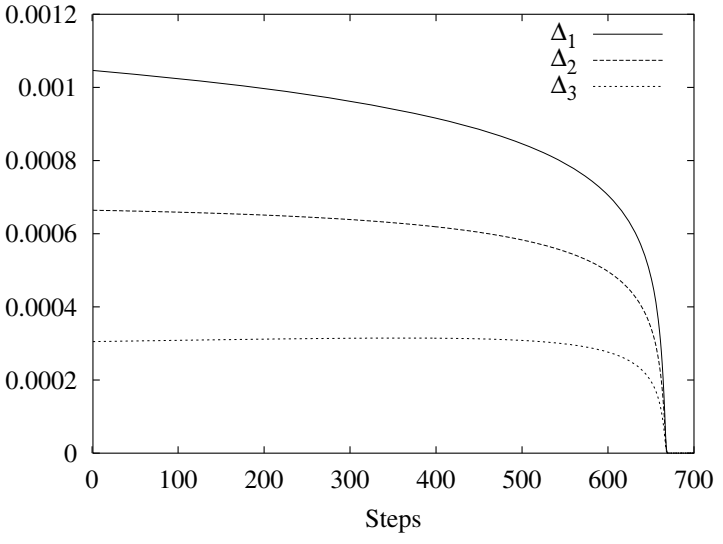


Fig. 4. Resource diffusion in three ASs topology

that the diffusion stops simultaneously in all the three ASs. This is a direct consequence of the random policy used in the selection of the AS that receives the resource.

In Experiment 3 we study the impact of freeloaders on the resource diffusion in a three ASs topology with $1/\rho = 100000$. We consider scenarios with and without freeloaders. Parameters settings are reported in the Table 5 ("Case 1"). In Fig. 5 it is possible to note that in the case with freeloaders the resource diffusion is slower. The impact is emphasized in Fig. 6 where we plot the extra amount of transfers required to get the same number of resources in AS_1 due to freeloaders. This index is computed as $\#Steps_{with}(\alpha_1) - \#Steps_{without}(\alpha_1)$. Fig. 6 illustrates also the results for "Case 2" where we increased the portion of freeloaders by setting $\xi_1 = 0.25$, as reported in Table 5 ("Case 2"). The impact of freeloaders is even stronger and the index increases.

Table 5. Parameters Experiment 2 and 3

Case 1 Without freeloaders	s	α^0	β^0	γ^0	ξ
AS_1	0.5	0.25	0.21	0.24	0.0
AS_2	0.36	0.5	0.38	0.12	0.0
AS_3	0.14	0.7	0.16	0.14	0.0
With freeloaders					
AS_1	0.5	0.25	0.6	0.15	0.15
AS_2	0.36	0.5	0.4	0.1	0.05
AS_3	0.14	0.7	0.2	0.1	0.2
Case 2 Without freeloaders					
AS_1	0.5	0.25	0.45	0.3	0.0
With freeloaders					
AS_1	0.5	0.25	0.6	0.15	0.25

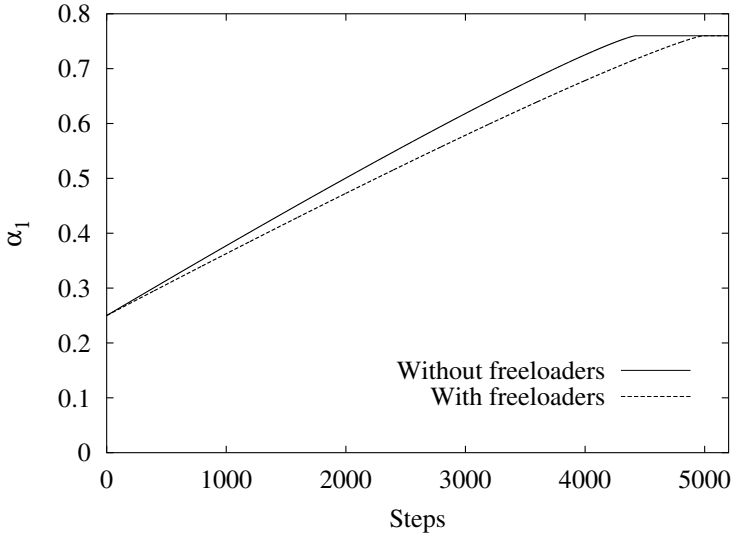


Fig. 5. Impact of freeloaders in AS_1 (three ASs topology)

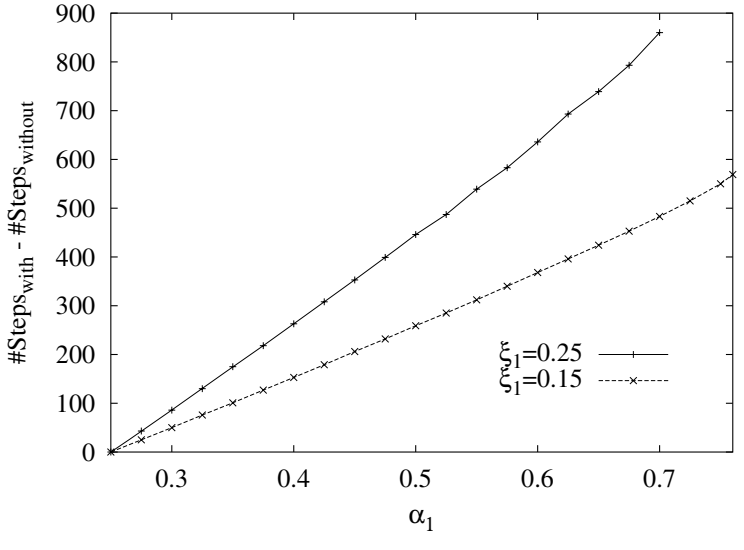


Fig. 6. Impact of freeloaders in AS_1 (three ASs topology): Case 1 lower plot, Case 2 upper plot

5 Conclusions and Future Work

The proposed model allows to analyze the diffusion of a resource in a non-homogeneous environment where peers are grouped into Autonomous Systems. The model gives an

analytical representation of some known important phenomenon like the impact of freeloaders. Furthermore the model takes into account resource popularity and peers behavior in different ASs.

This work is the first step towards a project that aims at studying the impact of P2P file sharing applications on the overall network traffic, by considering the geographic location of peers and the communication costs.

References

1. The true picture of peer-to-peer file sharing. Technical Report, Cachelogic Research (2004)
2. Global Index (GI). Technical Report, http://www.skype.com/skype_p2pexplained.htm
3. Gaeta, R., Gribaudo, M., Manini, D., Sereno, M.: Analysis of Resource Transfers in Peer-to-Peer File Sharing Applications using Fluid Models. Performance Evaluation: An International Journal, Peer-to-Peer Computing Systems 63(3), 147–264 (2006)
4. Manini, D., Gribaudo, M.: Modelling Search, Availability, and Parallel Download in P2P File Sharing Applications with Fluid Model. In: Proceeding of the 14th International Conference on Advanced Computing and Communication ADCOM 2006, Mangalore, India (2006)
5. Li, J., Chou, P.A., Zhang, C.: An efficient Mechanism for Content Distribution in a Peer-to-Peer Network. In: Microsoft Research, MSR-TR-2004-100 (2004)
6. Yang, X., de Veciana, G.: Service Capacity of Peer to Peer Networks. In: Proceedings of INFOCOM 2004, Hong Kong (March 2004)
7. Qiu, D., Srikant, R.: Modeling and Performance Analysis of Bit Torrent-Like Peer-to-Peer Networks. In: Proceedings of ACM SIGCOMM 2004, Portland, USA (2004)
8. Hossfeld, T., Leibnitz, K., Pries, R., Tutschku, K., Tran-Gia, P., Pawlikowski, K.: Information diffusion in eDonkey-like P2P networks. In: Proceedings of Australian Telecommun. Networks and Applications Conference (ATNAC), Bondi Beach, Australia (2004)
9. Leibnitz, K., Hoßfeld, T., Wakamiya, N., Murata, M.: Modeling of Epidemic Diffusion in Peer-to-Peer File-Sharing Networks. In: Proceedings of Proceedings of the Second International Workshop on Biologically Inspired Approaches to Advanced Information Technology Bio-ADIT 2006, Osaka, Japan (2006)
10. Massouli, L., Vojnovic, V.: Coupon Replication Systems. In: Proceedings of the ACM SIGMETRICS 2005, Banff, Alberta, Canada (2005)
11. Stutzbach, D., Zhao, S., Rejaie, R.: Characterizing Files in the Modern Gnutella Network. In: Proceedings of Multimedia Systems Journal (March 2007)
12. Ge, Z., Figueiredo, D.R., Jaiswal, S., Kurose, J., Towsley, D.: Modeling Peer-Peer File Sharing System. In: Proceedings of INFOCOM 2003, San Francisco, USA (April 2003)
13. Carofiglio, G., Gaeta, R., Garetto, M., Giaccone, P., Leonardi, E., Sereno, M.: A Fluid-Diffusive Approach for Modelling P2P Systems. In: Proceedings of the 14-th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems - MASCOTS 2006, Monterey, CA, USA (2006)
14. Shridharbhai Trivedi, K.: Probability and Statistics with Reliability, Queuing and Computer Science Applications. Prentice Hall, Englewood Cliffs (1982)

A Hessenberg Markov Chain for Fast Fibre Delay Line Length Optimization

J. Lambert[1], W. Rogiest[2], B. Van Houdt[1], D. Fiems[2], C. Blondia[1], and H. Bruneel[2]

[1] PATS Research Group, IBBT & University of Antwerp, Middelheimlaan 1, B-2020 Antwerp - Belgium
{joke.lambert,benny.vanhoudt,chris.blondia}@ua.ac.be
[2] SMACS Research Group, TELIN Department, Ghent University, Sint-Pietersnieuwstraat 41, B-9000 Gent - Belgium
{wouter.rogiest,dieter.fiems,herwig.bruneel}@ugent.be

Abstract. In this paper we present an approach to compute the invariant vector of the $N+1$ state Markov chain P presented in (Rogiest *et al.*, Lecture Notes in Computer Science, NET-COOP 2007 Special Issue, pp. 4465:185-194) to determine the loss rate of an FDL buffer consisting of N lines, by solving a related Hessenberg system (i.e., a Markov chain skip-free in one direction). This system is obtained by inserting additional time instants in the sample paths of P and allows us to compute the loss rate for various FDL lengths by solving a single system. This is shown to be especially effective in reducing the computation time of the heuristic LRA algorithm presented in (Lambert *et al.*, Proc. NAEC 2005, pp. 545-555) to optimize the FDL lengths, where improvements of several orders of magnitude can be realized.

1 Introduction

The rapid growth of the Internet traffic has resulted in an increasing demand for bandwidth as well as support for diverse service demands. As the electronic switches and routers are becoming the bottlenecks of the backbone network, these networks must evolve to new architectures based upon all optical switching. Optical burst switching (OBS) has been proposed [11], [13], [14] as an efficient and flexible switching method and it combines the benefits of optical circuit switching and optical packet switching. Although wavelength conversion reduces the need for OBS network buffering [7], contention can still arise when two or more data bursts arrive at the same output wavelength. In optical buffering, fibre delay lines (FDLs) are used to delay a burst for a specified amount of time, which is determined by the length of the delay line. Several architectures that make use of an FDL buffer have already been identified [3], [4], [8]. The main performance measure of an FDL buffer is its loss rate and analytic and simulation based results concerning the loss rates have been reported in [2], [5], [6], [9], [10], [12].

While the typical approach is to build an analytic model based on the evolution of the scheduling horizon, a more effective modeling approach was developed in [1] by focusing on the waiting time only. This model is applicable for Bernoulli

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 101–113, 2008.

arrival processes, general burst size distributions and arbitrary sets of FDLs. As the waiting time of a burst corresponds to the length of one of the FDLs, the computational complexity in [12] depended solely on the number of FDLs N, which is typically small, whereas with the scheduling-horizon-based analysis the complexity was a function of the length of the longest delay line (in slots).

Eventhough N is typically small, the approach taken in [12] has to be reperformed for each set of FDL lengths, which can be very time-consuming in an FDL length optimization procedure. In this paper we develop a Hessenberg reduction of the model presented in [12], which not only allows us to assess the loss rate more quickly for a single set of FDL lengths, but also enables us to compute the loss rate for various FDL length settings at the same time. This significantly reduces the time required to perform an FDL length optimization.

The remainder of this paper is structured as follows. Section 2 introduces the general notion of FDL buffers. We present the traffic model and the corresponding Markov chain in Section 3. From this we derive the transition probabilities and loss probabilities. Section 4 makes use of the memoryless nature of the arrival process to reduce the Markov chain to a Hessenberg, i.e., skip-free in one direction, Markov chain. In Section 5 we present a comparative study between the different solution methods discussed in this paper.

2 Fibre Delay Lines

In this paper, we study a single Wavelength Division Multiplexing (WDM) channel and assume contention for it is resolved by means of a Fibre Delay Line (FDL) buffer, which can delay, if necessary, data packets, called optical bursts (OBs), until the channel becomes available again. Unlike conventional electronic buffers, however, it cannot delay bursts for an arbitrary period of time, but it can only realize a discrete set of N delay values. In [9,2] it is shown that this leads to voids on the outgoing channel. We do not attempt to fill these voids (as this would alter the order of the bursts). We denote the N delay values as $a_1 \leq a_2 \leq \ldots \leq a_N$ and define $a_0 = 0$. Traditionally, there are two possibilities for the delay values, either all fibres have the same length, i.e., $a_k = A$ with $k = 1, 2, \ldots, N$, or the values are equidistant, i.e., $a_k = kD$ with $k = 1, \ldots, N$, where D is termed the buffer granularity. The analysis in this paper, however, is done for arbitrary lengths. We define the set $\mathcal{A}$ as $\mathcal{A} = \{a_0, a_1, \ldots, a_N\}$.

Define the scheduling horizon H at time t as the time difference between t' and t, where t' $(t' \geq t)$ is the earliest time at which all OBs present at time t will have left the system. When the $k-$th burst sees a scheduling horizon $H_k > 0$ upon arrival, with $a_i < H_k \leq a_{i+1}$ for some $0 \leq i \leq N$ (and with $a_{N+1} = \infty$), it will have to be delayed by a_{i+1} time units (if $i < N$, otherwise it is dropped), possibly creating a void on the outgoing channel (which occurs if $i < N$ and $a_i < H_k < a_{i+1}$). Bursts that observe a scheduling horizon $H_k = 0$ do not experience any delay. Since the length of the longest delay line corresponds to the maximum achievable delay a_N, an OB will be dropped (that is, lost) if the burst sees a scheduling horizon larger than a_N upon arrival.

3 Analysis

As the reduction procedure proposed in this paper heavily builds on the approach developed in [12], we start with a (brief) discussion of this approach.

3.1 Traffic Model

Both the model developed in [12] and the reduction model assume Bernoulli arrivals, i.e., a new OB arrival occurs in a slot with probability p independently from slot to slot. In reality, arrivals destined for an output port may not form a Bernoulli process and the losses do not occur uniformly over time, typically the majority of the losses may be expected to occur in bursts during so-called overload periods. Our Bernoulli assumption, with p large such that the load is close to or above one, therefore corresponds to assuming that these overload periods have a substantial duration and the arrivals during such a period may be approximated by a Bernoulli process. When we dimension the FDL buffer in this manner, we therefore aim for the best possible structure to minimize losses during overload periods as these contain most of the losses.

Arriving bursts are either accepted upon arrival or dropped. We number the OBs in the order at which they arrive, but only assign an index to the OBs that are accepted. With each accepted OB k, we associate an inter-arrival time T_k, that captures the time between the kth arrival and the next, being the arrival of (i) burst $k+1$, if this next burst is accepted, or (ii) a burst without number, if this next burst is dropped. For the assumed Bernoulli arrival process, these inter-arrival times are independent and geometrically distributed (with mean $1/p$):

$$P(T_k = n) = t_n = q^{n-1}p \qquad n \geq 1, \tag{1}$$

where we denote $q = 1 - p$. With each accepted burst, we also associate a burst size B_k and we assume that consecutive OB lengths are uncorrelated with a common distribution. Therefore, we can define general probabilities:

$$P(B_k = n) = b_n \qquad n \geq 1, \tag{2}$$

with $0 \leq b_n \leq 1$ and $\sum_n b_n = 1$. To simplify the notations, we will write $\bar{b}_n = \sum_{k=1}^{n} b_k$ and $\bar{t}_n = \sum_{k=1}^{n} t_k$.

3.2 Markov Chain

As mentioned, we will track the system's performance by means of the waiting time of an OB. We associate the waiting time W_k with the kth burst and define it as the time between the acceptance of OB k and the start of its transmission. As described in [12] the next burst is either accepted or dropped resulting in two possible scenarios for the waiting time transitions. These two scenarios are based on the horizon value as seen by the next arriving OB. Let $\bar{H}_k$ be the value of the scheduling horizon as seen by the burst, following burst k. Note that this

is not the value of the horizon as seen by the $k-$th arrival. This horizon value is given by

$$\bar{H}_k = (W_k + B_k - T_k)^+, \tag{3}$$

where $(x)^+$ denotes $\max(x, 0)$.

Scenario 1. We have a lossless transition in case $\bar{H}_k \leq a_N$ with the result that the waiting time of the accepted burst equals

$$W_{k+1} = \lceil W_k + B_k - T_k \rceil_{\mathcal{A}}, \tag{4}$$

where we adopted the relation $\lceil x \rceil_{\mathcal{A}} = \inf\{y \in \mathcal{A} \mid y \geq x\}$, for $x \leq a_N$.

Scenario 2. The burst will be dropped if $\bar{H}_k > a_N$. In this case the burst following burst k is not assigned an index, and possibly, even more bursts are dropped before burst $k+1$ is accepted. As a result of this (and of the memoryless nature of the arrival process), the waiting time of burst $k+1$ no longer relates to W_k and is defined as

$$W_{k+1} = \lceil a_N - \tilde{T}_k \rceil_{\mathcal{A}}, \tag{5}$$

where $\tilde{T}_k$ is the time until the arrival of the next accepted burst, i.e., the reactivation time. In case of geometric inter-arrival times, $\tilde{T}_k$ has a shifted geometric distribution:

$$P(\tilde{T}_k = n) = \tilde{t}_n = q^n p \qquad n \geq 0. \tag{6}$$

We are now able to build up the Markov chain, associated with the evolution of the waiting times. Remark that the waiting time can only take on $N+1$ different values $a_i \in \mathcal{A}$, therefore, the Markov chain consists of $N+1$ states. This chain is characterized by a transition matrix P with probabilities $p_{n,m}$:

$$p_{n,m} = P(W_{k+1} = a_m \mid W_k = a_n) \qquad 0 \leq n, m \leq N. \tag{7}$$

Corresponding to the scenarios discussed above, the probabilities $p_{n,m}$ can be split up into two separate contributions, i.e., the first term corresponds to the event of a lossless transition and the second term corresponds to the event of a transition with loss:

$$\begin{aligned} p_{n,m} &= P(a_n + B_k - T_k \leq a_N, a_m = \lceil a_n + B_k - T_k \rceil_{\mathcal{A}}) \\ &\quad + P(a_n + B_k - T_k > a_N, a_m = \lceil a_N - \tilde{T}_k \rceil_{\mathcal{A}}) \\ &= P(a_{m-1} - a_n < B_k - T_k \leq a_m - a_n) \\ &\quad + P(B_k - T_k > a_N - a_n) P(a_N - a_{m-1} > \tilde{T}_k \geq a_N - a_m), \end{aligned} \tag{8}$$

where $a_{-1} = -\infty$. In order to reduce the computation time it is useful to introduce

$$U_n = P(B_k - T_k \leq n) = \sum_{i=1}^{n} b_i (1 - q^{i-n-1}) + q^{-n-1} B(q), \tag{9}$$

where the sum over n disappears if $n \leq 0$ and $B(z)$ is the probability generating function of B_k evaluated in $z = q$, i.e., $B(z) = E[z^{B_k}] = \sum_{n=1}^{\infty} b_n z^n$. Adopting these notations, (8) can be expressed as

$$p_{n,m} = U_{a_m - a_n} - U_{a_{m-1} - a_n} + (1 - U_{a_N - a_n})(\bar{t}_{a_N - a_{m-1}} - \bar{t}_{a_N - a_m}).$$

The stationary probability vector $\pi = (\pi_0, \ldots, \pi_N)$ satisfies $\pi = \pi P$ and $\pi e = 1$, where e is column vector with all entries equal to one.

3.3 Loss Probability

Up to now, we considered only accepted OBs and even left the dropped bursts unnumbered. As a consequence the loss probability can be computed from the expected number of losses generated after each accepted packet. For geometric inter-arrival times the expected number of such losses is given by

$$E_{\text{loss}} = \sum_{k=1}^{\infty} \sum_{i=0}^{N} \pi_i p b_k (a_i + k - a_N - 1, 0)^+. \tag{10}$$

Namely, define the periods during which the system can accept new burst as available periods, and the periods during which the system drops arriving bursts as unavailable periods. Hence, $(a_i + k - a_N - 1, 0)^+$ in equation (10) denotes the length of the unavailable period, following an accepted burst, which had to wait a_i slots and which had a size equal to k. Finally, the loss probability is determined as follows

$$p_{\text{loss}} = \frac{E_{\text{loss}}}{1 + E_{\text{loss}}}. \tag{11}$$

4 Skip-Free in One Direction Markov Chain

By exploiting the memoryless nature of the arrival process, we will reduce the Markov chain characterized by P, to a Markov chain that is skip-free to the left, i.e., the transition probability matrix $\bar{P}$ of the new chain has the following structure:

$$\bar{P} = \begin{bmatrix} \bar{p}_{0,0} & \bar{p}_{0,1} & \bar{p}_{0,2} & \cdots & \bar{p}_{0,N} \\ \bar{p}_{1,0} & \bar{p}_{1,1} & \bar{p}_{1,2} & \cdots & \bar{p}_{1,N} \\ 0 & \bar{p}_{2,1} & \bar{p}_{2,2} & \cdots & \bar{p}_{2,M} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \ldots & \bar{p}_{N,N-1} & \bar{p}_{N,N} \end{bmatrix}. \tag{12}$$

We shall refer to the transition matrix P, defined in Section 3.2, as the dense matrix and to the transition matrix $\bar{P}$ (equation (12)) as the Hessenberg matrix. Besides the computational advantage of the Hessenberg form, which allows us to compute the invariant vector of $\bar{P}$ in $O(n^2)$ time, an even more important property of this reduction is that a_N, the length of the longest delay line, will only

affect two entries in $\bar{P}$, being $\bar{p}_{N,N-1}$ and $\bar{p}_{N,N}$. This will allow us to express the invariant vectors $\bar{\pi}$ of $\bar{P}$ for different a_N values in terms of the invariant vector of the system with $a_N = a_{N-1} + 1$. This feature will greatly simplify the optimization step of the LRA algorithm (see Section 5.1).

The idea behind the reduction works as follows. Suppose P makes a transition from state n to state m without a loss, where $a_m = \lceil a_n - T_k + B_k \rceil_{\mathcal{A}}$. Then, $\bar{P}$ will first go from state n to state $n-s$, with $a_{n-s} = \lceil a_n - T_k \rceil_{\mathcal{A}}$ in s steps (from state n via $n-1$, $n-2$, etc. to $n-s$). Afterwards, it will jump from state $n-s$ to state m in one step. Thus, we insert s additional time instants for each lossless transition from state n to state m.

The reduction for transitions that do involve losses is somewhat more involving. As with the lossless transitions, the chain characterized by $\bar{P}$ starts by visiting state $n-s_1$ (in s_1 steps) after which it will jump to state N (in one step). From state N, we repeatedly travel down (one step at a time) to some state $n-s_j$ (where $a_{n-s_j} = \lceil a_N - \tilde{T}_{k,j} \rceil_{\mathcal{A}}$, with $\tilde{T}_{k,j}$ having a shifted geometric distribution with parameter p), for $j \geq 2$ and jump back to state N as long as $\tilde{T}_{k,j} < B_{k,j}$, with $B_{k,j}$ distributed as B_k. As soon as $\tilde{T}_{k,j} \geq B_{k,j}$, we go from state $n-s_j$ to state m, with $m = \lceil a_N - \tilde{T}_{k,j} + B_{k,j} \rceil_{\mathcal{A}}$. We can split the original loss transitions in this manner, because conditioned on $\tilde{T}_{k,j} \geq B_{k,j}$, one easily checks that $\tilde{T}_{k,j} - B_{k,j}$ is distributed according to a shifted geometric distribution as required.

For ease of notation we introduce

$$F_n = P(B_k - T_k \leq 0 \ \ \& \ \ T_k \leq n) = \sum_{i=1}^{n} t_i \bar{b}_i. \tag{13}$$

Due to the upper Hessenberg form we have $\bar{p}_{n,m} = 0$ if $m < n-1$. Furthermore, there is a transition from state n to state $n-1$ in case the inter-arrival time is larger than $a_n - a_{n-1}$ (for $n = 1, \ldots, N$), i.e., $\bar{p}_{n,n-1} = 1 - \bar{t}_{a_n - a_{n-1}}$. For $0 < n < N$, we find, analogous to the derivation of the transition probabilities in Section 3.2, $\bar{p}_{n,n} = F_{a_n - a_{n-1}}$, whereas for $0 < n < m < N$

$$\begin{aligned}\bar{p}_{n,m} &= \sum_{i=1}^{a_n - a_{n-1}} t_i (\bar{b}_{i+a_m - a_n} - \bar{b}_{i+a_{m-1} - a_n}) \\ &= q^{a_n - a_m}(F_{a_m - a_{n-1}} - F_{a_m - a_n}) - q^{a_n - a_{m-1}}(F_{a_{m-1} - a_{n-1}} - F_{a_{m-1} - a_n}),\end{aligned}$$

and for $0 \leq m \leq N-1$ we find $\bar{p}_{0,m} = U_{a_m} - U_{a_{m-1}}$.

Only in the last column we need to add a contribution that corresponds to the event of a transition with loss. This leads to the probability $\bar{p}_{0,N} = 1 - U_{a_{N-1}}$, for $0 < n < N$ we get

$$\begin{aligned}\bar{p}_{n,N} &= \sum_{i=1}^{a_n - a_{n-1}} t_i (1 - \bar{b}_{i+a_{N-1} - a_n}) \\ &= \bar{t}_{a_n - a_{n-1}} - q^{a_n - a_{N-1}}(F_{a_{N-1} - a_{n-1}} - F_{a_{N-1} - a_n}),\end{aligned}$$

and finally $\bar{p}_{N,N} = \bar{t}_{a_N - a_{N-1}}$. Remark that the transition probabilities $\bar{p}_{n,m}$ are somewhat easier to compute than the transition probabilities $p_{n,m}$.

The stationary probability vector of $\bar{P}$ will be denoted as $\bar{\pi} = (\bar{\pi}_0, \ldots, \bar{\pi}_N)$. To compute the loss probability we need the stationary probability vector π of P, where we will prove the following relation between π and $\bar{\pi}$:

$$\pi_j = \frac{\sum_{i=0}^{N} \bar{\pi}_i \hat{p}_{i,j}}{\sum_{k=0}^{N} \sum_{i=0}^{N} \bar{\pi}_i \hat{p}_{i,k}}, \tag{14}$$

where $\hat{p}_{i,j}$ is defined as

$$\begin{aligned}
\hat{p}_{0,m} &= \bar{p}_{0,m} = U_{a_m} - U_{a_{m-1}}, && 0 \le m < N \\
\hat{p}_{n,m} &= 0, && m < n \\
\hat{p}_{n,n} &= \bar{p}_{n,n} = F_{a_n - a_{n-1}}, && 0 < n < N \\
\hat{p}_{n,m} &= \bar{p}_{n,m} = q^{a_n - a_m}(F_{a_m - a_{n-1}} - F_{a_m - a_n}) \\
&\quad - q^{a_n - a_{m-1}}(F_{a_{m-1} - a_{n-1}} - F_{a_{m-1} - a_n}), && 0 < n < m < N \\
\hat{p}_{0,N} &= U_{a_N} - U_{a_{N-1}} \\
\hat{p}_{n,N} &= q^{a_n - a_N}(F_{a_N - a_{n-1}} - F_{a_N - a_n}) \\
&\quad - q^{a_n - a_{N-1}}(F_{a_{N-1} - a_{n-1}} - F_{a_{N-1} - a_n}), && 0 < n < N \\
\hat{p}_{N,N} &= F_{a_N - a_{N-1}}.
\end{aligned} \tag{15}$$

Recall, the idea behind the Hessenberg reduction makes that for each sample path of the dense system P (of Section 3.2), there is a corresponding path for $\bar{P}$. The probabilities $\hat{p}_{i,j}$ are exactly those contributions to $\bar{P}$ for which the visit to state j also occurs in the corresponding sample path of P.

Let $\ddot{p}_{i,j}$ be the remaining probability $\bar{p}_{i,j} - \hat{p}_{i,j}$ (hence, $\bar{P} = \hat{P} + \ddot{P}$), then equation (14) can be proven as follows. Due to the sample path correspondence we have $P = (I - \ddot{P})^{-1}\hat{P}$, while the steady state vector of $\bar{P}$ satisfies

$$\begin{aligned}
& \bar{\pi} = \bar{\pi}(\hat{P} + \ddot{P}) \\
\Leftrightarrow\ & \bar{\pi}(I - \ddot{P}) = \bar{\pi}\hat{P} \\
\Leftrightarrow\ & \bar{\pi} = \bar{\pi}\hat{P}(I - \ddot{P})^{-1} \\
\Leftrightarrow\ & \bar{\pi}\hat{P} = \bar{\pi}\hat{P}(I - \ddot{P})^{-1}\hat{P} \\
\Leftrightarrow\ & \bar{\pi}\hat{P} = \bar{\pi}\hat{P}P.
\end{aligned}$$

Therefore we can conclude that the steady state vector π can be computed as described in equation (14).

5 Numerical Results

Since in practical cases the number of delay lines is small (typically 5 to 10), the gain of the reduction to the Hessenberg matrix, as discussed in Section 4, will be limited when considering a single set of FDL lengths. However, the contribution of our new model lies in its applicability in optimization studies, where the gain of the computational complexity will be more explicit. In this section we will make a comparative study for a specific optimization algorithm, namely the Largest Reduction Addition algorithm. This study illustrates that the exploitation of the skip-free to the left structure of the transition matrix places us in a position to deal with larger systems, while drastically reducing the computation times.

5.1 Largest Reduction Addition (LRA) Algorithm

In [10] we studied how the loss rate of the classic equidistant FDL buffer can be improved by considering alternate delay line structures, which means that the delay line lengths a_k are not necessarily a multiple of the constant D. Therefore we developed and compared three different heuristic algorithms. Due to performance and computational reasons, the Largest Reduction Addition (LRA) algorithm was superior to the other two algorithms. When applying the LRA algorithm, we assume that the size of the buffer N and the maximum packet length $B_{\max}$ are known. Besides we define $\mathcal{A}_0 = \{a_0\}$ and the addition of the delay value a_{n+1} to the set $\mathcal{A}_n$ is denoted by $\mathcal{A}_{n+1} = \{\mathcal{A}_n, a_{n+1}\}$. In each step of the LRA algorithm an FDL is added. The length of this FDL is chosen as the one whose addition causes the largest reduction in the loss rate. The interval in which we have to look for the optimal a_{n+1}, given the set $\mathcal{A}_n$, will be given by $I_{n+1} = [a_n + 1, a_n + B_{\max} - T_{\min}]$, where $T_{\min}$ denotes the minimal inter-arrival time. By default the LRA algorithm sets the minimal value of the optimization domain to $a_n + 1$ (various numerical experiments, not reported here, have indicated that the optimum was always found beyond $a_n + 1$). To upper bound the interval one can reason as follows. Consider the FDL set $\mathcal{A}_n$ and assume that the last accepted burst experienced a delay of a_n time units. The highest observable horizon value equals a_n plus $B_{\max}$, the largest possible burst size, minus the minimal inter-arrival time $T_{\min}$. Hence, the next burst will always be accepted if the buffer has an additional delay line a_{n+1} with length $a_{n+1} = a_n + B_{\max} - T_{\min}$. Therefore, selecting a_{n+1} beyond this value will only cause more losses as such an a_{n+1} value would result in a higher horizon value without being able to accept more bursts.

The LRA algorithm consists of the following N steps:

- Step 1: an FDL buffer consisting of one FDL is constructed. Its length is given by $a_1 = \text{argmin}_{\omega \in I_1} \left(p_{\text{loss}}^{\{\mathcal{A}_0, \omega\}} \right)$, where $p_{\text{loss}}^{\mathcal{A}}$ denotes the loss probability for the specific set $\mathcal{A}$ of FDLs.
- Step $n = 2$ to N: The set $\mathcal{A}_{n-1}$ is updated to $\mathcal{A}_n$ by adding one delay value. The length of the delay value is chosen as $a_n = \text{argmin}_{\omega \in I_n} \left(p_{\text{loss}}^{\{\mathcal{A}_{n-1}, \omega\}} \right)$.

In [10] the computation of the loss probabilities in each step and for each of the ω-values was based on the horizon approach. Using the approach taken in [12], one significantly reduces the computation time. However, most of the computations have to be redone when altering the value of ω. In the next section, we will show that for the Hessenberg reduction, we can retrieve the loss probability for all ω values by solving a single Markov chain (in $O(n^2)$ time).

5.2 A Fast LRA Implementation Based on the Hessenberg Matrix

The structure of the Hessenberg matrix provides a fast implementation of the LRA algorithm. We denote the transition matrix that characterizes the Markov chain corresponding with the set $\mathcal{A}_n$ as $\bar{\boldsymbol{P}}^{(n)}$ and the corresponding probabilities

$\bar{p}_{i,j}$, respectively $\hat{p}_{i,j}$, as $\bar{\boldsymbol{p}}_{i,j}^{(n)}$, respectively as $\hat{\boldsymbol{p}}_{i,j}^{(n)}$. Assume the set $\mathcal{A}_{n-1}$ is known and we have to determine the optimal length for a_n which has to be searched in the interval I_n. We first construct the transition matrix $\bar{P}^{(n,1)}$ corresponding with the delay line length $a_n = a_{n-1} + 1$. It is not hard to verify that

$$\bar{P}^{(n,1)} = \left[\begin{array}{ccc|c|c} & & & \hat{\boldsymbol{p}}_{0,n-1}^{(n-1)} & \bar{\boldsymbol{p}}_{0,n}^{(n)} \\ & \bar{\boldsymbol{P}}_l^{(n-1)} & & \hat{\boldsymbol{p}}_{1,n-1}^{(n-1)} & \bar{\boldsymbol{p}}_{1,n}^{(n)} \\ & & & \vdots & \vdots \\ & & & \hat{\boldsymbol{p}}_{n-1,n-1}^{(n-1)} & \bar{\boldsymbol{p}}_{n-1,n}^{(n)} \\ \hline 0 & \dots & 0 & \bar{\boldsymbol{p}}_{n,n-1}^{(n)} & \bar{\boldsymbol{p}}_{n,n}^{(n)} \end{array}\right],$$

where $\bar{\boldsymbol{P}}_l^{(n-1)}$ represents the $n \times (n-1)$ matrix found in the upper left corner of $\bar{\boldsymbol{P}}^{(n-1)}$. So we can construct the transition matrix $\bar{P}^{(n,1)}$ by recomputing at most $n+2$ elements. Furthermore, the probabilities $\hat{p}_{i,j}^{(n,1)}$ can be extracted immediately from the transition matrix $\bar{P}^{(n,1)}$ except for $0 \leq i \leq n = j$. Thus, recomputing $2(n+1)+1$ elements suffices to determine $\pi^{(n,1)}$ and its corresponding loss probability $p_{\text{loss}}^{(n,1)} = p_{\text{loss}}^{\{\mathcal{A}_{n-1}, a_{n-1}+1\}}$. For the other possible delay line lengths within the interval I_n, i.e., $a_n = a_{n-1} + x$, with $x > 1$, we can easily determine the steady state vector $\pi^{(n,x)}$ without the need to compute $\bar{P}^{(n,x)}$ or $\bar{\pi}^{(n,x)}$. This can be explained as follows. Since the transition matrix $\bar{P}^{(n,x)}$ differs only in two entries from $\bar{P}^{(n,1)}$, namely in $\bar{p}_{n,n-1}^{(n,x)}$ and $\bar{p}_{n,n}^{(n,x)}$, the steady state vector $\bar{\pi}^{(n,x)}$ can be expressed in terms of $\bar{\pi}^{(n,1)}$:

$$\bar{\pi}_i^{(n,x)} = \frac{\bar{\pi}_i^{(n,1)}}{\sum_{j=0}^{n} \bar{\pi}_j^{(n,x)}} \qquad i = 0, \dots, n-1 \tag{16a}$$

$$\bar{\pi}_n^{(n,x)} = \frac{\theta^{(n,x)}}{\sum_{j=0}^{n} \bar{\pi}_j^{(n,x)}}, \tag{16b}$$

where

$$\theta^{(n,x)} = \frac{\sum_{k=0}^{n-1} \bar{\pi}_k^{(n,1)} \bar{p}_{k,n}^{(n,1)}}{(1 - \bar{t}_x)}. \tag{17}$$

This can be used to determine the steady state vector $\pi^{(n,x)}$ based on equation (14):

$$\pi_i^{(n,x)} = \frac{\sum_{k=0}^{i} \bar{\pi}_k^{(n,1)} \bar{p}_{k,i}^{(n,1)}}{\sum_{j=0}^{n} \pi_j^{(n,x)}} \qquad i = 0, \dots, n-1 \tag{18a}$$

$$\pi_n^{(n,x)} = \frac{\sum_{k=0}^{n-1} \bar{\pi}_k^{(n,1)} \hat{p}_{k,n}^{(n,x)} + \theta^{(n,x)} \hat{p}_{n,n}^{(n,x)}}{\sum_{j=0}^{n} \pi_j^{(n,x)}}. \tag{18b}$$

Remark that we do not have to compute $\bar{\pi}^{(n,x)}$ to get $\pi^{(n,x)}$, but we can compute $\pi^{(n,x)}$ immediately from $\bar{\pi}^{(n,1)}$ and the probabilities $\hat{p}_{i,n}^{(n,x)}$ with $0 \leq i \leq n$. Therefore we have to recompute only $n+1$ elements for each value of x.

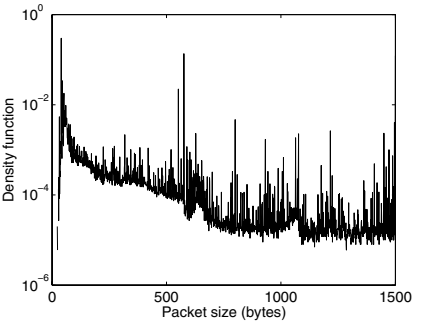


Fig. 1. Trace-based IP packet length distribution as captured on the AIX link

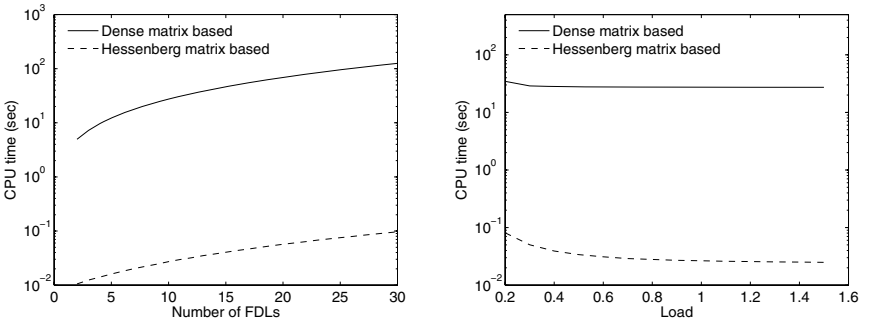
At the end of each step of the LRA algorithm, we determine x_{opt} such that $p_{\text{loss}}^{(n,x_{\text{opt}})} = \min_{x \in I_n} \left(p_{\text{loss}}^{(n,x)} \right)$ and we denote the corresponding transition matrix $\bar{P}^{(n,x_{\text{opt}})}$ as $\bar{\boldsymbol{P}}^{(n)}$ and the corresponding probabilities $\bar{p}_{i,j}^{(n,x_{\text{opt}})}$, respectively $\hat{p}_{i,j}^{(n,x_{\text{opt}})}$, as $\bar{\boldsymbol{p}}_{i,j}^{(n)}$, respectively $\hat{\boldsymbol{p}}_{i,j}^{(n)}$.

Notice, for the dense matrix approach of [12], we can only reuse some of the intermediary results of the calculation of $\boldsymbol{P}^{(n-1)}$ when computing the transition matrices $P^{(n,x)}$.

5.3 Execution Times of the Fast LRA Algorithm

As explained in Section 3.1, we focus on overload periods during which we assume Bernoulli arrivals with p large such that the load is close to or above one. Therefore, when regarding the loss probabilities computed in this section, we should keep in mind that these are the loss rates during the overload periods and not the long-run loss probabilities, which will be several orders of magnitude smaller. Clearly, the results and conclusions drawn in this section remain significative when the overload loss rates are scaled down to the long-run situation.

The study in [10] made use of packet traces collected by the NLANR (National Laboratory for Applied Network Research). More specifically, in this paper we will use an IP packet trace coming from the following link: AIX (a measurement point that sits at the interconnection point of NASA Ames and the MAE-West interconnection of Metropolitan Fiber Systems). The distribution of the packet sizes of the considered trace is depicted in Figure 1. Furthermore we use an FDL buffer with $N = 10$ FDLs. In order to keep the computation time in [10] limited, we had to define the length of a single slot equal to a number of bytes (more specifically, 50 bytes), as a result the packet sizes distribution had to be clustered (such that all packets had a multiple of 50 bytes as their packet length). With the approach taken in [12], clustering is no longer required as the packet length distribution has no impact on the size of the state space of the Markov chain. In other words, we can select the slot length equal to one byte. The main objective of this section exists in demonstrating the substantial computational gain that



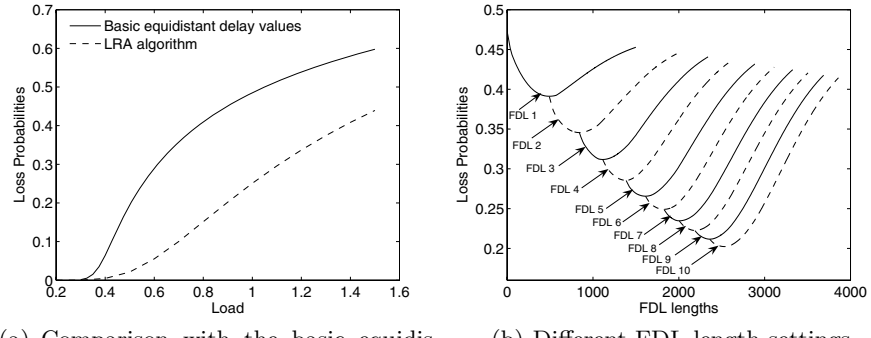
(a) Function of the number of FDLs (b) Function of the load

Fig. 2. Execution times for the different strategies that can be used to solve the LRA algorithm

is realized by the Hessenberg reduction over the dense matrix solution of [12] when running the LRA algorithm.

Figure 2(a) shows the execution times as a function of the number of FDLs, with a load $\rho = 0.9$. From this figure we can conclude that the Hessenberg solution offers a gain of several orders of magnitude over the dense matrix approach. Similar results are observed in Figure 2(b) that shows the execution times as a function of the load in case there are 10 FDLs. In order to quantify the loss reduction typically realized by the LRA algorithm, Figure 3(a) shows the loss probability as a function of the load for both equidistant delay values, with $D = B_{\max} - 1$, and for the combination of delay values found by the LRA algorithm.

Figure 3(b) shows the loss probabilities for different FDL length settings as they are calculated during the N steps of the LRA algorithm with load $\rho = 0.9$. The curve of FDL 1 shows the loss rate as influenced by the length of the first line. A



(a) Comparison with the basic equidistant situation (b) Different FDL length settings

Fig. 3. Loss probabilities determined with the LRA algorithm

Table 1. FDL lengths a_k and differences d_k between subsequent FDL lengths determined with the LRA algorithm

	$\rho = 0.2$		$\rho = 0.9$	
FDL k	a_k	d_k	a_k	d_k
1	1499	1499	491	491
2	2998	1499	841	350
3	4497	1499	1114	273
4	5996	1499	1386	272
5	7495	1499	1612	226
6	8994	1499	1827	215
7	10493	1499	2005	178
8	11992	1499	2187	182
9	13491	1499	2360	173
10	14990	1499	2531	171

minimum is realized at $a_1 = 491$ bytes and serves as a starting point for the second step of the LRA algorithm. During the second step, the optimal length for the second FDL is found in $a_2 = 841$ (FDL 2 in Figure 3(b)). This procedure is repeatedly executed until N FDL lengths are found. Note, the LRA algorithm is a heuristic algorithm to find useful non-equidistant FDL lengths, but it does not necessarily realize a global minimum. The resulting FDL sets and differences between subsequent FDL lengths (i.e., $d_k = a_k - a_{k-1}$) are displayed in Table 1. The results for the scenario with a low load of $\rho = 0.2$ are also included in this table.

The experimental results of this section indicate that for low loads the performance of the LRA algorithm coincides with the equidistant FDL buffer, whereas for higher loads we can realize a substantial reduction in loss when using non-equidistant delay values. This observation was already made in [10] based on the clustered trace, confirming our intuition that generally a clustered trace should suffice to get a good understanding of the system behavior.

6 Conclusion

This paper presented a novel approach to compute the invariant vector of the $N + 1$ state Markov chain presented in [12], based on a Hessenberg reduction, and resulting in a fine-tuned fibre delay line length optimization scheme. While previous work already allowed to compute the loss rate of a general FDL buffer [12], it was not adapted to the iterative scheme typically associated with optimization and as such inefficient to determine the optimal FDL length. Applying the Hessenberg reduction to the Largest Reduction Addition algorithm [10], we showed how the exploitation of the specific structure of the transition matrix (skip-free to the left) allows to compute the loss rate for an entire range of FDL lengths at the same time. This places us in a position to deal with larger systems, while drastically reducing the computation times. Finally, a comparative study showed how non-equidistant FDL sets are especially useful during overload

periods, however, the results and conclusions remain of course significative when the overload loss rates are scaled down to the long-run situation.

References

1. Almeida, R.C., Pelegrini, J.U., Waldman, H.: A generic-traffic optical buffer modeling for asynchronous optical switching networks. Communications Letters, IEEE 9(2), 175–177 (2005)
2. Callegati, F.: Optical buffers for variable length packet switching. IEEE Communications Letters 4, 292–294 (2000)
3. Chlamtac, et al.: CORD: Contention resolution by delay lines. IEEE Journal on Selected Areas in Communications 14, 1014–1029 (1996)
4. Haas, Z.: The staggering switch: An electronically controlled optical packet switch. Journal on Lightwave Technology 11, 925–936 (1993)
5. Hong, D., Poppe, F., Reynier, J., Bacelli, F., Petit, G.: The impact of burstification on TCP throughput in optical burst switching networks. In: Proc. of the 18th International Teletraffic Congress (ITC), Berlin, Germany (September 2003)
6. Van Houdt, B., Laevens, K., Lambert, J., Blondia, C., Bruneel, H.: Channel utilization and loss rate in a single-wavelength fibre delay line (FDL) buffer. In: Proceedings of IEEE Globecom 2004, paper OC05-07, Dallas, USA (November 2004)
7. Hunter, D., Andonovic, I.: Approaches to optical Internet packet switching. IEEE Communications Magazine 38(9), 116–120 (2000)
8. Hunter, D., Cornwell, W.D., Gilfedder, T.H., Franzen, A., Andonovic, I.: SLOB: A switch with large optical buffers for packet switching. IEEE/OSA Journal on Lightwave Technology 16, 1725–1736 (1998)
9. Laevens, K., Bruneel, H.: Analysis of a single wavelength optical buffer. In: Proceedings of Infocom, San Francisco (April 2003)
10. Lambert, J., Van Houdt, B., Blondia, C.: Single-wavelength optical buffers: non-equidistant structures and preventive drop mechanisms. In: Proceedings of NAEC 2005, Riva del Garda, pp. 545–555 (2005)
11. Qiao, J., Yoo, M.: Optical burst switching: A new paradigm for an optical Internet. Journal on High-Speed Networks 8, 69–84 (1999)
12. Rogiest, W., Fiems, D., Laevens, K., Bruneel, H.: Tracing an optical buffer's performance: An effective approach. In: Chahed, T., Tuffin, B. (eds.) NET-COOP 2007. LNCS, vol. 4465, pp. 185–194. Springer, Heidelberg (2007)
13. Turner, J.: Terabit burst switching. Journal on High-Speed Networks 8, 3–16 (1999)
14. Xiong, Y., Vandenhoute, M., Chankaya, H.: Control architecture in optical burst-switched WDM-networks. IEEE Journal on Selected Areas in Communications 18, 1838–1851 (2000)

Stochastic Fatigue Models for Efficient Planning Inspections in Service of Aircraft Structures

Nicholas Nechval[1], Konstantin Nechval[2], Gundars Berzinsh[1], Maris Purgailis[1], and Uldis Rozevskis[1]

[1] University of Latvia, Mathematical Statistics Department, Raina Blvd 19, LV-1050 Riga, Latvia
nechval@junik.lv
[2] Transport and Telecommunication Institute, Applied Mathematics Department, Lomonosov Street 1, LV-1019 Riga, Latvia
konstan@tsi.lv

Abstract. For important fatigue-sensitive structures of aircraft whose breakdowns cause serious accidents, it is required to keep their reliability extremely high. In this paper, we discuss inspection strategies for such important structures against fatigue failure. The focus is on the case when there are fatigue-cracks unexpectedly detected in a fleet of aircraft within a warranty period (prior to the first inspection). The paper examines this case and proposes stochastic models for prediction of fatigue-crack growth to determine appropriate inspections intervals. We also do not assume known parameters of the underlying distributions, and the estimation of that is incorporated into the analysis and decision-making. Numerical example is provided to illustrate the procedure.

Keywords: Aircraft, fatigue crack, inspection interval.

1 Introduction

Fatigue is one of the most important problems of aircraft arising from their nature as multiple-component structures, subjected to random dynamic loads. The analysis of fatigue crack growth is one of the most important tasks in the design and life prediction of aircraft fatigue-sensitive structures (for instance, wing, fuselage) and their components (for instance, aileron or balancing flap as part of the wing panel, stringer, etc.). An example of in-service cracking from B727 aircraft [1] (year of manufacture 1981; flight hours not available; flight cycles 39,523) is given on Fig. 1.

Several probabilistic or stochastic models have been employed to fit the data from various fatigue crack growth experiments. Among them, the Markov chain model [2], the second-order approximation model [3], and the modified second-order polynomial model [4]. Each of the models may be the most appropriate one to depict a particular set of fatigue growth data but not necessarily the others. All models can be improved to depict very accurately the growth data but, of course, it has to be at the cost of increasing computational complexity. Yang's model [3] and the polynomial model [4] are considered more appropriate than the Markov chain model [2] by some researchers

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 114–127, 2008.

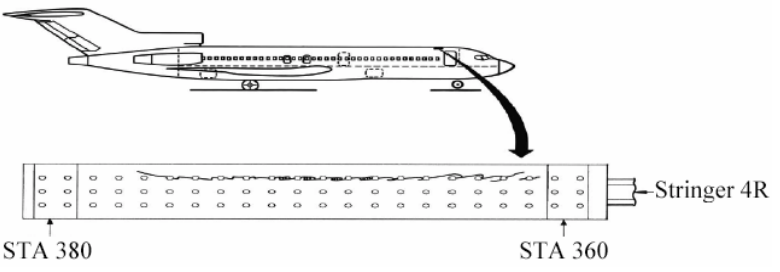


Fig. 1. Example of in-service cracking from B727 aircraft

through the introduction of a differential equation which indicates that fatigue crack growth rate is a function of crack size and other parameters. The parameters, however, can only be determined through the observation and measurement of many crack growth samples. If fatigue crack growth samples are observed and measured, descriptive statistics can then be applied directly to the data to find the distributions of the desired random quantities. Thus, these models still lack prediction algorithms. Moreover, they are mathematically too complicated for fatigue researchers as well as design engineers.

A large gap still needs to be bridged between the fatigue experimentalists and researchers who use probabilistic methods to study the fatigue crack growth problems.

2 Problem Description

Let us assume that a fatigue-sensitive component has been found cracked on *n* aircrafts within a warranty period. The cracking had not yet caused an accident, but the safety experts have told the manager that had this item failed, an accident was possible. It is clear that the part would have to be redesigned and replaced. The manager's dilemma is that redesigning the part, manufacturing the new design, and installing it in the fleet would take, say, at least two years. The manager must decide how to manage risk for the next two years. The alternatives include doing nothing and accepting the risk of continued cracking and the possibility of an accident. An inspection program is usually instigated, which should reduce the risk of failure, but due to uncertainties in aircraft loading histories, provides no direct measurement of the criticality of the detected cracks. Generally, such a program would lead to some aircraft being grounded, eliminating risk for those aircraft and reducing overall risk, but reducing operational capability. This would leave precious few aircraft to spare before the service's ability to accomplish its mission became impaired. In such a scenario, the decision process involves a complex probability problem concerning the likelihood of additional failures and acceptable risk. To compound the difficulty little guidance is provided in aircraft design specifications for this situation. The situation presented is not uncommon.

The purpose of this paper is to present a more accurate stochastic crack growth analysis method, while maintaining the simplicity of the proposed stochastic fatigue models, for the above problem. We discuss the optimal relationship between the

inspection time and the prespecified minimum level of reliability. To illustrate the proposed technique, a numerical example is given.

3 Paris-Erdogan Law

The basis of most of the fatigue models is the Paris-Erdogan law [5] relating the rate of growth of crack size a to N cycles:

$$\frac{\mathrm{d}a(N)}{\mathrm{d}N} = q[a(N)]^b \tag{1}$$

in which q and b are parameters depending on loading spectra, structural/material properties, etc. We fit $\mathrm{d}a/\mathrm{d}N$ vs $a(N)$ with a function that we can integrate between limits (initial crack size, a_0, and any given crack size, a) to get a life prediction.

In the linear region (see Fig. 2) we use the Paris-Erdogan Equation (1) as follows.

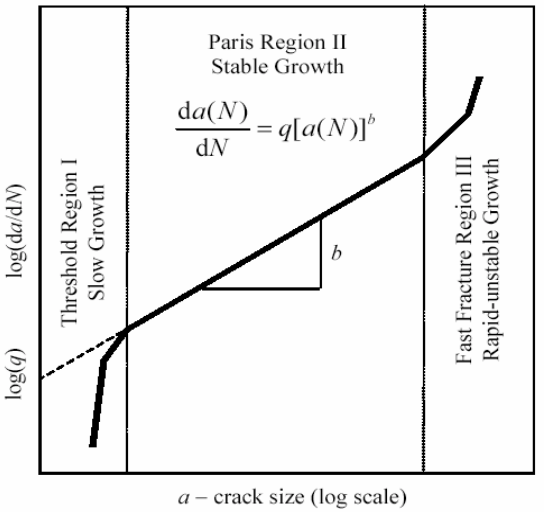


Fig. 2. Crack growth rate versus crack size curve (I = near-threshold region; II = linear region; III = instability region)

Integrating

$$\int_{N_0}^{N} \mathrm{d}N = \int_{a_0}^{a(N)} \frac{\mathrm{d}a}{qa^b}, \tag{2}$$

we have

$$N - N_0 = \frac{1}{q(-b+1)}\left[a(N)^{-b+1} - a_0^{-b+1}\right]. \tag{3}$$

Thus, the crack growth equation representing the solution of the differential equation for the Paris-Erdogan law is given by

$$N - N_0 = \frac{1}{q(b-1)}\left(\frac{1}{a_0^{b-1}} - \frac{1}{a(N)^{b-1}}\right). \tag{4}$$

3.1 Sensitivity Analysis

Consider the solution of the differential equation for the Paris-Erdogan law written in the form of (4) as:

$$N(a) = \frac{1}{q(b-1)}\left(\frac{1}{a_0^{b-1}} - \frac{1}{a^{b-1}}\right), \tag{5}$$

where a_0 is the initial crack size at N_0=0. The derivatives of the number of load cycles with respect to the parameters q and b read:

$$\frac{\mathrm{d}N(a)}{\mathrm{d}q} = -\frac{N(a)}{q} \tag{6}$$

and

$$\frac{\mathrm{d}N(a)}{\mathrm{d}b} = \frac{1}{b-1}\left[\frac{1}{q}\left(\frac{\ln a}{a^{b-1}} - \frac{\ln a_0}{a_0^{b-1}}\right) - N(a)\right]. \tag{7}$$

From this one can see that the number of cycles to reach a certain crack size is very sensitive to changes of the parameter q.

From an engineering standpoint the fatigue life of a component or structure consists of two periods (this concept is shown schematically in Fig. 3).

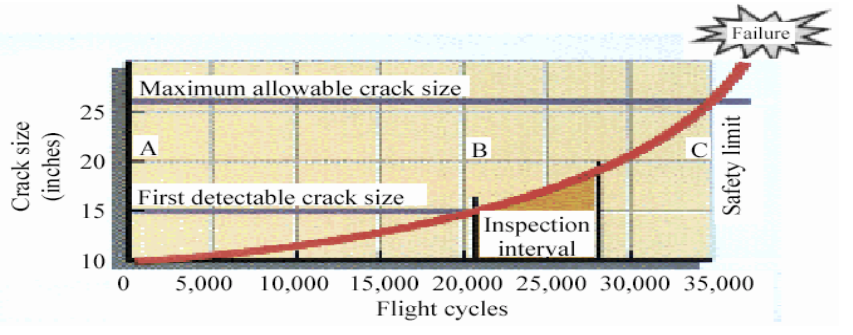


Fig. 3. Schematic fatigue crack growth curve (Crack initiation period (A-B); Crack propagation period (B-C))

Periodic inspections of aircraft are common practice in order to maintain their reliability above a desired minimum level. The appropriate inspection intervals are

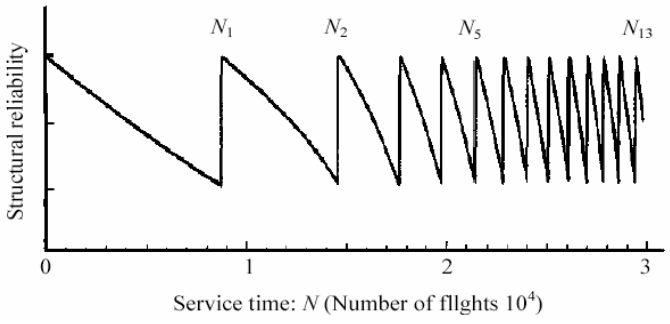


Fig. 4. Inspection schedule and structural reliability

determined so that the fatigue reliability of the entire aircraft structure remains above the minimum reliability level throughout its service life. Fig. 4 illustrates this case.

4 Statistical Variability of Fatigue-Crack Growth

The traditional analytical method of engineering fracture mechanics (EFM) usually assumes that crack size, stress level, material property and crack growth rate, etc. are all deterministic values which will lead to conservative or very conservative outcomes. However, according to many experimental results and field data, even in well-controlled laboratory conditions, crack growth results usually show a considerable statistical variability (as shown in Fig. 5).

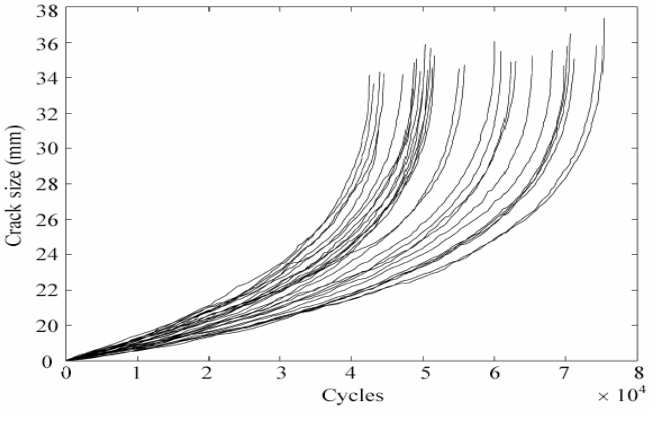


Fig. 5. Constant amplitude loading fatigue test data curves

Yet more considerable statistical variability is the case under variable amplitude loading (as shown in Fig. 6).

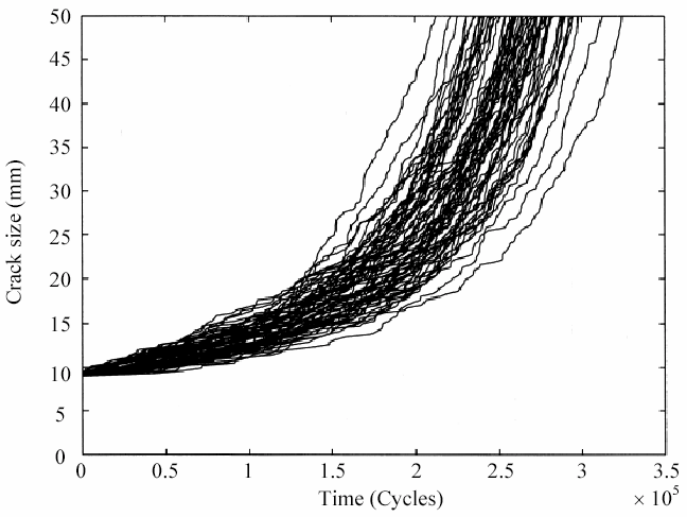


Fig. 6. Variable amplitude loading fatigue test data curves

The basis of most data analyses seems to be to take logarithms in (1) and estimate b and q by least squares in the equation

$$\ln(\mathrm{d}a(N)/\mathrm{d}N) = \ln q + b \ln a(N). \tag{8}$$

Unfortunately to use this equation estimates of d$a(N)$/dN are required. Estimates of derivatives are notoriously unreliable. If several repetitions of an experiment under the same conditions are made it is not always clear how to combine the results. Moreover, as a regression model the properties of the estimates of the coefficients in (8) are not the same as those of estimates of the coefficients in (4). Thus it is sensible to ask why the estimation does not proceed directly from the data on crack size and cycles through equation (4).

It is interesting to note that if b were known q could be estimated from a straight line

$$\frac{1}{a_0^{b-1}} - \frac{1}{a^{b-1}} = (b-1)q(N - N_0) \tag{9}$$

and indeed such a plot for a few values of q is indicative of the nature of the Paris-Erdogan equation in a particular case.

During the service of the components being assessed, there may be uncertainties in the applied loading conditions, extrapolation of the material data to service conditions, component dimensions, and nature, size and location of detected (postulated) defects, etc. These uncertainties/variations are critical inputs to the crack growth assessment and can be taken into account using probabilistic methodologies.

The purpose of this paper is to present a more useful stochastic fatigue crack growth models by using the solution of the Paris-Erdogan law equation, which result

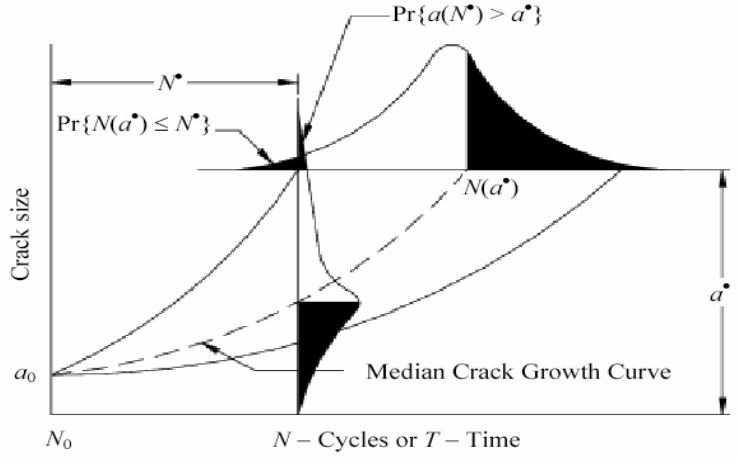


Fig. 7. Schematic diagram of crack size distribution and random time distribution

in a simple analytical solution for either the distribution of the service time to reach any given crack size or the distribution of the crack size at any service time.

The probability that crack size $a(N)$ will exceed any given crack size $a^{\bullet}$ in the service interval $(N_0, N^{\bullet})$, $\Pr\{a(N^{\bullet})>a^{\bullet}\}$ (see Fig. 7), is frequently referred to as crack exceedance probability and can be found based on the stochastic fatigue crack growth model. In addition to this probability distribution of crack size, the probability distribution of cycles (or time) for a crack to grow from size a_0 to $a^{\bullet}$, $\Pr\{N(a^{\bullet})\leq N^{\bullet}\}$, can also be found based on the above model. In fact, the probability that service time $N(a^{\bullet})$ will be within the interval $(N_0, N^{\bullet})$ for crack size to reach $a^{\bullet}$ is identical to $\Pr\{a(N^{\bullet}) >a^{\bullet}\}$. That is $\Pr\{N(a^{\bullet})\leq N^{\bullet}\} = \Pr\{a(N^{\bullet}) > a^{\bullet}\}$.

5 Stochastic Fatigue-Crack Growth Parameter Variability Models

These models allow one to describe the uncertainties in one or two parameters of the solution (4) of the differential equation (1) for the Paris-Erdogan law via parameters modelled as random variables in order to characterize the random properties, which seem to vary from specimen to specimen (see Fig. 5). In other words, the stochastic fatigue-crack growth parameter variability models (with respect to the parameters b and q modelled as random variables) are given by

$$N - N_0 = \frac{1}{Q(B-1)}\left(\frac{1}{a_0^{B-1}} - \frac{1}{a^{B-1}}\right), \tag{10}$$

where $(N - N_0)$ is a joint random variable of B and Q. In fact, these models are suited to account for this type of variability. The ones however cannot explain the variability of the crack growth rate during the crack growth process. In particular, crack growth

data (crack size versus service time and vice versa) may been analyzed using Eq. (10) by considering, for instance, two different approaches:

(i) B is identical for each specimen and Q varies from specimen to specimen, referred to as Case 1;
(ii) both B and Q vary from specimen to specimen, referred as Case 2.

For Case 1, with B=1, the crack growth data for each specimen are best fitted by equation

$$N - N_0 = \frac{\ln\left[\dfrac{a(N)}{a(N_0)}\right]}{Q} \tag{11}$$

to obtain a sample value of Q, where $a(N) \equiv a$, $a(N_0) \equiv a_0$. For Case 2 Equation (10) is used to best fit the crack growth data for each specimen to obtain a set of sample values of B and Q. From the statistical standpoint, B is considered to be a deterministic value and Q to be a statistical (random) variable in Case 1, while both B and Q are considered to be statistical variables in Case 2. It is found that the lognormal or Weibull distribution provides a reasonable fit for B and Q in both cases.

5.1 Weibull Crack Growth Parameter Variability Model

Consider Case 1. The Weibull probability distribution function, $F(q;\sigma,\delta)$, *of* Q is expressed as

$$F(q;\sigma,\delta) = \begin{cases} 1-\exp[-(q/\sigma)^{\delta}], & q \geq 0, \\ 0, & \text{otherwise}, \end{cases} \tag{12}$$

in which $F(q;\sigma,\delta)$ is the probability that Q is smaller than or equal to an arbitrary value q; σ and δ are distribution parameters representing the scale parameter and the shape parameter, respectively.

5.2 Crack Exceedance Probability

For B=1, the probability that crack size $a(N)$ will exceed any given (say, maximum allowable) crack size $a^{\bullet}$ can be derived and expressed as

$$\Pr\{a(N) > a^{\bullet}\} = \Pr\left\{Q > \frac{\ln[a^{\bullet}/a(N_0)]}{N - N_0}\right\} = \exp\left[-\left(\frac{\ln[a^{\bullet}/a(N_0)]}{\sigma(N - N_0)}\right)^{\delta}\right]. \tag{13}$$

For B=b≠1 the maximum allowable crack size exceedance probability for a single crack is given by

$$\Pr\{a(N) > a^{\bullet}\} = \Pr\left\{Q > \frac{[a(N_0)]^{-(b-1)} - [a^{\bullet}]^{-(b-1)}}{(b-1)(N - N_0)}\right\}$$

$$= \exp\left[-\left(\frac{[a(N_0)]^{-(b-1)} - [a^\bullet]^{-(b-1)}}{\sigma(b-1)(N-N_0)} \right)^{\delta} \right]. \tag{14}$$

It will be noted that the crack exceedance probability can be used for assigning sequential in-service inspections [6].

6 Stochastic Fatigue-Crack Growth Lifetime Variability Models

These models allow one to characterize the random properties, which seem to vary during crack growth (see Fig. 6), via crack growth equation with a stochastic noise V dependent, in general, on the crack size a:

$$\frac{1}{(b-1)q}\left(\frac{1}{a_0^{b-1}} - \frac{1}{a^{b-1}} \right) = N - N_0 + V \tag{15}$$

$$\ln\left(\frac{1}{(b-1)q}\left(\frac{1}{a_0^{b-1}} - \frac{1}{a^{b-1}} \right) \right) = \ln(N-N_0) + V \tag{16}$$

$$\frac{1}{a_0^{b-1}} - \frac{1}{a^{b-1}} = (b-1)q(N-N_0) + V, \tag{17}$$

$$\ln\left(\frac{1}{a_0^{b-1}} - \frac{1}{a^{b-1}} \right) = \ln[(b-1)q(N-N_0)] + V, \tag{18}$$

and so on, where $V \sim \mathrm{N}(0, \sigma^2(b,q,N))$, $a_0 \equiv a(N_0)$, $a \equiv a(N)$. They are suited to account for this type of variability. The ones however cannot explain the variability of the crack growth rate from specimen to specimen.

6.1 Crack Exceedance Probability

If $V \sim \mathrm{N}(0, \sigma^2)$ in (15), then the probability that crack size $a(N)$ will exceed any given (say, maximum allowable) crack size $a^\bullet$ can be derived and expressed as

$$\Pr\{a(N) > a^\bullet\} = \Phi\left(\left[N - N_0 - \frac{[a_0]^{-(b-1)} - [a^\bullet]^{-(b-1)}}{(b-1)q} \right] \Big/ \sigma \right), \tag{19}$$

where $\Phi(.)$ is the standard normal distribution function,

$$\Phi(\eta) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\eta} \exp(-x^2/2)dx. \tag{20}$$

If $V \sim N(0,[(b-1)\sigma(N-N_0)^{1/2}]^2)$ in (17), then the probability that crack size $a(N)$ will exceed any given (say, maximum allowable) crack size $a^{\bullet}$ can be derived and expressed as

$$\Pr\{a(N) > a^{\bullet}\} = \Phi\left(\left[\frac{(b-1)q(N-N_0) - ([a_0]^{-(b-1)} - [a^{\bullet}]^{-(b-1)})}{(b-1)\sigma(N-N_0)^{1/2}}\right]\right). \tag{21}$$

In this case, the conditional probability density function of a is given by

$$f(a;a_0,N_0,N) = \frac{a^{-b}}{\sigma[2\pi(N-N_0)]^{1/2}}$$

$$\times \exp\left(-\frac{1}{2}\left[\frac{(a_0^{-(b-1)} - a^{-(b-1)}) - (b-1)q(N-N_0)}{(b-1)\sigma(N-N_0)^{1/2}}\right]^2\right). \tag{22}$$

6.2 Data Analysis for a Single Crack

Consider the regression model corresponding to (17). Because the variance is non-constant (17) is a non-standard model; however, on dividing by $(N-N_0)^{1/2}$ the model becomes

$$\frac{a_0^{1-b} - a^{1-b}}{(N-N_0)^{1/2}} = (b-1)q(N-N_0)^{1/2} + W, \tag{23}$$

where W is normally distributed with mean zero and standard deviation $(b-1)\sigma$. Thus if b is known the estimator of $(b-1)q$ is just the least-squares estimator of the coefficient in Equation (23) and the estimate of $(b-1)\sigma$ is just the estimate of the variance of the regression. It remains to determine what to do about b.

Given the data describing a single crack, say a sequence $\{(a_i, N_i)\}_{i=1}^{n}$, it is easy to construct a log-likelihood using the density given by (22) and estimate the parameters b, q and σ by maximum likelihood. The log-likelihood is

$$L(b,q,\sigma;\{(a_i,N_i)\}) = -b\sum_{i=1}^{n}\ln a_i - n\ln\sigma - \frac{1}{2}\sum_{i=1}^{n}\left(\frac{a_0^{1-b} - a_i^{1-b} - (b-1)q(N_i-N_0)}{(b-1)\sigma(N_i-N_0)^{1/2}}\right)^2. \tag{24}$$

Inspection shows that this differs from the standard least-squares equation only in the term $-b\sum \ln a$, where the subscript i has been dropped. The likelihood estimators are obtained by solving the equations

$$dL/db = 0; \quad dL/dq = 0; \quad dL/d\sigma = 0. \tag{25}$$

In this case the equations have no closed solution. However, it is easy to see that the estimators for q and σ given b are the usual least-squares estimators for the coefficients in (23) conditioned on b,

$$\hat{q}(b) = \frac{1}{b-1}\left(n a_0^{1-b} - \sum_{i=1}^{n} a_i^{1-b} \right) \left(\sum_{i=1}^{n} (N_i - N_0) \right)^{-1}, \tag{26}$$

$$\hat{\sigma}^2(b) = \frac{1}{n(b-1)^2} \sum_{i=1}^{n} \frac{[a_0^{1-b} - a_i^{1-b} - \hat{q}(b)(b-1)(N_i - N_0)]^2}{N_i - N_0}, \tag{27}$$

and on substituting these back in the log-likelihood gives a function of b alone,

$$L(b) = -b \sum_{i=1}^{n} \ln a_i - n \ln[\hat{\sigma}(b)] - n/2. \tag{28}$$

Thus the technique is to search for the value of b that maximizes $L(b)$ by estimating q and σ as functions of b and substituting in $L(b)$. In this study a simple golden-section search worked very effectively.

6.3 Pooling Data

When several experiments have been performed it is possible to combine the log-likelihoods from each experiment to give estimators of the parameters of interest. Suppose that several experiments have been performed. Each experiment is labelled with j, j runs from 1 to m, and yields n_j observations. The data are then a set of sequences $\{(a_{jk},N_{jk})\}$, with j=1, …, m, k=1, …, n_j. The log-likelihood for the whole set of experiments is simply the sum of the log-likelihoods for the individual cracks; writing $L_j(b_j,q_j,\sigma_j)$ for the log-likelihood for the jth crack gives

$$L_j(b_j, q_j, \sigma_j; \{(a_{jk}, N_{jk})\}) = -b_j \sum_{k=1}^{n_j} \ln a_{jk} - n_j \ln \sigma_j$$

$$- \frac{1}{2} \sum_{k=1}^{n_j} \left(\frac{a_0^{1-b_j} - a_k^{1-b_j} - (b_j - 1) q_j (N_k - N_0)}{(b_j - 1)\sigma_j (N - N_0)^{1/2}} \right)^2 \tag{29}$$

and

$$L = \sum_{j=1}^{m} L_j(b_j, q_j, \sigma_j; \{(a_{jk}, N_{jk})\}). \tag{30}$$

The global log-likelihood can be used to investigate explicit parametric models for the parameters, or simply as a way to pool data. Estimation by maximum likelihood proceeds exactly as above; the q_j and σ_j are obtained as ordinary least-squares estimators from equations like (26) and (27), one for each crack, and substituted back into the log-likelihood to yield

$$L(b_1, b_2, ..., b_m) = -\sum_{j=1}^{m} b_j \sum_{k=1}^{n_j} \ln a_{jk} - \sum_{j=1}^{m} n_j \ln[\hat{\sigma}_j(b_j)] - \frac{1}{2} \sum_{j=1}^{m} n_j. \tag{31}$$

When the cracks are all assumed to be independent with distinct parameters the estimators from the joint log-likelihood are precisely those obtained by estimating from each separately as outlined above.

If a common value of b is used and the q_j and σ_j are assumed to absorb most of the experimental variability, the joint log-likelihood reduces to

$$L(b) = -b\sum_{j=1}^{m}\sum_{k=1}^{n_j}\ln a_{jk} - \sum_{j=1}^{m} n_j \ln[\hat{\sigma}_j(b)] - \frac{1}{2}\sum_{j=1}^{m} n_j. \tag{32}$$

7 Stochastic Fatigue-Crack Growth Parameter and Lifetime Variability Models

These models allow one to describe the uncertainties in the fatigue-crack growth of the Paris-Erdogan law via crack growth equation with a stochastic noise dependent, in general, on the crack size, and parameters modelled as random variables in order to characterize the random properties, which seem to vary both from specimen to specimen and during crack growth (see Fig. 6). In other words, the stochastic fatigue-crack growth parameter and propagation lifetime variability model (with respect to the parameters B and Q, modelled as random variables, and the stochastic noise V dependent, in general, on the crack size a) may be given, for example, as

$$N - N_0 = \frac{1}{(B-1)Q}\left(\frac{1}{a_0^{B-1}} - \frac{1}{a^{B-1}}\right) + V. \tag{33}$$

7.1 Crack Exceedance Probability

In this case, the probability that crack size $a(N)$ will exceed any given (say, maximum allowable) crack size $a^{\bullet}$ can be derived and expressed as

$$\Pr\{a(N) > a^{\bullet}\} = \mathrm{E}\left\{\exp\left[-\left(\frac{[a(N_0)]^{-(b-1)} - [a^{\bullet}]^{-(b-1)}}{\sigma(b-1)(N - N_0 + V)}\right)^{\delta}\right]\right\}. \tag{34}$$

8 Example

Let us assume that a fatigue-sensitive component (outboard longeron) has been found cracked on n=10 aircraft within a warranty period. Here a fleet of ten aircraft have all been inspected (see Table 1).

It is assumed that cracks start growing from the time the aircraft entered service. For typical aircraft metallic materials, an initial discontinuity size (a_0) found through quantitative fractography is approximately between 0.02 and 0.05 mm. Choosing a typical value for initial discontinuity state (e.g., 0.02 mm) is more conservative than choosing an extreme value (e.g., 0.05 mm). This implies that if the lead cracks can be attributed to unusually large initiating discontinuities then the available life increases.

Table 1. Inspections results

Aircraft (i)	Flight hours (N_i)	Crack size (mm) (a_i)
1	3000	1
2	2300	0.5
3	2200	1
4	2000	2
5	1500	0.8
6	1500	1.5
7	1300	1
8	1100	1
9	1000	1
10	800	1

We test a goodness of fit of the data of Table 1 with the Weibull fatigue-crack growth parameter variability model (see Case 1), where

$$\hat{Q}_i(b) = \frac{a_0^{1-b} - a_i^{1-b}}{(b-1)(N_i - N_0)}, \quad i = 1(1)n, \tag{35}$$

with a common value of b, a_0=0.02, and N_0=0.

8.1 Goodness-of-Fit Testing

We assess the statistical significance of departures from the Weibull model by performing empirical distribution function goodness-of-fit test.

We use the S statistic [7]. For complete datasets, the S statistic is given by

$$S(b) = \frac{\sum_{i=[n/2]+1}^{n-1} \left(\frac{\ln(\hat{Q}_{i+1}(b)/\hat{Q}_i(b))}{M_i} \right)}{\sum_{i=1}^{n-1} \left(\frac{\ln(\hat{Q}_{i+1}(b)/\hat{Q}_i(b))}{M_i} \right)} = 0.43, \tag{36}$$

where $[n/2]$ is a largest integer $\le n/2$, $\hat{Q}_i$ is the ith order statistic, the values of M_i are given in Table 13 [7]. The rejection region for the α level of significance is $\{S > S_{n;1-\alpha}\}$.

The percentage points for $S_{n;1-\alpha}$ were given in [7]. The value of b is one that minimizes $S(b)$. For this example, b = 0.87 and

$$S = 0.43 < S_{n=10;\,1-\alpha=0.95} = 0.69. \tag{37}$$

Thus, there is not evidence to rule out the Weibull model. Using the relationship (4), the inspection results can be extrapolated from the expected initial crack size, a_0, to the time of the next inspection when the maximum allowable crack size is equal to $a^{\bullet}$=10 mm as presented in Table 2.

Table 2. Predicted next inspection results

Aircraft	Maximum allowable crack size $a^{\bullet}$ (mm)	Next inspection time (flight hours)
1	10	5626
2	10	5503
3	10	4126
4	10	3033
5	10	3030
6	10	2477
7	10	2438
8	10	2063
9	10	1875
10	10	1500

9 Conclusion

In this paper, to capture the scatter of the fatigue crack growth data, the stochastic models that adopted the solution of the crack growth equation, proposed by Paris and Erdogan, and randomized one by including random factors into it are suggested. These stochastic models allow us to obtain the crack exceedance probability as well as the probability of random time to reach a specified crack size. Once the appropriate stochastic model is established, it can be used for the fatigue reliability prediction of structures made of the tested material. As such the models presented here provides a fast and computationally efficient way to predict the fatigue lives of realistic structures.

Acknowledgments. This research was supported in part by Grant No. 06.1936 and Grant No. 07.2036 from the Latvian Council of Science and the National Institute of Mathematics and Informatics of Latvia.

References

1. Jones, R., Molent, L., Pitt, S.: Studies in Multi-Site Damage of Fuselage Lap Joints. J. Theor. Appl. Fract. Mech. 32, 18–100 (1999)
2. Bogdanoff, J.L., Kozin, F.: Probabilistic Models of Cumulative Damage. Wiley, New York (1985)
3. Yang, J.N., Manning, S.D.: A Simple Second Order Approximation for Stochastic Crack Growth Analysis. Engng. Fract. Mech. 53, 677–686 (1996)
4. Wu, W.F., Ni, C.C., Liou, H.Y.: Random Outcome and Stochastic Analysis of Some Fatigue Crack Growth Data. Chin. J. Mech. 17, 61–68 (2001)
5. Paris, R., Erdogan, F.: A Critical Analysis of Crack Propagation Laws. Journal of Basic Engineering 85, 528–534 (1963)
6. Nechval, N.A., Nechval, K.N., Vasermanis, E.K.: Statistical Models for Prediction of the Fatigue Crack Growth in Aircraft Service. In: Varvani-Farahani, A., Brebbia, C.A. (eds.) Fatigue Damage of Materials 2003, pp. 435–445. WIT Press, Southampton, Boston (2003)
7. Kapur, K.C., Lamberson, L.R.: Reliability in Engineering Design. Wiley, New York (1977)

Effective Minimization of Acyclic Phase-Type Representations*

Reza Pulungan** and Holger Hermanns

Department of Computer Science, Saarland University
D-66123, Saarbrücken, Germany
Tel.: +49(681)3025478; Fax: +49(681)3025636
{pulungan,hermanns}@cs.uni-sb.de

Abstract. Acyclic phase-type distributions are phase-type distributions with triangular matrix representations. They constitute a versatile modelling tool, since they (1) can serve as approximations to any continuous distributions, and (2) exhibit special properties and characteristics, which usually result in some ease in analysis. The size of the matrix representation has a strong influence on computational efforts needed when analyzing these distributions. This representation, however, is not unique, and two representations of the same distribution can differ drastically in size. This paper proposes an effective procedure to aggregate the size of the matrix representation without altering the distribution.

Keywords: Stochastic Models; Markov Models.

1 Introduction

Phase-type (PH) distributions are enjoying increasing attention in various fields of computer and engineering sciences. They are frequently used as stochastic modelling aids in areas such as queueing theory [23,2], computer network design [9,21], and reliability analysis [8], for instance in the analysis of dynamic fault trees [22,5].

In this paper, we deal with continuous-time PH distributions. They are a versatile and tractable class of probability distributions, retaining the principal analytical tractability of exponential distributions in a more general setting: the class of PH distributions is topologically dense [20] on the support set $[0, \infty)$. Therefore they can be used to approximate other probability distributions or the trace of empirical distributions obtained from experimental observations.

Any PH distribution agrees with the distribution of the absorption time in some Markov chain with an absorbing state [23]. Such a Markov chain is usually

* This work is supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS). See www.avacs.org for more information.

** Corresponding author.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 128–143, 2008.

the basis of numerical or analytical analysis for models involving that PH distribution, and it is therefore called the *representation* of that distribution. These representations are not unique: distinct absorbing Markov chains may represent the same distribution, and any PH distribution is represented by infinitely many distinct absorbing Markov chains. The representations differ in particular with respect to their size, i.e. the number of states and transitions. Thus, for a given PH distribution, an obvious question to pose is what the minimal-size representation of that distribution may be—and how to construct it.

The nowadays standard approach for aggregating a Markov chain without altering its stochastic properties is based on *lumpability*, in various flavors [7,19,12,4]. This aggregation is equally applicable to absorbing Markov chains, but the computed fix-point is not guaranteed to be the minimal representation in the above sense.

Therefore, the problem of identifying and constructing smaller-sized representations remains one of the most interesting theoretical research questions in the field of PH distributions. In this paper, we focus on the class of acyclic PH (APH) distributions, namely distributions with upper triangular matrix representations. Like PH distributions, also APH distributions are topologically dense on the support set $[0, \infty)$ [20].

Practically, identifying smaller representations of APH distributions can have compelling computational impact. This is especially apparent in modelling formalisms that support compositionality, like PEPA [19], IMC [18], or dynamic fault trees [5]. In such formalisms, complex models are built from smaller and simpler components. The size of some complex model is roughly the product of the individual components' sizes. In this setting, using smaller representations for the components can significantly alleviate the blow-up of the state space during composition. In some cases, this may turn an otherwise intractable model into one with tractable model size.

The systematic study of APH representations was pioneered by Cumani [11], and later by O'Cinneide [26,27], who identified minimality conditions, but without algorithmic considerations. In fact, the quest for an algorithm to construct the minimal representation of any APH distribution has recently seen considerable advances: He and Zhang provided an algorithm for computing minimal representations of APH distributions [16]. This algorithm involves converting the given APH distribution to a representation that only contains the poles of the distribution. This representation is not necessarily an APH distribution, but a matrix-exponential distribution [14]. If this is the case, another state and its total outgoing rate are determined and added to the representation. This is performed one by one until an APH representation is obtained. This results in a representation of provably minimal size. This algorithm involves solving a system of nonlinear equations for each additional state. Since nonlinear programming is difficult, the practicality of this method for large models is not obvious, and has not been investigated so far.

The algorithm developed in this paper addresses the very same problem, but in the opposite way. Instead of adding states to a representation until it

becomes an APH representation, we eliminate states from a representation as we proceed, until no further elimination is possible. Each elimination of a state involves solving a system of linear equations. The resulting algorithm is of cubic complexity in the size of the original state space, and does only involve standard numerical steps. The algorithm is guaranteed to return a smaller representation. However, unlike the algorithm of He and Zhang, minimality of the final result is not guaranteed for arbitrary APH distributions.

In summary, this paper contributes an efficient algorithm to minimize APH distributions. The algorithm is easy to implement and straightforward to parallelize. It only consists of vector-matrix multiplications and the solution of well-conditioned systems of linear equations. To illustrate the effectiveness of our approach, we study a dynamic fault tree scenario with a prototype implementation of the algorithm. As we will discuss, the results are promising.

A full version of this paper including detailed proofs is available in [28].

2 Phase-Type Distributions

This section provides the preliminaries of PH distributions. This discussion is focused on APH distributions, their canonical forms and the transformation from arbitrary APH distribution to the canonical forms.

2.1 Parameterization Definition

Let $\{X(t) \in \mathcal{I} | t \in \mathbb{R}^+\}$ be a homogeneous Markov process defined on a discrete and finite state space $\mathcal{I} = \{s_1, s_2, \cdots, s_n, s_{n+1}\}$ and with time parameter $t \in \mathbb{R}^+ := [0, \infty)$. The Markov process is a finite continuous-time Markov chain (CTMC). We view the structure of such a CTMC as a tuple $\mathcal{M} = (S, \mathbf{R})$ where S is a finite set of states, and $\mathbf{R}$ a rate matrix $\mathbf{R} : S \times S \to \mathbb{R}^+$. The rate matrix $\mathbf{R}$ is related to the corresponding infinitesimal generator matrix by: $\mathbf{Q}(s, s') = \mathbf{R}(s, s')$ if $s \neq s'$ else $\mathbf{Q}(s, s) = -\sum_{s' \neq s} \mathbf{R}(s, s')$ for all $s, s' \in S$. If state s_{n+1} is absorbing (i.e., $\mathbf{Q}(s_{n+1}, s_{n+1}) = 0$) and all other states s_i are transient (i.e., $\mathbf{Q}(s_i, s_i) < 0$), the infinitesimal generator matrix of the Markov chain can be written as

$$\mathbf{Q} = \begin{bmatrix} \mathbf{A} & \underline{A} \\ \mathbf{0} & 0 \end{bmatrix}.$$

Matrix $\mathbf{A}$ is called a *PH-generator* and it is non-singular because the first n states in the Markov chain are transient. Vector $\underline{A}$ is a column vector where its component $\underline{A}_i$, for $1 \leq i \leq n$, represents the transition rate from state s_i to the absorbing state. Since $\mathbf{Q}$ is an infinitesimal generator matrix, $\underline{A} = -\mathbf{A}\underline{e}$ where $\underline{e}$ is an n-dimensional column vector whose components are all equal to 1. Let $\underline{\pi} = (\underline{\alpha}, \alpha_{n+1})$ be the initial probability distribution of the CTMC. Then the probability distribution of the time to absorption in the CTMC is called a *phase-type* distribution. The pair $(\underline{\alpha}, \mathbf{A})$ is called the representation of the PH distribution and $PH(\underline{\alpha}, \mathbf{A})$ is used to denote the PH distribution that has representation $(\underline{\alpha}, \mathbf{A})$. The dimension of PH-generator $\mathbf{A}$ is called the *order of*

the representation. All PH representations we are dealing with are assumed to be irreducible. A representation is *irreducible* if for the specified initial distribution all transient states are visited with non-zero probability.

The PH distribution is completely characterized by its (cumulative) distribution function and by its Laplace-Stieltjes transform (LST). The LST is a rational function. When expressed in irreducible ratio, the denominator of the LST has degree no more than n. This degree of the denominator is called the *degree of the distribution.* The zeros of the denominator polynomial are called the *poles* of the distribution. It is known [23,25] that a given PH distribution has more than one irreducible representation. The order of a *minimal irreducible representation*, namely a representation with the least possible number of states, is referred to as the *order of the PH distribution.* O'Cinneide in [25] showed that the order of a PH distribution may be different from but at least as great as its degree.

2.2 Acyclic Phase-Type Distributions

An interesting subset of the family of PH distributions is the family of acyclic PH distributions. The family can be identified by their triangular representations. A *triangular representation* is a representation whose PH-generator, under some permutation of its components, is an upper triangular matrix.

In [26], O'Cinneide proved a theorem which characterizes APH distributions in terms of the properties of their density functions and LSTs. In particular he showed that an APH distribution has only real poles. Thus, any general PH representation—possibly containing cycles—represents an APH distribution (and hence has an acyclic representation) whenever all of the poles of its LST are real numbers.

Example 1 (Fault Tolerant System). Fault trees [17] are used to model fault-tolerant systems and to analyze their reliability. The fault tree in Fig. 1–(left) models a fault-tolerant system, which consists of two identical components. The redundancy of the components is introduced to increase the system's reliability. The system fails if both components fail. Further, each component is comprised

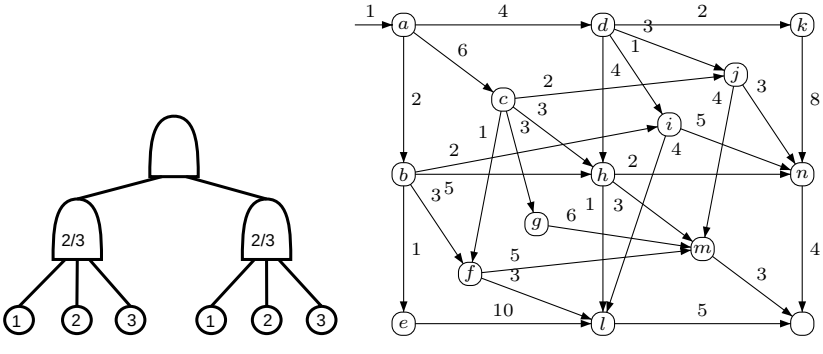


Fig. 1. A Fault Tree (left) and its APH Distribution (right)

of three subcomponents, whose failure times are governed by exponential distributions with rate 1, 2 and 3, respectively. A component is operational if it has at least two functional subcomponents.

The fault tree gives rise to an APH distribution whose representation is depicted in Fig. 1–(right). The representation is obtained by applying a formalization of the dynamic fault tree semantics [22,5]. Note that this representation is the result of a previous aggregation by weak bisimulation algorithm. This representation is not minimal. In the subsequent sections, we will provide the minimal representation.

2.3 Acyclic Canonical Forms

Let an infinitesimal generator matrix of the form

$$\begin{bmatrix} -\lambda_1 & \lambda_1 & 0 & \cdots & 0 \\ 0 & -\lambda_2 & \lambda_2 & \cdots & 0 \\ 0 & 0 & -\lambda_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -\lambda_n \end{bmatrix},$$

be denoted by $\mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n)$. A representation $(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n))$ is referred to as a *bidiagonal representation*.

Cumani in [11], presented canonical forms of APH representations. In particular, he proved that every APH representation has a bidiagonal representation of the same or less order. Aside from the bidiagonal representation, he also provided two other canonical forms and straightforward procedures to transform one to others.

A similar theorem was proved by O'Cinneide in [26]. Using the invariant polytopes method [25], he provided the characterization of APH distributions and proceeded to prove that such distributions are PH distributions with ordered bidiagonal representations. The following theorem provides the first of Cumani's canonical forms (called the series canonical form).

Theorem 1 ([11]). *Let $PH(\underline{\alpha}, \mathbf{A})$ be an APH representation and let the diagonal components of $-\mathbf{A}$ be $\lambda_n \geq \lambda_{n-1} \geq \cdots \geq \lambda_1$, so ordered. Then there is an ordered bidiagonal representation such that $PH(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n)) = PH(\underline{\alpha}, \mathbf{A})$.*

Since all of the diagonal components of matrix $-\mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ are of a particular order (namely ascending), the first canonical form is also called the *ordered bidiagonal representation*.

Example 2 (Ordered Bidiagonal Representation). Fig. 2 depicts the ordered bidiagonal representation (first canonical form) of the triangular representation shown in Fig. 1–(right).

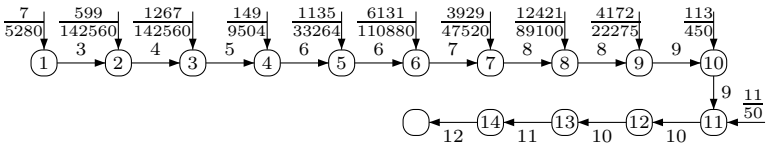


Fig. 2. An Ordered Bidiagonal Representation

2.4 Transformation Algorithms

The proof of Theorem 1, provided by Cumani in [11], is constructive and can be turned into an algorithm to transform any acyclic representation to its ordered bidiagonal representation. However, the transformation algorithm is not polynomial in time, for it examines individual paths of the original acyclic representations.

In [24], O'Cinneide presented another similar algorithm which is based on the concept of PH-simplicity and PH-majoration. We will not treat the concepts in detail in this paper. He and Zhang in [15] provided a better performing algorithm, which they called the *spectral polynomial algorithm*, to obtain the bidiagonal representation of a given APH distribution. The algorithm does not make any assumptions on the PH-simplicity of the given generator matrix $\mathbf{A}$ and may result in a bidiagonal representation of smaller order.

Let $PH(\underline{\alpha}, \mathbf{A})$ be an APH distribution and let $\{-\lambda_1, -\lambda_2, \cdots, -\lambda_n\}$ be the eigenvalues of $\mathbf{A}$. Denote the i-th column of matrix $\mathbf{P}$ by $\mathbf{P}_{*,i}$, then $\mathbf{AP} = \mathbf{PBi}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ can be expressed by

$$\mathbf{AP}_{*,i} = -\lambda_i \mathbf{P}_{*,i} + \lambda_{i-1}\mathbf{P}_{*,i-1},\ i = 1, \cdots, n,$$

with $\mathbf{P}_{*,0} = \underline{0}$. The system of equations can be rewritten as

$$\mathbf{P}_{*,i} = \frac{1}{\lambda_i}(\mathbf{A} + \lambda_{i+1}\mathbf{I})\mathbf{P}_{*,i+1},\ i = 1, \cdots, n-1.$$

For the matrix $\mathbf{P}$ to be of unit row-sums, we have to set $\mathbf{P}_{*,n} = -\mathbf{A}\underline{e}/\lambda_n$ (consult [15] for more details). If there exists a sub-stochastic vector such that $\underline{\beta} = \underline{\alpha}\mathbf{P}$, then $PH(\underline{\alpha}, \mathbf{A}) = PH(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n))$. When the bidiagonal generator matrix is in canonical form, namely if $\lambda_n \geq \lambda_{n-1} \geq \cdots \geq \lambda_1$, then such sub-stochastic vector always exists because $\mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ PH-majorizes $\mathbf{A}$.

The spectral polynomial algorithm has complexity $\mathcal{O}(n^3)$ where n is the order of the given acyclic representation.

3 Reducing the Order of Representations

In this section, we propose a procedure to reduce the order of acyclic representations. Roughly the procedure is as follows: (1) Given an APH distribution with representation $(\underline{\alpha}, \mathbf{A})$, we transform the representation into an ordered bidiagonal representation $(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n))$ by using spectral polynomial algorithm

of He and Zhang. The order of the new representation is at most the same as the order of the original one. (2) Once an ordered bidiagonal representation is obtained, some of its states can be removed without affecting its distribution function. In the rest of the section, we will show that a smaller representation can be obtained by removing some unnecessary states from the ordered bidiagonal representation. A method for identifying and removing those unnecessary states will be provided. The resulting representation is also an ordered bidiagonal representation of fewer number of states.

3.1 The *L*-Terms

Let $L(\lambda) = \frac{s+\lambda}{\lambda}$ be the reciprocal of the LST of an exponential distribution with rate $-\lambda$. We refer to a single expression of $L(\cdot)$ as an *L-term*. The LST of an ordered bidiagonal representation $(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n))$ can be written as

$$\begin{aligned}\tilde{f}(s) &= \frac{\underline{\beta}_1}{L(\lambda_1)\cdots L(\lambda_n)} + \frac{\underline{\beta}_2}{L(\lambda_2)\cdots L(\lambda_n)} + \cdots + \frac{\underline{\beta}_n}{L(\lambda_n)}, \\ &= \frac{\underline{\beta}_1 + \underline{\beta}_2 L(\lambda_1) + \cdots + \underline{\beta}_n L(\lambda_1)L(\lambda_2)\cdots L(\lambda_{n-1})}{L(\lambda_1)L(\lambda_2)\cdots L(\lambda_n)}. \end{aligned} \tag{1}$$

Note that the LST expression in Equation (1) may not be in irreducible ratio form. The LST is produced in such a way that the denominator polynomial corresponds exactly to the sequence of the transition rates of the ordered bidiagonal representation. Hence, the degree of the denominator polynomial is the same as the order of the ordered bidiagonal representation.

3.2 Reducing the Order of Bidiagonal Representations

Observing Equation (1), if we are to remove a state from the ordered bidiagonal representation, we will have to find a common L-term in both of the numerator and denominator polynomials of Equation (1). Removing such a common L-term from the numerator and denominator polynomials means removing the corresponding state from the representation.

However, such removal of a state can only be carried out if after the removal, the initial probability distribution $\underline{\beta}$ is redistributed in a correct manner, namely the new initial probability distribution, say $\underline{\gamma}$, is a sub-stochastic vector. This procedure of identifying and properly removing a state from an ordered bidiagonal representation is described formally in the following lemma.

Lemma 1. *Let $(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n))$ be an ordered bidiagonal representation of order n. If for some $1 \leq i \leq n$, $\underline{\beta}_1 + \underline{\beta}_2 L(\lambda_1) + \cdots + \underline{\beta}_i L(\lambda_1)L(\lambda_2)\cdots L(\lambda_{i-1})$ is divisible by $L(\lambda_i)$ then*

$$PH(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n)) = PH(\underline{\gamma}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_{i-1}, \lambda_{i+1}, \cdots, \lambda_n))$$

if $\underline{\gamma}$ is a sub-stochastic vector. $(\underline{\gamma}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_{i-1}, \lambda_{i+1}, \cdots, \lambda_n))$ is an ordered bidiagonal representation of order $n-1$.

Proof. Assume that the conditions are true. Then we can find a common L-term in both the numerator and denominator polynomials of the LST of representation $(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n))$ because the rest of the numerator polynomial is divisible by $L(\lambda_i)$. Removing this common L-term reduces the order by 1 and hence we obtain a new representation $(\underline{\gamma}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_{i-1}, \lambda_{i+1}, \cdots, \lambda_n))$.

Since the new representation is constructed from the same LST (albeit simplified), it constitutes the same PH distribution. □

The first observation we can gain from inspecting the form of Equation (1) and Lemma 1 is that $L(\lambda_1)$ cannot divide the numerator polynomial if $\underline{\beta}_1 \neq 0$. Hence, Erlang distributions and the convex combination of Erlang distributions with the same parameter are always irreducible.

To reduce the order of an ordered bidiagonal representation, we need to check two conditions in Lemma 1, namely the divisibility of the numerator polynomial and the sub-stochasticity of the resulting initial probability vector. For this, let

$$R(s) = \underline{\beta}_1 + \underline{\beta}_2 L(\lambda_1) + \cdots + \underline{\beta}_i L(\lambda_1)L(\lambda_2)\cdots L(\lambda_{i-1}). \tag{2}$$

By ordinary algebra, we can check whether $R(s)$ is divisible by $L(\lambda_i)$ simply by checking whether $R(-\lambda_i) = 0$.

The sub-stochasticity of the resulting initial probability vector, on the other hand, can be checked while computing it. Referring back to Lemma 1, let us denote the generator matrix $\mathbf{Bi}_1(\cdot) := \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ and the generator matrix $\mathbf{Bi}_2(\cdot) := \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_{i-1}, \lambda_{i+1}, \cdots, \lambda_n)$. The two ordered bidiagonal representations represent the same PH distribution, hence

$$\begin{aligned} PH(\underline{\beta}, \mathbf{Bi}_1(\cdot)) &= PH(\underline{\gamma}, \mathbf{Bi}_2(\cdot)), \\ 1 - \underline{\beta}\exp(\mathbf{Bi}_1(\cdot)t)\underline{e_n} &= 1 - \underline{\gamma}\exp(\mathbf{Bi}_2(\cdot)t)\underline{e_{n-1}}, \end{aligned} \tag{3}$$

where $\underline{e_n}$ is a column vector of dimension n whose components are all equal to 1. To compute vector $\underline{\gamma}$ from vector $\underline{\beta}$, we need $n-1$ equations relating their components. We can evaluate Equation (3) at $n-1$ different t values to obtain such required system of equations. However, such function evaluations can be costly. To avoid this, we proceed differently.

For a PH representation $(\underline{\alpha}, \mathbf{A})$, the i-th derivative, for $i > 0$, of its distribution function is given by

$$F^{(i)}(t) = -\underline{\alpha}\mathbf{A}^i \exp(\mathbf{A}t)\underline{e}.$$

Evaluating these derivatives at $t = 0$ allows us to avoid computing the exponential of matrices. Hence, the components of vector $\underline{\gamma}$ can be computed by solving the following system of equations

$$\underline{\beta}\mathbf{Bi}_1(\cdot)^i\underline{e_n} = \underline{\gamma}\mathbf{Bi}_2(\cdot)^i\underline{e_{n-1}}, \quad i = 0, \cdots, n-2. \tag{4}$$

Once Equation (4) is solved, the sub-stochasticity of $\underline{\gamma}$ can be determined simply by checking that all of its components are nonnegative real numbers. The equation and the sub-stochasticity of $\underline{\gamma}$ are the sufficient conditions for $\mathbf{Bi}_1(\cdot) = \mathbf{Bi}_2(\cdot)$. For the necessity proof we refer to Lemma 2 in Section 3.5.

3.3 The Algorithm

Let such a state which corresponds to the L-term $L(\lambda_i)$ in Lemma 1 be called a *removable* state. Thus, a state is removable in an ordered bidiagonal representation, if the numerator polynomial of the LST is divisible by the L-term corresponding to the state and if removing the state results in a valid initial probability distribution.

Lemma 1 can be turned into an algorithm that reduces the order of a given APH representation.

Algorithm 1 (Reducing Acyclic Representation).

1: ***function*** REDACYCREP($\underline{\beta}, \mathbf{Bi}$)
2: $n \leftarrow$ ORDEROF($\underline{\beta}, \mathbf{Bi}$)
3: $i \leftarrow 2$
4: ***while*** $i \leq n$ ***do***
5: ***if*** REMOVABLE($i, (\underline{\beta}, \mathbf{Bi})$) ***then***
6: $(\underline{\beta}', \mathbf{Bi}') \leftarrow$ REMOVE($i, (\underline{\beta}, \mathbf{Bi})$)
7: $n \leftarrow n - 1$
8: ***else***
9: $i \leftarrow i + 1$
10: ***end if***
11: $(\underline{\beta}, \mathbf{Bi}) \leftarrow (\underline{\beta}', \mathbf{Bi}')$
12: ***end while***
13: ***return*** $(\underline{\beta}, \mathbf{Bi})$
14: ***end function***

Algorithm 1 inputs an ordered bidiagonal representation and outputs its reduced ordered bidiagonal representation. Function ORDEROF$(\cdot)$ returns the order of the given representation. Function REMOVABLE$(\cdot)$ returns TRUE if state s_i is removable from the given representation. Removing it means redistributing the initial probability distribution, whose computation was shown in Equation (4). This is basically what function REMOVE$(\cdot)$ does, namely removing state s_i from the given representation.

The algorithm proceeds by checking each state whether it is removable or not. If it is removable then it is eliminated. Further, the algorithm terminates once all states have been checked and the removable ones have been eliminated. Hence, the algorithm does what it is supposed to do and terminates.

Let n be the order of the given APH representation. The spectral polynomial algorithm of He and Zhang entails n matrix-vector multiplications, thus amounts to $\mathcal{O}(n^3)$. Checking whether state s_i is removable needs an evaluation of Equation (2), which requires $\left(i^2 - i\right)$ multiplications and $\left(\frac{i^2+i}{2} - 1\right)$ additions. At worst this costs $\mathcal{O}(n^2)$.

If state s_i is removable, eliminating it entails solving Equation (4). We observe that for any bidiagonal generator $\mathbf{Bi}$ of dimension d, $\mathbf{Bi}(s_i, s_i) = -\mathbf{Bi}(s_i, s_{i+1})$ for $0 < i < d$. Now, since both matrices in the system of equations are bidiagonal generators, this observation allows us to show—by induction on n—that

Equation (4) can be transformed into

$$\underline{b} = \mathbf{A}\underline{\gamma}^T \tag{5}$$

where $\mathbf{A}$ is an upper triangular matrix and $\underline{b}$ is a column vector which is obtained by evaluating the left hand side of Equation (4) $n-1$ times. A detailed analysis shows that this transformation requires $(\frac{3n^2+5n}{2})$ multiplications and (n^2) additions, and is thus of order $\mathcal{O}(n^2)$ time. Equation (5), on the other hand, can be solved in $\mathcal{O}(n^2)$ time, because $\mathbf{A}$ is upper triangular, which means we only need to apply backward substitutions.

Since the procedure for checking whether a state is removable and then removing it is actually carried out at most n times, hence the overall time complexity of the algorithm is $\mathcal{O}(n^3 + n(n^2 + n^2 + n^2)) = \mathcal{O}(n^3)$.

Example 3 (Reduced Acyclic Representation). Inputting the model depicted in Fig. 1 into Algorithm 1, it is first transformed into an ordered bidiagonal representation. The ordered bidiagonal representation has the same number of states as the original representation. It is depicted in Fig. 2.

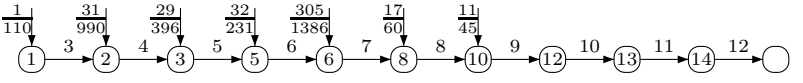


Fig. 3. A Minimal Acyclic Representation of Fig. 2

The ordered bidiagonal representation after all its removable states are eliminated is shown in Fig. 3. It is a minimal representation, because its order is the same as its algebraic degree.

3.4 Non-minimal Representation

In the previous examples, the algorithm reduces the acyclic representation to a minimal representation. This can be verified by the fact that the order of the representation depicted in Fig. 3 is the same as its degree. The degree is precisely the number of L-terms which are not the divisor of the numerator polynomial, which in this case is the same as the number of states in the representation.

However, to arrive at a minimal result is not always possible. To shed some light on this, consider the representations depicted in Fig. 4. Both representations have the same distribution. Given representation in Fig. 4–(left), our algorithm cannot produce smaller representations, because both states 4 and 5 are not removable. However, the representation in Fig. 4–(right) is a minimal representation, and it is smaller than what the algorithm returns.

The reason behind this deficiency lies in the fact that the algorithm is bound to the set of present total outgoing rates (as L-terms), while in reality the representation depends on the interplay of total outgoing rates and initial probability distribution. These dependencies are in full generality difficult to detect, because

Fig. 4. Non-Minimal (left) and Minimal (right) Acyclic Representations

we are then left with the problem of finding matches over a continuous domain of candidates, akin to the nonlinearity of the problem encountered by He and Zhang [16].

3.5 The Use of Order Reduction

Let $Red(\underline{\alpha}, \mathbf{A})$ be the reduced representation of APH representation $(\underline{\alpha}, \mathbf{A})$ obtained by the application of Algorithm 1.

Lemma 2. *Let* $(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n))$ *be an ordered bidiagonal representation. For an arbitrary* $\lambda_x \geq \lambda_i$*, for* $1 \leq i \leq n$*, we can find an ordered bidiagonal representation* $(\underline{\beta}', \mathbf{Bi}(\lambda_1, \cdots, \lambda_i, \lambda_x, \lambda_{i+1}, \cdots, \lambda_n))$ *that has the same distribution.*

The proof of this lemma can be found in [28]. Note that there is a unique way of incorporating a proper new state with a particular total outgoing rate to an existing ordered bidiagonal representation. Furthermore, since the ordered bidiagonal representation is a canonical form, for the particular sets of total outgoing rates, the obtained ordered bidiagonal representation is unique. Note that λ_x must not be less than λ_1, since in that case the resulting representation will have a different distribution.

Lemma 3. *Let* $(\underline{\alpha}, \mathbf{A})$ *be an APH representation. Furthermore, let the algebraic degree and the order of* $PH(\underline{\alpha}, \mathbf{A})$ *be the same. Then the order of* $Red(\underline{\alpha}, \mathbf{A})$ *is the same as the order of* $PH(\underline{\alpha}, \mathbf{A})$*.*

The proof of this lemma can be found in [28]. Lemma 3 basically establishes that applying Algorithm 1 to any APH representation whose algebraic degree is equal to the order of the distribution is certain to result in a minimal representation. The algorithm can also be applied to APH representations whose algebraic degree is different from order of the distribution, although in this case it is not guaranteed to result in a minimal representation.

4 Weak Bisimulation

One may wonder how likely it is to encounter a model which is reducible by our algorithm—or by lumping. Actually, Commault and Mocanu in [10] showed that for any pre-specified structured PH representation of order n, the set of parameter values producing PH distributions of algebraic degree less than n has measure zero. In other words, the chance to find a reducible Markov chain appears very low. This is indeed true for models obtained from parameter estimation mechanism such as fitting methods.

However, this measure based interpretation is often misleading. Many cases have been reported that show the (sometimes tremendous) reduction obtained by lumping a model. The question thus is: why are aggregation techniques for Markov chains such as lumping successful after all? They are indeed successful for representations which are constructed out of a structured behavioral representation. The reason lies in the way models are constructed which usually results in very specific and particular parameter values:

Constructive mechanisms used in building Markovian models from smaller components usually involve convolution, choice and composition operations which correspond to stochastic operations: sum, minimum (or convex combination) and maximum, respectively. These operations often produce models with repeatable, symmetric, and similar sub-structures, which enable reduction by the aggregation techniques. In the context of APH, for instance, the idea of *core series* in [29] identifies these sub-structures in the set of paths to the absorbing state.

To shed some light on the relation of our reduction algorithm to lumpability, we here consider weak bisimilarity. The notion of weak bisimilarity is formalized in the following definition, which is a variation of the original definition [6,4], accounting for the case of absorbing states (and otherwise treating the chain as unlabelled). For $C \subseteq S$, let $\mathbf{R}(s, C) = \sum_{s' \in C} \mathbf{R}(s, s')$. If $\mathcal{R}$ is an equivalence relation on S, $S/\mathcal{R}$ is the partition of S induced by $\mathcal{R}$ and for $s \in S$, $[s]_{\mathcal{R}}$ is the class which contains s.

Definition 1 (Weak Bisimulation). *For $\mathcal{M} = (S, \mathbf{R})$ let $\mathcal{R}$ be an equivalence relation on S. $\mathcal{R}$ is a* weak bisimulation *on $\mathcal{M}$ if for all $s_1 \mathcal{R} s_2$: $\mathbf{R}(s_1, C) = \mathbf{R}(s_2, C)$ for all $C \in S/\mathcal{R}$ with $C \neq [s_1]_{\mathcal{R}}$ and if absorbing states are only related to absorbing states. States s_1 and s_2 are weakly bisimilar, denoted $s_1 \approx s_2$ if and only if there exists a weak bisimulation $\mathcal{R}$ on $\mathcal{M}$ such that $s_1 \mathcal{R} s_2$.*

Weak bisimulation differs from the more prominent notion of strong bisimulation (or ordinary lumpability [7]), in that transitions which do not cross class boundaries are not considered. This can be seen as the 'class generalization' of the fact that in a CTMC the values of $\mathbf{R}(s, s)$ (i.e., loops at states) are irrelevant for the diagonal entries of $\mathbf{Q}$.

States that are weakly bisimilar in a CTMC model can be lumped by moving to the quotient induced by this equivalence, thus producing an aggregated CTMC model. An algorithm for computing the weak bisimulation quotient is at hand [3]. It has cubic complexity, in the number of states of the original model.

Weak bisimulation and lumpability play an important role in CTMC modelling and analysis. Weakly bisimilar states and models possess the same probabilistic reachability properties [4], which means that their probability distributions of reaching certain subsets of the state space are the same. Hence weakly bisimilar models are exchangeable so far as their reachability properties are concerned. Weak bisimulation can be used to identify bisimilar states in a model, and by lumping them, to reduce the order of the model. Weak bisimulation can also be applied to PH representations, without affecting the distribution:

Lemma 4. *Let* $(\underline{\alpha}, \mathbf{A})$ *be a PH representation. If* $(\underline{\beta}, \mathbf{B})$ *is obtained by lumping weak bisimilar states in* $(\underline{\alpha}, \mathbf{A})$*, then* $PH(\underline{\alpha}, \mathbf{A}) = PH(\underline{\beta}, \mathbf{B})$ *and the order of* $(\underline{\beta}, \mathbf{B})$ *is at most the order* $(\underline{\alpha}, \mathbf{A})$*.*

The proof is a straightforward consequence of the fact that weak bisimulation does not alter transient probabilities of state classes [4], together with the particular role played by absorbing states in our variation of weak bisimulation. We now make precise the relation between our order reduction and weak bisimulation.

Lemma 5. *Let* $(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n))$ *be an ordered bidiagonal representation of order* n*. If*

1. $\underline{\beta}_2 \neq 0$*, and*
2. $\underline{\beta}_1 + \underline{\beta}_2 L(\lambda_1) = (\underline{\beta}_1 + \underline{\beta}_2) L(\lambda_i), i \in \{2, \cdots, n\}$ *and* $\underline{\beta}_j = 0$ *for all* $2 < j \leq i$*,*

then both $(\underline{\beta}, \mathbf{Bi}(\lambda_1, \lambda_2, \cdots, \lambda_n))$ *and* $(\underline{\gamma}, \mathbf{Bi}(\lambda_1, \cdots, \lambda_{i-1}, \lambda_{i+1}, \cdots, \lambda_n))$ *represent the same PH distribution, for some sub-stochastic vector* $\underline{\gamma}$ *of dimension* $n - 1$*. Moreover, there is a weak bisimulation relating their respective Cox representations.*

The proof of this lemma can be found in [28]. It says that once an acyclic model is transformed into its ordered bidiagonal representation, it may be possible to reduce the state space by weak bisimulation. In the circumstances described in the lemma, our order reduction coincides with weak bisimulation aggregation on the Cox representation of the APH distribution.

5 Implementation

We have implemented the reduction procedure as described in Algorithm 1 in C++. In order to handle the sensitivity of the initial probability distribution of the ordered bidiagonal representation in relation to the distribution it represents, we have to resort to using rational numbers in the implementation. For this purpose we use GMP library [1]. In this section we report the result of the implementation.

Fig. 5 depicts a model of fault-tolerant parallel processors (FTPP) [13]. The FTPP consists of four network elements (NE), which are fully connected to each others and four groups of processors. In each of these groups, one processor is originally powered down and is used as a spare. Each NE is attached to four processors, one from each group. Overall in the system, there are as many processors in a single group as the number of NEs. At least two of these processors are required to be operational for the whole system to be considered operational, otherwise it is considered failed. Each processor and each NE may experience failures which are governed by some exponential distributions. Furthermore, a failed NE brings down the four processors connected to it. Based on the fault tree of the FTPP model, we obtain the absorbing CTMC model associated with the distribution of the time to system's failure. A more detailed discussion of this reliability model can be found in [5].

The resulting absorbing CTMC model is an APH representation. We use Algorithm 1 to reduce the state space (order) of the representation. Table 1 summarizes the result of the experiments. We have three FTPP models where we vary the number of NEs and thus the number of processors in each group. For each model, the table provides the size of the state space before and after the reduction. Note that the state spaces before the reduction have been minimized according to a stochastic weak bisimulation algorithm. In general, the reduction algorithm produces state spaces which are orders of magnitude smaller than the original ones. The computation time (in seconds) of the reduction (RED) procedure is shown in the last column of the table. The numbers in the brackets are the computation times required to the transformation (TRA) to the ordered bidiagonal representations by the spectral polynomial algorithm.

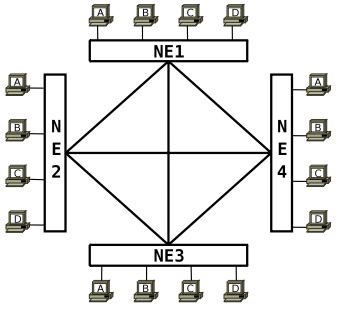


Fig. 5. An FTPP Model

The models prior to the reduction are obtained by composing smaller components. During the composition, the size of the representation blows up fast, because it is basically a cross-product operation over the smaller representations. However, when many of the smaller components are stochastically similar, the composition alters the representation drastically while the stochastic behavior remains the same. We expect the reduction algorithm to be most useful in such circumstances. The FTPP examples we provide above demonstrates this. Moreover, for instance in [29], it is shown that the minimal representation of the compositions of several Erlang distributions is much smaller than the original representation.

Table 1. State Space Reduction of the FTPP Models

#NE	Before Reduction		After Reduction		Time (sec.)
	States	Trans.	States	Trans.	RED (TRA)
4	65	304	22	39	<1 (1)
5	391	2956	41	76	13 (92)
6	1728	17211	64	121	944 (18062)

6 Conclusions

This paper has introduced an algorithm to reduce acyclic representations of PH distributions (viz. acyclic absorbing CTMCs). In each iteration, the algorithm requires quadratic time (in the current number of states), to reduce the representation by at least one state, as long as no further reduction is possible. We have highlighted that the resulting CTMC is not necessarily minimal, and have clarified the relation to weak bisimulation lumping.

In practice, one may wonder whether it is beneficial to run an overall cubic algorithm to reduce the matrix representation of a PH distribution. This, of course depends on the application context. If one intends to numerically compute the absorption probability at many time points (or at a single large time point) it might very well be worthwhile to run the suggested algorithm as a preprocessing step. This step reduces the dimensions of the involved matrices and vectors, and hence speeds up the subsequent (usually uniformization-based) iterations.

Further, if the CTMC representation is used in a concurrency context, then a one-state saving in a single component—prior to exploring the crossproduct with other components—saves states in the order of the entirety of all other components. This can in general lead to an exponential saving in size of the overall system.

References

1. The GNU Multiple Precision Arithmetic Library (2008), `http://gmplib.org/`
2. Asmussen, S.: Phase-type representations in random walk and queueing problems. Annals of Probability 20(2), 772–789 (1992)
3. Baier, C., Hermanns, H.: Weak bisimulation for fully probabilistic processes. In: Grumberg, O. (ed.) CAV 1997. LNCS, vol. 1254, pp. 119–130. Springer, Heidelberg (1997)
4. Baier, C., Katoen, J.-P., Hermanns, H., Wolf, V.: Comparative branching-time semantics for Markov chains. Information and Computation 200(2), 149–214 (2005)
5. Boudali, H., Crouzen, P., Stoelinga, M.: A compositional semantics for dynamic fault trees in terms of interactive Markov chains. In: Namjoshi, K.S., Yoneda, T., Higashino, T., Okamura, Y. (eds.) ATVA 2007. LNCS, vol. 4762, pp. 441–456. Springer, Heidelberg (2007)
6. Bravetti, M.: Revisiting interactive Markov chains. Electronic Notes in Theoretical Computer Science 68(5) (2002)
7. Buchholz, P.: Exact and ordinary lumpability in finite Markov chains. Journal of Applied Probability 31, 59–75 (1994)
8. Chakravarthy, S.R., Krishnamoorthy, A., Ushakumari, P.V.: A k-out-of-n reliability system with an unreliable server and phase type repairs and services: the (N,T) policy. Journal of Applied Mathematics and Stochastic Analysis 14(4), 361–380 (1992)
9. Chakravarthy, S.R., Ravi, K.: A Stochastic Model for a Computer Communication Network Node with Phase Type Timeout Periods, ch. 14. In: Numerical Solutions of Markov Chains, pp. 261–286. Marcel Dekker (1991)
10. Commault, C., Mocanu, S.: A generic property of phase-type representations. Journal of Applied Probability 39, 775–785 (2002)
11. Cumani, A.: Canonical representation of homogeneous Markov processes modelling failure time distributions. Microelectronics and Reliability 2(3), 583–602 (1982)
12. Derisavi, S., Hermanns, H., Sanders, W.H.: Optimal state-space lumping in Markov chains. Information Processing Letters 87(6), 309–315 (2003)
13. Dugan, J.B., Bavuso, S.J., Boyd, M.A.: Dynamic fault-tree models for fault-tolerant computer systems. IEEE Transaction on Reliability 41(3), 363–377 (1992)
14. Fackrell, M.W.: Characterization of Matrix-Exponential Distributions. PhD thesis, School of Applied Mathematics, The University of Brisbane (2003)

15. He, Q.-M., Zhang, H.: Spectral polynomial algorithms for computing bi-diagonal representations for phase type distributions and matrix-exponential distributions. Stochastic Models 2(2), 289–317 (2006)
16. He, Q.-M., Zhang, H.: An Algorithm for Computing Minimal Coxian Representations. INFORMS Journal of Computing, ijoc.1070.0228 (2007)
17. Henley, E.J., Kumamoto, H.: Probabilistic Risk Assessment. IEEE Computer Society Press, Los Alamitos (1992)
18. Hermanns, H.: Interactive Markov Chains: The Quest for Quantified Quality. LNCS, vol. 2428. Springer, Heidelberg (2002)
19. Hillston, J.: A compositional approach to performance modelling. Cambridge University Press, Cambridge (1996)
20. Johnson, M.A., Taaffe, M.R.: The denseness of phase distributions. Purdue School of Industrial Engineering Research Memoranda 88-20, Purdue University (1988)
21. Khayari, R.E.A., Sadre, R., Haverkort, B.R.: Fitting world-wide web request traces with the EM-algorithm. Performance Evaluation 52(2-3), 175–191 (2003)
22. Manian, R., Dugan, J.B., Coppit, D., Sullivan, K.J.: Combining various solution techniques for dynamic fault tree analysis of computer systems. In: Proceedings of the 3rd IEEE International Symposium on High-Assurance Systems Engineering (HASE 1998), pp. 21–28. IEEE Computer Society Press, Los Alamitos (1998)
23. Neuts, M.F.: Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach. Dover (1981)
24. O'Cinneide, C.A.: On non-uniqueness of representations of phase-type distributions. Communications in Statistics: Stochastic Models 5(2), 247–259 (1989)
25. O'Cinneide, C.A.: Characterization of phase-type distributions. Communications in Statistics: Stochastic Models 6(1), 1–57 (1990)
26. O'Cinneide, C.A.: Phase-type distributions and invariant polytopes. Advances in Applied Probability 23(43), 515–535 (1991)
27. O'Cinneide, C.A.: Triangular order of triangular phase-type distributions. Communications in Statistics: Stochastic Models 9(4), 507–529 (1993)
28. Pulungan, R., Hermanns, H.: Effective minimization of acyclic phase-type representations. Reports of SFB/TR 14 AVACS 38, SFB/TR 14 AVACS (March 2008), ISSN: 1860-9821, http://www.avacs.org
29. Pulungan, R., Hermanns, H.: The minimal representation of the maximum of Erlang distributions. In: Proceedings of the 14th GI/ITG Conference on Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB 2008), pp. 207–221. VDE Verlag (2008)

A Response Time Distribution Model for Zoned RAID

Abigail S. Lebrecht*, Nicholas J. Dingle, and William J. Knottenbelt

Department of Computing, Imperial College London,
South Kensington Campus, SW7 2AZ, United Kingdom
{asl102,njd200,wjk}@doc.ic.ac.uk

Abstract. RAID systems are widely deployed, both as standalone storage solutions and as the building blocks of modern virtualised storage platforms. An accurate model of RAID system performance is therefore critical to understanding storage system performance. To this end, this paper presents a queueing network-based model of RAID systems comprised of zoned disks and operating at RAID level 0-1 or 5. The contribution over previous work is twofold. Firstly, our analysis approximates full I/O request response time distributions rather than just mean values. This provides the ability to reason about response time quantiles and higher moments of response time – both of which are useful in the context of modern quality of service requirements. Secondly, we validate our model against measurements from a real RAID system rather than a software simulation. The close agreement between predicted and observed response time distributions gives a high level of confidence in the validity of our model.

1 Introduction

There is an unrelenting business demand for fast, reliable storage. For example, the CIO of Chevron Corporation, the world's fifth largest energy company, claims that Chevron accumulates data at a rate of 2TB per day [1]. Similarly, the news agency Reuters estimates that it manages 1.4PB of data, a figure which is growing at an annual rate of 50%.

Much of this data is ultimately stored on RAID systems, which are deployed either as standalone storage solutions or as the building blocks of virtualised storage infrastructures. The detailed understanding of RAID system performance is therefore critical to determining whether or not application-level quality of service demands will be met by a given storage infrastructure.

In this paper, we propose a novel queueing model for the analysis of RAID systems comprised of zoned disks[1]. The inputs to this analysis are a specified

* Corresponding author. Tel.: +44 20 7594 8385.

[1] On modern hard drives there are more blocks on cylinders on the outside of the platter than those closer to the centre. Cylinders with the same number of blocks are grouped together in zones. Disks rotate with a constant angular velocity and therefore data throughput is higher for outer zones than for inner ones.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 144–157, 2008.

I/O request arrival rate, an I/O request access profile, a given RAID configuration and physical disk parameters. The primary output of this analysis is an approximation to the cumulative distribution function of I/O request response time. From this, it is straightforward to calculate response time quantiles, as well as the mean, variance and higher moments of I/O request response time. This improves on the state-of-the-art in RAID performance models [2,3,4,5,6], all of which yield approximations to the mean response time only.

The first step in building a good RAID model is to derive an accurate model of the behaviour of a single disk drive. To do this, we abstract a disk drive as an M/G/1 queue and model the service time as the sum of the random variables of seek time, rotational latency and data transfer time. In doing so, we take into account the properties of zoned disks. Section 2 describes this zoned disk drive queueing model in detail.

In line with previous work [7], we then abstract a RAID system as a fork-join queueing network. This comprises several queues, each of which represents one disk drive in the array. Incoming I/O requests *fork* into subtasks which are processed across the drives; upon completion of all subtasks, they *join*, signalling completion of the I/O request. Section 3 presents our analytical approximation of such a fork-join queueing network.

We tailor our basic fork-join approximation to account for the I/O request patterns associated with particular request types and request sizes under different RAID levels. We focus on RAID 0-1 (mirrored stripes) and RAID 5 (distributed parity), which are the two most commonly used RAID levels. Section 4 derives the resulting novel cumulative distribution functions of I/O request response time for RAID 0-1 and 5. Models for RAID 0 (striping without redundancy) and RAID 6 (double distributed parity) can also be derived from these results.

To test the accuracy of our resulting models, we validate them against RAID device measurements in Section 5. Section 6 concludes and considers directions for future work.

2 Disk Model

In making performance predictions for a disk array or storage system, it is fundamental to model disk service time accurately. To this end, we model a disk drive as an M/G/1 queue where the service time density is the convolution of seek time, rotational latency and data transfer time densities. Defining random variables for seek time, S, rotational latency, R, and block transfer time, T, we describe their distributions below.

2.1 Seek Time

A seek, S, is the time taken for the disk head to move from the cylinder where it is currently located, C_2, to the cylinder containing a target sector, C_1. We define a random variable, $D = |C_1 - C_2|$, as the seek distance. Seek time can then be

defined in terms of seek distance. Specifically [8],

$$S(D) = \begin{cases} 0 & \text{if } D = 0 \\ a + b\sqrt{D} & \text{otherwise} \end{cases}$$

where a and b are constants defined in terms of the disk geometry, and are given by:

$$a = \frac{minseek\ \sqrt{Cyls - 1} - maxseek}{\sqrt{Cyls - 1} - 1}$$
$$b = \frac{maxseek - minseek}{\sqrt{Cyls - 1} - 1}$$

Here *Cyls* is the total number of cylinders on the disk, *minseek* is the track-to-track seek time and *maxseek* is the full-stroke seek time.

The disk model must reflect the layout of a zoned disk accurately. As cylinders get closer to the disk edge, their circumference increases and the number of sectors per cylinder increases. Therefore, a random request has an increased probability of being directed to a sector on an outer cylinder. Let C be a random variable representing the cylinder number of a randomly selected disk sector. Then the probability distribution of C can be approximated by assuming that the number of sectors per track increases linearly [9]. That is,

$$f_C(x) = \frac{\alpha + \beta x}{\gamma} \qquad x = 0, 1, \ldots, Cyls - 1$$

with constants α, β and γ defined as:

$$\alpha = \frac{SEC[0]}{spb}$$
$$\beta = \frac{SEC[Cyls - 1] - SEC[0]}{(Cyls - 1)\ spb}$$
$$\gamma = \alpha(Cyls - 1) + \frac{\beta}{2}(Cyls - 1)^2$$

where $SEC[0]$ and $SEC[Cyls - 1]$ are the number of sectors on the innermost and outermost tracks respectively and *spb* is the number of physical sectors per logical block. α represents the number of logical blocks on the innermost track and β charts the rate of increase in blocks per cylinder.

The probability density function (pdf) of seek distance is calculated by assuming the two random variables, C_1 and C_2, as two distinct cylinder numbers, and calculating the seek distance between all possible cylinder numbers. This is split into two terms, one for the case when $C_1 \leq C_2$ and one for the case where $C_1 > C_2$:

$$f_D(x) = \int_0^{Cyls-1-x} f_C(y) f_C(x + y) dy + \int_x^{Cyls-1} f_C(y) f_C(y - x) dy \quad (1)$$

The full expansion of Equation (1) appears in [9]. The cumulative distribution function (cdf) of seek time, $F_S(t)$, can be defined in terms of the cdf of $f_D(x)$, $F_D(x)$, as [8]:

$$F_S(t) = F_D\left(\left(\frac{t-a}{b}\right)^2\right)$$

2.2 Rotational Latency

Rotational latency, R, is the time to rotate to the angle of a target sector. R has a uniform distribution with a range between 0 and the time for a full disk revolution, R_{max} [3].

2.3 Data Transfer Time

The time to transfer k logical blocks on cylinder x of a zoned disk can be approximated as [9]:

$$t(x) = \frac{k\ spb\ R_{max}}{\alpha + \beta x}$$

Denoting T_k as the random variable of the time to transfer k blocks of data, its cdf is:

$$\begin{aligned} F_{T_k}(t) &= \int P(T_k \le t \mid C = x) f_C(x) dx \\ &= \int_{\max(\phi_k(t),0)}^{Cyls-1} f_C(x) dx \end{aligned} \tag{2}$$

where

$$\phi_k(t) = \frac{k\ spb\ R_{max}}{\beta t} - \frac{\alpha}{\beta}$$

calculates the minimum cylinder number for which k logical blocks of data can be transferred in under t ms. The solution of the integral in Equation (2) is a function of t with a domain bounded between the minimum and maximum possible k-block transfer times. If t_{min} and t_{max} are the times to transfer to a single physical sector on the outermost and innermost tracks respectively, Equation (2) expands to:

$$F_{T_k}(t) = \begin{cases} 0 & \text{if } t < k\ spb\ t_{min} \\ \frac{1}{2(t_{max}-t_{min})^2\gamma}\left(p + \frac{q}{t} + \frac{r}{t^2}\right) & \text{if } k\ spb\ t_{min} \le t \le k\ spb\ t_{max} \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

with

$$\begin{aligned} p &= (Cyls-1)t_{max}(2(t_{max}-t_{min})\alpha + (Cyls-1)(t_{max}-2t_{min})\beta) \\ q &= ((Cyls-1)t_{max}(-2\ k\ spb(t_{max}-t_{min})t_{min}\alpha + (Cyls-1)k\ spb\ t_{max}\beta \\ &\quad +(1-Cyls)k\ spb(t_{max}-2t_{min})t_{min}\beta)) \\ r &= (1-Cyls)(Cyls-1)k^2 spb^2 t_{max}^2 t_{min}^2 \beta \end{aligned}$$

The Laplace transform of the disk's response time pdf is derived using the Pollaczek-Khintchine transform equation for M/G/1 queues [10]:

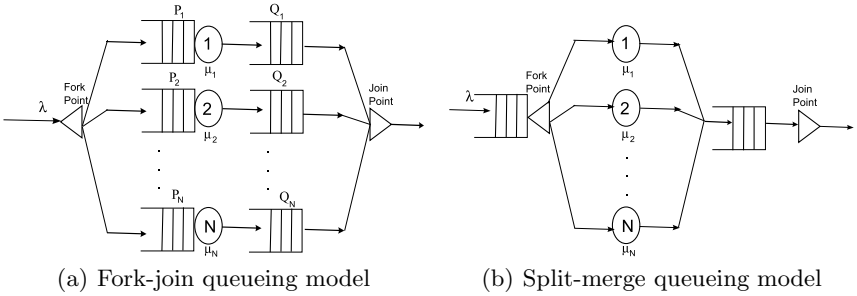
$$W^*(\theta) = \frac{(1-\rho)\theta X^*(\theta)}{\lambda X^*(\theta) - \lambda + \theta}$$

Here $X^*(\theta)$ is the Laplace transform of the service time pdf, which in our case is the product of the Laplace transforms of the pdfs of S, R and T, i.e. $S^*(\theta)R^*(\theta)T_k^*(\theta)$. Also, $\rho = \frac{\lambda}{\mu}$, where λ is the I/O request arrival rate to the disk and μ is the mean service rate, which in our case is given by $\frac{1}{E[R]+E[S]+E[T_k]}$. As $W^*(\theta)$ is unlikely to have an analytical inversion, we invert it numerically using the Euler method [11] to obtain the response time pdf $f_W(t)$. The cumulative distribution function $W(t)$ is also easily obtained by inverting $W^*(\theta)/\theta$.

3 The Fork-Join Queue

Fork-join queues have been widely employed as an appropriate queueing abstraction of the operation of disk arrays [2]. Given N queues, (see Figure 1(a)), each incoming job is split into N subtasks at the fork point. Each of these subtasks queues for service at a parallel service node before joining a queue for the join point. When all N subtasks in the job are at the head of their respective join queues, they rejoin (synchronise) at the join point.

It is difficult to model job response times in a fork-join synchronisation analytically. Indeed, to date, exact analytical results exist only for the mean response time of a two server system consisting of homogeneous M/M/1 queues [12]. Approximate results for mean response times for M/M/1 and M/G/1 fork-join queues are more abundant [12,13,14,15,5]. However, they all have limitations in the context of our present application. In particular, none of these results have yet been extended to find higher moments or full response time distributions. Furthermore, some are not applicable to M/G/1 queues (necessary to support our disk service time model) or heterogeneous servers (necessary to support the modelling of heterogeneous disks), and some can be very computationally



(a) Fork-join queueing model (b) Split-merge queueing model

Fig. 1. Fork-join vs. split-merge queueing models

intensive for a large number of queues (necessary when modelling very large disk arrays). Our present work addresses these issues.

Our approach is inspired by Harrison and Zertal's method for approximating the mean of the maximum of multiple random variables [16]. This gives an approximation to a fork-join synchronisation by assuming it to be a similar queueing network, the split-merge queue [17] (see Figure 1(b)). In the split-merge queue, a job splits into N subtasks which are serviced in parallel. Only when all the subtasks finish servicing and rejoin can the next job split into subtasks and start servicing. This will lead to a slower mean response time than its fork-join equivalent.

An exact solution for the cumulative distribution function of job response time in a split-merge queue can be found by utilising properties of Order Statistics [18,19]. Any random variables, $X_1, X_2, \ldots, X_n$ can be reordered as $X_{(1)}$, $X_{(2)}, \ldots, X_{(n)}$, where $X_{(1)} \leq X_{(2)} \leq \ldots \leq X_{(n)}$. Then $X_{(1)}, X_{(2)}, \ldots, X_{(n)}$ are the order statistics of $X_1, X_2, \ldots, X_n$.

The cdf of $X_{(n)}$, the maximum order statistic (corresponding to job response time in a split-merge queue), is calculated exactly as:

$$\begin{aligned} F_{X_{(n)}}(x) &= P(X_{(n)} \leq x) \\ &= P(X_{(i)} \leq x \ \forall i) \end{aligned}$$

Thus, if $X_1, X_2, \ldots, X_n$ are independent with cdfs $F_i(x)$,

$$F_{X_{(n)}}(x) = \prod_{i=1}^{n} F_i(x) \tag{4}$$

Applying this to a disk array, consider an n-block I/O request sent to an array of n homogeneous disks. If each disk processes a 1-block request and has a response time cdf of $W(t)$, then the approximate response time cdf of the I/O request is $(W(t))^n$. The next section generalises this to deal with variable size I/O requests and I/O request sizes under various RAID levels.

4 RAID Model

The results derived in Equation (4) above suffice to calculate the response time cdf for read or write requests to an n-disk RAID 0 system in which each request consists of a multiple of n blocks. However, not every I/O request leads to an access to all disks, being influenced by I/O request size and type, and also by RAID level. Below we deal with RAID levels 0-1 and 5, for both read and write requests.

Our model is designed to accept a homogeneous stream of I/O requests of a given size and type. We further assume that all the service time distributions on all disks are identically distributed. For the sake of notational simplicity, let $W(t, \gamma, \mu)$ define the cdf of the response time distribution of a single M/G/1 queue (disk), γ is the arrival rate at an individual disk and μ is the mean service rate. We assume there are n disks in the array and that the arrival rate of logical I/O requests to the disk array as a whole is λ.

4.1 RAID 0-1

Read Requests. Assuming an efficient RAID controller, a b-block read on RAID 0-1 can read data from either primary or mirror disks. With $b \geq n$, we thus utilise all n disks of the array (and not $\frac{n}{2}$ disks) to give better performance results for medium and large sized requests. However, if $b < n$ only b disks are utilised at any time. To account for this, we view the system as a b-queue fork-join queue. The arrival rate at the disks needs to be modified since each request only arrives at b of the n disks.

Therefore the cdf of the response time distribution for a read on a RAID 0-1 system is:

$$\begin{cases} \left(W\left(t, \frac{\lambda b}{n}, \frac{1}{E[R]+E[S]+E[T_1]}\right)\right)^b & \text{if } b < n \\ \left(W\left(t, \lambda, \frac{1}{E[R]+E[S]+E[T_{\frac{b}{n}}]}\right)\right)^n & \text{otherwise} \end{cases}$$

Write Requests. A b-block write must account for each request being written on both a primary and mirror disks. The corresponding response time cdf is defined as:

$$\begin{cases} \left(W\left(t, \frac{2\lambda b}{n}, \frac{1}{E[R]+E[S]+E[T_1]}\right)\right)^{2b} & \text{if } 2b < n \\ \left(W\left(t, \lambda, \frac{1}{E[R]+E[S]+E[T_{\frac{2b}{n}}]}\right)\right)^n & \text{otherwise} \end{cases}$$

4.2 RAID 5

Read Requests. A read request under RAID 5 is modelled in the same way as the equivalent read request under RAID 0-1. Note that, since RAID 5 distributes data (and parity) across all disks, a $b \geq n$ read request will access all n disks, despite the stripe size of $n-1$ disks. The response time cdf is:

$$\begin{cases} \left(W\left(t, \frac{\lambda b}{n}, \frac{1}{E[R]+E[S]+E[T_1]}\right)\right)^b & \text{if } b < n \\ \left(W\left(t, \lambda, \frac{1}{E[R]+E[S]+E[T_{\frac{b}{n}}]}\right)\right)^n & \text{otherwise} \end{cases}$$

Write Requests. The behaviour of a RAID 5 write depends on the size of the request, with different methods used to update the parity.

Small Partial Stripe Write If a request consists of $b < \frac{n-1}{2}$ blocks (i.e. a small partial stripe write), then parity is calculated using [20]:

$$new\_parity = new\_data \oplus old\_data \oplus old\_parity$$

where $\oplus$ is the exclusive-or (XOR) operator. This is a read-modify-write operation. Each of the b blocks and parity must be transferred twice, first to read the old data and parity, then to write the new data and parity. When the old

data and parity have been read from all disks, a new request will be issued to write the new data and parity to the same disks. This request is given priority in the queue, so at least one disk (the last to complete the pre-read) will just have completed reading a data or parity block that now needs to be re-written. Therefore we add a full disk rotation into the service time distribution. However, it is likely that by the time the last disk has completed its pre-read, the remaining disks will have started servicing the next I/O request in their queues. These disks will need to re-seek to write the new data and parity. Therefore, we assume that b disks seek again on the second request, while one disk needs a complete rotation only.

The request to pre-read will have a mean service time of $E[R]+E[S]+E[T_1]$. The arrival rate at each of $b+1$ disks for both requests (i.e. the pre-read and data transfer operations) is $\lambda(b+1)/n$. Combining both arrival streams, we approximate the cdf of the response time as:

$$\left(W\left(2t, \frac{2\lambda(b+1)}{n}, \frac{1}{\frac{(2b+1)(E[R]+E[S])+R_{max}}{2(b+1)}+E[T_1]}\right)\right)^{b+1}$$

The mean service time in the above is calculated by averaging the mean services times of the pre-read and data transfer operations. Thus, the pdfs of seek time and rotational latency are altered to:

$$f'(t) = \begin{cases} \frac{1}{2(b+1)} & \text{if } x = 0 \\ \frac{2b+1}{2(b+1)} f(t) & \text{otherwise} \end{cases}$$

where $f(t)$ represents the probability density function of seek time or rotational latency.

Large Partial Stripe Write. If $\frac{n-1}{2} \leq b < n-1$ (i.e. a large partial stripe write), then to minimise disk accesses the parity is calculated by reading only from the disks that are not being written to. The new parity is calculated by XOR-ing the data that will be written with the data from the disks that will remain unchanged. This is a reconstruct-write operation. The first request pre-reads $n-1-b$ blocks of data for the calculation of the new parity. When all $n-1-b$ disks complete their pre-read, a new request is sent to the other $b+1$ disks to write the new data and parity. The arrival rate for the pre-read will be $\lambda(n-1-b)/n$, and we compute the time to complete this phase as the slowest of the $n-1-b$ queues. The arrival rate of the write request will be $\lambda(b+1)/n$ to $b+1$ queues. Both requests will have the same mean service time of $E[R]+E[S]+E[T_1]$. Averaging the number of queues we are finding the maximum of $(n/2)$, we approximate the response time cdf of the two requests required (pre-read and data transfer) as:

$$\left(W\left(2t, \lambda, \frac{1}{E[R]+E[S]+E[T_1]}\right)\right)^{n/2}$$

Full Stripe Write. If a request consists of a number of complete stripes (i.e. $b \bmod (n-1) = 0$), no pre-reads are needed to calculate the parity. All the disks are utilised, with either the new data block or the new parity block written to each disk. The response time cdf is:

$$\left(W\left(t, \lambda, \frac{1}{E[R] + E[S] + E[T_{\frac{b}{n-1}}]}\right)\right)^n$$

Full Stripe Followed by Small Partial Stripe Write. If $b > n-1$ and $0 < b \bmod (n-1) < \frac{n-1}{2}$, at least one full stripe write will occur followed by a small partial stripe write. Let $k = \lfloor \frac{b}{n-1} \rfloor$ and $b_{mod} = b \bmod (n-1)$. We assume that there are two types of requests to be averaged. The first request involves k full stripe writes, followed by a parity pre-read to $b_{mod}+1$ disks. The second request writes the new data and parity to $b_{mod}+1$ disks. The response time cdf is then approximated as:

$$\left(W\left(2t, \frac{\lambda(n+b_{mod}+1)}{n}, \frac{1}{\frac{(n+b_{mod})(E[R]+E[S])+R_{max}}{n+b_{mod}+1} + E[T_{\frac{k}{2}+\frac{b_{mod}+1}{n}}]}\right)\right)^{\frac{n+b_{mod}+1}{2}}$$

Full Stripe Followed by Large Partial Stripe Write. If $\frac{n-1}{2} \leq b \bmod (n-1) < n-1$, at least one full stripe write will occur followed by a large partial stripe write. The initial request will be to write k blocks to all disks and then pre-read an additional block on $n - b_{mod} - 1$ disks. The second request, issued upon the completion of the first, writes the new data and parity to the remaining $b_{mod}+1$ disks. If one of the $n - b_{mod} - 1$ pre-reading disks complete service last, then all the disks to be written to will need to seek to write the new data. However, it is possible that, despite the smaller mean service time, one of the $b_{mod}+1$ disks will complete last. This disk will not need to seek or wait for rotation at all, and will be in position to write the new data immediately. To account for these possibilities, we assume that $b_{mod}+0.5$ disks will need to seek again in the second request. The cdf is:

$$\left(W\left(2t, \frac{\lambda(n+b_{mod}+1)}{n}, \frac{1}{\frac{(n+b_{mod}+0.5)(E[R]+E[S])}{2n} + E[T_{\frac{k+1}{2}}]}\right)\right)^{\frac{n+b_{mod}+1}{2}}$$

5 Validation

To provide confidence in our analytical models, we validate them against device measurements. Our experimental platform consists of an Infortrend A16F-G2430 RAID system containing 4 Seagate ST3500630NS disks. Each disk has 60801 cylinders. A sector is 512 bytes and we have derived from measurement that the time to write a single physical sector on the innermost and outermost tracks are 0.012064ms (t_{max}) and 0.005976ms (t_{min}) respectively. The stripe width on the

array is configured as 128KB, which we define as the block size. Therefore there are 256 sectors per block. The time for a full disk revolution is 8.33ms. A track to track seek takes 0.8ms and a full-stroke seek requires 17ms for a read request; the same measurements are 1ms and 18ms respectively for write requests [21].

To obtain response time measurements, we implemented a benchmarking program that issues read and write requests using a master process and a number of child processes. The master process constructs a list of arrival times according to a specified distribution (here a negative exponential distribution to conform to the model assumptions) and then monitors the system clock until the first generated arrival time occurs. When it does, a child process is spawned which performs the read/write operation and measures the time taken. This leaves the master process free to spawn further child processes at the calculated arrival times without the need for it to wait for previously-issued operations to complete.

Throughout, it was necessary to minimise the effects of buffering and caching as these are not represented in the model. We therefore disabled the write-back cache on the RAID system and set the read-ahead buffer to 0KB. Furthermore, devices are opened with the `O_DIRECT` flag set. For each of the experiments presented below, 100 000 requests were issued with an arrival rate of $\lambda = 0.02$ requests per millisecond to random logical locations. The resulting cumulative distribution functions of the response times were calculated using the statistical package `R`.

5.1 Data Transfer Model

The data transfer model in Equation (3) can be validated by setting the number of sectors to write to a large enough number that the seek and rotation time will be insignificant in comparison to the transfer time. We thus write 100MB in each transfer to a single ST3500630NS disk connected directly to a separate test machine. This ensures that any overheads imposed by the RAID controller are bypassed. We also ensure that no queueing occurs by waiting until a request

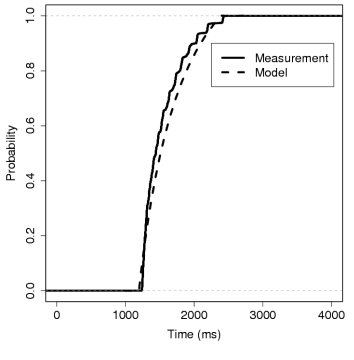


Fig. 2. Data transfer time of 100MB requests on a Seagate ST3500630NS disk, compared with analytical model of zoned data transfer time

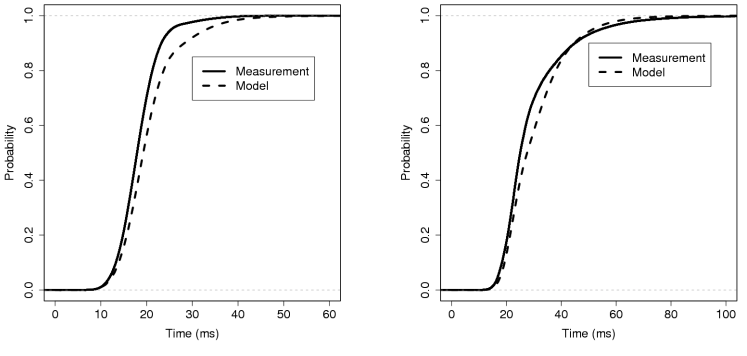
Table 1. Response time mean and variance comparison for read and write requests on RAID 0-1 and RAID 5 ($\lambda = 0.02$ requests/ms)

		Blocks	Mean measured (ms)	Mean model (ms)	Variance measured (ms^2)	Variance model (ms^2)
RAID 0-1	read	2	18.3	20.4	20.3	40.1
	write	2	29.0	30.3	164.9	119.8
RAID 5	read	3	22.6	24.2	58.9	66.4
	write	1	51.9	48.4	286.2	466.8
		2	50.0	50.1	257.9	411.5
		3	31.2	30.3	186.3	119.8
		4	70.9	69.0	2445.8	975.1
		5	65.5	66.3	2256.3	848.0

completes before issuing another. Figure 2 compares the cumulative distribution function of the analytical data transfer time model with device measurements. The effects of disk zoning are clearly evident in the measurements, which are a close match to the analytical model.

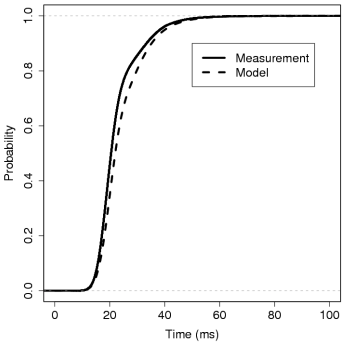
5.2 RAID 0-1

Figure 3(a) displays the measured and modelled cdfs for read requests on a four disk RAID 0-1 system, and Figure 3(b) shows the corresponding cdf for write requests. We observe good agreement between model and measurement. Table 1 further illustrates the accuracy of the model, comparing mean and variance for the model and measured results.

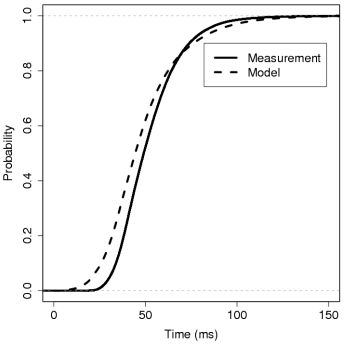


(a) 2-block read request (b) 2-block write request

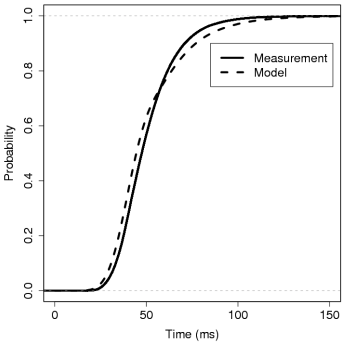
Fig. 3. Cumulative distribution functions of RAID 0-1 I/O request time on a 4 disk RAID system ($\lambda = 0.02$ requests/ms)



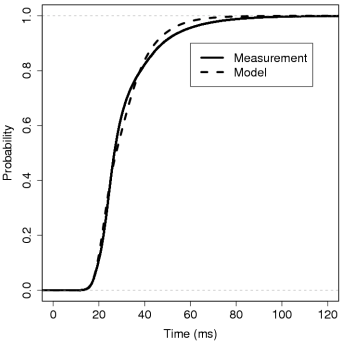
(a) 3-block read request



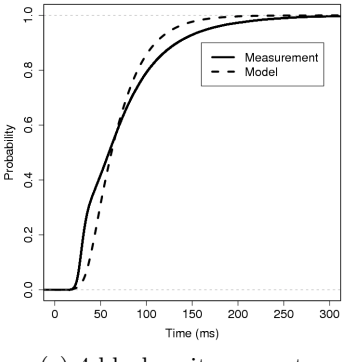
(b) 1-block write request



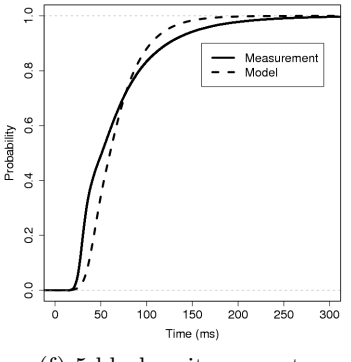
(c) 2-block write request



(d) 3-block write request



(e) 4-block write request



(f) 5-block write request

Fig. 4. Cumulative distribution functions of RAID 5 I/O request time on a 4 disk RAID system ($\lambda = 0.02$ requests/ms)

5.3 RAID 5

The analytical model assumes that all writes begin with a full stripe write, or, if the request is for fewer blocks than a full stripe, then all blocks in the request are written to the same stripe. Consequently, for the measurements presented here, the

alignment of I/O request starting locations on the RAID system was constrained to ensure that this assumption held. We note that more general alignments will occur in practice; extending the model to account for this is part of our future work.

Figure 4 displays the measured and modelled cdfs of I/O request response time on a four disk RAID 5 system. Figure 4(a) compares the distributions for 3-block read requests. Figure 4(b) is a small partial stripe write and Figure 4(c) is a large partial stripe write. Figures 4(d), 4(e) and 4(f) are full stripe requests, full stripes followed by a small partial stripe, and full stripes followed by a large partial stripe, respectively.

Table 1 compares means and variances in all the above cases, and again we note good agreement between the measured and modelled results. The slight discrepancies between measured and modelled variances are caused by the model's simple abstraction of the array's behaviour. Specifically, for partial stripe request sizes, each request will not transfer to all disks in the array in a single request. To approximate this, the model transfers a fraction of a single block to each disk. This enables accurate mean response time results, but the variance suffers.

6 Conclusion and Future Work

In this paper we have developed new methods for modelling the performance of RAID systems. Our analytical queueing models enable, for the first time, the calculation of an approximation to the response time distribution of I/O request response time in these systems. These results are validated against device measurements from a real RAID system, demonstrating the accuracy of the analytical models.

There are three ways in which we will seek to relax constraints on our model, making it more applicable in the context of real I/O workloads and systems. Firstly, caching is not yet supported in our model, and therefore in our measurements all caching (read-ahead and write-through) was disabled both on the disk and on the RAID controller. However, we appreciate the important role that caching at both these levels plays in I/O performance and we will therefore seek to incorporate it. Secondly, as discussed in Section 5.3, we currently constrain the alignment of RAID 5 write requests. In the future, we intend to extend the RAID 5 write model to describe requests that start with a partial stripe, followed by further data. Finally, we have assumed Markovian arrivals in our model, and have generated request streams that conform to this assumption for our measurements. We intend to compare the model response times with response times generated from real I/O traces. With these constraints removed, we can then extend the model to allow I/O request streams consisting of mixed request types and sizes. This will involve converting our analytical model into a multi-class queueing system.

Acknowledgements

We are grateful to Peter Harrison and Soraya Zertal for helpful discussions. This work is supported by EPSRC research grant EP/F010192/1.

References

1. Gantz, J.F.: The expanding digital universe: A forecast of worldwide information growth through 2010. White paper, IDC (2007)
2. Lee, E.K.: Performance Modeling and Analysis of Disk Arrays. PhD thesis, University of California, Berkeley (1993)
3. Chen, S., Towsley, D.: A performance evaluation of RAID architectures. IEEE Transactions on Computers 45, 1116–1130 (1996)
4. Harrison, P.G., Zertal, S.: Queueing models of RAID systems with maxima of waiting times. Performance Evaluation 64, 664–689 (2007)
5. Varki, E.: Response time analysis of parallel computer and storage systems. IEEE Transactions on Parallel and Distributed Systems 12, 1146–1161 (2001)
6. Varki, E., Merchant, A., Xu, J., Qiu, X.: Issues and challenges in the performance analysis of real disk arrays. IEEE Transactions on Parallel and Distributed Systems 15, 559–574 (2004)
7. Chen, P.M., Lee, E.K., Gibson, G.A., Katz, R.H., Patterson, D.A.: RAID: High-Performance, Reliable Secondary Storage. ACM Computing Surveys 26, 145–185 (1994)
8. Chen, S., Towsley, D.: The design and evaluation of RAID 5 and parity striping disk array architectures. IEEE Transactions on Parallel and Distributed Systems 17, 58–74 (1993)
9. Zertal, S., Harrison, P.G.: Multi-RAID queueing model with zoned disks. In: High Performance Computing and Simulation Conference (HPCS 2007) (2007)
10. Harrison, P.G., Patel, N.M.: Performance Modelling of Communication Networks and Computer Architectures. Addison-Wesley, Reading (1993)
11. Abate, J., Whitt, W.: The Fourier-series method for inverting transforms of probability distributions. Queueing Systems Theory and Applications 10, 5–88 (1992)
12. Nelson, R., Tantawi, A.N.: Approximate analysis of fork/join synchronization in parallel queues. IEEE Transactions on Computers 37, 739–743 (1988)
13. Varma, S., Makowski, A.M.: Interpolation approximations for symmetric fork-join queues. In: Proc. Performance 1993, pp. 245–265 (1994)
14. Thomasian, A., Tantawi, A.N.: Approximate solutions for M/G/1 fork/join synchronization. In: Proc. WSC 1994, pp. 361–368 (1994)
15. Varki, E.: Mean value technique for closed fork-join networks. In: Proc. ACM SIGMETRICS, pp. 103–112 (1999)
16. Harrison, P.G., Zertal, S.: Queueing models with maxima of service times. In: Proc. TOOLS Conference, pp. 152–168 (2003)
17. Duda, A., Czachórski, T.: Performance evaluation of fork and join synchronization primitives. Acta Informatica 24, 525–553 (1987)
18. David, H.A.: Order Statistics. John Wiley and Sons, Inc., Chichester (1981)
19. Lebrecht, A.S., Knottenbelt, W.J.: Response time approximations in fork-join queues. In: 23rd UK Performance Engineering Workshop (UKPEW) (2007)
20. Patterson, D.A., Gibson, G., Katz, R.H.: A case for redundant arrays of inexpensive disks (RAID). In: Proc. International Conference on Management of Data (SIGMOD) (1988)
21. Seagate: Barracuda ES Data Sheet (2007) `http://www.seagate.com/docs/pdf/datasheet/disc/ds_barracuda_es.pdf`

Exact Sojourn Time Distribution in an Online IPTV Recording System

Tobias Hoßfeld[1], Kenji Leibnitz[2], and Marie-Ange Remiche[3]

[1] University of Würzburg, Institute of Computer Science,
Am Hubland, 97074 Würzburg, Germany
hossfeld@informatik.uni-wuerzburg.de
[2] Osaka University, Graduate School of Information Science and Technology
1-5 Yamadaoka, Suita, Osaka 565-0871, Japan
leibnitz@ist.osaka-u.ac.jp
[3] Université Libre de Bruxelles, CoDE/SMG, CP 165/15,
Av. F.D. Roosevelt, 50; B-1050 Bruxelles, Belgium
mremiche@ulb.ac.be

Abstract. In this paper we analytically derive the sojourn time of a user accessing an online IPTV recording service. Basically, the system consists of a server (or server farm) and the bandwidth at which a user can download recorded files depends on the number of other concurrently downloading users. In particular, we consider both cases, one in which the user's download speed is limited by his own access bandwidth, whereas in the other case the server is the bottleneck and all downloading users share the server's upload capacity. Our model includes user impatience which depends on the actual download speed. A user may abort his download attempt if a threshold duration is exceeded. The file size distribution is obtained by measurements of offered files at an existing server.

1 Introduction

With the recent advancements in access technology for end-users, the Internet is becoming an attractive medium for distributing large volume contents. This is one of the reasons that *Internet-based Television* (IPTV) has gained enormous popularity as a means of delivering high-quality video images, especially since it is capable of offering additional interactive features in contrast to conventional TV, such as chats, rankings, or discussion forums of the shows. Unlike television programs that are broadcast as terrestrial, cable, or satellite signals, IPTV transmits the video data as stream of IP packets and permits the user to ubiquitously access his favorite TV show on-demand, whenever, wherever, and on whatever device (TV, PC, portable player). It is, therefore, not really surprising that a wide range of new free and commercial services are currently being offered over the Internet, e.g. *YouTube*, *Joost*, or *Zattoo* which have become extremely popular. For instance, as of February 2008 the free Japanese service *GyaO* counted about 18.3 million registered subscribers [1], of which about 3-4 million are estimated to be active weekly, in spite of the fact that access to the service is only limited from within Japan.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 158–172, 2008.

As discussed in [2], these previously mentioned IPTV systems differ significantly in their network architecture (server-based vs. peer-to-peer), as well as in their video delivery mechanism. Basically, there are three major categories of IPTV content distribution methods. First, there is *live TV streaming*, in which the current live TV program is packetized and simply streamed as application-layer multicast to the connected overlay peers, which then share the stream with other peers. Then, there are *Video-on-Demand* (VoD) systems that are not restricted to any broadcasting schedules of TV stations, but permit the user to browse a catalogue of available video files and play the content entirely upon request. For both solutions, respectively, Zattoo and Joost are two popular services, both based on peer-to-peer technology. However, as they operate on proprietary protocols and architectures, an analytical evaluation for such systems can only be done from the edge of the network by studying the user's perceived *quality of service* (QoS) or *quality of experience* (QoE).

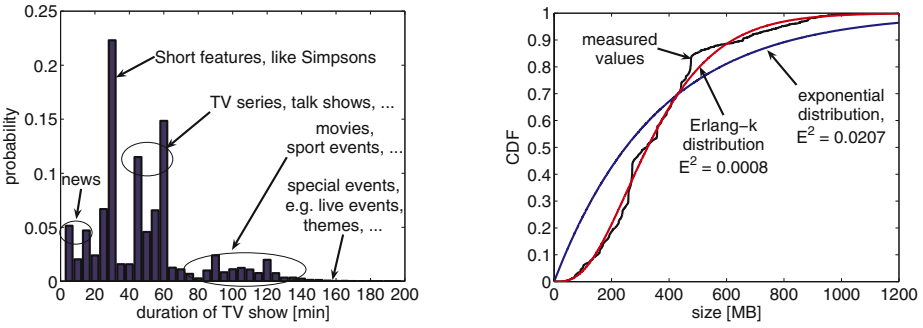
On the other hand, *network-based TV recorders* operate basically in the same way as home hard-disc video recorders with the only difference that the content is recorded and stored at some remote server in the Internet. An example for such a service is *OnlineTVRecorder* (OTR) in Germany [3]. The live TV program is recorded at the OTR server and registered users can download their previously programmed shows and later view them offline on a PC or handheld device. We will describe OTR in more detail in Section 2.

In this paper we extend our previous work in [4] on modeling the performance of an OTR video delivery service by means of a Markov model to derive the stationary sojourn time, which corresponds to the time until a typical user completes the download of a file. Since the file sizes are very large, the download duration may take longer than the user is willing to wait. For this reason, we include the user impatience in our model, where a waiting or downloading user may leave the system before completing the download. Queuing models with user impatience can be found in [5,6], however, those models can not be applied to our system. For further discussion of the existing literature, see [4].

The remainder of this paper is organized as follows. In Section 2, we describe the basic operation of an OTR server and provide measurements for characterizing the content offered at the server. This is required to approximate the service time in our analytical model, which we define in Section 3. Next, we obtain the stationary sojourn time of a typical customer in Section 4, defined as the time needed to completely download the desired video file. In Section 5 we numerically investigate the influence of the system parameters on the sojourn time and conclude our work in Section 6.

2 Model of the OTR Recording System

The access to OTR is provided by a portal at its main web page [3]. There, a registered user has the possibility to choose programs he wishes to record from an *electronic program guide* (EPG) or can download previously recorded shows. The recordings can be either downloaded directly from the main server, from



(a) Distribution of TV show durations (b) CDF of files sizes

Fig. 1. Measurements of offered files at an OTR server

user-created mirror sites, or alternatively via P2P file-sharing networks (eMule or BitTorrent). However, the majority of clients are using the HTTP-based server download platforms and in this work we will focus on this type of file transfer. In the following, when we refer to OTR server (or simply *server*), we treat the main server and the mirror servers in the same way, since their basic operation is the same with the only difference that mirror sites usually offer only a subset of available recordings after a slight delay.

As mentioned above, the recorded files are offered at a server via HTTP which can be freely accessed over the Internet. In order to safeguard licensing restrictions and prevent unauthorized access, a user may only download files that he had previously recorded. For this reason, the content is offered encoded which can not be played directly, but must be decoded prior to playback. When requesting a file, the user may begin immediately his download if the server has available download slots, else he will be put on a waiting queue. OTR also offers prioritizing certain users who provide donations, but we assume that all users are treated equal in this paper. Eventually, as over time other users will finish their downloads and leave the system, the waiting user will commence his download.

We investigated the actual file sizes of video files offered at an OTR server by measurements, which were taken in April 2007 consisting of 11563 randomly selected file samples from 19 different TV channels. According to the information provided by OTR, standard video files are encoded at a resolution of 512 × 384 pixels at a video bitrate of about 750 kbps and an audio bitrate of 128 kbps [3]. The measured data contains only standard quality video files and consists of approximately 80% encoded in DivX format and 20% in Windows Media Video (WMV) format.

Fig. 1(a) shows the probability distribution of the TV show durations in minutes. The majority of the files (95%) are discretized in units of 5 min. We can distinguish 4 major categories of TV shows. Most files are short features (e.g. animation series) of about 30 min and shorter files may be for instance news reports. Another peak can be found between 45-60 min, which is the usual duration of TV dramas or other periodical shows. Movies usually have the duration between 90-120 min and very few larger recordings of special events exist, like broadcasts of live sports events.

In this paper, we are more interested in the file size distribution than the duration of the shows themselves in order to approximate the download time. Fig. 1(b) shows that the actual file size distribution has a mean of 368.31 MB and standard deviation of 196.82 MB. It can be well fitted by an Erlang-distribution with only small residual mean squared error $E^2 = 0.0008$. We will assume an exponential file size distribution for the sake of analytical tractability in spite of its slightly higher residual error, cf. Fig. 1(b), however, the equations developed here could be extended to the Erlang case, too.

3 Description of the Model

Let us assume an OTR server responsible of managing the demands of a maximum finite number of $\underline{N}$ customers. In the following we will describe the model of the behavior of one of these servers.

We assume first that user requests arrive at the server according to a Poisson process with parameter $\lambda > 0$. When a request arrives and the system has free download slots, the client immediately proceeds with the download. Then, the user becomes a *downloading client* and we also say that the customer is *served*.

We may further assume that the server has a total fixed upload bandwidth of capacity C. In our model, the capacities are normalized by C, thus without loss of generality the normalized server capacity is 1. This bandwidth is equally shared among a maximum of $n^\star$ simultaneously downloading clients. If there were more than $n^\star$ simultaneous clients, the exceeding customers would wait for a downloading slot to become free. We refer to those clients as *waiting clients*. The total number of downloading and waiting clients at the server is thus in this particular setting finite (with a maximum number equal to $\underline{N}$).

When downloading a file, the access bandwidth of the client may be the bottleneck, that is R/C, where R is the maximum physical download bandwidth of the customer and C the real capacity at the server. We assume this bandwidth R is the same for all customers. Clearly, the condition $R/C < 1$ must hold.

The average rate at which clients complete their downloads also depends on the file size. We assume here that the file size is randomly distributed following an exponential distribution of parameter μ reflecting the measurement results described in Section 2 and normalized by the system capacity.

While in the system, a client might become *impatient* and decides to leave the system after a random amount of time. We assume that the average impatience duration depends on the speed of the download. That is, when there are less than $n^\star$ customers in the system, the impatience duration is distributed according to an exponential random amount of time with average θ_1^{-1}. In the other case, i.e., there are more than $n^\star$ customers in the system, the impatience duration for customers being served remains the same, but the average impatience time for waiting customers is $\theta_2^{-1} < \theta_1^{-1}$.

Our objective is to derive the time needed for an arbitrary customer to successfully complete the download of a file. We call this the *sojourn time* of a customer.

4 Derivation of the Stationary Sojourn Time Distribution

The changeover point $N^\star$ reflects whether the user's access bandwidth or the server's capacity is the limiting factor. Let first $N^\star$ be such that

$$\frac{C}{N^\star - 1} > R \qquad \text{and} \qquad \frac{C}{N^\star} \leq R\,. \tag{1}$$

We assume that $N^\star < n^\star$, otherwise the resulting model is trivial.

We are interested in computing the exact sojourn time a customer spends in the system in order to completely download the entire file. The method we will apply actually consists of solving a system of differential equations and is inspired by the work of Sericola et al. in [7] or Masuyama and Takine in [8]. However, our system is more complex since it integrates impatience and different service behaviors according to the actual number of customers in the system. We first derive equations for the remaining sojourn time distribution of an observed customer, then we establish the stationary distribution of the number of customers in the system and finally obtain the stationary distribution of the sojourn time of an arbitrary customer.

4.1 Remaining Sojourn Time

When the system counts less than $N^\star$ customers, any new customer is served at an average speed of $(R/C\mu)^{-1}$. Obviously, in this case the bandwidth is not shared. However, as soon as the system counts more than $N^\star$ clients, say n_d clients, the bandwidth of our observed customer shrinks to $(\mu/n_d)^{-1}$ on average. It is important to know exactly when this happens.

It is clear that as soon as our observed customer is in service, i.e. downloading the file, he will continue until either the file is completely downloaded or the customer becomes impatient and leaves the system before completion. Nevertheless, we still need to take the arriving customers following his arrival into account, even if they do not directly interfere with our observed customer's sojourn time, i.e. if they are waiting customers. Indeed, these waiting customers will eventually become downloading customers. Accordingly, the service rate will remain at a level $\mu/n^\star$, even when a customer in service leaves the system.

Imagine now our customer entering the system counting already more than $n^\star$ clients. Our observed customer becomes, thus, a waiting customer. It is then important to know exactly how many clients were waiting prior to his arrival in order to exactly determine when his service will start. In the same manner we also need to record how many clients arrive after his arrival, in order to determine the subsequent speed of service.

Owing to the necessity to keep track of the actual number of the other clients in the system and, therefore, to differentiate between the system behavior, we define the following three different conditional random variables.

For $K \in \{0, 1, \ldots, n^\star - 1\}$, we define the random variable $W(K, 1)$ as the remaining time a customer needs in order to completely download the file he requested, given that there are K customers in service. For $K \in \{n^\star, \ldots, \underline{N} - 1\}$

and $N \in \{n^\star + 1, \ldots, \underline{N}\}$, we define the two random variables $W(K,1)$ and $W(K,0,N)$ according to whether our customer is in service or not. The r.v. $W(K,1)$ is the remaining sojourn time of the observed customer when the system counts K customers. The r.v. $W(K,0,N)$ is the remaining sojourn time of the observed customer when there are K customers in the system and our observed client is waiting at position N to be served, i.e. $N - n^\star$ customers need to leave the system before our customer starts downloading the file. Note that these $N - n^\star$ clients have to be clients in service or waiting customers located in front of our observed customer in the queue.

We denote by $\mathcal{E}(x)$ an exponentially distributed random variable with mean $1/x$ and formulate the following theorem, considering all possible cases that can occur.

Theorem 1. *For $K \in \{0, \ldots, N^\star - 1\}$, the remaining sojourn time of a customer $W(K,1)$ in a system that counts K customers is such that:*

$$W(K,1) = \begin{cases} \mathcal{E}(\Lambda(K)) & w.p.\ \mu R/C(\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K-1,1) & w.p.\ K(\mu R/C + \theta_1)(\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K+1,1) & w.p.\ \lambda(\Lambda(K))^{-1} \end{cases} \tag{2}$$

where $\Lambda(K)$ is the exponentially distributed rate at which the next event occurs that changes the system state:

$$\Lambda(K) = (K+1)\,(\mu R/C + \theta_1) + \lambda. \tag{3}$$

When K belongs to $\{N^\star, \ldots, n^\star - 1\}$, we have:

$$W(K,1) = \begin{cases} \mathcal{E}(\Lambda(K)) & w.p.\ \mu((K+1)\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K-1,1) & w.p.\ (K/(K+1)\mu + K\theta_1)\,(\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K+1,1) & w.p.\ \lambda(\Lambda(K))^{-1} \end{cases} \tag{4}$$

where

$$\Lambda(K) = \mu + (K+1)\,\theta_1 + \lambda. \tag{5}$$

When $K \in \{n^\star, \ldots, \underline{N} - 2\}$, the remaining sojourn time of the observed customer, assuming he is already in service, is:

$$W(K,1) = \begin{cases} \mathcal{E}(\Lambda(K)) & w.p.\ \mu(n^\star\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K-1,1) & w.p.\ \frac{(n^\star-1)/n^\star\mu + (n^\star-1)\theta_1 + (K+1-n^\star)\theta_2}{\Lambda(K)} \\ \mathcal{E}(\Lambda(K)) + W(K+1,1) & w.p.\ \lambda(\Lambda(K))^{-1} \end{cases} \tag{6}$$

where

$$\Lambda(K) = \mu + n^\star\theta_1 + (K+1-n^\star)\,\theta_2 + \lambda. \tag{7}$$

When the observed customer is not in service and assuming he is at position $n^\star + 1$, we have:

$$W(K,0,n^\star+1) = \begin{cases} \mathcal{E}(\Lambda(K)) + W(K-1,1) & w.p.\ (\mu + n^\star\theta_1)(\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K-1,0,n^\star+1) & w.p.\ (K-n^\star)\theta_2(\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K+1,0,n^\star+1) & w.p.\ \lambda(\Lambda(K))^{-1}. \end{cases} \tag{8}$$

In case the observed customer is at position N with $N \in \{n^\star + 2, \ldots, K+1\}$, we have:

$$W(K,0,N) = \begin{cases} \mathcal{E}(\Lambda(K)) + W(K-1,0,N-1) & w.p.\ \frac{\mu + n^\star\theta_1 + (N-(n^\star+1))\theta_2}{\Lambda(K)} \\ \mathcal{E}(\Lambda(K)) + W(K-1,0,N) & w.p.\ (K+1-N)\,\theta_2(\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K+1,0,N) & w.p.\ \lambda(\Lambda(K))^{-1}. \end{cases} \tag{9}$$

In both cases described by (8) and (9), the term $\Lambda(K)$ is used as defined in (7). When K equals $\underline{N}-1$, the remaining sojourn time of the observed customer, already in service, is:

$$W(\underline{N}-1,1) = \begin{cases} \mathcal{E}(\Lambda(\underline{N}-1)) & w.p.\ \mu\,(n^\star\Lambda(\underline{N}-1))^{-1} \\ \mathcal{E}(\Lambda(\underline{N}-1)) + W(\underline{N}-2,1) & w.p.\ \frac{(n^\star-1)/n^\star\mu + (n^\star-1)\theta_1) + (\underline{N}-n^\star)\theta_2}{\Lambda(\underline{N}-1)} \end{cases} \tag{10}$$

where

$$\Lambda(\underline{N}-1) = \mu + n^\star\theta_1 + (\underline{N}-n^\star)\,\theta_2. \tag{11}$$

When the observed customer is not in service, then assuming he is at position $n^\star+1$, we have:

$$W(\underline{N}-1,0,n^\star+1) = \begin{cases} \mathcal{E}(\Lambda(\underline{N}-1)) + W(\underline{N}-2,1) \\ \qquad w.p.\ (\mu + n^\star\theta_1)(\Lambda(\underline{N}-1))^{-1} \\ \mathcal{E}(\Lambda(\underline{N}-1)) + W(\underline{N}-2,0,n^\star+1) \\ \qquad w.p.\ (\underline{N}-(n^\star+1))\,\theta_2(\Lambda(\underline{N}-1))^{-1}. \end{cases} \tag{12}$$

In case the observed customer is at position N with $N \in \{n^\star+2, \ldots, \underline{N}\}$, we have:

$$W(\underline{N}-1,0,N) = \begin{cases} \mathcal{E}(\Lambda(\underline{N}-1)) + W(\underline{N}-2,0,N-1) \\ \qquad w.p.\ (\mu + n^\star\theta_1 + (N-(n^\star+1))\,\theta_2)\,(\Lambda(\underline{N}-1))^{-1} \\ \mathcal{E}(\Lambda(\underline{N}-1)) + W(\underline{N}-2,0,N) \\ \qquad w.p.\ (\underline{N}-N)\,\theta_2(\Lambda(\underline{N}-1))^{-1}. \end{cases} \tag{13}$$

where the definition of $\Lambda(K)$ found in Equation (11) is used in (12) and (13).

Proof. We only establish a formal proof of Equation (8), since the proof for all other equations can be obtained following a similar argument.

We compute the remaining sojourn time of an observed customer, given that the observed customer is in a system counting K other clients with $K \geq n^\star$. Moreover, we assume that our tagged customer's service has not yet started. However, as soon as one of the $n^\star$ clients already in service leaves the system, our observed customer will begin with his download. We, thus, compute the remaining sojourn time of our observed customer given that he is at position $n^\star+1$. In this case, because of the memoryless property of the exponential distribution, the next event (arrival or departure) takes place after an exponentially distributed time with parameter $\Lambda(K)$. We have, as stated in Equation (7):

$$\Lambda(K) = \mu + n^\star\theta_1 + (K+1-n^\star)\,\theta_2 + \lambda. \tag{14}$$

Indeed, we may observe one of the $n^\star$ clients in service, either finishing their download (at a rate $\mu/n^\star$), or becoming impatient (at a rate θ_1). The remaining customers including our observed customer, thus, those $K+1-n^\star$ customers that have not yet started downloading their file, may become impatient at a rate θ_2. Of course, we may still observe the arrival of a new customer at a rate λ.

If the departure of a served customer occurs, then our observed customer will become served. This happens with the probability $(\mu + n^\star\theta_1)\left(\Lambda(K)\right)^{-1}$ and corresponds to the first case in (8). The second case corresponds to where a waiting customer becomes impatient, leaving our observed customer still waiting for service, but the system counts one customer less. This happens with probability $(K-n^\star)\,\theta_2\left(\Lambda(K)\right)^{-1}$. The last case corresponds to the arrival of a new user. □

Let us remark the following point. Since we are interested in computing the sojourn time of a customer, defined as the total time needed to download his desired file, we only consider successfully completed downloads and the event that our observed customer leaves the system due to impatience is not taken into account in any of the equations in Theorem 1.

Let W be the remaining sojourn time of a typical customer. We define the following conditional probabilities. For $K \in \{0, \ldots, n^\star - 1\}$

$$\begin{aligned} R(y \,|\, K, 1) &= P\left[W > y \,|\, X = K, S = 1\right] \\ &= P\left[W(K,1) > y\right], \end{aligned} \tag{15}$$

thus, the complementary remaining sojourn time distribution of a customer in service ($S = 1$) in a system counting K users ($X = K$). For $K \in \{n^\star, \ldots, \underline{N} - 1\}$

$$\begin{aligned} R(y \,|\, K, 1) &= P\left[W > y \,|\, X = K, S = 1\right] \\ &= P\left[W(K,1) > y\right] \end{aligned} \tag{16}$$

$$\begin{aligned} R(y \,|\, K, 0, N) &= P\left[W > y \,|\, X = K, S = 0, P = N\right] \\ &= P\left[W(K,0,N) > y\right] \end{aligned} \tag{17}$$

where $N \in \{n^\star + 1, \ldots, K+1\}$ and P is the position of the observed user in the queue of waiting users, since $S = 0$ indicates that our observed customer is not in service, yet.

We now establish the system of differential equations in the next theorem, which is given without proof.

Theorem 2. *The conditional complementary probability distributions $R(y \,|\, K, 1)$ and $R(y \,|\, K, 0, N)$ respect the following differential equations.*

If $0 \leq K < N^\star$:

$$\begin{aligned} \partial_y R\left(y \,|\, K, 1\right) = &- \Lambda(K) R\left(y \,|\, K, 1\right) + K\left(R/C\mu + \theta_1\right) R\left(y \,|\, K-1, 1\right) \\ &+ \lambda R\left(y \,|\, K+1, 1\right). \end{aligned} \tag{18}$$

If $N^\star \leq K < n^\star$:

$$\begin{aligned}\partial_y R\left(y \,|\, K,1\right) = &- \Lambda(K) R\left(y \,|\, K,1\right) \\ &+ \left(\frac{K}{K+1}\mu + K\theta_1\right) R(y \,|\, K-1,1) + \lambda R\left(y \,|\, K+1,1\right).\end{aligned} \tag{19}$$

If $n^\star \leq K < \underline{N} - 1$:

$$\begin{aligned}\partial_y R\left(y \,|\, K,1\right) = &- \Lambda(K) R\left(y \,|\, K,1\right) + \lambda R\left(y \,|\, K+1,1\right) \\ &+ \left(\frac{n^\star - 1}{n^\star}\mu + \left(n^\star - 1\right)\theta_1 + \left(K + 1 - n^\star\right)\theta_2\right) R\left(y \,|\, K-1,1\right).\end{aligned} \tag{20}$$

$$\begin{aligned}\partial_y R\left(y \,|\, K,0,n^\star+1\right) = &- \Lambda(K) R\left(y \,|\, K,0,n^\star+1\right) + \left(\mu + n^\star\theta_1\right) R\left(y \,|\, K-1,1\right) \\ &+ \left(K - n^\star\right)\theta_2 \, R\left(y \,|\, K-1,0,n^\star+1\right) \\ &+ \lambda R\left(y \,|\, K+1,0,n^\star+1\right)\end{aligned} \tag{21}$$

Moreover, if $n^\star + 1 < N \leq K + 1$:

$$\begin{aligned}\partial_y R\left(y \,|\, K,0,N\right) = &- \Lambda(K) R\left(y \,|\, K,0,N\right) + \lambda R\left(y \,|\, K+1,0,N\right) \\ &+ \left(\mu + n^\star\theta_1 + \left(N - 1 - n^\star\right)\theta_2\right) R\left(y \,|\, K-1,0,N-1\right) \\ &+ \left(K + 1 - N\right)\theta_2 R\left(y \,|\, K-1,0,N\right).\end{aligned} \tag{22}$$

Finally, we have

$$\begin{aligned}\partial_y R\left(y \,|\, \underline{N} - 1,1\right) = &- \Lambda\left(\underline{N} - 1\right) R\left(y \,|\, \underline{N} - 1,1\right) \\ &+ \left(\frac{n^\star - 1}{n^\star}\mu + \left(n^\star - 1\right)\theta_1 + \left(\underline{N} - n^\star\right)\theta_2\right) R\left(y \,|\, \underline{N} - 2,1\right)\end{aligned} \tag{23}$$

$$\begin{aligned}\partial_y R\left(y \,|\, \underline{N} - 1,0,n^\star+1\right) = &- \Lambda\left(\underline{N} - 1\right) R\left(y \,|\, \underline{N} - 1,0,n^\star+1\right) \\ &+ \left(\underline{N} - 1 - n^\star\right)\theta_2 R\left(y \,|\, \underline{N} - 2,0,n^\star+1\right) \\ &+ \left(\mu + n^\star\theta_1\right) R\left(y \,|\, \underline{N} - 2,1\right)\end{aligned} \tag{24}$$

and

$$\begin{aligned}\partial_y R\left(y \,|\, \underline{N} - 1,0,N\right) = &- \Lambda\left(\underline{N} - 1\right) R\left(y \,|\, \underline{N} - 1,0,N\right) \\ &+ \left(\underline{N} - N\right)\theta_2 R\left(y \,|\, \underline{N} - 2,0,N\right) \\ &+ \left(\mu + n^\star\theta_1 + \left(N - 1 - n^\star\right)\theta_2\right) R\left(y \,|\, \underline{N} - 2,0,N-1\right)\end{aligned} \tag{25}$$

when $n^\star + 1 < N \leq \underline{N}$.

For $n^\star \le i < \underline{N}$, we define the vectors $\mathbf{R}(y,i)$ of size $(i-n^\star+2)\times 1$ as follows:

$$\mathbf{R}(y,i) = \begin{pmatrix} R(y\,|\,i,0,n^\star+1) \\ R(y\,|\,i,0,n^\star+2) \\ \cdots \\ R(y\,|\,i,0,i+1) \\ R(y\,|\,i,1) \end{pmatrix}. \tag{26}$$

Accordingly, we define the vector $\mathbf{R}(y)$ as composed as follows:

$$\mathbf{R}(y) = \begin{pmatrix} R(y\,|\,0,1) \\ R(y\,|\,1,1) \\ R(y\,|\,n^\star-1,1) \\ \mathbf{R}(y,n^\star) \\ \mathbf{R}(y,n^\star+1) \\ \cdots \\ \mathbf{R}(y,\underline{N}-1) \end{pmatrix} \tag{27}$$

which has the dimension $\frac{1}{2}\left((n^\star)^2 - (2\,\underline{N}+1)\,n^\star + 3\,\underline{N} + \underline{N}^2\right)$.

Theorem 2 can now be written as

$$\partial_y \mathbf{R}(y) = A\mathbf{R}(y) \tag{28}$$

$$\mathbf{R}(0) = \mathbf{1}, \tag{29}$$

where $\mathbf{1}$ is a vector of appropriate size consisting of 1. The matrix A is defined as composed of the following blocks:

$$A = \begin{pmatrix} C_0 & A_0 & 0 & \ldots & 0 \\ B_1 & C_1 & A_1 & \ldots & 0 \\ 0 & B_2 & C_2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & C_{\underline{N}-1} \end{pmatrix}, \tag{30}$$

where for $0 \le i < n^\star$

$$C_i = -\Lambda(i) \tag{31}$$

$$A_i = \lambda \tag{32}$$

$$B_i = \begin{cases} i\,(R/C\mu + \theta_1) & \text{if } 0 < i < N^\star \\ i/\,(i+1)\,\mu + i\,\theta_1 & \text{if } N^\star \le i < n^\star \end{cases} \tag{33}$$

When i belongs to $\{n^\star, \ldots, \underline{N}-1\}$, we have $C_i = -\Lambda(i)I$ with I the identity matrix of appropriate size $(i-n^\star+2)\times(i-n^\star+2)$. We have

$$B_{n^\star} = \begin{pmatrix} \mu + n^\star\theta_1 \\ \frac{n^\star-1}{n^\star}\mu + (n^\star-1)\,\theta_1 + \theta_2 \end{pmatrix} \tag{34}$$

$$B_{n^\star+1} = \begin{pmatrix} \theta_2 & \mu + n^\star\theta_1 \\ \mu + n^\star\theta_1 + \theta_2 & 0 \\ 0 & \frac{n^\star-1}{n^\star}\mu + (n^\star-1)\,\theta_1 + 2\,\theta_2 \end{pmatrix} \tag{35}$$

For $2 \leq i \leq \underline{N} - 1 - n^\star$ the matrix $B_{n^\star+i}$ is of size $(i+2) \times (i+1)$ and such that:

$$\begin{array}{ll} B_{n^\star+i}(j,j) & = (i-(j-1))\,\theta_2 \qquad \text{for } 1 \leq j \leq i \\ B_{n^\star+i}(j+1,j) & = \mu + n^\star\,\theta_1 + j\,\theta_2 \qquad \text{for } 1 \leq j \leq i \\ B_{n^\star+i}(1,i+1) & = \mu + n^\star\,\theta_1 \\ B_{n^\star+i}(i+2,i+1) & = \frac{n^\star-1}{n^\star}\mu + (n^\star-1)\,\theta_1 + (i+1)\,\theta_2, \end{array} \tag{36}$$

and other elements being equal to 0. For $0 \leq i \leq (\underline{N}-1) - n^\star$, the matrix A_i is a matrix of size $(i+2) \times (i+3)$, whose elements are

$$\begin{array}{rl} A_{n^\star+i}(j,j) = \lambda & \qquad \text{for } 1 \leq j \leq i+1 \\ A_{n^\star+i}(i+2,i+3) = \lambda & \end{array} \tag{37}$$

and others elements being equal to 0.

The system of differential equations (28) with initial conditions (29) can be easily solved and we get:

$$R(y) = \exp(Ay). \tag{38}$$

4.2 Stationary Distribution of the Number of Users in the System

Our objective is to compute the stationary distribution of the time a customer needs in order to completely download a file. When a customer enters the system, he may find the system already occupied with $K < \underline{N}$ customers. This section aims at computing the stationary distribution of the number of customers in the system at the arrival instant of the observed customer. Due to the PASTA property, this stationary distribution is simply equal to the stationary distribution of the number of customers in the system at any time.

Let $\{X(t);\, t \in \mathbb{R}^+\}$ be the Markov process counting the number of customers in the system. As previously mentioned, the corresponding stationary random variable is given by X. We denote by the vector π the corresponding stationary distribution, that is

$$\pi(K) = P\left[X = K\right], \tag{39}$$

with $K \in \{0, \ldots, \underline{N}\}$. We define Q as the generator associated to the process $\{X(t);\, t \in \mathbb{R}^+\}$. The elements of Q are:

$$\begin{array}{ll} Q(i,i+1) = \lambda & \text{for } 0 \leq i \leq \underline{N} - 1 \\ Q(i,i-1) = i\,R/C\mu + i\,\theta_1 & \text{for } 1 \leq i \leq N^\star \\ \qquad\quad = \mu + i\,\theta_1 & \text{for } N^\star + 1 \leq i \leq n^\star \\ \qquad\quad = \mu + n^\star\theta_1 + (i-n^\star)\,\theta_2 & \text{for } n^\star + 1 \leq i \leq \underline{N} \end{array} \tag{40}$$

The diagonal elements of Q are such that $Q\mathbf{1} = \mathbf{0}$, with $\mathbf{0}$ being a vector consisting of entries 0 and of appropriate size. Other elements of Q are zeros. Clearly, the process $\{X(t);\, t \in \mathbb{R}^+\}$ is a birth-and-death process. Accordingly, let ρ_i be

$$\begin{array}{ll} \rho_i = \lambda/\left((i+1)\,R/C\mu + (i+1)\,\theta_1\right) & \text{for } 0 \leq i < N^\star \\ \quad = \lambda/\left(\mu + (i+1)\,\theta_1\right) & \text{for } N^\star \leq i < n^\star \\ \quad = \lambda/\left(\mu + n^\star\,\theta_1 + (i+1-n^\star)\,\theta_2\right) & \text{for } n^\star \leq i < \underline{N} - 1. \end{array} \tag{41}$$

We obtain the stationary probabilities for $K \in \{1, \ldots, \underline{N}\}$

$$\pi(K) = \pi(0) \prod_{i=0}^{K-1} \rho_i \qquad \text{with} \qquad \pi(0) = \left(1 + \sum_{K=1}^{\underline{N}} \prod_{i=0}^{K-1} \rho_i\right)^{-1}. \tag{42}$$

4.3 Stationary Total Sojourn Time

When a customer enters the system, he may either be immediately served or is placed last in the queue depending on the current number of customers present in the system. The complementary total sojourn time distribution of a customer respects the following equation:

$$P[W > y] = \sum_{K=0}^{n^\star - 1} \pi(K) R(y \mid K, 1) + \sum_{K=n^\star}^{\underline{N}-1} \pi(K) R(y \mid K, 0, K+1). \tag{43}$$

For $i \in \{0, \ldots, \underline{N} - 1 - n^\star\}$, we define the vectors $\underline{\pi}(i)$ as:

$$\underline{\pi}(i) = \mathbf{e}(i+2, i+1)\, \pi(n^\star + i), \tag{44}$$

where $\mathbf{e}(i, j)$ is a row vector of size i full of 0's except element j which is equal to 1. Accordingly, we define $\mathbf{\Pi}$ as composed of

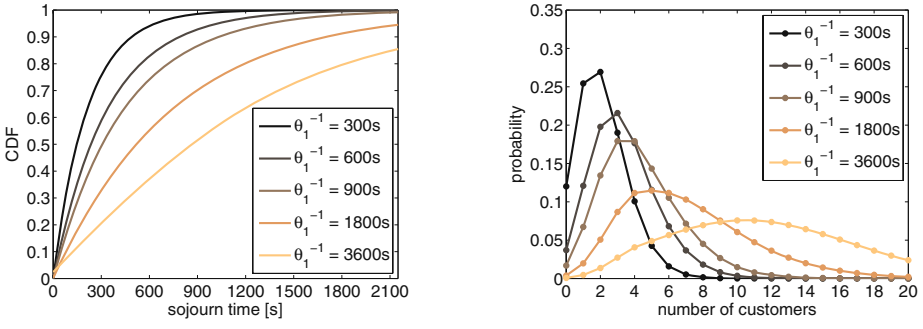
$$\mathbf{\Pi} = \left(\pi(0)\ \pi(1) \ldots \pi(n^\star - 1)\ \underline{\pi}(n^\star) \ldots \underline{\pi}(\underline{N} - 1)\right). \tag{45}$$

Then, using Equation (38), we obtain the complementary distribution of the user's sojourn time as

$$P[W > y] = \mathbf{\Pi} \exp(Ay)\mathbf{1}. \tag{46}$$

5 Numerical Evaluation

In this section we will provide some numerical results and briefly discuss the influence of some of the important parameters on the system behavior. Let us consider an OTR mirror server site with a total capacity of $C = 20$ Mbps. Note that while the analytical model used rates normalized by the server's capacity, we give absolute values here, as we are interested in the actual downloading durations and they are more meaningful for verifying the plausibility of the results. The average file size μ^{-1} is 359.87 MByte and the maximum download rate R of all users is 4 Mbps as specified by the ITU G.992.2 standard for ADSL Lite. Since we assume this is a mirror site operated by a private person, it is reasonable to assume that the maintainer only limits access to a relatively small number of concurrently served downloads, e.g. $n^\star = 10$ and $\underline{N} = 20$, due to the following consideration. Many existing mirror servers indicate the current queue length and the expected waiting time until a download slot will become free. In the above scenario, a user at the first position of the waiting queue would have to wait 20 min and at the last (i.e. 10th) position would require



(a) CDF of sojourn time (b) Distribution of number of users

Fig. 2. Influence of the impatience threshold θ_1^{-1}

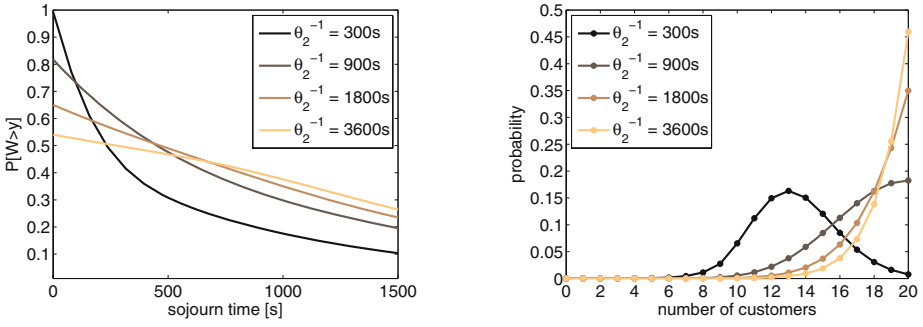
a waiting time of 200 min. A requesting user who would be facing to wait so long before service would obviously select a different mirror site beforehand. As further parameters, if not indicated otherwise, we assume a request arrival rate of $\lambda = 10^{-2}$ requests/s as well as the impatience thresholds of $\theta_1^{-1} = 2\,\text{h}$ and $\theta_2^{-1} = 1\,\text{h}$ for downloading and waiting users, respectively. Furthermore, we verified the accuracy of our numerical implementation by simulations.

5.1 Influence of Impatience during Downloading

Let us first consider the impatience threshold θ_1 and its influence on the sojourn time of a successful customer as shown in Fig. 2. On the left, in Fig. 2(a), the CDF of the sojourn time as computed from Equation (46) is shown with different impatience thresholds for downloading users θ_1^{-1}. Darker lines represent smaller values of θ_1^{-1}. Obviously, the sojourn time increases when the patience threshold increases. This can be explained by the fact that when θ_1^{-1} is small, only small files are actually downloaded and users downloading larger files will have a large tendency to abort their attempts, so on average the sojourn time in the system will be small. This is also suggested when we look at the steady state distribution of the number of users in the system, see Fig. 2(b). For all considered values, in particular when θ_1^{-1} is small, the probability of finding an empty system upon arrival, i.e. $\pi(0)$, is greater than zero. A larger θ_1^{-1} shifts the weight of the distribution toward a larger number of customers. In the case of $\theta_1^{-1} = 3600\,\text{s}$ we can see that $\pi(\underline{N}) > 0$, leading to blocking of potentially new arrivals.

5.2 Influence of Impatience during Waiting

We now investigate the influence of the impatience threshold of waiting users θ_2 on the sojourn time. The numerical results are shown in Fig. 3. Note that in contrast to Fig. 2(a) the left figure (Fig. 3(a)) now shows the complementary CDF of the sojourn time. It is remarkable that the probabilities $P[W > 0]$ may



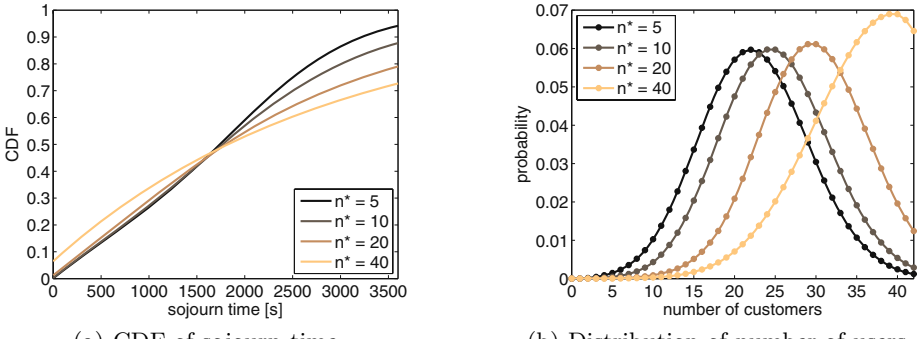
(a) CCDF of sojourn time (b) Distribution of number of users

Fig. 3. Influence of the impatience threshold θ_2

be less than 1, if the waiting impatience time θ_2^{-1} is large. Again, we can interpret this result better by looking at the corresponding distributions of customers as shown in Fig. 3(b). When $\theta_2^{-1} = 300\,\mathrm{s}$ the system is already serving $n^\star$ customers and there are on average 3 waiting users in the queue, indicated by the highest probabilities $\pi(i)$ for $i \in \{12, 13, 14\}$. However, since the impatience time is small, queued users quickly leave the system again. On the other hand for larger impatience values of θ_2^{-1}, especially when $\theta_2^{-1} > 900\,\mathrm{s}$, users wait longer in the queue and the probability is very large to find the system fully occupied. Note that $P[W = 0] = P[X = \underline{N}]$, i.e., the probability that the sojourn time is zero is equal to the probability that there are $\underline{N}$ customers in the system.

5.3 Influence of the Number of Available Download Slots

Finally, we investigate how the number of available download slots affects the sojourn time. We now look at a slightly different scenario, with $\lambda = 1/80\,\mathrm{s}^{-1}$,



(a) CDF of sojourn time (b) Distribution of number of users

Fig. 4. Influence of the number of download slots $n^\star$

$\underline{N} = 42$ and for values of $n^\star = 5, 10, 20, 40$ to emphasize the effect. Although the curves for the CDF of the sojourn time do not intersect at the same point, cf. Fig. 4(a), the intersection point y' for any two curves lies in a small range between 1600 s and 1700 s. With larger $n^\star$ the download bandwidth decreases and together with a higher patience while downloading than while waiting, the probability for sojourn times $y > y'$ beyond this intersection point increase with $n^\star$. The CDF of the sojourn time displays a similar behavior with blocking for large $n^\star$ as already observed for θ_2. Note that a value of $n^\star = 40$ and $\underline{N} = 42$ means that the maximum waiting queue length is 2.

6 Conclusion

In this paper we analytically derived the sojourn time of an arbitrary user, who successfully completes downloading a file from an IPTV server. Since the data volume of the requested file is very large, it is necessary to also take the user behavior in terms of his impatience into account, as this is a phenomenon frequently observed in actual systems. We investigated the effects of the impatience thresholds of waiting and downloading users, as well as the number of available downloading slots. Further numerical evaluations that we performed, e.g. the impact of the arrival rate λ, were also studied, but left out of this paper due to lack of space. In the future, we wish to perform a more detailed evaluation of the influencing parameters and extend the model to consider a more accurate file size distribution as well as heterogeneous users with different access bandwidths.

References

1. USEN Corporation: Broadband business and video & contents business monthly results (March 2008), `http://www.usen.com/ir/english/`
2. Hoßfeld, T., Leibnitz, K.: A qualitative measurement survey of popular internet-based IPTV systems (submitted for publication) (2008)
3. OnlineTVRecorder, `http://www.onlinetvrecorder.com/`
4. Hoßfeld, T., Leibnitz, K., Remiche, M.A.: Modeling of an online TV recording service. ACM SIGMETRICS Performance Evaluation Review 35(2), 15–17 (2007)
5. Brandt, A., Brandt, M.: On the M(n)/M(n)/s queue with impatient calls. Perform. Eval. 35(1-2) (1999)
6. Gromoll, H.C., Robert, P., Zwart, B., Bakker, R.: The impact of reneging in processor sharing queues. In: Proc. of SIGMETRICS 2006/Performance 2006, pp. 87–96. ACM Press, New York (2006)
7. Sericola, B., Guillemin, F., Boyer, J.: Sojourn times in the M/PH/1 processor sharing queue. Queueing Systems 50(1), 109–130 (2005)
8. Masuyama, H., Takine, T.: Sojourn time distribution in a MAP/M/1 processor-sharing queue. Operations Research Letters 31(5), 406–412 (2003)

An Empirical Case-Study of a Central-Server-Model on System Performance

Stefan Schreieck[1], Reinhard German[2], and Kai-Steffen Hielscher[2]

[1] ZEDV (Computing Center),
University of Applied Sciences Kempten, Bahnhofstraße 61, 87435 Kempten
stefan.schreieck@fh-kempten.de
[2] Informatik 7 (Computer Networks and Communication Systems),
University of Erlangen-Nuremberg, Martensstraße 3, 91058 Erlangen
{german,ksjh}@cs.fau.de

Abstract. This paper investigates from a practical point of view the applicability and the limits of a simple Central-Server-Model for performance modeling. To model the prolific webserver at the University of Applied Sciences in Kempten the study uses a Central-Server-Queuing-Theory-Model consisting of a single CPU server/queue and a single Disk server/queue. First the webserver is monitored and the model is calibrated using real data. Then performance is predicted using the model and the outcomes are compared to reality. In addition the model is validated analyzing the calculated and the measured response times with the aid of replaying a sequence of requests. A key point the paper deals with, is the question which and how much information gets lost using such a simple model.

1 Introduction

Building models from real world computer systems is never an easy task. On the one hand you must have the knowledge of fundamental modeling techniques and on the other hand you must be capable to use them in a promising way. One essential problem is the level of detail you should consider in your models. If the architecture of your model is too simple, the results you will achieve are not very realistic and mostly the model is good for nothing. But if you take every single element of the real world system into consideration, your model gets extremely complex and you have to put a high effort into solving it. This was and is a key problem through all times in modeling. As described in [1] as a conclusion you must trade off between the complexity of the model and the results you can get, i.e. you need from it.

Most modeling studies start with a straightforward, so called product form analytical model of a system. Attractive values from such models can be calculated relatively effortless. There exist algebraic methods which allow deriving basic results like utilizations, throughput, waiting and response times. To make analytical modeling more reliable and flexible several extensions have been developed for various systems, e.g. different arrival processes and queuing algorithms

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 173–188, 2008.

or multiclass techniques [2]. If a model could not be solved with analytical methods, according to circumstances it could be an alternative to map the system on a Markov Chain, building up state spaces [3]. Unfortunately, with increasing facets in the model, the number of possible states grows extremely fast and the upper limit for Markov models comes with a so called state space explosion. For highly detailed and therefore complicated or in any other way special models the only back door is a simulation study. The most applied kind of simulation is the discrete event simulation. Constructing such simulation models is a tedious task, executing, i.e. solving them is even more time-consuming. In addition to the long execution times of various simulation runs, another drawback is that they don't provide an exact solution. The outcome of a simulation model is one possible result for a particular scenario.

As a consequence it seems worth, in contrast to papers like [1], to change the angle of view and in place of varying a model, to have a closer look on the possibilities and limitations of a very simple model from a practical point of view. Perhaps it is possible to apply the economic Pareto-Principal [4] on computer systems models, too, i.e. with twenty percent modeling effort eighty percent modeling success could be achieved. Thus we do a case study with the webserver of the University of Applied Sciences Kempten.

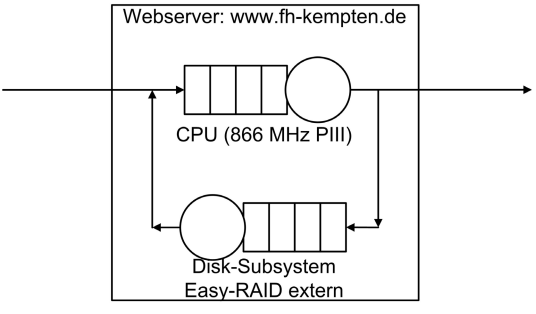


Fig. 1. Open Network Queuing Model of the webserver

According to this chaste open queuing network in Fig. 1 we are mainly interested in fundamental question like utilizations of CPU and disk subsystem under different load scenarios, throughputs and the corresponding response times. Therefore we monitor the server for one week and collect lots of data. Analyzing the gathered information, we can parameterize our model and use it for prediction purposes. To be able to validate the model and judge about it, we conduct some experiments with a load generator. Varying the load applied on the server should lead to deeper insights concerning the quality of the Central-Server-Model in our case study and comparable modeling situations.

Solving the model, i.e. calculating measures like waiting queue lengths or response times is done by using the popular Mean Value Analysis (MVA) [2]. The MVA is a time averaged method whose calculations are based on the average

state, quasi steady-state of a system. Therefore it is often used in association with closed product-form networks, but there also exist variations for open queuing networks and several extensions, e.g. Priority-Scheduling. There is no need to build up a complete state space as usual for Markov-Models, calculations are done by an algorithm consisting of fundamental laws like *Little's Law*. For closed product-form models results are achieved from an recursive algorithm based on the *Arrival Theorem* which states that the performance measures of a system with N customers can be conducted from the values of the system for $N-1$ customers. For open product-form models an appropriate adaption exists [2]. In this context a central measure is the Service-Demand, i.e. the mean time period a request needs to be serviced by a server. In practice you need the system throughput (X_0) and utilization values from each server (U_i) to calculate the demands (D_i) as follows:

$$D_i = \frac{U_i}{X_0} \tag{1}$$

2 Monitoring the Webserver

Before we start to analyze any data or investigate our model, we want to provide some information concerning the webserver and the monitoring process first. These are the basic parameters of the webserver:

- 1 Processor PIII 866 MHz
- 512 MB RAM
- External Easy-RAID-System, connected through SCSI (RAID-Level 5, Capacity: 205 GB), divided into 3 partitions:
  - Swap, sda1
  - Root (/), sda2
  - Home (/home), sda3, (contains et al. webpages)
- 100 MBit/s network connection
- Operating system: SuSE-Linux 9.0
- Webserver: Apache 1.3 (incl. modules for Perl and PHP)

Data logging takes place in the period from Fri. 01.12.06, 00:00 h inclusively until Fri. 08.12.06, 00:00 h exclusively. Statistics about the attended requests, which provide a basis for the arrival rate, i.e. the throughput at the server and therefore at our model, comes from the Apache Logfile [5]. For an overview of the CPU and Disk Utilization the *iostat* tool is used [6]. Every minute an averaged value is written to a logfile on the *root* partition. If nothing contrary is mentioned all figures and diagrams are based on minute-by-minute averaged values.

3 Examination of the Webserver-Requests

During our monitoring period 368701 requests were applied to the Apache. This means a daily average of 52672 queries, respectively one request every 0.6 seconds. All in all 5296 different IP Addresses can be found in the logfiles, leading to an estimation of 69.9 queries from each single client.

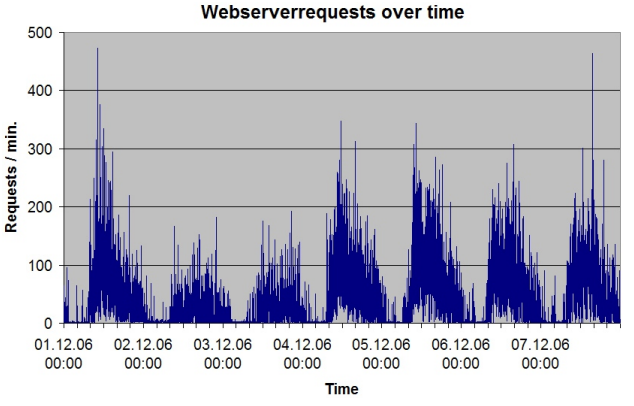


Fig. 2. Webserverrequests over time

On weekdays you become aware of a steep rising edge every morning, which culminates in the maximum utilization of the day in the period from about 10:00 h to 12:00 h. Sometimes you notice interim lower rates between 12:00 h and 13:00 h. At 15:00 h the number of requests decreases comparatively flatly until 01:00 h. From then on only few queries are recognized and from 03:30 h to 06:30 h the server experiences the period with the lowest request rate.

The increasing and decreasing phases are shaped fairly identically on the weekend. In opposite to the working days there are less requests and the duration of the period with higher request rates is longer and the maximum can be found at ca. 15:00 h.

As the MVA-Algorithm uses assumptions of averaged values, i.e. it is intended for quite constant load, facing our intention to solve the Central-Server-Model with it, these heavily varying arrival rates are suboptimal. Having a closer look on the queries, you can find a bursty structure with alternating periods of many and few requests caused by the behavior of the users. They usually surf to a special site and therefore request the main page with all embedded elements. After this a short break for looking and reading comes up and then the cycle starts again from the beginning.

Otherwise, unless you are modeling each singular request, a model will always use some aggregates and mean values according to its level of detail. In the end you must decide by means of your modeling goals and the heterogeneity of the load which level of averaged values is acceptable. Because it is not always an easy task to find adequate demands for different classes, we assume that only one averaged class should be used in our model. For this we want to have a look on the utilization of the server now.

4 Examination of the Utilization

Investigating the utilization of the webserver means primarily studying the consumption of the resource CPU. The mean load is amounting to 2.54 %. The distribution over time is shown in Fig. 3.

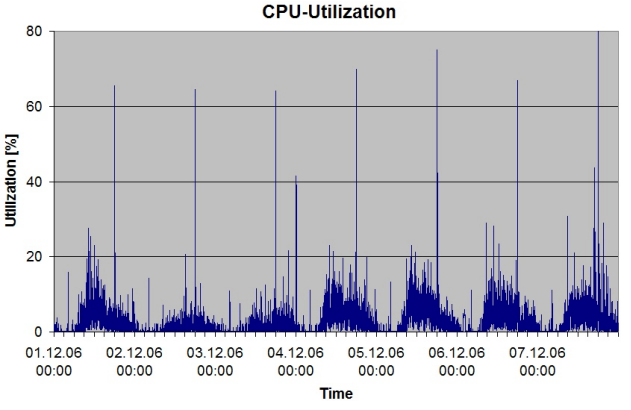


Fig. 3. CPU-Utilization over time

As we expected, the progression of the utilization curve follows the corresponding line of the requests. Obviously these two measures are highly correlated. Furthermore the amplitude of the utilization is directly influenced by the amount of queries processed by the webserver and reaches during busy periods 10 % to 20 %. Nevertheless there seem to be some special effects.

The maximum of 65 % utilization comes up every evening at 18:00 h. It results from the backup process which creates a tar-archive of the complete homepage and the database for writing out on a tape. Likewise you find a peak every night at 04:15 h which is caused by the *cron.daily* cronjob of the operating system. At this time the system logs (warn and message) are rotated and compressed. In addition some administration tasks on the packet and manpage databases are done. The third thing to be curious about is the peak on Monday, 04.12.06 at 00:00 h. Its source is the index updating process of the homepage's internal search function realized by *Perlfect Search* [7].

Apart from the CPU the disk subsystem is worth to be analyzed, too. On the one hand all information delivered to the users is stored on the hard disks and on the other hand the log writes from the Apache are stored on the disks. Furthermore operations on the disks are considered relatively time consuming for a computer system visible in average access times of about 10 ms. Figure 4 presents the utilization of the different partitions.

According to the explanations in the CPU section you see the maximum values every evening at 18:00 h from the backup process. During this task the demand on the *root* partition is very poor, but on the *home* partition arise peaks with about 8 %. Updating the search index causes also a notable value. *Root* is more challenged by the *cron.daily* cronjob which leads to ca. 2 % utilization. On the average the utilization of the whole disk subsystem is amounting to 0.05 %.

It is quite evident that there are many special tasks demanding resources at different levels and varying times on the server. These will cause another gap to our model. If you build up a model you are usually interested in the capability of the offered service, which means for example the maximum number of concurrent

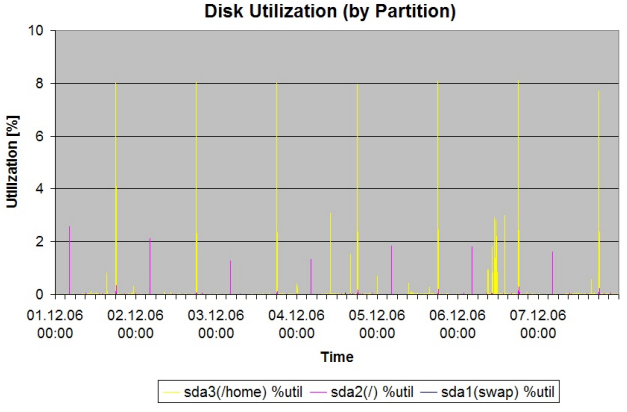


Fig. 4. Disk-Utilization over time

users during normal operation. For a simple model you can't regard such sporadic administrative tasks and in the majority of cases it is not necessary because on the one hand these batch jobs are executed in periods of low utilization and on the other hand the duration of these jobs is mostly very short. In the end it will be your decision what you want to model and if you have to take a special effect into consideration.

If you want to see curves corresponding to the requests applied to the server, because of the small values you have to scale up the y-axis. In Fig. 5 sporadic peaks can be found which belong to activities on the *swap* space. This gives the impression that the rather poor 512 MB RAM of the server seem to be enough for stable operating. All other activities are equally distributed to the *root* and the *home* partition.

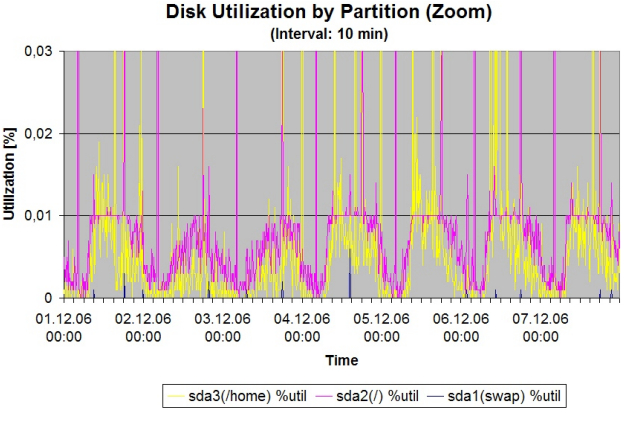


Fig. 5. Disk-Utilization scaled up

5 Parametrizing and Solving the Model

As mentioned above we want to solve our Central-Server-Model by using the MVA-Algorithm. Therefore we need demands for the CPU-Server and the Disk-Server in the model. These are calculated by the quotient of the utilization of the respective device and the system throughput. Because of the heavily alternating arrival rate, determining suitable, i.e. representative averaged values in the monitored data isn't an easy task. To avoid the influence of noise it is a good idea to choose an interval with intensive utilization. Furthermore the additional demand caused by the monitoring with *iostat* carries less weight during a busy period. Due to our plan to use one single average class in the model, it is advisable to pick not a too short period in the monitored data, to cover all essential effects.

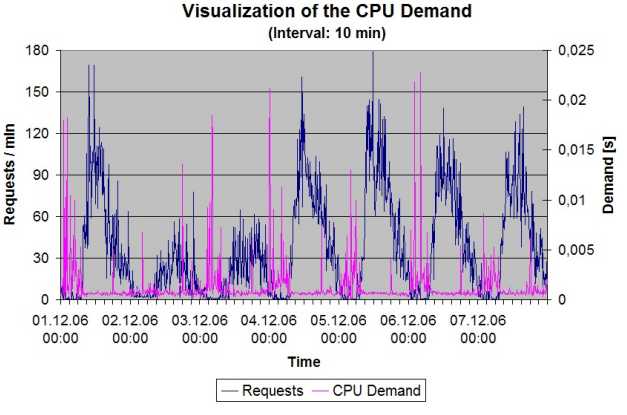


Fig. 6. CPU-Demand over time

Based on the results of the sections 3 and 4 you can see in Fig. 6 how the CPU-Demand evolves over time. Every time special tasks are active, like backup or cron-job, tall peaks occur, which we don't want to model. The same behavior can be found in the disk diagram. Both demand curves show considerable fluctuating during periods with low arrival rates, which should be discarded for the estimation of a demand. Fortunately during busy periods the demands remain on a constant level. This makes hope for useful outcomes from our single class model. After taking everything into consideration we decide to use the acquired data in the interval from Tuesday, 05.12.06 from 10:00 h inclusively to 12:00 h exclusively and we obtain the values from Table 1.

Based on the results of our calibration interval we decide to check the quality of the model against measured values. Therefore we apply our model with the demands calculated above to the minute-by-minute averaged arrival rates recorded by the Apache-Log.

In Fig. 7 we illustrated the difference between the measured utilization and the one calculated by the model. As you can see, one time the model outperforms

Table 1. Performance measures of the webserver

arrival rate (λ)	
system throughput (X_0)	$\frac{14572}{7200} = 2.0239\, s^{-1}$
averaged utilizations (U_i)	
CPU	7.75 %
Disk	0.02 %
averaged demands (D_i)	
CPU	0.0383 s
Disk	$1.14E^{-4}$ s
mean queue length (Q_i)	
CPU	0.08401
Disk	0.0002
mean residence times (R_i)	
CPU	41.5 ms
Disk	0.1 ms
response time	41.6 ms

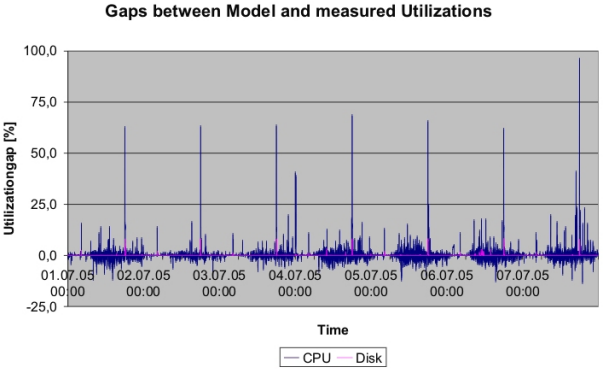


Fig. 7. Difference from computed to real Utilization of CPU and Disk

and one time it underperforms reality. Of course there are big errors during the special tasks mentioned above, like the backup process. All in all the utilizations predicted by the model are too small. For the CPU we calculate a small error of 7.9 % and for the disk subsystem an error of 86.4 %. However the remarkable aberration for the disks is mainly caused by the small values which on the one hand suffer from a limited accuracy of measurement and on the other hand are influenced by our logging activities which account for some third-party utilization.

Unfortunately the processing time for a request measured by the Apache in seconds is too imprecise for a comparison with the model's results. The imple-

mentation of this feature is more useful in conjunction with Apache 2, which allows measuring processing times in microseconds [8]. Therefore we can not validate our model the usual way using this performance measure. However we are able to compute response times according to the measured arrival rates. Our model states that on average it takes 0.032927 s CPU-Time and 0.000114 s Disk-Time to answer an ordinary request.

6 Testing with Httperf

Alternatively to the Apache-Log we want to use the load generator *httperf* to validate our model because it is able to produces some response time statistics [9]. Under various rates we want to replay the sequence of requests logged in the already selected two hour interval in different experiments, each starting by night at 04:30 h to avoid additional requests from real users. One the one hand this gives us the chance to verify diverse modeling assumptions against the real world, like proportionality between the arrival rate and the utilization, and on the other hand we become capable to check our model by comparing the measured results with the theoretical outcomes of our model. Because synthetic load is produced we have the power to control and influence the experiments in a promising manner, e.g. in one test we apply only requests for static sites to the server to see the impact of dynamic generated content on the server utilization.

Figure 8 shows the infrastructure we used for the tests. From two Linux-PCs (Pentium IV, 2.4 GHz, 512 MB RAM, httperf-0.8) in a subnet we drive the subsequent *httperf*-experiments of Table 2 through a Novell router into the backbone of the university network and apply requests on the webserver.

Table 2. With *httperf* conducted experiments

No.	Client	time	number of requests	option
V1	Test-PC 2	Sat. 04:30 h	14572	rate=2
V2	Test-PC 1	Sun. 04:30 h	14572	rate=4
V3	Test-PC 2	Mon. 04:30 h	29144	rate=8
V4	Test-PC 2	Tue. 04:30 h	29144	rate=16
V5	Test-PC 1	Wed. 04:30 h	29144	rate=16 (noatime)
V6	Test-PC 1	Thu. 04:30 h	14572	rate=14
	Test-PC 2	Thu. 04:30 h	14572	rate=14
V7	Test-PC 1	Fri. 04:30 h	28076	rate=8 (static)

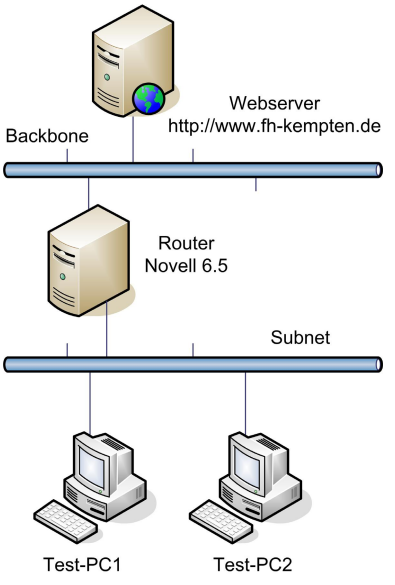


Fig. 8. Testing Infrastructure

7 Test Results

As anticipated, at the beginning of each experiment the rising number of requests per minute causes an increasing CPU-utilization. Likewise at the end of the test the utilization shrinks with the arrival rate. A short start-up period at the beginning leads to a moderate fluctuating utilization. Small aberrations from the with *httperf* projected requests per minute come from the queries of other users or roboters (spiders).

A comparison between the average CPU-Utilizations measured in the several tests is pretty close to theory. Doubling the arrival rate causes doubling utilization values, with the exception of early full utilization at a rate of 28. Nevertheless it's justified to describe the CPU as a load independent server.

A conspicuous fact is that the utilization from the original requests is a little bigger than the reproduced load. This effect can be explained with the bursty character of http-requests against the absolutely uniform applied queries with *httperf*. To serve a big number of concurrent calls the Apache has higher costs to spawn new child processes and handle them. Having a look on the amount of different PIDs in the Apache-Log backs this assumption.

As noticed in [10] with every single request the access time in the inodes of the corresponding files is updated. Deactivating this feature by mounting the partition with the option *noatime* [11] seems to have no noticeable influence on the CPU (V4, V5), whereas the effort to serve requests for static sites decreases from ca. 26 % to about 2 %. Because of this huge discrepancy we can ascertain that the fraction of script sites requested from the webserver in normal operation

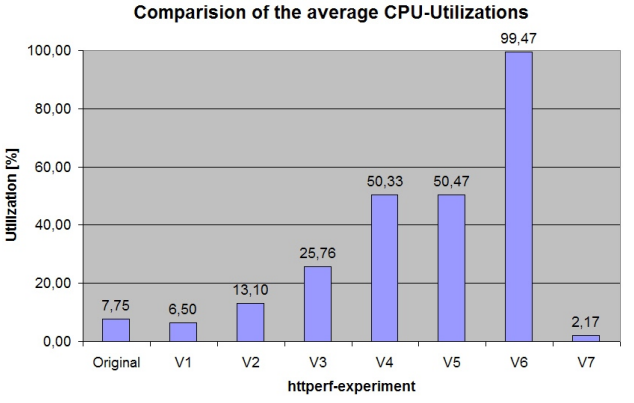


Fig. 9. Comparison of the average CPU-Utilizations

must be nearly constant. Otherwise we would have noticed remarkable fluctuations for the utilizations and demands. To reduce the gap from the model to the real world system it might be a good idea to introduce a second class for dynamic generated content by using the test results to determine adequate demands.

Monitoring the disk it is worth to note that the utilization slightly decreases during the experiment. This can be traced back to the recurrence of some requests in conjunction with buffering effects. With a subsequent replay of the same experiment we can prove that the utilization tends towards zero.

Figure 10 shows that the utilization of the disks rises with an increasing arrival rate, too. Although both the *root* and the *home* partition get busier, a proportional effect can't be stated. The growth is too small. Hence you can say that the disk subsystem is not a perfect load independent server as stated by the Central-Server-Model.

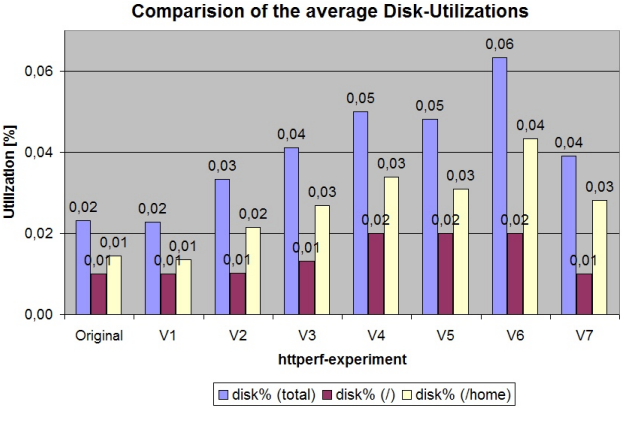


Fig. 10. Comparison of the average Disk-Utilizations

The advantage of turning off the access time updates is insignificant for our RAID system (V4, V5). In contrast to the CPU, a distinction between ordinary static files and dynamic generated content is not necessary. The demands for the disks are nearly equal. Examining the read and write requests on each partition, you get an adequate result. Whilst the *swap* space isn't accessed in any experiment, the augmentation of the requests applied to *root* and *home* is marginal. As observed for the original monitoring interval the tests require only few reads, which leads to the assumption that most content of the websites is cached in memory. Writes account for the bigger part of transactions and can be deduced from logging activities.

Taking the kilobytes per second written into consideration, it becomes obvious that the data volume increases in the manner of the arrival rate. This leads to the conclusion that there must be implemented an optimizing technique. If you examine the read and write requests on the various partitions over time, both, the writes on the *root* and on the *home* partition show patterns suggesting a buffering mechanism. In other words instead of delivering every single write request to the disks, several writes are accumulated and submitted to the RAID as one job.

Because of the low utilization we obtain for the disk subsystem, the assumption of a load independent server isn't a disaster, but at this point our model represents the real world imperfectly. However, due to the low utilization these results might be taken with a pinch of salt with regard to errors in measurements.

8 Httperf-Tests Modeling Results

Due to the with httperf conducted experiments we are able to use the results of the different experiments to reparameterize the Central-Server-Model and to compute such fundamental indicators like the response time. This way it is possible to compare the coherence of the various tests and check them against the measured response times of *httperf*. Furthermore we obtain some reference values for our original model which allows drawing additional conclusions about its quality. Let us first have a look on the individual utilizations, throughputs and demands by means of Table 3.

Aspiring computed response times we need appropriate values for calculating the demands of the CPU-Server and the Disk-Server. To achieve comparable results we do two things. First we use only the first cycle of the requests from experiments which apply each query twice. Second we drop about 25 % of the measured data from the beginning of each test to eliminate the start-up period, because the original monitoring doesn't show any start-up period of course. The complete throughput of the systems is made up of the requests send by *httperf* in addition with the ones from spiders and other users.

Having a look at Fig. 11 you can see, apart from the bigger original demand, which we have already drawn back on the burstiness of the requests, as expected for a load independent server, we achieve a nearly perfect result: all other demands reach the same level with no regard to the applied arrival rate. The low

Table 3. Performance measures from the various tests

No.	U_{CPU} [%]	U_{Disk} [%]	$X_0[\frac{\text{Req.}}{\text{s}}]$	D_{CPU} [s]	D_{Disk} [s]
Original	7.75	0.02	2.0239	0.0383	$1.14E^{-4}$
V1	6.50	0.02	2.0136	0.0323	$1.13E^{-4}$
V2	13.10	0.03	4.0250	0.0325	$8.26E^{-5}$
V3	25.76	0.04	8.0324	0.0321	$5.13E^{-5}$
V4	50.33	0.05	16.0159	0.0314	$3.12E^{-5}$
V5	50.47	0.05	16.0977	0.0315	$3.01E^{-5}$
V6	99.47	0.06	27.7262	0.0355	$2.26E^{-5}$
V7	2.17	0.04	8.0043	0.0027	$4.88E^{-5}$

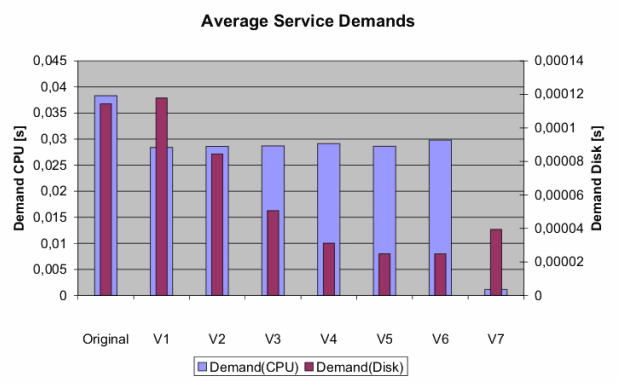


Fig. 11. Comparison of the computed demands, Original and different tests

demand from test V7 isn't contradictory to the rest because it represents only queries for static sites. It is only listed to illustrate the difference to the experiments with dynamic content generation. To forecast the influence of changes in the constitution of the processed requests, introducing a second class will be indispensable.

For the RAID system we obtain an antithetic picture. According to the observed utilization values the demands decrease with increasing system throughput. But facing the small values, the impact on the overall outcomes will be insignificant.

Now we are able to compare the response times measured by *httperf* for the various experiments to the calculated response times from the demands of the individual test. With no doubt we can state by means of Fig. 12 that in each experiment the real response time is bigger than the calculated one. In addition to this, the difference between the two values grows with an increasing arrival rate. The contrary result of V6 should be accepted with caution because at full load you can assume that the real webserver drops some requests leading to a shorter response time compared to the calculated value.

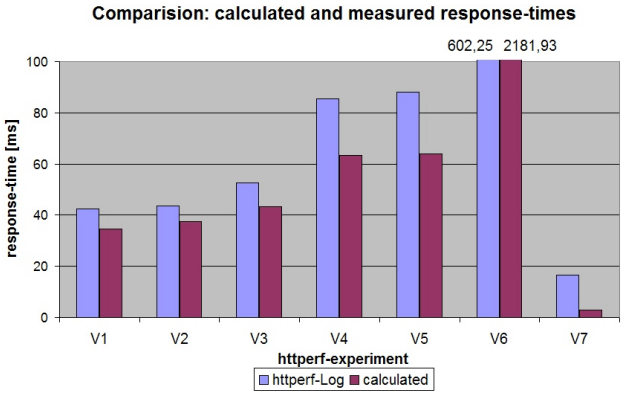


Fig. 12. Comparison between with MVA calculated and with *httperf* measured response-times

One evident reason for the gap between measured and calculated response times is given by the network infrastructure. The measurements of *httperf* include the times required to transport the data packets on the network from the Test-PCs to the server and vice versa. This information is neglected by our model. Covering network delays is a special modeling problem which can't be matched easily due to the huge diversity of network infrastructures between a server and its users [12].

Therefore problems arise from the limited view of our model concerning the choice of servers being taken into consideration. For a service like http, which consists in large parts of network activities, it would make sense to include for example an extra server for the network interface, e.g. the network channels, to cover other elementary components producing delays [12]. Summing up the results of our response time comparison we get a relative error between the calculated and the measured response time from about 14 % to 27 %.

9 Conclusion

Starting with the probably most popular analytical queuing network model for computer systems, the single class Central-Server-Model consisting of one server for the CPU and one for the disks, we investigated the practical applicability of such a simple model by the means of a case study of the prolific webserver at the University of Applied Sciences in Kempten. During the whole modeling process we repeatedly challenged the model to discover gaps between the model and the real world system. This task was supported by carrying out several experiments.

It is a good idea to begin a modeling task of a webserver with a short look at the filetypes stored on the server. This way you will became aware of content with wide varying demands from the early beginning. In our example we observed at the CPU-Utilization a possible error from pure static files to a normal mix with dynamic content by a factor of about 12, leading to a response time lasting three

times longer. Besides files with very different size could also be a reason for using more than one average class for all requests.

Furthermore the model in conjunction with the MVA-Algorithm assumes a steady load on the server. The bursty behavior of http requests is at best modeled indirectly taking the bigger utilization into consideration. Special tasks like a backup process can hardly be covered by the analytic model. This means that the model can only handle long time average situations. Having a close look on very short periods is not possible. In one of our experiments we observed ca. 1.25 % higher CPU-Utilization.

Especially for a server system whose service is mainly based on network connections, the two servers in the model limit its capabilities. Facing the low utilization of the webserver's disks, the network channels suggest themselves to cause an even bigger delay. Due to our tests we estimate the proportion of the network form 14 % to 27 %. Besides the RAID system turned out not to be a load independent server, against the assumption of the model.

To come to conclusion our simple Central-Server-Model has the ability to imitate the webserver from low to medium arrival rates pretty precise. For higher loads the discrepancy to the real world increases. Taking into consideration the hardly assessable network delays between server and user even such noticeable aberrations could be tolerated. In the end you could recommend using a simple Central-Server-Model to nearly every modeler to get a rough feeling for the performance of a server. If it turns out to be sufficient depends on the one hand on the main properties of the server, e.g. the stability of the load, and on the other hand of the modeling goal, i.e. what questions meant to be answered by the model. Otherwise you have to improve your model meeting the challenge of increasing complexity.

References

1. Agrawala, A.K., Larsen, R.L.: Experience with the Central-Server-Model on a lightly loaded system. In: Symposium on the Simulation of Computer Systems IV (1976)
2. Menascé, D.A., Almeida, V.A.F., Dowdy, L.W.: Performance by Design, pp. 337–448. Prentice-Hall, Englewood Cliffs (2004)
3. Bolch, G., Greiner, S., de Meer, H., Trivedi, K.S.: Queuing Networks and Markov Chains, Modelling and Performance Evaluation with Computer Science Applications, pp. 123–184. John Wiley & Sons Inc., Chichester (2006)
4. Reh: Pareto's Principle - The 80-20 Rule. About Inc. (2007), `http://management.about.com/cs/generalmanagement/a/Pareto081202.htm`
5. Apache Foundation: Apache HTTP Server, Custom Log Formats (1999 - 2005), `http://httpd.apache.org/docs/mod/mod_log_config.html`
6. Godard: Report Central Processing Unit (CPU) statistics and input/output statistics for devices and partitions. iostat - Linux man page
7. Perlfect Search - Free site indexer and search engine script, `http://www.perlfect.com/freescripts/search/`
8. Apache Foundation: Apache Module mod_log_config (2006) `http://httpd.apache.org/docs/2.0/mod/mod_log_config.html`

9. Mosberger, Jin: httperf - A Tool for Measuring Web Server Performance, http://www.hpl.hp.com/personal/David_Mosberger/httperf.html
10. Schreieck: Leistungsbezogene Systemanalyse eines Webservers und Betrachtung von Modellierungsaspekten. Interner Bericht FAU CS7 (2005)
11. Mourani: The atime and noatime attribute, Securing and Optimizing Linux: RedHat Edition - A Hands on Guide. OpenDocs (2000), http://www.linux.com/guides/solrhe/SecuringOptimizing-Linux-RH-Edition-v1.3/chap6sec73.shtml
12. Dilley, J., Friedrich, R., Jin, T., Rolia, J.: Measurement Tools and Modeling Techniques for Evaluating Web Server Performance. HP Labs Technical Reports 96-161 (1996)
13. Hielscher, Schreieck, German: Analyse und Modellierung einer produktiven verteilten Webanwendung. Bericht 263, 3. GI/ITG-Workshop, MMBnet 2005 (2005)
14. Menascé, D.A., Peraino, R., Dinh, N., Dinh, Q.: Planning the Capacity of a Web Server: An Experience Report. In: Int. CMG Conference (1999)

A Tandem Queueing Model for Delay Analysis in Disconnected Ad Hoc Networks

Ahmad Al Hanbali, Roland de Haan, Richard J. Boucherie, and Jan-Kees van Ommeren

Stochastic Operations Research, University of Twente, The Netherlands*
{hanbali,haanr,r.j.boucherie,j.c.w.vanommeren}@ewi.utwente.nl

Abstract. Ad hoc network routing protocols may fail to operate in the absence of an end-to-end connection from source to destination. This deficiency can be resolved by so-called delay-tolerant networking which exploits the mobility of the nodes by letting them operate as relays according to the store-carry-and-forward paradigm.

In this work, we analyze the delay performance of a small mobile ad hoc network by considering a tandem queueing system. We present an exact packet-level analysis by applying ideas from the polling literature. Due to the state-space expansion, this analysis cannot efficiently be applied for all model parameter settings. For this reason, an analytical approximation is constructed and its excellent performance has extensively been validated. Numerical results on the mean end-to-end delay show that the switch-over time distribution impacts this metric only through its first two moments. Finally, we study delay optimization under power control.

Keywords: Tandem queueing model, Ad hoc networks, Delay-tolerant networking, Autonomous server, Performance analysis.

1 Introduction

End-to-end connectivity is not a natural property of ad hoc networks. For instance, nodes may vary their transmission power, nodes may move, nodes may enter the sleep mode, or nodes may suffer from hardware failures. As a result, the network structure changes dynamically and this may lead to undesired situations of nodes becoming disconnected from parts of the network. The traditional store-and-forward routing protocols cannot be employed in highly disconnected ad hoc networks. A solution for this problem is to exploit the mobility of nodes present in the network. Such an approach has been proposed in the pioneering paper of Grossglauser and Tse [1] as an alternative to the store-and-forward paradigm and it is now known as the store-*carry*-and-forward paradigm in the context of delay-tolerant networking (DTN) [2]. In DTN, the incurred delay to

* In the Netherlands, the 3 universities of technology have formed the 3TU.Federation. This article is the result of joint research in the 3TU.Centre of Competence NIRICT (Netherlands Institute for Research on ICT).

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 189–205, 2008.

send data between nodes can be very large and unpredictable due to the disconnection problem. Applications of such can be found in, e.g., disaster relief networks, rural networking, environmental monitoring networks, vehicular networks, and interplanetary networks.

An important aspect of DTN is the so-called contact opportunity between nodes. Two nodes are said to be in contact if they are within transmission range of one another and thus packet exchange between them is possible. The duration of a contact impacts the performance under such a networking approach. Another key factor for the performance is the inter-contact time, which is defined as the time duration between two consecutive contacts of node pairs. The inter-contact time mainly depends on the mobility of the nodes.

We will analyze the performance of DTN by taking into account, unlike [3,4,5], that the transmission of packets may fail due to the short contact time and a retransmission is required. Also, we assume that a source node generates a stream of packet arrivals instead of only one packet, as in [5,6,7]. In addition, we are interested in a more practical case of small, finite-size networks, rather than in asymptotic cases (see, e.g., [1,7]). As a primary step towards understanding such networks, we study a model comprising a fixed source and destination node, and a single mobile node operating as a relaying device. Although it is a small model, it contains the main characteristics of a DTN and it is already non-trivial from an analytical perspective. Moreover, we emphasize that the analysis carried out in this paper can readily be extended to model, e.g., multiple relay nodes with single-copy packet approach [8], h-hop ($h \geq 2$) relay routing schemes, or mobile source and destination nodes.

The network model of our interest is reminiscent of a two-queue tandem model with a single alternating server. Such a tandem model has been analyzed under various servicing strategies (see, e.g., [9]). Typically, these strategies are based on the assumption that the server can be controlled. However, in the mobility-driven model of our interest, the server is autonomous and there is no possibility to control its movement. The research efforts on models with time-limited service periods are also closely related to our work. In a two-queue setting, [10] analyzes the model via boundary value techniques. Unfortunately, the analysis along these lines for more than two queues appears intractable. Time-limited service models have also been studied in the context of polling systems (see, e.g., [11,12]). However, also in these models, there exists a notion of server control, since it is assumed that whenever a queue becomes empty the server moves to another queue.

Our main interest is in the end-to-end delay in the network described above. We study this network at the packet level by considering the two-queue tandem model as a particular kind of polling system with customer routing. Specifically, it is a time-limited polling system extended with the feature that the server remains at a queue even if it becomes empty. We perform an exact analysis for this system by extending the techniques developed in [12] and [13]. Due to the state-space expansion, the computation time of the joint queue-length probabilities may grow large for certain model parameters. Therefore, as a complementary

tool, we present an analytical approximation for the case that the service requirements at each queue are exponential. The queue-length process at the second queue is then analyzed in isolation as a workload process with Poisson batch arrivals. The key element is to approximate the batch size distribution as closely as possible. Numerical experiments show the excellent performance of the approximation for a broad range of parameter settings. These experiments further show that the mean sojourn time is insensitive to third and higher moments of the switch-over times. Finally, several guidelines are given for delay optimization by power control.

The rest of the paper is organized as follows. Section 2 gives the model description, discusses the stability, and presents exact results for the sojourn time in the source node and the mobile node. Section 3 proposes and analyzes an approximation for the sojourn time in the mobile queue. In Section 4, we numerically validate the accuracy of the approximation and present additional results which give insight in the delay of the network. Section 5 concludes the paper.

2 Model and Exact Results

2.1 Model

We consider a tandem model consisting of 3 first-in-first-out (FIFO) systems with unlimited queue, Q_i, $i = 1, 2, 3$, in which customers arrive to Q_1 and subsequently require service at Q_2 before reaching their destination at Q_3. The special feature of the model is that Q_2 alternates between positions L_1 and L_2 such that customers at Q_1 are served only when Q_2 is at L_1 and customers at Q_2 are served only when Q_2 is at L_2. In addition, Q_2 incurs a switching time from L_i to L_j $(i \neq j,\ i, j \in \{1, 2\})$ during which the server at neither Q_1 nor Q_2 is available. Q_3 is a sink and will not be included in our analysis.

Given a random variable (rv) X, $X(t)$ will denote its distribution function, $\tilde{X}(s)$ its Laplace-Stieltjes Transform (LST). Customers arrive to Q_1 according to a Poisson process with arrival rate λ. The service requirement S_i at Q_i has general distribution $S_i(\cdot)$ and mean $1/\beta_i$. We assume that the service requirements are independent and identically distributed (iid) rvs.

Movement of Q_2 is autonomous. Q_2 remains at location L_1 (resp. L_2) a (random) time of duration $X_n^{L_1}$ (resp. $X_n^{L_2}$) before it migrates to L_2 (resp. L_1) during its n-th visit. After the n-th visit to L_1, Q_2 incurs a switch-over time $C_n^{1,2}$ from L_1 to L_2, and similarly a switch-over time $C_n^{2,1}$ after the n-th visit to L_2. We assume that $C_n^{1,2}$ $(C_n^{2,1})$ is an iid sequence with general distribution $C^{1,2}(\cdot)$ $\left(C^{2,1}(\cdot)\right)$ and mean $c^{1,2}$ $(c^{2,1})$. Thus, the location of Q_2 is driven by an underlying continuous-time, discrete-state, process $\{L(t) : t \geq 0\}$ with state space $\{-2, -1, 0, 1\}$. More precisely, $L(t) = 1$ $(L(t) = 0)$ when Q_2 is at L_1 (resp. L_2) at time t, and $L(t) = -1$ $(L(t) = -2)$ when Q_2 switches from L_1 to L_2 (L_2 to L_1). Without loss of generality, let $L(0) = 1$. We further assume that $X_n^{L_1}$ $(X_n^{L_2})$ is an iid sequence of common exponential distribution with rate α_1 (α_2). Furthermore, we assume $\{X_n^{L_1}, X_n^{L_2}, C_n^{1,2}, C_n^{2,1}\}$ are iid and mutually independent, and also independent of the inter-arrival times and service requirements.

During the availability of the server at Q_1 and Q_2, the server alternates between service and idle periods depending on whether customers are present. When the server is active at the end of a visit of Q_2 to L_1 or L_2, service will be preempted. At the beginning of the next visit of Q_2, the service time will be re-sampled according to $S_i(\cdot)$. This discipline is commonly referred to as *preemptive-repeat-random*. Let $N_i(t)$ denote the number of customers in Q_i, $i = 1, 2$, at time t. Assume $N_i(0) = 0$, $i = 1, 2$.

Further, we note that in the analysis we will use visit (time) rather than contact (time) to refer to (the duration of) a contact opportunity as to be in line with the common practice in the polling literature. Also, it is worth pointing out that the term customer throughout this paper will designate packet.

Our objective is to analyze the sojourn time of a customer in the tandem system and at the individual queues Q_1 and Q_2. First, we will state the stability conditions for the tandem system. Second, we discuss several results for the sojourn time and queue length at Q_1 which will be required in the analysis later. Next, we determine the joint queue-length probabilities for the tandem model at specific instants. These probabilities can be related to the time-equilibrium probabilities. Finally, applying Little's law, the mean sojourn time at Q_2 is obtained.

2.2 Stability Condition

The tandem model is stable if each customer in the system can be served in a finite period of time. Stability is considered on a per-queue basis as service capacity cannot be exchanged between the queues. The system is stable if and only if all the queues in the system are stable.

Let a cycle define the time that separates two consecutive server visits to a queue. Due to the independence assumptions on our rvs, cycle lengths are iid, with generic rv $C := X^{L_1} + X^{L_2} + C^{1,2} + C^{2,1}$. For an individual queue to be stable, we must have that on average the number of customer arrivals per cycle is smaller than the number of customers that can be served at most per cycle. This latter number for Q_i will be denoted by $N^i_{\max}$, $i = 1, 2$, and is geometrically distributed (due to the exponential visit times and preemptive-repeat-random discipline), i.e., $\mathbb{P}(N^i_{\max} = k) = p_i(1-p_i)^k$, $k = 0, 1, 2, \ldots$, where $p_i = \mathbb{P}(\text{service is preempted at } Q_i) = 1 - \tilde{S}_i(\alpha_i)$, $i = 1, 2$. Thus, the stability condition for Q_i, $i = 1, 2$, reads

$$\rho_i := \frac{\mathbb{E}[\text{arrivals per cycle to } Q_i]}{\mathbb{E}[N^i_{\max}]} = \lambda \mathbb{E}[C] \cdot \frac{1 - \tilde{S}_i(\alpha_i)}{\tilde{S}_i(\alpha_i)} < 1, \qquad (1)$$

where ρ_i is referred to as generalized load at Q_i and $\mathbb{E}[C] = 1/\alpha_1 + 1/\alpha_2 + c^{1,2} + c^{2,1}$, the mean cycle time. Notice that under stability the arrival rate to Q_2 and Q_1 are equal.

2.3 Queue One

Recall that the server visit process is autonomous and that service is according to the preemptive-repeat-random discipline. It is then easily seen that Q_1 in

isolation is an M/G/1 queue with on-off server with arrival rate λ, mean service time $1/\beta_1$, exponential on-period X^{L_1} with rate α_1, and off-period R^{off} equal to the switch-over times plus the server visit time to Q_2 at L_2, i.e., $R^{off} = C^{1,2} + C^{2,1} + X^{L_2}$. By a renewal reward argument P_{on}, the probability that the server is on, satisfies $P_{on} = (\alpha_1 \mathbb{E}[C])^{-1}$ and $P_{off} := 1 - P_{on}$.

The M/G/1 queue with on-off server has been extensively studied in the literature (see, e.g., [14,15]). Let us state here only the results that are relevant for our analysis. The LST of the sojourn time of a customer is denoted by $\tilde{D}_1(s)$ and follows from a decomposition argument [15]

$$\tilde{D}_1(s) = \tilde{W}_1(s)\tilde{S}^{eff}(s) \ , \tag{2}$$

where $\tilde{W}_1(s)$ and $\tilde{S}^{eff}(s)$ denote the LST of the waiting time of a customer (until it is taken into service for the first time) and the effective service time (including possible service interruptions), respectively. These latter LSTs are given by [15]

$$\tilde{W}_1(s) = \tilde{W}_{M/G/1}(s)\big(P_{on} + P_{off}\tilde{R}_e^{off}(s)\big), \tag{3}$$

$$\tilde{S}^{eff}(s) = \frac{(\alpha_1 + s)(\alpha_2 + s) \cdot \tilde{S}_1(\alpha_1 + s)}{(\alpha_1 + s)(\alpha_2 + s) - \alpha_1\alpha_2(1 - \tilde{S}_1(\alpha_1 + s)) \cdot \tilde{C}^{1,2}(s)\tilde{C}^{2,1}(s)} \ , \tag{4}$$

where $Re(s) > 0$, $\tilde{R}_e^{off}(s)$ denotes the LST of the residual time of an off-period and $\tilde{W}_{M/G/1}(s)$ is the LST of the waiting time in the "corresponding" M/G/1 queue with service time with LST $\tilde{S}^{eff}(s)$.

It follows that the probability generating function (p.g.f.) of N_1, the number of customers at Q_1, which we denote by $F_1(\cdot)$, can be expressed as function of $\tilde{D}_1(\cdot)$ using the so-called functional form of Little's law (see [16] for a general proof for FIFO queues with non-anticipating arrivals) as follows

$$F_1(z) = \tilde{D}_1\big(\lambda(1 - z)\big), \quad |z| \leq 1 \ . \tag{5}$$

Let us denote by $F^{\{-2,1\}}(\cdot)$ the p.g.f. of the number of customers at the end of an off-period, i.e., at the transition of $L(t)$ from -2 to 1. It can then be shown in a couple of steps by using Eq. (5), the PASTA property and conditioning on the position of the server, that

$$F^{\{-2,1\}}(z) = \tilde{W}_{M/G/1}\big(\lambda(1 - z)\big) \cdot \tilde{S}^{eff}\big(\lambda(1 - z)\big) \cdot \tilde{R}^{off}\big(\lambda(1 - z)\big) \ . \tag{6}$$

This function will be required in Section 3 in the approximative analysis for Q_2. For more details on its derivation, we refer to [17].

2.4 Queues in Tandem

Joint Queue-length Probabilities at the End of a Server Visit. In this section, we will determine the queue-length distribution at the end of a server visit at each queue of the tandem model. The analysis builds on the work of Eisenberg [18] and involves setting up an iterative scheme. This iterative approach was introduced by Leung [19] for the study of a probabilistically-limited polling model.

Later, this model was extended in [12] to a time-limited polling model and in [13] for a time-limited model in which the server remains at a queue even if it becomes empty. A key role in the iterative scheme is played by the (auxiliary) p.g.f.'s $\phi_k^i(\mathbf{z})$ and $\phi_k^{s,i}(\mathbf{z})$ for $\mathbf{z} := (z_1, z_2)$, which will be explained below. In the final step of the iteration scheme $\gamma^i(\mathbf{z})$, the p.g.f. of the queue-length distribution *at the end of a server visit* to Q_i, is obtained as a function of $\phi_k^{s,i}(\mathbf{z})$.

Let us consider a service visit to a tagged queue i. We will recursively relate the number of customers present at the end of this visit to the number present at the beginning. To this end, we mark three types of events that may occur during a server visit viz., a customer arrival to an empty Q_i, a service completion at Q_i, and the departure of a server from Q_i. Further, we define for $k \geq 0$, $N_k^i := (N_{k,1}^i, N_{k,2}^i)$, where $N_{k,j}^i$, $j = 1, 2$, denotes the number of customers at Q_j just after the epoch of the k-th marked event at Q_i. We let N_0^i refer to the number of customers present at the arrival of the server to Q_i. Finally, we denote by the rv $\kappa_i \geq 1$ the number of marked events that occurs during a visit time of Q_i. It is immediately clear that the sequence $\{N_k^i\}_{k=0}^{\infty}$ is a Markov chain. We may then define for $k \geq 1$

$$\phi_k^i(\mathbf{z}) := \mathbb{E}[\mathbf{z}^{N_k^i} \mathbf{1}_{\{\kappa_i > k\}}] \ , \tag{7}$$

where $\mathbf{1}_{\{A\}}$ is the indicator function of event A ($\mathbf{1}_{\{A\}} = 1$, if A is true, and 0 otherwise). That is, $\phi_k^i(\mathbf{z})$ is the joint p.g.f. of the number of customers at all queues at the k-th marked event epoch at Q_i and marked event k is not the final marked event during the visit (i.e., marked event $k+1$ will occur). Similarly, we define for $k \geq 1$

$$\phi_k^{s,i}(\mathbf{z}) := \mathbb{E}[\mathbf{z}^{N_k^i} \mathbf{1}_{\{\kappa_i = k\}}] \ . \tag{8}$$

That is, $\phi_k^{s,i}(\mathbf{z})$ is the joint p.g.f. of the number of customers at all queues at the k-th marked event epoch at Q_i and k is the final marked event (i.e., marked event k is a server departure event, and marked event $k+1$ will not occur). Let $N(T)$ be the number of arrivals during a random period T, I_1 be the (exponential) inter-arrival time of customers at Q_1, and $C^{i,j}(\mathbf{z})$ be the p.g.f. of the number of arrivals during a switch-over time from Q_i to Q_j. Then, by analogy with the results of De Haan et al. [13] for a time-limited polling system, $\phi_k^i(\mathbf{z})$ and $\phi_k^{s,i}(\mathbf{z})$, $i = 1, 2,\ k = 1, 2, \ldots$, are recursively given by

$$\begin{aligned} \phi_k^1(\mathbf{z}) &= \phi_{k-1}^1(\mathbf{z}) \mid_{z_1=0} \cdot \mathbb{E}[\mathbf{z}^{N(I_1)} \mathbf{1}_{\{X^{L_1} > I_1\}}] \cdot z_1 \\ &\quad + \left(\phi_{k-1}^1(\mathbf{z}) - \phi_{k-1}^1(\mathbf{z}) \mid_{z_1=0}\right) \cdot \mathbb{E}[\mathbf{z}^{N(S_1)} \mathbf{1}_{\{X^{L_1} > S_1\}}] \cdot z_2/z_1 \ , \end{aligned} \tag{9}$$

$$\phi_k^2(\mathbf{z}) = \left(\phi_{k-1}^2(\mathbf{z}) - \phi_{k-1}^2(\mathbf{z}) \mid_{z_2=0}\right) \cdot \mathbb{E}[\mathbf{z}^{N(S_2)} \mathbf{1}_{\{X^{L_2} > S_2\}}] \ / z_2 \ , \tag{10}$$

and

$$\begin{aligned} \phi_k^{s,1}(\mathbf{z}) &= \phi_{k-1}^1(\mathbf{z}) \mid_{z_1=0} \cdot \mathbb{E}[\mathbf{z}^{N(X^{L_1})} \mathbf{1}_{\{X^{L_1} < I_1\}}] \\ &\quad + \left(\phi_{k-1}^1(\mathbf{z}) - \phi_{k-1}^1(\mathbf{z}) \mid_{z_1=0}\right) \cdot \mathbb{E}[\mathbf{z}^{N(X^{L_1})} \mathbf{1}_{\{X^{L_1} < S_1\}}] \ , \end{aligned} \tag{11}$$

$$\begin{aligned} \phi_k^{s,2}(\mathbf{z}) &= \phi_{k-1}^2(\mathbf{z}) \mid_{z_2=0} \cdot \mathbb{E}[\mathbf{z}^{N(X^{L_2})}] \\ &\quad + \left(\phi_{k-1}^2(\mathbf{z}) - \phi_{k-1}^2(\mathbf{z}) \mid_{z_2=0}\right) \cdot \mathbb{E}[\mathbf{z}^{N(X^{L_2})} \mathbf{1}_{\{X^{L_2} < S_2\}}] \ , \end{aligned} \tag{12}$$

where

$$\begin{aligned}
\phi_0^i(\mathbf{z}) &= \gamma^{3-i}(\mathbf{z})C^{3-i,i}(\mathbf{z}) \ , \\
\mathbb{E}[\mathbf{z}^{N(I_1)}\mathbf{1}_{\{X^{L_1}>I_1\}}] &= \lambda/(\lambda+\alpha_1) \ , \\
\mathbb{E}[\mathbf{z}^{N(S_i)}\mathbf{1}_{\{X^{L_i}>S_i\}}] &= \tilde{S}_i(\alpha_i+\lambda(1-z_1)) \ , \\
\mathbb{E}[\mathbf{z}^{N(X^{L_1})}\mathbf{1}_{\{X^{L_1}<I_1\}}] &= \alpha_1/(\lambda+\alpha_1) \ , \\
\mathbb{E}[\mathbf{z}^{N(X^{L_i})}\mathbf{1}_{\{X^{L_i}<S_i\}}] &= \alpha_i \cdot \frac{1-\tilde{S}_i(\alpha_i+\lambda(1-z_1))}{\alpha_i+\lambda(1-z_1)} \ , \\
\mathbb{E}[\mathbf{z}^{N(X^{L_2})}] &= \alpha_2/(\alpha_2+\lambda(1-z_1)) \ .
\end{aligned}$$

Equation (9) can be explained by the following observations. First, the time between the $(k-1)$-th and the k-th marked event epoch (and thus also the number of arriving customers) depends on whether at least one customer was present at the $(k-1)$-th marked event epoch. This explains why the equation consists of two parts. Second, the number of customers at all queues at a marked event epoch is equal to the number present at the previous marked event epoch adjusted for the arrivals and departures in the meantime. Equation (10) consists only of one part due to the fact that once Q_2 is empty, it will remain empty during the rest of that visit. Along the same lines as Eqs. (9) and (10), Eqs. (11) and (12) are derived. Finally, we note that $\phi_0^i(\mathbf{1}) = 1$ for $\mathbf{1}$=(1,1), while $\phi_k^i(\mathbf{1}) \leq 1$, for all $k = 1, 2, \ldots$, since the k-th marked event might not occur at all during a visit to Q_i.

Notice that the final marked event is always a departure of the server from Q_i. Therefore, we can write for the number of customers at the queues at the end of a server visit to Q_i

$$\gamma^i(\mathbf{z}) = \mathbb{E}[\mathbf{z}^{N^i_{\kappa_i}}] = \lim_{K\to\infty} \sum_{k=1}^{K} \mathbb{E}[\mathbf{z}^{N^i_k}\mathbf{1}_{\{\kappa_i=k\}}] = \lim_{K\to\infty} \sum_{k=1}^{K} \phi_k^{s,i}(\mathbf{z}) \ . \quad (13)$$

For computational convenience, we will numerically invert the joint p.g.f. of the number of customers at the queues using the Discrete Fourier Transform. To apply this technique, we only need the values of the joint p.g.f. to be known at a finite number of points (z_1, z_2), with $z_i = \omega_i^{k_i}$, where $\omega_i = exp(-2\pi I/J_i)$. Here I is the imaginary unit and J_i refers to the number of discrete points used for Q_i in the transformation process. Hence, we replace $z_i,\ i = 1, 2$, in the expressions above by $\omega_i^{k_i}$, so that all expressions become functions of $\mathbf{k} = (k_1, k_2)$. Also, let $\check{\gamma}^i(\mathbf{k})$ refer to $\gamma^i(\mathbf{z})$, with $z_i = \omega_i^{k_i}$. We start the iteration process with an empty system and then compute $\check{\gamma}^1(\mathbf{k})$ using Eq. (13) up to a finite K such that $1 - \check{\gamma}^i(\mathbf{0}) < \delta$, for some $\delta > 0$. Next, we compute $\check{\gamma}^2(\mathbf{k})$ in the same way and then we continue with $\check{\gamma}^1(\mathbf{k})$ again, and so on. The iteration process is stopped when $\forall\ \mathbf{k} : |\mathrm{Re}(\check{\gamma}^i(\mathbf{k})) - \mathrm{Re}(\hat{\gamma}^i(\mathbf{k}))| < \epsilon,\ i = 1, 2$, for some $\epsilon > 0$, where $\hat{\gamma}^i(\mathbf{k})$ refers to the previously obtained value for $\check{\gamma}^i(\mathbf{k})$. The standard values for the convergence parameters that have been used are $\epsilon = 10^{-6}$ and $\delta = 10^{-9}$. Finally, via the inverse transform, the joint queue-length probabilities at visit completion instants can be found. We note that for these probabilities to be

exact, we need at least that $J_i \to \infty,\ i = 1, 2$. However, the strength of the approach is that the probabilities are already close to the exact probabilities for small values of J_i. It should be noted that when the system load increases, the values J_i must typically be increased to guarantee the accurate computation of the probabilities. Thus, this iterative approach appears mainly applicable to systems with a light to moderate load.

Mean Sojourn Time at Queue Two. The sojourn time is related to the *time-equilibrium* queue-length probabilities. These probabilities can be obtained from the queue-length probabilities at the end of a server visit due to the exponential visit times. This is done by conditioning on the position of the server. Notice that the server is either at some queue or switching from one queue to another. Let us consider the queue-length process and condition on the server being at some tagged queue Q_i. This induced process consists of a series of exponentially distributed periods with positive jumps between each two periods. Due to the PASTA property, the system state just before these (Poisson) jump epochs equals exactly the time-average system state. Moreover, the system state at these epochs is exactly the state as observed by the server when it departs from Q_i. Thus, we have that a departing server observes the system in steady-state conditioned on the queue it departs from. Let us further denote the p.g.f. of the number of customers present at a random instant during a switch-over time from Q_{3-i} to Q_i by $C_R^i(\mathbf{z})$. It can readily be found that

$$C_R^i(\mathbf{z}) = \gamma^i(\mathbf{z}) \cdot \frac{1 - \tilde{C}^{3-i,i}\big(\lambda(1 - z_1)\big)}{c^{3-i,i} \cdot \lambda(1 - z_1)}. \tag{14}$$

Hence, by conditioning on the position of the server, we may write for $P(\mathbf{z}) := \mathbb{E}[z_1^{N_1} z_2^{N_2}]$, the joint p.g.f. of the time-equilibrium queue lengths,

$$P(\mathbf{z}) = \frac{1}{\mathbb{E}[C]} \sum_{i=1}^{2} \left(\gamma^i(\mathbf{z}) \cdot \frac{1}{\alpha_i} + C_R^i(\mathbf{z}) \cdot c^{3-i,i} \right). \tag{15}$$

The mean sojourn time at Q_2 then follows from the mean number of customer at Q_2, $\mathbb{E}[N_2]$, and applying Little's law

$$\mathbb{E}[D_2] = \frac{\mathbb{E}[N_2]}{\lambda} = \frac{1}{\lambda} \cdot \frac{d}{dz_2} P(1, z_2) \mid_{z_2=1} . \tag{16}$$

Remark 1. We note that using the distributional form of Little's law also higher moments can be obtained for the total sojourn time in the tandem system. However, this form cannot be applied to the individual sojourn time at Q_2, since the arrival process to Q_2 does not satisfy the non-anticipating property [16].

3 Approximation

In this section, we present an approximation for $\tilde{D}_2(s)$, the LST of the sojourn time of a customer in the mobile queue Q_2, so that we may also deal with the

situations in which the exact approach is no longer computationally feasible. We consider the workload process in Q_2 when $L(t) = 0$, i.e. Q_2 is served. This will be done under the additional assumption that the service times are exponentially distributed at both queues with rate β_1 and β_2 respectively. It turns out that this process corresponds to the workload process in an $M/M/1$ queue with batch arrivals. The sojourn time of a customer in Q_2 then equals the sum of

- the sojourn time of a customer in the batch arrival queue,
- the time a customer is at Q_2 but $L(t) \neq 0$, i.e. Q_2 is not served.

We emphasize that in this case both the preemptive-resume and preemptive-repeat-random disciplines are stochastically identical. For the sake of simplicity, in the following we will consider the preemptive-resume discipline.

3.1 The Workload in Queue Two

To study the workload process at Q_2, we split the time into disjoint intervals which begin at the time instants that the $L(t)$-process jumps from state -2 to 1 (i.e., the start of an on-period at Q_1). Denote the starting points of these intervals by $\{Z_n, n = 1, 2, \ldots\}$, with, by convention, $Z_1 = 0$. Let the n-th cycle of $L(t)$ denote the time interval $[Z_n, Z_{n+1}[$, with duration $X_n^{L_1} + C_n^{1,2} + X_n^{L_2} + C_n^{2,1}$. Let $V(t)$ denote the workload (i.e., virtual waiting time) of Q_2 present at time t. Without loss of generality, we assume that $V(t)$ is left-continuous, i.e., arrivals are not counted as being in the system until just after they arrive. A sample path of the evolution of $V(t)$ as function of $L(t)$ is shown in Figure 1.

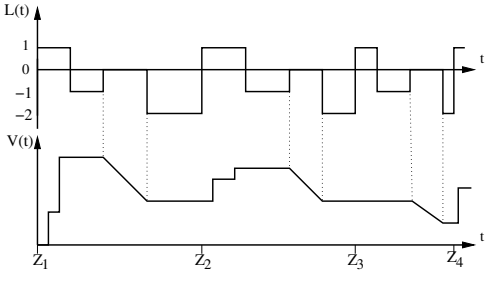


Fig. 1. Evolution of $L(t)$ and workload $V(t)$ of Q_2

Let W_n^B denote the workload present in Q_2 at time Z_n. Based on the evolution of $L(t)$, it is easily seen that

$$W_{n+1}^B = \left(W_n^B + \sum_{i=1}^{K_n} S_{2,i} - X_n^{L_2} \right)^+, \quad n \geq 0, \tag{17}$$

where $(.)^+ = \max(., 0)$, K_n is the total number of arrivals to Q_2 (or departures from Q_1) during $X_n^{L_1}$ and $S_{2,i}$ is the service requirement of a customer in Q_2.

Note that $S_{2,i}$ is independent of $X_n^{L_2}$ and that K_n depends on $N_1(Z_n)$, the number of customers at Q_1 at time Z_n. Therefore, K_n, $n = 1, 2, \ldots$, are correlated. For the sake of model tractability, we need the following

Assumption: K_n, $n = 1, 2, \ldots$, *are iid and independent of* $\{X_m^{L_2} : m \leq n\}$.

By this assumption, Eq. (17) represents the workload seen by the first customer of a batch in a queue with Poisson batch arrivals with rate α_2, independent batch size K_n, and exponential service requirement with rate β_2.

Let $\tilde{G}(s) := E\left[e^{-s\sum_{i=1}^{K_n} S_{2,i}}\right]$, i.e., $\tilde{G}(s)$ denotes the LST of the service requirement of a batch. It is well known that the batch arrival queue is stable when $-\alpha_2\tilde{G}'(0) = \alpha_2\mathbb{E}[K_n]/\beta_2 < 1$. It can readily be verified that the latter condition is equivalent to the condition in (1) for Q_2. Furthermore, the LST of the steady-state distribution of W_n^B can be written as

$$\tilde{W}^B(s) = \left(1 + \alpha_2\tilde{G}'(0)\right)\frac{s}{s - \alpha_2 + \alpha_2\tilde{G}(s)}. \tag{18}$$

By conditioning on K_n, we find that

$$\tilde{G}(s) = \mathbb{E}\left[\left(\tfrac{\beta_2}{\beta_2+s}\right)^{K_n}\right]. \tag{19}$$

Finally, let $\tilde{V}^j(s)$ denote the LST of the workload seen by the j-th customer within a batch upon arrival (including the work brought in by itself). Since the service requirement of customers is independent of the workload present in the queue upon arrival and its distribution is exponential with rate β_2, $\tilde{V}^j(s)$ reads

$$\tilde{V}^j(s) = \tilde{V}^{j-1}(s)\frac{\beta_2}{\beta_2 + s}, \quad j = 1, 2, \ldots, \tag{20}$$

with $\tilde{V}^0(s) = \tilde{W}^B(s)$. Moreover, since K_n are iid, $\mathbb{P}(J = j)$, the probability that a customer is the j-th customer within the batch is equal to the fraction of customers who are j-th arrival in their own batch, which gives $\mathbb{P}(J = j) = \mathbb{P}(K_n \geq j)/\mathbb{E}[K_n]$.

Removing the condition on the customer position in a batch, the LST of the sojourn time of an arbitrary customer in the batch arrival queue is given by

$$\tilde{V}(s) = \beta_2\tilde{W}^B(s)\frac{1 - \tilde{G}(s)}{s\mathbb{E}[K_n]}. \tag{21}$$

It remains to compute $\mathbb{E}[z^{K_n}]$ in order to find $\tilde{G}(s)$, and eventually $\tilde{W}^B(s)$. This will be done next using the matrix-geometric approach.

3.2 The p.g.f. of the Batch Size Distribution

As remarked in the previous section, K_n is the total number of departures from Q_1 during the n-th cycle and depends on the queue length of Q_1 at time Z_n. To

compute the p.g.f. of K_n, we first assume that Q_1 has a limited queue of $M-1$ customers. This queue is denoted by Q_1^M. Later, we will let M tend to infinity to get the final results.

As we need the arriving batch size distribution in steady state, we assume that Q_1^M is in steady state at time Z_n. The probability that there are i customers in Q_1^M at Z_n is denoted by $b_M(i)$. Under the assumption that the unlimited Q_1 is stable, $\lim_{M\to\infty} b_M(i) = b(i)$ with $\sum_{i\geq 0} b(i)z^i = F^{\{-2,1\}}(z)$ (see Eq. (6)). Let $b_M = (b_M(0), \ldots, b_M(M-1))$ denote the steady-state distribution of the finite capacity Q_1^M.

Let $(N_1(t), D(t))$ denote the two dimensional, continuous-time process with discrete state-space $\{0, 1, \ldots, M-1\} \times \{0, 1, \ldots\} \cup \{(M, 0)\}$, where $N_1(t)$ represents the number of customers in Q_1 at time t, and $D(t)$ the number of departures from Q_1 until t. State $(M, 0)$ is absorbing. We refer to this absorbing Markov chain by **AMC**. The absorption of **AMC** occurs when the server leaves the queue which happens with rate α_1. By setting the probability that the initial state of **AMC** at $t = 0$ is $(i, 0)$ to $b_M(i)$, the probability that the absorption of **AMC** occurs from one of the states $\{(i, k) : i = 0, 1, \ldots, M-1\}$ equals the steady-state batch size distribution $\mathbb{P}(K_n = k)$. The non-zero transition rates of **AMC** can be written as

$$\begin{array}{lll} q\big((i,j),(i+1,j)\big) = \lambda, & 0 \leq i \leq M-2, & j \geq 0, \\ q\big((i,j),(i-1,j+1)\big) = \beta_1, & 1 \leq i \leq M-1, & j \geq 0, \\ q\big((i,j),(M,0)\big) = \alpha_1, & 0 \leq i \leq M-1, & j \geq 0. \end{array} \tag{22}$$

Let us order the infinite number of **AMC** states as: $(0,0), \ldots, (M-1,0), (0,1)$ $, \ldots, (M-1,1), \ldots$, and finally $(M,0)$. It is easily seen that the transition matrix $\mathbf{P}$ of **AMC** can be written as

$$\mathbf{P} = \left(\begin{array}{c|c} \mathbf{Q} & \mathbf{R} \\ \hline \mathbf{0} & 0 \end{array}\right),$$

where $\mathbf{Q}$ is an upper-bidiagonal block matrix of infinite dimension, $\mathbf{0}$ is the row vector with all zero entries and $\mathbf{R} = (\alpha_1, \ldots, \alpha_1)^T$. The blocks of $\mathbf{Q}$'s diagonal are all equal to $\mathbf{A}$, a M-by-M bidiagonal matrix with diagonal $(-\lambda-\alpha_1, -\lambda-\alpha_1-\beta_1, \ldots, -\lambda-\alpha_1-\beta_1, -\alpha_1-\beta_1)$ and upper-diagonal $(\lambda, \ldots, \lambda)$. The blocks of $\mathbf{Q}$'s upper-diagonal are all equal to $\mathbf{B}$, a M-by-M lower-diagonal matrix with lower-diagonal $(\beta_1, \ldots, \beta_1)$.

Next, we will derive $\mathbb{P}(K_n = k)$ as function of the inverse of $\mathbf{Q}$. Since $\mathbf{Q}$ is an upper-bidiagonal block matrix, $\mathbf{Q}^{-1}$ is an upper-triangular block matrix. It is easy to verify that the entries of $\mathbf{Q}^{-1}$ are equal to $\mathbf{U}_{m,l} = \big(-\mathbf{A}^{-1}\mathbf{B}\big)^{l-m}\mathbf{A}^{-1}$ for $m \geq 0$ and $l \geq m$. Note that the matrix $\mathbf{A}$ is invertible since it is upper-bidiagonal with strictly negative diagonal entries.

From the theory of absorbing Markov chains, given that the initial state vector of **AMC** is b_M, the probability that the absorption occurs at one of the states $\{(i, k) : i = 0, 1, \ldots, M-1\}$ is given by (see, e.g., [20]) $P(K_n = k) = -\alpha_1 b_M(\mathbf{U}_{0,k})e = -\alpha_1 b_M\big(-\mathbf{A}^{-1}\mathbf{B}\big)^k\mathbf{A}^{-1}e$, where e denote the M-dimensional

column vector with all entries equal to one. After some algebra, we find

$$E_M[z^{K_n}] = -\alpha_1 b_M(\mathbf{A}+z\mathbf{B})^{-1}e, \quad |z| \le 1. \tag{23}$$

Therefore, it remains to find $(\mathbf{A}+z\mathbf{B})^{-1}$.

Now, define $\mathbf{Q}(z) := (\mathbf{A}+z\mathbf{B})$, let $u^T = (1,0,\ldots,0)$ and let $v^T = (0,\ldots,0,1)$. Observe that $\mathbf{Q}(z) = \mathbf{T}(z) + \beta_1 uu^T + \lambda vv^T$, where $\mathbf{T}(z)$ is a M-by-M tridiagonal Toeplitz matrix with diagonal entries equal to $(-\lambda-\beta_1-\alpha_1)$, upper-diagonal entries equal to λ, and lower-diagonal entries $z\beta_1$. Let t^*_{ij} denote the (i,j)-entry of $\mathbf{T}^{-1}(z)$. By applying the Sherman-Morrison formula [21, pp. 76] we find that the (i,j)-entry of $\mathbf{Q}^{-1}(z)$ gives for $i,j = 1,\ldots,M$,

$$q^*_{ij} = m_{ij} - \lambda\frac{m_{iM}m_{Mj}}{1+\lambda m_{MM}}, \quad \text{where } m_{ij} = t^*_{ij} - \beta_1\frac{t^*_{i1}t^*_{1j}}{1+\beta_1 t^*_{11}}. \tag{24}$$

The inverse of a tridiagonal Toeplitz matrix has been computed in closed-form (see [22, Sec. 3.1]). Following that same approach, we obtain t^*_{ij} as function of r_1 and r_2, the distinct roots of $\lambda r^2 - (\lambda+\beta_1+\alpha_1)r + \beta_1 z$ with $|r_1| < |r_2|$. Inserting the values of t^*_{ij} into (23) and letting $M \to \infty$ yield that (for a detailed derivation see [17, Sec. 3.2])

$$\mathbb{E}[z^{K_n}] = \frac{\alpha_1}{\lambda(1-r_1)(r_2-1)}\left[1+\beta_1\frac{1-z}{\lambda r_2-\beta_1}F^{\{-2,1\}}(r_1)\right], \tag{25}$$

where $F^{\{-2,1\}}(.)$ is given in (6). Inserting $z = \beta_2/(\beta_2+s)$ into (25) gives the closed form of $\tilde{G}(s)$, the LST of the service requirement of a total batch (see (19)), which in turn gives the closed form of $\tilde{W}^B(s)$.

3.3 Sojourn Time in Queue Two

Let H_0 denote the sojourn time of a customer in the batch arrival queue. Moreover, let $\{H_t : t \ge 0\}$ denote the remaining sojourn time of a customer in Q_2 if the server would be continuously working at Q_2 from time t onwards. In other words, H_t decreases at rate 1 when $L(t) = 0$ and H_t is constant when $L(t) \in \{-2,-1,1\}$ at time t. Let Y denote the number of service interruptions during the sojourn time of a customer.

The visit periods have an exponential length with rate α_2. Now, given that $H_0 = v$, the number of interruptions has a Poisson distribution with

$$\mathbb{E}[z^Y | H_0 = v] = e^{-\alpha_2 v(1-z)}. \tag{26}$$

The duration of these interruptions are independent and are given by $\Xi = C^{2,1} + X^{L_1} + C^{1,2}$. Furthermore, Ξ^*, the time it takes before H_t actually starts decreasing after time 0, satisfies $\Xi^* = X_e^{L_1} + C^{1,2}$, where $X_e^{L_1}$ is the residual time of X^{L_1}. Note that $X_e^{L_1}$ and X^{L_1} are identically distributed with common exponential distribution. It is easily seen that $D_2 = \Xi^* + H_0 + \sum_{i=1}^{Y}\Xi_i$. By conditioning on H_0 and Y, we find for the LST of D_2,

$$\tilde{D}_2(s) = \mathbb{E}\big[e^{-s\Xi^*}\big]\mathbb{E}\big[e^{-s(\sum_{i=1}^{Y}\Xi_i + H_0)}\big] = \mathbb{E}\big[e^{-s\Xi^*}\big]\mathbb{E}\big[e^{-sH_0}e^{-\alpha_2 H_0(1-\tilde{\Xi}(s))}\big].$$

where $\tilde{\Xi}(s) = \frac{\alpha_1}{\alpha_1+s}\tilde{C}^{1,2}(s)\tilde{C}^{2,1}(s)$. Since H_0 equals the sojourn time in the batch arrival queue, we find (see, Eq. (21))

$$\tilde{D}_2(s) = \frac{\alpha_1\tilde{C}^{1,2}(s)}{\alpha_1+s} \times \tilde{W}^B\big(\Delta(s)\big) \times \frac{\beta_2}{\mathbb{E}[K_n]} \times \frac{1-\tilde{G}\big(\Delta(s)\big)}{\Delta(s)}, \tag{27}$$

where $\Delta(s) := s + \alpha_2(1-\tilde{\Xi}(s))$.

4 Numerical Evaluation

The evaluation of the model will be done in three parts. First, we will extensively validate the accuracy of the approximation. Second, we consider the impact of the switch-over time distribution on the mean end-to-end delay in the network. Finally, we study the optimization of the end-to-end delay in the network by adjusting the visit time parameters via power control. Throughout this section, the distribution of the switch-over times of Q_2, $C^{1,2}$ and $C^{2,1}$, are assumed identically distributed according to an exponential distribution with mean $c^{1,2} = c^{2,1}$.

4.1 Model Validation

We validate the approximate model developed in Section 3.3 for the mean sojourn time at Q_2 by comparison with the results for the exact model of Section 2.4. Note that the exact expression of the LST of the sojourn time at Q_1 was derived in Section 2.3. The approximation assumption is K_n, $n = 1, 2, \ldots$, are iid and also independent of $\{X_m^{L_2} : m \leq n\}$.

Let us introduce some notation. Let $\mathbb{E}[D_2^{app}]$ (resp. $\mathbb{E}[D_2^{exa}]$) denote the mean sojourn time in Q_2 using the approximate (resp. exact) model given in Sect. 3.3 (resp. in Sect. 2.4). Let R_2 denote the absolute relative difference between the approximate and exact mean sojourn time in Q_2, i.e., $R_2 := |1 - \mathbb{E}[D_2^{app}]/\mathbb{E}[D_2^{exa}]|$. Further, we note that the load at Q_1 and Q_2 can be rewritten as (see (1))

$$\rho_i = \frac{\lambda}{\beta_i}\left(\frac{\alpha_1+\alpha_2}{\alpha_{3-i}} + 2\alpha_i c^{1,2}\right), \quad i = 1, 2.$$

Figure 2(a) displays R_2 as a function of λ for different values of $c^{1,2}$ with $\beta_1 = \beta_2 = 1$ and $\alpha_1 = \alpha_2 = 0.1$. Thus, in this scenario the load at Q_1 and Q_2 are equal ($\rho_1 = \rho_2$). Observe that R_2 increases with λ and that the approximate model is accurate for $\rho_1 = \rho_2 < 0.5$. This is because the probability that Q_1 is empty upon the departure of the server from Q_1 decreases with λ. For this reason, the auto-correlation of K_n increases with λ. Moreover, Figure 2(a) shows that R_2 decreases with $c^{1,2}$ for $\rho_1 = \rho_2$ (e.g., for $\rho_1 = 0.5$, $R_2 = 15\%$ when $c^{1,2} = 1$ and $R_2 = 8\%$ when $c^{1,2} = 20$). This is because in the case where $\rho_1 = \rho_2$, λ decreases with $c^{1,2}$.

Figure 2(b) shows the mean sojourn time in Q_2 using the approximate and exact models. Observe that the approximation gives an upper bound for $\mathbb{E}[D_2^{exa}]$.

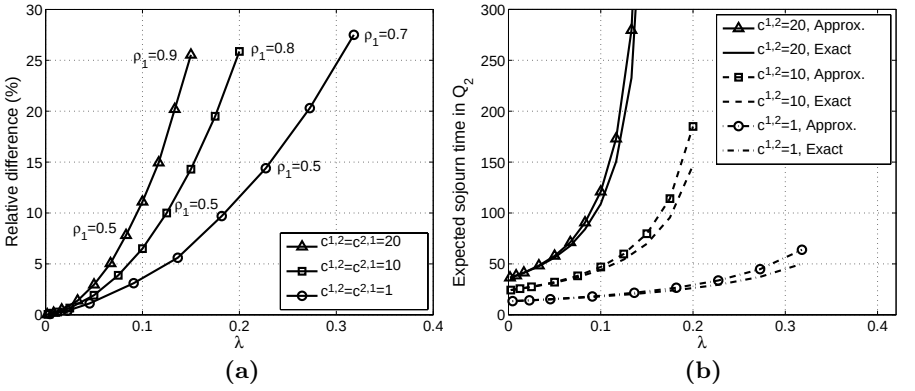


Fig. 2. For $\beta_1 = \beta_2 = 1$, $\alpha_1 = \alpha_2 = 0.1$: **(a)** Relative difference R_2 as a function of λ. **(b)** Mean sojourn time in Q_2 as a function of λ.

This observation is in support of the result in [23] which proves that in the correlated $M/G/1$ a positive correlation between the service requirement and the last inter-arrival reduces the mean sojourn time. We should emphasize that also in our model K_n and the last inter-arrival are positively correlated, i.e., an increase of the last inter-arrival time induces stochastically an increase of K_n. Finally, we found that for a given ρ_2, the approximate model is more accurate for higher values of ρ_1 (e.g., when $\rho_2 = 0.6$, $R_2 = 9.6\%$ for $\rho_1 = 0.25$, while $R_2 = 4.7\%$ for $\rho_1 = 0.75$).

We conclude that the approximate model has the following properties:

- It is accurate for *low and moderate* load at Q_1 and Q_2;
- It gives an upper bound for the sojourn time at Q_2;
- It is accurate for *high* load at Q_1 and *moderate* load at Q_2.

4.2 Impact of the Switch-Over Times Distribution on Sojourn Time

We note that in the analysis the distribution of the switch-over time was assumed to be arbitrary. This section studies the impact of the distribution of the switch-over times on the end-to-end sojourn time of a customer in the network. This will be done by considering the following two different distributions of the switch-over times in such a way that they share the same first two moments: two-phase hyper-exponential and two-phase Coxian distribution.

We evaluated the mean sojourn time as a function of $SCOV_s := Var(C^{1,2})/(c^{1,2})^2$, the squared coefficient of variation of the switch-over times. Through an extensive numerical evaluation, we observed that the mean sojourn time is equal for the two different distributions. This suggests that the mean end-to-end sojourn time depends on the distribution of the switch-over times only through their first two moments. Additional experiments for Weibull distributed switch-over times were performed and were in support of this conjecture. The latter

experiments were done by simulation since the LST of the Weibull distribution is not known in closed form. We note that this conjecture is well known in the context of single-server queue with vacations and several polling models, but to the best of the authors' knowledge it is novel in the context of tandem models with autonomous servers.

4.3 Insight on the Optimal End-to-End Sojourn Time

In this section, we study the evolution of α_2^{opt}, the optimal value of α_2, that yields the minimum value of the end-to-end sojourn time under the constraints of zero switch-over time, i.e., $C^{1,2} = 0$, and constant cycle length, i.e., $\mathbb{E}[C] = 1/\alpha_1 + 1/\alpha_2$ is constant. Note that under these constraints α_1 decreases when α_2 increases. Since the mean sojourn time in Q_1 decreases with α_2 and mean sojourn time at Q_2 increases with α_2, α_2^{opt} exists and it is unique. The adjustment of α_1 and α_2 can be done in practice by controlling the transmission power of the stations.

The optimal value α_2^{opt} can be computed by applying the numerical optimization package of MAPLE to the approximate mean sojourn time in (27). However, in practice one might prefer a more simple and intuitive rule that provides a value for α_2 which yields a close to optimal mean sojourn time. Therefore, we will discuss two alternative, heuristic optimization approaches.

The first heuristic selects the values of α_1 and α_2 such that the load is balanced at both queues, i.e., $\rho_1 = \rho_2$. This gives:

$$\alpha_i = (\beta_1 + \beta_2)/(\beta_{3-i} \cdot \mathbb{E}[C]),\ i = 1, 2\ . \tag{28}$$

The second heuristic chooses α_1 and α_2 based on the analysis of a tandem model of two M/M/1 queues with shared service capacity. This means that the servers at both queues are always present, but serving at rate ν at Q_1 and at rate $1 - \nu$ at Q_2. Then, the optimal ν, say ν^*, is the one that minimizes the end-to-end sojourn in such a tandem model, which we denote by $\mathbb{E}[D]^{M/M/1}$ and equals simply

$$\mathbb{E}[D]^{M/M/1} = 1/(\beta_1\nu - \lambda) + 1/(\beta_2(1-\nu) - \lambda)\ . \tag{29}$$

We choose the ratio α_1/α_2 equal to $(1 - \nu^*)/\nu^*$, such that the fraction of time that the server is at Q_1 in our model equals the optimal rate ν^* in the M/M/1 tandem model.

In Table 1, we present the results of this comparison. Here, α_2^{opt}, α_2^{LB} and $\alpha_2^{M/M/1}$ refer to the choice of α_2 in the optimal case, in the load balancing heuristic, and in the M/M/1 tandem heuristic, respectively. Further, we present the relative differences in mean sojourn time using the two heuristics (denoted by ϵ^{LB} and $\epsilon^{M/M/1}$) with respect to the optimal mean sojourn time, $\mathbb{E}[D]^{opt}$. Table 1 displays the performance of those heuristics when β_1 is increased while β_2, λ and $\mathbb{E}[C]$ are kept constant. We note that for the symmetric case, $\beta_1 = \beta_2$, the heuristics would also give the optimal solution $\alpha_1 = \alpha_2$. The performance using load balancing worsens rapidly when β_1 is increased. Also the M/M/1 tandem heuristic deviates from the

Table 1. Comparison of α_2 and $\mathbb{E}[D]$ for different optimization approaches for $\beta_2 = 1$, $\lambda = 0.1$, and $\mathbb{E}[C] = 10$

β_1	1.1	2	3	6	11	16
α_2^{opt}	0.194	0.174	0.166	0.156	0.150	0.148
α_2^{LB}	0.190	0.150	0.133	0.117	0.109	0.106
$\alpha_2^{M/M/1}$	0.195	0.167	0.154	0.138	0.128	0.123
$\mathbb{E}[D]^{opt}$	14.47	12.82	12.14	11.42	11.08	10.94
ϵ^{LB} (%)	<0.1	3.9	8.6	17.2	23.6	26.3
$\epsilon^{M/M/1}$ (%)	<0.1	0.4	0.9	2.3	3.8	4.7

optimum, but the relative differences remain small. We note that other values of $\mathbb{E}[C]$ were considered for which similar results were found.

We can conclude that balancing the load is not a good solution for end-to-end sojourn time optimization unless $\beta_1 \approx \beta_2$. However, using an optimization heuristic based on a simple tandem model of two M/M/1 queues will give nearly optimal results for the mean sojourn time.

5 Conclusions

This study is part of a research effort towards developing analytical models for quantifying the end-to-end delay in a delay-tolerant network. We consider here a network consisting of a fixed source node, a fixed destination node, and a mobile relay node. A closed-form expression has been derived for the delay at the packet's source node. Next, an iterative approach has been developed for the joint queue-length distribution of the source and the relay node. In addition, a simple approximate model has been proposed for the delay analysis at the relay node. The approximate model has extensively been validated and shows excellent results. Numerical results on the mean end-to-end delay show that the switch-over time distribution impacts this metric only through its first two moments. Moreover, load balancing is not an effective tool for delay optimization, while the M/M/1 tandem heuristic is near optimal.

In future work, we will study scenarios where multiple relay nodes coexist in the network. We anticipate that the exact and approximate models can be extended at least to cover the situations for which only a single copy of a packet exists at a time.

References

1. Grossglauser, M., Tse, D.: Mobility increases the capacity of ad hoc wireless networks. ACM/IEEE Transactions on Networking 10, 477–486 (2002)
2. Delay Tolerant Networking Research Group, `http://www.dtnrg.org`
3. Chaintreau, A., Hui, P., Crowcroft, J., Diot, C., Gass, R., Scott, J.: Impact of human mobility on the design of opportunistic forwarding algorithm. In: Proc. of IEEE INFOCOM, Barcelona, Spain (2006)

4. Groenevelt, R., Nain, P., Koole, G.: The message delay in mobile ad hoc networks. Performance Evaluation 62, 210–228 (2005)
5. Small, T., Haas, Z.J.: Resource and performance tradeoffs in delay-tolerant wireless networks. In: Proc. ACM SIGCOM Workshop on Delay-Tolerant Networks, Philadelphia, PA, USA (2005)
6. Ibrahim, M., Al Hanbali, A., Nain, P.: Delay and resource analysis in manets in presence of throwboxes. Performance Evaluation 64, 933–947 (2007)
7. Zhang, E., Neglia, G., Kurose, J., Towsley, D.: Performance modeling of epidemic routing. Computer Networks 51, 2867–2891 (2007)
8. Spyropoulos, T., Psounis, K., Raghavendra, C.S.: Efficient routing in intermittently connected mobile networks: The single-copy case. ACM/IEEE Transactions on Networking (available online) (to appear, 2007)
9. Wang, C., Wolff, R.: Work-conserving tandem queues. Queueing Systems 49, 283–296 (2005)
10. Coffman, E.G., Fayolle, G., Mitrani, I.: Two queues with alternating service periods. In: Performance 1987: Proc. of the 12th IFIP WG 7.3 Intnl. Symposium on Computer Performance Modelling, Measurement and Evaluation, pp. 227–239 (1988)
11. Frigui, I., Alfa, A.: Analysis of a time-limited polling system. Computer Communications 21(6), 558–571 (1998)
12. Leung, K.: Cyclic-service systems with non-preemptive time-limited service. IEEE Transactions on Communications 42, 2521–2524 (1994)
13. de Haan, R., Boucherie, R.J., van Ommeren, J.C.W.: A polling model with an autonomous server. Research Memorandum 1845, University of Twente (2007)
14. Doshi, B.: Queueing systems with vacations - a survey. Queueing Systems 1, 29–66 (1986)
15. Katayama, T.: Waiting time analysis for a queueing system with time-limited service and exponential timer. Naval Research Logistics 48, 638–651 (2001)
16. Zazanis, M.: A Palm calculus approach to functional versions of Little's law. Stochastic Processes and their Applications 74(7), 195–201 (1998)
17. Al Hanbali, A., de Haan, R., Boucherie, R.J., van Ommeren, J.C.W.: A tandem queueing model for delay analysis in disconnected ad hoc networks. Research Memorandum 1861, University of Twente (2007)
18. Eisenberg, M.: Queues with periodic service and changeover times. Operation Research 20, 440–451 (1972)
19. Leung, K.: Cyclic-service systems with probabilistically-limited service. IEEE Journal on Selected Areas in Communications 9, 185–193 (1991)
20. Gaver, D.P., Jacobs, P.A., Latouche, G.: Finite birth-and-death models in randomly changing environments. Advances in Applied Probability 16, 715–731 (1984)
21. Press, W., Flannery, B., Teukolsky, S., Vetterling, W.: Numerical Recipes in C: The Art of Scientific Computing. Cambridge University Press, Cambridge (1992)
22. Dow, M.: Explicit inverses of Toeplitz and associated matrices. ANZIAM J. 44, E185–E215 (2003)
23. Borst, S., Boxma, O., Combé, M.: Collection of customers: a correlated M/G/1 queue. Performance Evaluation 20, 47–59 (1992)

Exact Asymptotic Analysis of Closed BCMP Networks with a Common Bottleneck

Jonatha Anselmi and Paolo Cremonesi

Politecnico di Milano,
Dipartimento di Elettronica e Informazione
Via Ponzio 34/5, I-20133 Milan, Italy
anselmi@elet.polimi.it

Abstract. In this paper, we prove the asymptotic equivalence between closed, open and mixed multiclass BCMP queueing networks. Under the assumption that the service demands of a given station, for sufficiently large population sizes, are greater than the ones of all the other stations, we prove that as the total number of customers *semi-proportionally* grows to infinity the underlying Markov chain of a closed network converges to the underlying Markov chain of a suitable open or mixed network. The equivalence theorem lets us extend the state of the art exact asymptotic theory of queueing networks considering a general population growth and including the case in which stations have load-dependent rates of service, and provides a natural technique for the approximate on-line solution of closed networks with large populations. We also show the validity of Little's law in the limit.[1]

1 Introduction

Closed, multiclass queueing networks have been widely used to analytically model the behavior of computer systems with application constraints, e.g. a finite number of threads, (see, e.g., [7,23]). In the frameworks of Internet services, for instance, a closed, multiclass queueing network model for multi-tier Internet services and applications has been recently proposed in [23] where the queues are represented by the different tiers of the application and the customers issue many requests with think times in between. In such frameworks, an immediate, on-line solution of several queueing network models is required. In fact, the high variability of workload characteristics [6,9] and the dynamic environment in which applications and services are deployed require continuous network and system re-configurations in order to meet quality of service constraints and optimize performance metrics. The fulfillment of quality constraints and the optimization of performance metrics can be approached through the formulation of non-linear optimization problems [1,7]. Such problems make a large use of performance indices in their objective functions and constraints, and require the on-line solution of several queueing network models. An other scenario in which an

[1] This work has been supported by the technical and financial support of Neptuny.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 206–220, 2008.

immediate solution is required is in the hierarchical modeling of large networks. The Method of Layers [21], for instance, is based on the iterative solution of several closed networks representing different layers of software servers. In general, a closed network must be solved whenever a flow-equivalent station [8] is introduced in the model. Unfortunately, the computational complexity of existing solution algorithms for closed multiclass models is prohibitively expensive and for real world systems, which are characterized by a *large* number of customers [23], their use is impractical. Moreover, the numerical instabilities characterizing existing algorithms strongly affect the accuracy of the solution [18] and this further motivates the discovery of efficient techniques for the *on-line* and stable solution of closed multiclass queueing networks with large populations.

The asymptotic theory of queueing networks concerns the study of queueing network models when some input parameter is large or grows to infinity, e.g. the total number of customers. Such asymptotic theory can support the above requirements because it is aimed to provide simple closed-form formulas which can be used as accurate estimates of network performance indices [3,4] or also as accurate starting conditions of iterative solution techniques, e.g. [22], which have been shown critical for convergence issues [17]. The asymptotic analysis also provides immediate useful information on the behavior of a network and can be used at design-time to pursue given performance goals. A wide literature has been developed concerning the asymptotic analysis of BCMP networks [5] with load-independent stations (see, e.g., [3,4]), but little appeared about networks with *load-dependent* stations. This type of station is useful to model a service center where its processing speed depends on its queue length and it is used in many applications. For instance, load-dependent stations can represent multiple-server queues or flow-equivalent stations [8] which are used for the hierarchical modeling of large/multitiered networks. Flow-equivalent stations are used to efficiently speed up the evaluation of different alternatives for model input parameters and they are also important for the approximate solution of non-BCMP networks (see, e.g., [8,2]).

McKenna and Mitra [15] analyze multiclass networks with load-dependent stations expliciting the normalizing constant [5] with an integral using the Gamma function and obtain an asymptotic series expansion under the "normal usage" assumption which essentially imposes that no queue is heavily loaded, i.e. most of the customers are in a delay station. Analogously, Mei and Tier [16] provide different expansions assuming the existence of a delay and also of a large number of stations. The asymptotic formula provided for the normalizing constant is approximate and no bounds are provided for the accuracy of their results.

In this paper, we propose a very different approach which lets us prove the exact asymptotic behavior of multiclass BCMP networks extending the results presented in [3] with respect to load-dependent stations and a more general type of population growth that we call *semi-proportional*. This type of growth essentially keeps constant the number of customers of some classes and lets the others proportionally grow to infinity. We analyze the case in which the service demands of a given station, for sufficiently large population sizes, are greater

than the ones of all the other stations. In the real world, this assumption is met for a variety of applications. For instance, the current trend of many Internet applications addresses a multitiered architecture and quite often it happens, e.g. [24], that the application tier is the more overloaded one. This essentially reflects our assumption because each tier usually represents a queue [23]. Our approach is based on the analysis of the Markov chain underlying a queueing network. Exploiting the well-known insensitivity property of BCMP networks, we provide a theorem stating the asymptotic equivalence between closed, open and mixed networks. In particular, we prove that, as the total number of customers grows to infinity, the underlying Markov chain of a closed network is equivalent to the underlying Markov chain of a suitable open or mixed network. Hence, the behavior of a closed network must converge, in the limit, to the behavior of a mixed or open network. We then use this equivalence to efficiently solve closed networks with large population sizes in an approximate manner and to derive exact asymptotic curves for system response time and throughput which capture very well the network behavior even for small population sizes. We also show the validity of Little's law [14] also when the number of customers grows to infinity.

The structure of the paper is as follows. Section 2 introduces the model under investigation and some definitions. Section 3 illustrates our main result proving the asymptotic equivalence between closed, open and mixed BCMP networks under the semi-proportional population growth. In Section 4 we provide asymptotic formulas for the efficient solution of networks with large population sizes as well as the rationale behind Little's law in the limit. Numerical results are given in Section 5 and, finally, Section 6 draws conclusions of our work.

2 Model and Definitions

We consider multiclass BCMP queueing networks [5]. There are M stations and customers are partitioned into R classes. If not otherwise specified, indices r and s will implicitly range from 1 to R and indices i and j from 1 to M indexing, respectively, network classes and stations. $p_{ij,r}$ is the (constant) probability that upon completing service at station i a class-r customer goes to station j. $p_{0i,r}$ and $p_{j0,r}$ are, respectively, the probability that a class-r customer entering from outside visits station i and the probability that a class-r customer leaves the network after completion at station j.

If the network is open, we denote by

- λ_r, the mean overall class-r customers arrival rate from outside,
- $\lambda = \sum_r \lambda_r$, the mean overall arrival rate from outside,
- $\lambda_{0,ir}$, the mean class-r customers arrival rate from outside to station i
- λ_{ir}, the mean class-r customers arrival rate to station i which can be obtained by solving the linear system

$$\lambda_{ir} = \lambda_{0,ir} + \sum\nolimits_j \lambda_{jr} p_{ji,r}, \ \forall i, r. \tag{1}$$

If the network is closed, we denote by

– N_r, the constant number of class-r customers circulating in the network,
– $\mathbf{N} = (N_1, N_2, \ldots, N_R)$, the total population vector,
– $N = \sum_r N_r$, the total number of customers without class distinction.

Let $\mathbf{n} = (\mathbf{n}_1, \ldots, \mathbf{n}_M)$ be a state of the network where $\mathbf{n}_i = (n_{i1}, \ldots, n_{iR})$ denotes the population vector in station i and n_{ir} denotes the number of class-r customers in i. Note that for closed networks $N_r = \sum_i n_{ir}$.

$\pi(\mathbf{n})$ and $\pi_i(\mathbf{n}_i)$ are respectively the network steady-state probabilities [8] of being in state $\mathbf{n}$ and the steady-state probability of having $\mathbf{n}_i$ customers in station i. $\pi_i(n)$ is the steady-state probability of having, in i, n customers without class distinction.

μ_{ir} is the mean class-r *service rate* of station i and the quantity $1/\mu_{ir}$ is the average *service time*. e_{ir} is the mean number of visits of a class-r customer to station i and can be obtained through the following linear system

$$e_{ir} = p_{0i,r} + \sum\nolimits_j e_{jr} p_{ij,r}, \ \forall i, r. \tag{2}$$

In closed networks $p_{0i,r} = 0$ and for each r the previous system has only $M - 1$ independent equations, and its solution is determined up to a multiplicative constant assuming, for instance, $e_{1r} = 1, \ \forall r$ (see, e.g., [8]). $D_{ir} = e_{ir}/\mu_{ir}$ is the mean loading of station i for class-r customers (also called relative utilization) and for a closed network it can be interpreted as the average time spent by a class-r customer at station i during its full execution when using the network alone and visiting station 1 once, i.e. $e_{1r} = 1$.

Let $x_i(n)$ be an arbitrary positive function of the number of customers which visit i. $x_i(n)$ represents the load-dependent rate of service of i when there are n customers in i *relative* to the service rate when $n = 1$, i.e. $x_i(1) = 1$.

Let also $y_{ir}(n)$ be an arbitrary positive function of the number of class-r customers which visits i. $y_{ir}(n)$ represents the load-dependent rate of service of class-r customers in station i when there are n class-r customers in i *relative* to the service rate when there is only one class-r customer, i.e. $y_{ir}(1) = 1$.

We say that station i provides a *type 1* load-dependence if its load-dependence is a function of the total number of customers in i. Analogously, station i provides a *type 2* load-dependence if its load-dependence is a function of only the number of customers in i of a specific class. It is well-known that the model discussed above with stations providing type-1 or type-2 load-dependencies satisfy the BCMP assumptions [5]. We denote by $z_{ir}(n)$ the relative service rate of the load-dependence provided by i, i.e.

$$z_{ir}(n) = \begin{cases} x_i(n) & \text{If station } i \text{ relative service rate is of type 1,} \\ y_{ir}(n) & \text{otherwise.} \end{cases} \tag{3}$$

In this paper we analyze the special case in which there exists a population $\mathbf{N}'$ and a station, indexed in the following by m, such that

$$\lim_{n\to\infty} x_m(n) = \hat{x}_m, \ \frac{D_{mr}}{x_m(N)} > \frac{D_{ir}}{x_i(N)}, \quad i \neq m, \ N > N' \tag{4}$$

if relative service rate of m is of type 1, otherwise

$$\lim_{n\to\infty} y_{mr}(n) = \hat{y}_{mr},\ \frac{D_{mr}}{y_{mr}(N_r)} > \frac{D_{ir}}{y_{ir}(N_r)},\quad i \neq m,\ N_r > N'_r. \tag{5}$$

However, in Section 3 we provide new insights related to the more difficult case in which (4) and (5) are relaxed. No constraints are imposed on the convergence of $x_i(n)$, $i \neq m$, as $n \to \infty$.

Since we evaluate the network when the number of customers grows to infinity, in the following we refer to m as the *bottleneck* of the network because as $N \to \infty$ it is the station with the highest per-class loadings.

X_{ir}, Q_{ir} and C_{ir} respectively represent the *mean* throughput, queue length (number of customers) and response time of class-r customers in station i. We assume that the previous quantities refer to global quantities if station or class subscript is removed. For instance, X_r represents the *system* throughput for class-r customers while X represents the system throughput due to *all* classes.

3 Asymptotic Analysis

In this section we prove that a closed BCMP network is equivalent, as the number of customers grows to infinity, to a suitable open or mixed BCMP network with the bottleneck station removed. In Section 3.1 we discuss the underlying Markov chain of the models under investigation, in Section 3.2 we introduce our notion of semi-proportional population growth and in Section 3.3 we illustrate the asymptotic equivalence result.

3.1 Underlying Markov Chain

The underlying Markov chain of a BCMP network model has been specified in [5] where the states of the model take into account the service discipline adopted by stations (FCFS, PS, LCFS-PR or IS) and the distribution of their service times. However, in that work a simpler representation of network states is given aggregating some states. An aggregate state of the model is given by *only considering the number of per-class customers in the stations*, i.e. matrix $\mathbf{n}$. This representation reduces the number of possible states as well as the complexity of computational algorithms for the model solution. It is well-known (insensitivity property) that for such states and within the BCMP assumptions, any service time distributions yield, in terms of average performance indices, the same results of exponential service time distributions (see [5,19]). Hence, given a BCMP network we can assume that all stations do have an exponential service time distribution and describe the network behavior with a Markov chain built by considering only aggregate states without introducing any degree of approximation in terms of average performance indices. We henceforth consider this equivalent representation of the network because our focus is on the averages of performance indices. Thus, we introduce the following definition.

Definition 1. *Two Markov chains are called equivalent if there exists a bijection f between their state spaces such that each pair of states $(\mathbf{n}, f(\mathbf{n}))$ guarantees that the out-going transition rate from $\mathbf{n}$ to $\mathbf{n}'$ is equal to the out-going transition rate from $f(\mathbf{n})$ to $f(\mathbf{n}')$, for all states $\mathbf{n}'$.*

3.2 Semi-proportional Population Growth

As described in [3], let $\beta \equiv \beta(\mathbf{N}) = [\beta_1, \ldots, \beta_R]$ be the *population mix* vector corresponding to population $\mathbf{N}$ whose components are such that

$$\beta_r = \frac{N_r}{N}, \quad \sum\nolimits_r \beta_r = 1. \tag{6}$$

We study the asymptotic behavior of closed networks letting population N grow to infinity keeping constant the number of customers of some classes and letting the others *proportionally* grow to infinity. For simplicity and without loss of generality, in the following we assume that only classes $r \leq R' \leq R$ are allowed to grow to infinity and we let them proportionally grow keeping constant their mix

$$\beta'_r = \frac{N_r}{\sum_{s=1}^{R'} N_s}, \ \forall r \leq R'. \tag{7}$$

We call this type of growth *semi-proportional*. Note that the semi-proportional growth is a generalization of both proportional and *unbalanced* growths proposed in [3] where the authors respectively let R and exactly one class of customers grow to infinity.

In the following, we refer to class-r customers as *heavy* if and only if $r \leq R'$. It is easy to see that under this type of growth, we have

$$\lim_{N \to \infty} \beta(\mathbf{N}) = (\beta'_1, \ldots, \beta'_{R'}, 0, \ldots, 0) \tag{8}$$

which means, in general, that the proportion of network utilization due to non-heavy customers (in the limit) approaches zero.

3.3 Asymptotic Equivalence

Assume that $e_{mr} > 0, \ \forall r$. In this case, it is easy to see that the asymptotic behavior of the network can be described in terms of heavy customers only, i.e. considering a model in which non-heavy customers classes are removed. In fact, since the queue length of m, in the limit, grows to infinity and the number of non-heavy customers remains finite, eventually we have that stations $i \neq m$ will be populated by heavy customers only because non-heavy customers will eventually be *lost* in the queue of m. This means that the network asymptotic behavior, in this case, can be analyzed by considering heavy customers only. Thus, in the following we consider the case in which non-heavy customers do not visit station m, i.e.

$$e_{mr} = 0, \ \forall r > R'. \tag{9}$$

This implies that assumptions (4) and (5) now hold only for heavy customers classes. We now introduce our *asymptotic equivalence* theorem.

Theorem 1. *Given a population* $\mathbf{N}$*, as* N *semi-proportionally grows to infinity, the underlying Markov chain of a closed BCMP network is equivalent to the underlying Markov chain of an ergodic mixed BCMP network formed by the same set of stations with station* m *removed, the same routing probabilities* $p_{ij,r}$*, and,* $\forall r \leq R'$*, overall arrival rates* $\lambda_{0j,r} = \mu_{mr}\hat{z}_{mr}\beta'_r p_{mj,r}$ *and* $p_{i0,r} = p_{im,r}, \quad p_{0j,r} = p_{mj,r}$*.*

Proof. Given in Appendix A.

Hence, performance indices of station $i \neq m$ of a closed model converge, in the limit, to the performance indices of the associated station in the mixed model. Note that the equivalent network defined according to Theorem 1 is ergodic since the stability condition is verified by assumptions (4) and (5) (see, e.g., [12,13]). Hence, all queue lengths Q_{ir}, $i \neq m$, have a finite limit and, as a consequence, both Q_{mr} and C_{mr} grow to infinity proportionally to N_r. This also implies that the proportion of network utilization due to non-heavy customers is non-null even though their mix, in the limit, approaches zero.

The asymptotic throughput for heavy customers of class r is then given by

$$X_r(\beta) \equiv \lim_{N\to\infty} X_r(\mathbf{N}) = \sum_j \lambda_{0j,r} = \mu_{mr}\hat{z}_{mr}\beta'_r(1 - p_{mm,r}) \tag{10}$$

while for non-heavy customers the throughput is given by applying existing solution algorithms for mixed networks (see, e.g., [8]) on a reduced number of classes, i.e. $R - R'$. The following corollary is straightforward.

Corollary 1. *Given a population* $\mathbf{N}$ *and assuming* $R' = R$*, as* N *grows to infinity the underlying Markov chain of a closed BCMP network is equivalent to the underlying Markov chain of an ergodic open BCMP network formed by the same set of stations with station* m *removed, the same routing probabilities* $p_{ij,r}$*, arrival rates* $\lambda_{0j,r} = \mu_{mr}\hat{z}_{mr}\beta_r p_{mj,r}$ *and* $p_{i0,r} = p_{im,r}, \quad p_{0j,r} = p_{mj,r}$*.*

In this degenerate case, note that N grows *proportionally* to infinity because each component of $\beta(\mathbf{N})$ remains constant. According to (9) and assumptions (4) and (5), note also that in this case we must have $e_{mr} > 0$, $\forall r$, otherwise we would have multiple bottlenecks. This corollary is important because extends the equivalence discussed in [3] proving that the asymptotic equivalence between open and closed networks holds also for load-dependent stations.

It is easy to see that both Theorem 1 and Corollary 1 hold whenever the situation of a common bottleneck can appear: in fact, considering, for the sake of simplicity, the latter case in which $R' = R$, Corollary 1 can hold also if assumption (4) is not met (see the proof of Theorem 1) but a common bottleneck exists for all customer classes. Let us consider the case in which (4) is relaxed, i.e. in the case in which there exists no station m such that $D_{mr}/x_m(n) > D_{ir}/x_i(n)$, $\forall r$, $i \neq m$ and $\forall n \geq N'$ for some N'. In this setting and assuming load-independent stations, in [4] it has been conjectured the existence of some βs which, in the limit, yield the saturation of exactly one station, i.e. a common bottleneck. Given an input population $\mathbf{N}$, the existence of such situation, for

some common bottleneck m, can be quickly checked by applying Corollary 1. In fact, it sufficies to apply the corollary with respect to all stations and for each of them checking whether or not the equivalent open network is ergodic. If an ergodic network is found for some station m, then m is the common bottleneck and the network asymptotic behavior can be described by applying the analysis discussed in the next section. However, because of bottleneck shifting phenomena [4], in the setting discussed above we do not have a common bottleneck *for each possible* β as, instead, assumption (4) implies. Thus, we leave as future work the analysis of this more difficult setting in which (4) is relaxed.

4 Efficient Analysis of Networks with Large Populations

The computational complexity of exact solution algorithms for closed BCMP networks with load-dependent stations is prohibitively expensive. For instance, the complexity of the standard (load-dependent) MVA [18] is $O(RNM\prod_r N_r)$ for time and $O(RM\prod_r N_r)$ for space. Hence, for multiclass networks with large population sizes, storage and time requirements do not meet the issues discussed in the introduction. Moreover, current algorithms rely on numerical instabilities which strongly affect the accuracy of the solution and eventually yield unfeasible solutions, e.g. negative throughputs [18].

In the following, we exploit the equivalence results presented in Section 3 in order to efficiently solve networks with large population sizes. In Section 4.1 we assume that all customers visit the bottleneck station m while in Section 4.2 we analyze the case in which $e_{mr} = 0$ for some r. Section 4.3 shows exact asymptotic curves for system response time and throughput, and explains the rationale behind Little's law in the limit.

4.1 Open Network Approximation

Let us assume that $e_{mr} > 0,\ \forall r$, i.e. that all customers visit the bottleneck. Corollary 1 implies that a network with load-dependent stations and with sufficiently large per-class populations can be approximately solved exploiting solution algorithms for open BCMP networks which, in general, are extremely efficient. In fact, for a number of load-dependencies of practical interest there exist simple closed-form formulas for performance indices.

Hence, given a closed network with large population $\mathbf{N}$, we apply Corollary 1 removing the bottleneck station m and then we analyze the resulting open network. It is known, e.g. [5,13], that for computational purposes a better equivalent representation of an open network is obtained by considering the overall arrival rate

$$\lambda = \sum_r \lambda_r = \sum_r \sum_i \lambda_{0i,r} = \sum_r \mu_{mr}\hat{z}_{mr}\beta_r(1 - p_{mm,r}) \tag{11}$$

and making the replacement $p_{0i,r} \leftarrow p_{0i,r}\lambda_{0i,r}/\lambda_{0i}$. With this transformation and considering as aggregate state the total number of customers in each station

without class distinctions, i.e. vector $(n_1, \ldots, n_M)$, it is well-known that [5]

$$\pi(n_1, \ldots, n_M) = \prod_i \pi_i(n_i) = \prod_i \frac{\rho_i^{n_i}}{\prod_{t=1}^{n_i} x_i(t)} \; G_i \tag{12}$$

where

$$\rho_i = \lambda \sum_r \frac{e_{ir}}{\mu_{ir}}, \quad G_i^{-1} = \sum_{n_i=0}^{\infty} \frac{\rho_i^{n_i}}{\prod_{t=1}^{n_i} x_i(t)}. \tag{13}$$

Note that the mean number of visits e_{ir} in (13) can now be exactly computed through linear system (2). Since previous formulas state that each queue $i \neq m$ can be analyzed independently in isolation as a singleclass queue, they can be efficiently solved by applying existing literature related to the singleclass load-dependent queue (note that many load-dependencies yield closed-form solutions for performance indices, e.g. multiple-servers, Heffes stations [11], limited load-dependent stations [13]). Bottleneck queue-length Q_m is then simply given by $Q_m = N - \sum_{i \neq m} Q_i$, network throughput can be approximated by

$$X(\mathbf{N}) = \sum_r X_r(\mathbf{N}) = \sum_r \mu_{mr} x_m(Q_m) \beta_r (1 - p_{mm,r}) \tag{14}$$

which is clearly asymptotically correct and network response time is obtained by applying Little's law [14] accordingly.

4.2 Mixed Network Approximation

We now consider the case in which some customers do not visit the bottleneck m, i.e. assumption (9). If population $\mathbf{N}$ is large, then we can approximate the closed network with a mixed network as defined in Theorem 1 assuming that only the number of heavy customers grows to infinity. The mixed network is thus solved through standard algorithms, e.g. [8,13], and Theorem 1 tells us that as the number of heavy customers increases, performance indices converge to exact values for each finite population of non-heavy customers. Observe that the asymptotic solution is exact even if the number of non-heavy customers is non-large and, thus, we can perform the approximation even if the number of non-heavy customers is small.

Let us consider the case of load-independent queues. It is well-known that in order to compute performance indices of non-heavy customers, we have to solve a $(R - R')$-class closed queueing network with modified loadings

$$D_{ir} \leftarrow \frac{D_{ir}}{1 - \sum_{s=1}^{R'} \lambda_s D_{is}}, \; \forall r > R' \tag{15}$$

which take into account the uncontrolled effects of the open classes. Formula (15) can be generalized to load-dependent stations (see, e.g., [8,13]). In this case, the computational complexity we need to pay is higher than the open network case because it is implied the solution of a closed network. However, the solution of this closed network is characterized by a fewer number of classes, i.e. $R - R'$, and this can drastically reduce, in practice, the complexity of the analysis because the majority of existing exact algorithms is exponential in the number of classes.

4.3 Response Time Asymptotic Curves

Let us first assume that station m is load-independent and that $R' = R$. Since non-bottleneck stations behave, in the limit, as in an open network, exploiting the classic MVA recursive formula [20] we have

$$\begin{aligned}\lim_{N\to\infty} C(\mathbf{N}) &= \sum_{i\neq m} Q_i/\lambda + \lim_{N\to\infty} \textstyle\sum_r D_{mr}(1 + Q_m(\mathbf{N} - \mathbf{e}_r))\lambda_r/\lambda \\ &= \left[\sum_{i\neq m} Q_i + \textstyle\sum_r \beta_r \lim_{N\to\infty} (N - \sum_{i\neq m} Q_i)\right] N/\lambda = \lim_{N\to\infty} N/\lambda \end{aligned} \tag{16}$$

where $\lambda = \sum_r \lambda_r$, $\lambda_r = \beta_r / D_{mr}$ and, thus, we obtain that Little's law holds also in the limit. Note that the original formulation of Little's law assumes that N is finite [14]. Hence, the following asymptotic curve for system response time holds

$$C(\mathbf{N}) = N/\lambda. \tag{17}$$

To take into account the load-dependence of m, in (17) we make the replacement $D_{mr} \leftarrow D_{mr}/x_m(N - \sum_{i\neq m} Q_i)$ which yields, according to (4), an asymptotically correct formula (recall that in the limit $Q_m \to \infty$ and $x_m(Q_m) < \infty$).

The response time asymptotic curve obtained assuming $R = R'$ can be analogously written also when $R' < R$. For instance, assuming load-independent stations, we can write [8]

$$\lim_{N\to\infty} C_{ir}(\mathbf{N}) = D_{ir} \frac{1 + \sum_{s=R'+1}^{R} Q_{is}(N_{R'+1}, \ldots, N_R)}{1 - \sum_{s=1}^{R'} \lambda_s D_{is}}, \tag{18}$$

$i \neq m,\ r \leq R'$. In this case, however, the computational effort needed by the resulting asymptotic curves is higher than the case in which $R' = R$ because it requires the solution of a closed model with $R - R'$ classes. The response time asymptotic curve is then given by

$$C(\mathbf{N}) = \frac{N}{\sum_{r=1}^{R'} \lambda_r + \sum_{R'=1}^{R} X_r(N_{R'+1}, \ldots, N_R)} \tag{19}$$

where $X_r(N_{R'+1}, \ldots, N_R)$ denotes the system throughput of non-heavy customers of class r.

5 Numerical Results

In this section we illustrate some numerical results in order to show the accuracy of our asymptotic analysis. We remark that real systems do have thousands of customers and our analysis is mainly addressed to solve models with large population sizes. However, we were unable to quantify an accurate theoretical bound on the minimum number of customers able to guarantee a given precision threshold. Intuitively, if non-bottleneck loadings are low with respect to the bottleneck ones then this bound is low because non-bottleneck queue lengths

quickly converge to their asymptotic values, i.e. most of the jobs quickly tends to accumulate in the bottleneck. Conversely, if non-bottleneck loadings are high then this bound increases. Hence, we take into account this intuitive consideration by numerically evaluating the accuracy of the asymptotic curves discussed in previous section for different values of the ratio

$$\theta = \max_{r,i \neq m} \lim_{N \to \infty} \frac{D_{ir}}{D_{mr}} \frac{x_m(N)}{x_i(N)}. \tag{20}$$

Thus, as θ increases the highest non-bottleneck loading approaches to the one of the bottleneck. Note that $0 < \theta < 1$. Within the c-th experiment, let

$$E_c(C) = |C^{\mathrm{App},c} - C^{\mathrm{Ex},c}| \frac{1}{C^{\mathrm{Ex},c}} 100\% \tag{21}$$

be the percentage relative error of the approximate estimate $C^{\mathrm{App},c}$ of system response time with respect to its exact value $C^{\mathrm{Ex},c}$. As output measure we choose system response time because it is the less robust performance measure.

We first evaluate the accuracy of our analysis assuming a proportional population growth, i.e. $R' = R$. Let $\theta \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. For each θ, we randomly generate 500 two-class models choosing M between 2 and 10, non-bottleneck loadings in range $[0, 50]$ and proportionally increase N from 20 to 80 with step 20 where all N_r are randomly chosen in order to sum N. We did not consider models with three classes or higher population sizes because of the prohibitively expensive space and time requirements needed by our implementation of the (load-dependent) MVA supported by the multiprecision library [10] which overcomes numerical instabilities typically arisen by load-dependent stations. We assume that all stations, including the bottleneck, are (type-1) load-dependent. The load-dependent stations we consider are multiple-servers, Heffes [11], i.e. $x_i(n) = n/(n + \alpha_i)$, $\alpha_i > 0$, and step-wise stations, i.e. $x_i(n) = x_{i1}$ if $n < \alpha_i$ otherwise $x_i(n) = x_{i2}$. Within such load-dependencies, it is well-known that there exist simple closed-form formulas for the asymptotic queue-length: for instance, for Heffes stations $Q_i = (1 + \alpha_i)\rho_i/(1 - \rho_i)$. For multiple-server stations we assume a maximum of 40 servers, for Heffes and the step-wise stations we assume $\alpha \leq 40$. The loadings of the bottleneck station are generated in order to satisfy (4) and (20). In particular, to better stress the accuracy of our analysis, we ensure that condition (4) is verified for non-zero N's. This essentially slows the convergence speed of our formulas because m, for small populations, is not the most overloaded station.

In each point of the left-hand side graphs of Figure 1, we show the values of $E(C) = \sum_{c=1}^{500} E_c(C)/500$ for different population sizes and what we notice is that the value of θ does affect the accuracy of our asymptotic estimates which in any case quickly converge to exact values. Our asymptotic estimates capture very well the network dynamics even for small population sizes. We remark that in real-world systems $N = 80$ is small and for this value we obtain $E(C) = 4.5\%$. Thus, formula (17) can be adopted for the accurate, on-line and stable solution of models which currently would require a prohibitive amount of memory and time

arising, at the same time, numerical instabilities. The high errors obtained for $N = 20$ are due to the fact that for such too small populations our analysis tends to overestimate the queue length of non-bottleneck stations underestimating the one of the bottleneck m. We proved that this problem vanishes as N increases.

We now evaluate the accuracy of our analysis assuming a semi-proportional population growth. We randomly generate, for each θ, 500 three-class models choosing $R' = 1$ or $R' = 2$ with equal probability. Since the space and time requirements characterizing the (load-dependent) MVA let us solve only models with small population sizes, for this case we assume that all queues are load-independent. This lets us adopt a more efficient implementation of the MVA and evaluate the accuracy of our analysis exploring models with larger populations. Thus, we semi-proportionally increase N from 50 to 600 with step 50. The number of non-heavy customers is randomly chosen between 5 and 200. In each point of the right-hand side graphs of Figure 1, we show the values of $E(C)$ for different θs. We observe again the high accuracy and the fast convergence to exact values of formula (19). Note that for $N = 600$, we have $E(C) = 0.17\%$.

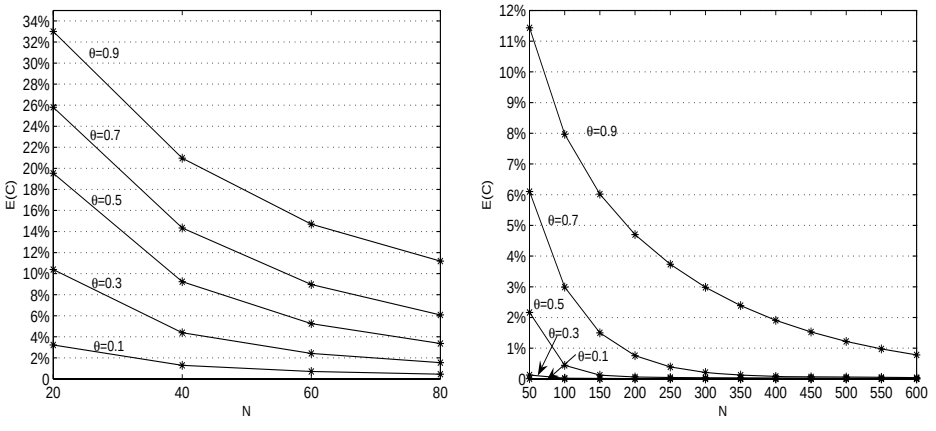


Fig. 1. $E(C)$ assuming a proportional (semi-proportional on the right) growth and load-dependent (respectively load-independent) stations

6 Conclusions

In this paper we proved the exact asymptotic behavior of closed BCMP networks with load-dependent stations under the semi-proportional population growth and assuming the existence of a common bottleneck for sufficiently large population sizes. We proved the asymptotic equivalence between closed and open or mixed networks showing that the performance indices of a closed model converge (in the limit) to the performance indices of a suitable open or mixed model. On the basis of this result, we provided asymptotically correct formulas for the efficient solution of closed networks with large population sizes as well as the behavior of the well-known Little's law in the limit. Numerical results showed

that the solution of closed multiclass network with load-dependent stations can be obtained *on-line* exploiting simple formulas which provide a good accuracy. Such formulas can be used in many frameworks: they are able to pursue the dynamic environment in which Internet applications are deployed, can be embedded in optimization problems which need the solution of several models in a short time and can be adopted to characterize flow-equivalent servers which are usually used in the hierarchical modeling of large networks.

References

1. Almeida, J., Almeida, V., Ardagna, D., Francalanci, C., Trubian, M.: Resource management in the autonomic service-oriented architecture. In: IEEE ICAC, pp. 84–92. ACM Press, New York (2006)
2. Anselmi, J., Casale, G., Cremonesi, P.: Approximate solution of multiclass queueing networks with region constraints. In: MASCOTS 2007, Istanbul, Turkey, IEEE Computer Society, Los Alamitos (2007)
3. Balbo, G., Serazzi, G.: Asymptotic analysis of multiclass closed queueing networks: Common bottlenecks. Perf. Eval. 26(1), 51–72 (1996)
4. Balbo, G., Serazzi, G.: Asymptotic analysis of multiclass closed queueing networks: Multiple bottlenecks. Perf. Eval. 30(3), 115–152 (1997)
5. Baskett, F., Chandy, K., Muntz, R., Palacios, F.: Open, closed, and mixed networks of queues with different classes of customers. J. ACM 22(2), 248–260 (1975)
6. Bennani, M.N., Menasce, D.A.: Assessing the robustness of self-managing computer systems under highly variable workloads. In: ICAC 2004, Washington, DC, USA, pp. 62–69. IEEE Computer Society Press, Los Alamitos (2004)
7. Bennani, M.N., Menasce, D.A.: Resource allocation for autonomic data centers using analytic performance models. In: ICAC, Washington, DC, USA, pp. 229–240. IEEE Computer Society, Los Alamitos (2005)
8. Bolch, G., Greiner, S., de Meer, H., Trivedi, K.S.: Queueing Networks and Markov Chains. Wiley-Interscience, Chichester (2005)
9. Chase, J.S., Anderson, D.C., Thakar, P.N., Vahdat, A.M., Doyle, R.P.: Managing energy and server resources in hosting centers. In: SOSP 2001 Proc., pp. 103–116 (2001)
10. Granlund, T.: Gnu mp: The gnu multiple precision arithmetic library (2000), `http://www.swox.se/`
11. Heffes, H.: Moment formulae for a class of mixed multi-job-type queueing networks. Bell System Technical Journal 61,5
12. Kant, K.: Introduction to Computer System Performance Evaluation. McGraw-Hill, New York (1992)
13. Lavenberg, S.: Computer Performance Modelling Handbook. In: Lavenberg, S.S. (ed.), Academic Press, New York (1983)
14. Little, J.D.C.: A proof of the queueing formula $l = \lambda w$. Operations Research 9, 383–387 (1961)
15. McKenna, J., Mitra, D.: Asymptotic expansions for closed markovian networks with state-dependent service rates. J. ACM 33(3), 568–592 (1986)
16. Mei, J.-D., Tier, C.: Asymptotic approximations for a queueing network with multiple classes. SIAM Journal on Applied Mathematics 54(4), 1147–1180 (1994)

17. Pattipati, K.R., Kostreva, M.M., Teele, J.L.: Approximate mean value analysis algorithms for queuing networks: existence, uniqueness, and convergence results. J. ACM 37(3), 643–673 (1990)
18. Reiser, M.: Mean-value analysis and convolution method for queue-dependent servers in closed queueing networks. Perf. Eval. 1, 7–18 (1981)
19. Reiser, M., Kobayashi, H.: Queueing networks with multiple closed chains: Theory and computational algorithms. IBM J. Res. Dev. 19(3), 283–294 (1975)
20. Reiser, M., Lavenberg, S.S.: Mean-value analysis of closed multichain queueing networks. Journal of the ACM 27(2), 313–322 (1980)
21. Rolia, J.A., Sevcik, K.C.: The method of layers. IEEE Trans. Softw. Eng. 21(8), 689–700 (1995)
22. Schweitzer, P.: Approximate analysis of multi-class closed networks of queues. In: Proceedings of the Int. Conf. on Stochastic Control and Optimization (1979)
23. Urgaonkar, B., Pacifici, G., Shenoy, P., Spreitzer, M., Tantawi, A.: Analytic modeling of multitier internet applications. ACM Trans. Web 1(1), 2 (2007)
24. Villela, D., Pradhan, P., Rubenstein, D.: Provisioning servers in the application tier for e-commerce systems. ACM Trans. Inter. Tech. 7(1), 7 (2007)

A Proof of Theorem 1

For simplicity, we assume that $p_{mm,r} = 0$, $\forall r$. Consider a closed BCMP network with population $\mathbf{N}$ and the mixed network defined in the theorem. We first analyze the case in which the mixed network has population size constraints with lost arrivals, i.e. inside the mixed network we impose that the class-r *heavy* customers cannot exceed N_r units. For each state of the network $\mathbf{n}$, if $\sum_i n_{ir} = N_r$, $r \leq R'$, then a class-r customer which attempts to enter the network is dropped and lost, otherwise it is accepted. Thus, for each finite N, there exists a non-null throughput of class-r dropped jobs. We show that the underlying Markov chains of the closed and the constrained mixed network are equivalent if we ignore the transition rates related to departures from m and external arrivals. Then we let population N grow to infinity keeping constant mix $\beta(N_1, \ldots, N_{R'})$ in order to show that the transition rates related to departures from m asymptotically converge to the transition rates related to departures of the mixed (unconstrained) network. Let $\Omega(\mathbf{N}) = \{\mathbf{n}^* \mid \sum_i n^*_{ir} = N_r, \forall r\}$ be the set of network states $\mathbf{n}^*$ of the closed network. Let $\Psi(\mathbf{N})$ be the set of network states $\mathbf{n}$ of the respective mixed network with station m removed and population size constraints $\mathbf{N}$, i.e.

$$\Psi(\mathbf{N}) = \left\{\mathbf{n} \mid \sum_{i \neq m} n_{ir} \leq N_r,\ r \leq R' \wedge \sum_{i \neq m} n_{ir} = N_r,\ r > R'\right\}, \qquad (22)$$

where to keep notation simple, we assume that $\mathbf{n}$ is a $M \cdot R$ dimensional matrix in which $n_{mr} = 0$, $\forall r$. Now we provide a bijection $f : \Psi(\mathbf{N}) \to \Omega(\mathbf{N})$, and show that each pair of states $(\mathbf{n}, \mathbf{n}^* = f(\mathbf{n}))$ provides the same in-going and out-going transition rates if we ignore the transition rates related to departures from the bottleneck station m in the closed model and departures from the network in the open model. The bijection f associates state $\mathbf{n}$ to state $\mathbf{n}^*$ as follows

$$\begin{aligned} n^*_{ir} &= n_{ir} && i \neq m,\ \forall r \\ n^*_{mr} &= N_r - \textstyle\sum_{i \neq m} n_{ir} && \forall r. \end{aligned} \qquad (23)$$

Within assumption (9), it is easy to see that $n^*_{mr} = 0$, $r > R'$. We define

$$\hat{\mu}_{ir} \equiv \hat{\mu}_{ir}(n_i) \equiv \begin{cases} \mu_{ir} x_i(n_i) & \text{If station } i \text{ has a type-1 load-dependence} \\ \mu_{ir} y_{ir}(n_{ir}) & \text{If station } i \text{ has a type-2 load-dependence} \end{cases} \tag{24}$$

as the *effective* service rate of station i when n_i customers visit i. We denote by $\mathbf{e}_{mr}$ the M-row and R-column matrix with the m-th item of the r-th column set to one and all the others set to zero. We also define $\delta(x) = 1$ if $x \neq 0$, otherwise 0. We now prove the equivalence between both Markov chains. The transition rate from state $\mathbf{n}$ to state $\mathbf{n} - \mathbf{e}_{ir} + \mathbf{e}_{jr}$ due to a class-r customer departure from i to j, $i \neq m,\ j \neq m$, is

$$p_{ij,r} \hat{\mu}_{ir} \delta(n_{ir}) n_{ir} / n_i \tag{25}$$

and analogously from state $\mathbf{n}^*$ to state $\mathbf{n}^* - \mathbf{e}_{ir} + \mathbf{e}_{jr}$ is

$$p_{ij,r} \hat{\mu}_{ir} \delta(n^*_{ir}) n^*_{ir} / n^*_i. \tag{26}$$

Bijection (23) ensures that both transition rates are equal. The transition rate from state $\mathbf{n}$ to $\mathbf{n} - \mathbf{e}_{ir}$ due to a class-r customer leaving the network from i is

$$p_{i0,r} \hat{\mu}_{ir} \delta(n_{ir}) n_{ir} / n_i \tag{27}$$

where $p_{i0,r} = p_{im,r}$, and analogously from state $\mathbf{n}^*$ to state $\mathbf{n}^* - \mathbf{e}_{ir} + \mathbf{e}_{mr}$ is

$$p_{im,r} \hat{\mu}_{ir} \delta(n^*_{ir}) n^*_{ir} / n^*_i. \tag{28}$$

Bijection (23) ensures that both transition rates are equal. The transition rate from state $\mathbf{n}$ to $\mathbf{n} + \mathbf{e}_{jr}$ due to an external class-r customer arrival to i is

$$p_{0j,r} \lambda_r \delta\left(N_r - \textstyle\sum_i n_{ir}\right) \tag{29}$$

where $p_{0j,r} = p_{mj,r}$, $\lambda_r = \hat{\mu}_{mr} \beta'_r$, and analogously from $\mathbf{n}^*$ to $\mathbf{n}^* + \mathbf{e}_{jr} - \mathbf{e}_{mr}$ is

$$p_{mj,r} \hat{\mu}_{mr} \delta(n^*_{mr}) n^*_{mr} / n^*_m. \tag{30}$$

Transition rates (29) and (30) are not equal since, for instance, $\beta'_r \neq n^*_{mr}/n^*_m$. We now let N semi-proportionally grow to infinity. Bijection (23) still holds by induction and for heavy customers we have

$$\lim_{N \to \infty} \frac{n^*_{mr}}{n^*_m} = \lim_{N \to \infty} \frac{N_r - \sum_i n_{ir}}{\sum_s \left(N_s - \sum_i n_{is}\right)} = \beta'_r \tag{31}$$

and $\delta\left(N_r - \sum_i n_{ir}\right) = \delta\left(n^*_{mr}\right) = 1$. In the limit we have that the ergodicity condition is satisfied by assumptions (4) and (5) (recall that the stability of a mixed network is unaffected by the presence of closed classes [19]) and this means that $Q_i < \infty$, $\forall i \neq m$, and no job can be dropped, i.e. the population size constraints are removed. As $N \to \infty$, states $\mathbf{n}$ and $\mathbf{n}^*$ have the same outgoing and in-going transition rates. Since the pair of states $(\mathbf{n}, \mathbf{n}^*)$ is generic, we conclude that both underlying Markov chains are equivalent and that the closed network converges to the mixed (ergodic) network defined in the theorem.

Multiclass G-Networks of Processor Sharing Queues with Resets

J.-M. Fourneau[1,2]

[1] INRIA Project MESCAL, LIG, UJF & CNRS,
Montbonnot, France
[2] PRiSM, Université de Versailles-Saint-Quentin & CNRS & UniverSud,
Versailles, France

Abstract. We consider an open queueing network of generalized queues with several class of customers and one class of signal. Each queue has an infinite capacity and one server. The service time is exponential. The service discipline is Processor Sharing. After its service completion a customer moves to another queue and may become a signal. When the signal enters a non empty queue it vanishes while it resets the queue when it enters an empty queue. We prove that the steady state distribution for such a network of queues has a product form solution. To the best of our knowledge it is the first multiclass network of generalized queues and resets with product form solution.

1 Introduction

In this paper we present a novel extension of G-networks of queues with several classes of customers and one class of signal. We introduce a new type of *"reset" customers* recently considered by Gelenbe and Fourneau in [17]. We prove that a new extension of these networks with a different type of resets still has a product form for its stationary regime. To the best of our knowledge, it is the first result on multiclass network with product form in the context of queues with customers and resets.

The first type of signal introduced by Gelenbe was described as a negative customer [9]. A negative customer deletes a positive customer in a queue at its arrival if it is possible. Positive customers are usual customers in classical queueing networks. A negative customer is never queued. Under typical assumptions (Poisson arrival for both types of customers, exponential service time for positive customers, Markovian routing, independence, open topology) Gelenbe proved that such a network has a product form solution for its steady-state behavior. The flow equation for these networks exhibits some uncommon properties: it is neither linear as in closed queueing networks nor contracting as in open queueing networks like Jackson networks [22]. Therefore the existence of a solution had to be proved [10] and a numerical algorithm had to be developed [3]. Network of positive and negative customers were introduced to model neural networks where neurones exchange inhibitory and exciting signals [8,13].

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 221–233, 2008.

Traditionally, queueing networks model systems with customers which circulate among a finite set of servers, waiting for service, obtaining service at each of the servers and moving from server to server. Such systems are either closed (and do not receive customers from the outside world), or open (with customers arriving from the outside world and then leaving when they are done with their work). Queueing models typically do not have provisions for some customers being used to eliminate other customers, or to redirect other customers among the queues. In other words, customers in traditional queueing networks cannot exert direct control on other customers. G-network models overcome some of the limitations of conventional queueing network models and still preserve the computationally attractive product form property of some Markovian queueing networks. They contain unusual customers such as negative customers which eliminate normal customers, catastrophes which flush all the customers out of a queue and triggers which move other customers from one queue to another [11,12]. Multiple class versions of these models have also been derived [4,5,15] to generalize BCMP theorem [2]. Currently there are several hundred references devoted to the subject, and a recent journal special issue [14] and a book [24] provide insight into some of the research issues, developments and applications in the area of networks of queues with customers and signals. G-networks have also been the source of new interesting questions and techniques (see for instance [21]). Note however that the composition of reversed process studied in [21] has been extended to networks of multiclass queues.

Most of the current applications of G-networks are currently in the field of random neural networks [8,13] because the product form solution leads to an efficient learning procedure. For instance Rubino and his coauthors have used random neural networks based on G-networks with positive and negative customers to obtain a subjective evaluation (i.e. based on users satisfaction) of Quality of Service for streaming videos on packet networks [23,25]. G-networks and Random Neural Networks were also used to build the learning process for Cognitive Packet Networks [16,19]. G-networks have been proposed to model virus propagation [20], agents [26] and work deletion in a queue [1].

In a former study [17], we have introduced a new type of signals in an open network of queues. These signals are denoted as resets and they do not delete customers. More precisely a signal has the following behavior. When it arrives at a queue, it has one of two following effects according to the size of the queue:

- If the queue is non-empty, the reset deletes a customer or triggers a customer movement.
- If the queue is empty then the queue is reset to a random queue length whose distribution is somehow related to the stationary distribution at that queue.

The transitions associated with resets are quite different from the transitions in the model of a network with positive and negative customers. But both systems have the same flow equation to define the loads. This equivalent flow equation leads to interesting questions on the nature of flow equations and how we can transform at the same time Markov chains and flow equation to keep product form solution [18,6]. More recently we have proved in [7] that a closed queueing

network with resets has a product form solution for a new type of resets. All the results proved since the seminal papers by Gelenbe in the early nineties [9,11,12] deal with open networks of queues. This assumption is necessary because the signals considered in these papers always delete at least one customer. Thus in a closed network such a signal leads to an empty network of queues which is an absorbing state. Thus we do not have product form for signals which only delete customers. However the three theorems in [7] show that the steady state solution has product form when the resets satisfy some constraints.

Here we generalize the G-network of queues with resets in another direction. We consider a network with several classes of customers, one class of signal, an open topology, processor sharing queues. A signal deletes a customer when it enters the queue if there is any. But if the queue is empty, it makes the queue jump to any arbitrary state (except one state). This last action is denoted as a reset.

The paper is as follows. In Section 2, we introduce the model and we prove that this model has product form solution for the steady-state distribution under the usual assumptions (Poisson arrivals, exponential service times for ordinary customers, Markovian customer movement). For the sake of readability the details of the proof are postponed into an appendix. In Section 3 we study the existence of a solution to the fixed point system which generalizes the usual flow equation.

2 A Network of Generalized Queues with Resets and Its Product Form Solution

We investigate generalized networks with an arbitrary number N of Processor Sharing queues. We consider K classes of positive customers and only one class of signal.

State: The state of the queueing network is represented by the vector $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)$, where component $\boldsymbol{x}_i$ denotes the state of queue i. As usual with multiple class PS queues, the state of queue i is given by vector $(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(p)})$, where component $x_i^{(k)}$ is the number of customers of class k in queue i. Furthermore, we note $||\boldsymbol{x}_i||$ the total number of customers in queue i. We denote by $(\boldsymbol{x}_i + e^{(k)})$ (resp. $(\boldsymbol{x}_i - e^{(k)})$) the state of queue i obtained by adding (resp. suppressing) one customer of class k. Similarly, we denote by $(\boldsymbol{x} + e_i^{(k)})$ (resp. $(\boldsymbol{x} + e_i^{(k)})$) the network state obtained by adding (resp. suppressing) a single customer of class k in queue i. Vector $(\boldsymbol{x} \oplus (i, 0))$ is equal to $\boldsymbol{x}$ except component $\boldsymbol{x}_i$ which is equal to vector $\mathbf{0}$ (i.e. queue i is empty).

External Arrivals: The external arrivals to the queues follow independent Poisson processes. The external arrival rate at queue i is denoted by $\lambda_i^{(k)}$ for positive customers of class k. For the sake of readability we assume that there is no external arrivals of signals into the queues but the extension is straightforward.

Service: The customers are served according to the processor sharing (PS) policy. The service times are assumed to be iid exponential with intensity $\mu_i^{(k)}$ for customers of class k in queue i. Thus, as usual with PS queues, there is a service completion of a class k customer in queue i with rate $\mu_i^{(k)} \frac{x_i^{(k)}}{||\boldsymbol{x}_i||}$.

Signals: Signals are not queued in the network. At its arrival at a queue, a signal destroys a positive customer if there is any and then vanishes instantaneously. Every customer is the queue has the same probability to be deleted by the signal. Thus, as usual with PS queues with signals which delete a customer, there is a deletion of a class k customer with rate $c\frac{x_i^{(k)}}{||\boldsymbol{x}_i||}$ where c is the arrival rate of signal at queue i.

Resets: If, at its arrival, the queue is already empty, a signal resets the queue. The queue jumps to an arbitrary state $\boldsymbol{x}_i$ with distribution $\tau_i()$ which will be precisely defined in the theorem. Note at this step that all states can be reached by a reset except the empty queue state.

Customer Movement: At its service completion time in queue i, and according to a Markovian transition matrix, a customer of class k may join queue j as a positive customer of class l with probability $P_{i,j}^{+(k,l)}$, or as a negative customer with probability $P_{i,j}^{-(k)}$. It may also leave the network with probability $d_i^{(k)}$. We assume that a customer cannot return to the queue it has just left neither as a positive customer nor as a signal: $P_{i,i}^{+(k,l)} = 0$ and $P_{i,i}^{-(k)} = 0$ for all i, k and l. As usual, we have:

$$\forall i,k \quad \sum_{j=1}^{N}\sum_{l=1}^{K} P_{i,j}^{+(k,l)} + \sum_{j=1}^{N} P_{i,j}^{-(k)} + d_i^{(k)} = 1. \tag{1}$$

Let $p(\boldsymbol{x})$ be the stationary probability distribution of the network state if it exists. The following result establishes the existence of a product form solution if a fixed point system has a solution which satisfies the stationary constraints.

Theorem 1. *Consider an arbitrary open G-network with p classes of positive customers and a single class of signal. Assume that the Markov chain is ergodic. Assume that the distribution $\tau_i(\boldsymbol{x}_i)$ is given by:*

$$\tau_i(\boldsymbol{x}_i) = \sum_{k=1}^{K} \alpha_i^{(k)} g_i(\boldsymbol{x}_i - e_i^{(k)}) \mathbb{1}_{\{x_i^{(k)}>0\}}, \tag{2}$$

where $\mathbb{1}_X$ is the indicator function. If the system of non-linear equations:

$$\begin{aligned} \rho_i^{(k)} &= \frac{\lambda_i^{(k)} + \sum_{j=1}^{N}\sum_{l=1}^{K} \mu_j^{(l)} \rho_j^{(l)} P_{j,i}^{+(l,k)} + \alpha_i^{(k)} \Lambda_i^-}{\mu_i^{(k)} + \Lambda_i^-}, \\ \alpha_i^{(k)} &= \frac{\rho_i^{(k)}}{\sum_{l=1}^{K} \rho_i^{(l)}}, \\ \Lambda_i^- &= \sum_{j=1}^{N}\sum_{l=1}^{K} \mu_j^{(l)} \rho_j^{(l)} P_{j,i}^{-(l)}, \end{aligned} \tag{3}$$

has a positive solution such that for all station i:

$$\sum_{k=1}^{K} \rho_i^{(k)} < 1, \tag{4}$$

then the system stationary distribution has a product form:

$$p(\boldsymbol{x}) = \prod_{i=1}^{N} g_i(\boldsymbol{x}_i), \tag{5}$$

where

$$g_i(\boldsymbol{x}_i) = (1 - \sum_{k=1}^{K} \rho_i^{(k)})||\boldsymbol{x}_i||! \prod_{k=1}^{K} \frac{(\rho_i^{(k)})^{x_i^{(k)}}}{x_i^{(k)}!}. \tag{6}$$

Proof: Before proceeding with the proof it is worthy to remark that g_i is a distribution probability. Therefore there is no need of a normalization constant in Eq. 5.

Similarly a simple counting argument shows that τ_i is a distribution probability.

The proof is based on the resolution of the Chapman Kolmogorov equation describing the steady-state on the process. We note $M_i^{(k)}(\boldsymbol{x}_i)$ the departure rate of customers of class k from the queue i. Since the service discipline considered is processor sharing, $M_i^{(k)}(\boldsymbol{x}_i)$ can be written as a function of $\mu_i^{(k)}$:

$$M_i^{(k)}(\boldsymbol{x}_i) = \mu_i^{(k)} \frac{x_i^{(k)}}{|\boldsymbol{x}_i|} \mathbb{1}_{\{||\boldsymbol{x}_i||>0\}}. \tag{7}$$

Let us write now the Chapman Kolmogorov equation:

$$\begin{aligned}
p(\boldsymbol{x}) \sum_{i=1}^{N} \left[\sum_{k=1}^{K} (\lambda_i^{(k)} + M_i^{(k)}(\boldsymbol{x}_i)) \right] = & \\
\sum_{i=1}^{N} \sum_{k=1}^{K} \lambda_i^{(k)} p(\boldsymbol{x} - e_i^{(k)}) \mathbb{1}_{\{x_i^{(k)}>0\}} & \quad [1] \\
+ \sum_{i=1}^{N} \sum_{k=1}^{K} M_i^{(k)}(\boldsymbol{x}_i + e^{(k)}) d_i^{(k)} p(\boldsymbol{x} + e_i^{(k)}) & \quad [2] \\
+ \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{K} \sum_{l=1}^{K} M_i^{(k)}(\boldsymbol{x}_i + e^{(k)}) P_{i,j}^{+(k,l)} p(\boldsymbol{x} + e_i^{(k)} - e_j^{(l)}) \mathbb{1}_{\{x_j^{(l)}>0\}} & \quad [3] \\
+ \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{K} \sum_{l=1}^{K} M_i^{(k)}(\boldsymbol{x}_i + e^{(k)}) P_{i,j}^{-(k)} p(\boldsymbol{x} + e_i^{(k)} + e_j^{(l)}) \frac{x_j^{(l)} + 1}{|\boldsymbol{x}_j| + 1} & \quad [4] \\
+ \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{K} M_i^{(k)}(\boldsymbol{x}_i + e^{(k)}) P_{i,j}^{-(k)} p\left((\boldsymbol{x} + e_i^{(k)}) \oplus (j,0) \right) \tau_j(\boldsymbol{x}_j) & \quad [5].
\end{aligned}$$

Let us give some explanations about the right-hand side of this equation. The first term corresponds to external arrivals of customers of class k in queue i. The second term corresponds to end of service and departure to the outside. The third term considers the case where a customer of class k leaves queue i for queue j as a customer of class l. In the fourth term we describe a customer of class k leaving queue i for queue j as a signal which deletes one customer. All customers have the same probability to be deleted by the signal. Thus the class of the deleted customer is distributed following the ratio of customers of class l in the queue before the deletion. Finally there is a transition from an empty queue l which receives a signal from queue i and which jumps to state $\boldsymbol{x}_j$ with probability $\tau_j(\boldsymbol{x}_j)$. The proof is detailed in the appendix. This system has, as usual with G-network, a non linear flow equation.

Lemma 1. *The flow equation (i.e. Equation 8) is satisfied.*

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)} = \sum_{i=1}^{N}\sum_{k=1}^{K}\mu_i^{(k)}\rho_i^{(k)}d_i^{(k)} + \sum_{i=1}^{N}\sum_{k=1}^{K}\Lambda_i^{-}\rho_i^{(k)}. \tag{8}$$

Proof: Consider again the fixed point system (Eq. 3).

$$\rho_i^{(k)}(\mu_i^{(k)} + \Lambda_i^{-}) = \lambda_i^{(k)} + \sum_{j=1}^{N}\sum_{l=1}^{K}\mu_j^{(l)}\rho_j^{(l)}P_{j,i}^{+(l,k)} + \alpha_i^{(k)}\sum_{j=1}^{N}\sum_{l=1}^{K}\mu_j^{(l)}\rho_j^{(l)}P_{j,i}^{-(l)}. \tag{9}$$

Now do the summation for all indices i and k and note that $\sum_{k=1}^{K}\alpha_i^{(k)} = 1$, for all i:

$$\begin{aligned}\sum_{i=1}^{N}\sum_{k=1}^{K}\rho_i^{(k)}\mu_i^{(k)} + \sum_{i=1}^{N}\sum_{k=1}^{K}\Lambda_i^{-}\rho_i^{(k)} &= \sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)} \\ &+ \sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{j=1}^{N}\sum_{l=1}^{K}\mu_j^{(l)}\rho_j^{(l)}P_{j,i}^{+(l,k)} \\ &+ \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{K}\mu_j^{(l)}\rho_j^{(l)}P_{j,i}^{-(l)}.\end{aligned}$$

After factorization:

$$\begin{aligned}\sum_{i=1}^{N}\sum_{k=1}^{K}\rho_i^{(k)}\mu_i^{(k)} + \sum_{i=1}^{N}\sum_{k=1}^{K}\Lambda_i^{-}\rho_i^{(k)} &= \sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)} \\ &+ \sum_{j=1}^{N}\sum_{l=1}^{K}\mu_j^{(l)}\rho_j^{(l)}\Big(\sum_{i=1}^{N}\sum_{k=1}^{K}P_{j,i}^{+(l,k)} + \sum_{i=1}^{N}P_{j,i}^{-(l)}\Big).\end{aligned}$$

Taking into account that the sum of routing probabilities is 1 (i.e. Eq. 1) we get:

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\rho_i^{(k)}\mu_i^{(k)} + \sum_{i=1}^{N}\sum_{k=1}^{K}\Lambda_i^{-}\rho_i^{(k)} = \sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)} + \sum_{j=1}^{N}\sum_{l=1}^{K}\mu_j^{(l)}\rho_j^{(l)}(1-d_j^{(l)}).$$

After cancellation of terms we obtain the flow equation. □

3 Existence of a Solution to the Fixed Point Equation

The proof of the existence of a fixed point of system (3) under general assumptions is crucial since existence of the product form is given under existence of a solution to that system of equations. In that section we use Brouwer's fixed point theorem to prove that a positive solution of system (3) exists. However we do not prove that the stability condition (i.e. $\sum_{k=1}^{K} \rho_i^{(k)} < 1$) is satisfied. Brouwer's fixed point theorem requires that we find a compact and convex set S such that function F is continuous on S and $F(S) \subseteq S$.

Before we give a sufficient condition of existence of a fixed point to mapping F, let us introduce some notions.

Definition 1. *A non negative matrix M is transient if and only if:*

$$\lim_{n \to +\infty} M^n = 0.$$

Property 1. If M is transient then $(I - M)$ is invertible.

Definition 2. *For two vectors u and v, we note $u \preceq v$ if each component of u is less or equal to the corresponding component of v.*

Let us now define F an operator on $(\mathbb{R}^+)^{N \times K}$ defined by its components $F_i^{(k)}$ which is applied on vector $\boldsymbol{q}$ whose components are $\boldsymbol{q}_i^{(k)} = \mu_i^{(k)} \rho_i^{(k)}$:

$$F_i^{(k)}(\boldsymbol{q}) = \frac{\lambda_i^{(k)} + \sum_{j=1}^{N} \sum_{l=1}^{K} \boldsymbol{q}_j^{(l)} P_{j,i}^{+(l,k)} + \frac{\sum_{j=1}^{N} \sum_{l=1}^{K} \boldsymbol{q}_j^{(l)} P_{j,i}^{-(l)} \rho_i^{(k)}}{\rho_i}}{1 + \frac{\sum_{j=1}^{N} \sum_{l=1}^{K} \boldsymbol{q}_j^{(l)} P_{j,i}^{-(l)}}{\mu_i^{(k)}}}.$$

We investigate the existence of a fixed point for F.

Theorem 2. *Consider an open G-network such that for every queue index i there exists a class index k such that $\lambda_i^{(k)} > 0$, system (3) has a solution in $(\mathbb{R}^+)^{N \times K}$.*

Proof: We define a new operator, say G, on $(\mathbb{R}^+)^{N \times K}$ by its components:

$$G_i^{(k)}(q) = \lambda_i^{(k)} + \sum_{j=1}^{N} \sum_{l=1}^{K} \boldsymbol{q}_j^{(l)} P_{j,i}^{+(l,k)} + \sum_{j=1}^{N} \sum_{l=1}^{K} \boldsymbol{q}_j^{(l)} P_{j,i}^{-(l)}.$$

Let λ be the vector with components $\lambda_i^{(k)}$. Using a matrix formulation, the definition of G may be restated as $G(q) = \lambda + q(P^+ + P^-)$. Since we are in an open G-network, $P^+ + P^-$ is transient, then $(I - P^+ - P^-)$ is invertible and G has a fixed point $\overline{\boldsymbol{q}}$ defined by:

$$\overline{\boldsymbol{q}} = \lambda (I - P^+ - P^-)^{-1}.$$

Now, we define $S0$ a subset of $(I\!R^+)^{N\times K}$ as follows:

$$S0 = \{q \in (I\!R^+)^{N\times K} : 0 \preceq q \preceq \overline{\boldsymbol{q}}\}.$$

$S0$ is compact and convex. Since the network is open, it exists i, k such that $\lambda_i^{(k)} > 0$ and consequently $\lambda \neq \mathbf{0}$. Thus $\overline{\boldsymbol{q}} \neq 0$ and the interior of S is not empty.

Clearly, $F \preceq G$ and G is non-decreasing , so for all $\boldsymbol{q}$ in $S0$ we have:

$$F(\boldsymbol{q}) \preceq G(\boldsymbol{q}) \preceq G(\overline{\boldsymbol{q}}) = \overline{\boldsymbol{q}},$$

and then $F(S0) \subseteq S0$.

But function F is not defined when $\rho_i = 0$ for some index i. And the limits depend of the ratio $\rho_i^{(k)}/\rho_i$ when both $\rho_i^{(k)}$ and ρ_i go to 0. Thus we must improve the definition of the set we consider.

Let us now define a componentwise lower bound of F.

$$\underline{\boldsymbol{q}}_i^{(k)} = \frac{\lambda_i^{(k)}\mu_i^{(k)}}{\mu_i^{(k)} + \sum_{j=1}^{N}\sum_{l=1}^{K}\overline{\boldsymbol{q}}_j^{(l)}P_{j,i}^{-(l)}}.$$

Again, it is clear that $\underline{\boldsymbol{q}} \preceq F(\boldsymbol{q})$ for all $\underline{\boldsymbol{q}} \preceq \boldsymbol{q} \preceq \overline{\boldsymbol{q}}$ and for all i there exists an index k such that $\underline{\boldsymbol{q}}_i^{(k)}$ is positive as by assumption $\lambda_i^{(k)}$ is positive. Therefore we define S as follows:

$$S = \{q \in (I\!R^+)^{N\times K} : \underline{\boldsymbol{q}} \preceq \boldsymbol{q} \preceq \overline{\boldsymbol{q}}\}.$$

Then S is compact and convex and has a non empty interior, F is continuous and $F(S) \subseteq S$. F satisfies assumptions of Brouwer's theorem and F has a fixed point (say $\boldsymbol{q}0$). □

We can remove the technical assumption of the previous theorem (i.e. $\forall\ i\ \exists\ \ k$ such that $\lambda_i^{(k)} > 0$). But the proof is much more complex and is based on another function H which is also a component-wise lower bound of F:

$$H_i^{(k)}(\boldsymbol{q}) = \frac{\lambda_i^{(k)}\mu_i^{(k)} + \mu_i^{(k)}\sum_{j=1}^{N}\sum_{l=1}^{K}\boldsymbol{q}_j^{(l)}P_{j,i}^{+(l,k)}}{\mu_i^{(k)} + \sum_{j=1}^{N}\sum_{l=1}^{K}\overline{\boldsymbol{q}}_j^{(l)}P_{j,i}^{-(l)}}.$$

The proof is omitted for the sake of readability.

Property 2. Remark that G is the fixed point system we get in a model of a network of PS queues with routing matrix $R = P^+ + P^-$ and same service and arrival rates. If $\overline{\boldsymbol{q}}$ implies that this network without signals is stable, then the G-network of PS queues with resets is also stable (i.e. $\sum_{k=1}^{K}\rho_i^{(k)} < 1$ for all i).

Proof: indeed we have proved that the fixed point of the G-network with reset is componentwise smaller than the fixed point of the network with usual customers: for all $\boldsymbol{q}$ in S, $F(q) \preceq \overline{\boldsymbol{q}}$. This is also true for $\boldsymbol{q}0$ as $\boldsymbol{q}0 \in S$. As the service rates $\mu_i^{(k)}$ are the same for both networks, this relation implies that the stability condition of the usual network of PS queues implies that $\sum_{k=1}^{K}\rho_i^{(k)} < 1$. □

Finally one can remark that the uniqueness of the steady-state distribution of an ergodic Markov chain implies the uniqueness of the solution of the fixed point system which satisfy the stability constraints. It remains to prove an algorithm to find the numerical solution of the fixed point. This is not an easy task because the equations are not linear and this property may lead to non convergent numerical algorithm even if we are very close to the solution [6]. A new version of the envelop algorithm developed in [3] is still under study to prove its convergence.

4 Conclusion

The first product form result for networks with positive and negative customers [9] had been the source of a large number of questions and problems in various fields of stochastic modelling. The first signals considered always delete customers. Thus G-networks have been proposed to model work deletion. For instance, a network of queues with jumps back to zero can model the removal of pieces which fail a statistical test based on the analysis of a sample.

Resets are completely different. When a signal enters a non empty queue, we have a customer deletion but when the queue is empty, the resets make the queue size positive with probability one. And the size reached after a signal reception is not upper bounded. Thus, we have a more complex dynamics than a single deletion.

This work is the first multiclass G-networks of queues with product form solution. So its main contributions are theoretical. A natural (and theoretical) question is to try to extend this result to other types of station such as LIFO, FIFO and Infinite Server queues. This is not that easy. First, to the best of our knowledge it was not possible to prove such a result for Infinite Server when the signals delete customers. All the theorems previously published on Multiclass G-networks with signals only consider PS, LIFO and FIFO queues [4,15]. Some results [5] are even more restrictive and only apply for PS queues. Furthermore the distribution τ_i must be dependent of the service discipline and every state can be reached by a reset. Note that we must consider the detailed state space which takes into account the position of the customer in the queue. Such an assumption makes the analysis very complex but also very interesting. Further works are necessary before we can use these theorems on real examples but we think that they open new directions to apply G-networks in stochastic modeling.

Acknowledgment. This work was supported by ANR research project (BLANC 2006) SMS.

References

1. Artalejo, J.R.: G-networks: a versatile approach for work removal in queueing networks. European J. Op. Res. 126, 233–249 (2000)
2. Baskett, F., Chandy, K., Muntz, R.R., Palacios, F.G.: Open, closed and mixed networks of queues with different classes of customers. Journal ACM 22(2), 248–260 (1975)

3. Fourneau, J.M.: Computing the steady-state distribution of networks with positive and negative customers. In: 13th IMACS World Congress on Computation and Applied Mathematics, Dublin (1991)
4. Fourneau, J.M., Gelenbe, E., Suros, R.: G-networks with multiple classes of positive and negative customers. Theoretical Computer Science 155, 141–156 (1996)
5. Fourneau, J.M., Kloul, L., Quessette, F.: Multiple class G-networks with jumps back to Zero. IEEE Mascots 95, 28–32 (1995)
6. Fourneau, J.M., Quessette, F.: Computing the Steady-State Distribution of G-networks with Synchronized Partial Flushing. In: Levi, A., Savaş, E., Yenigün, H., Balcısoy, S., Saygın, Y. (eds.) ISCIS 2006. LNCS, vol. 4263, pp. 887–896. Springer, Heidelberg (2006)
7. Fourneau, J.M.: Closed G-networks with resets: Product Form solution. In: 4th International Conference on the Quantitative Evaluation of Systems QEST 2007, Edinburgh, pp. 287–296 (2007)
8. Gelenbe, E.: Random neural Networks with Negative and Positive Signals and Product Form Solution. Neural Computation 1(4), 502–510 (1990)
9. Gelenbe, E.: Product form queueing networks with negative and positive customers. Journal of Applied Probability 28, 656–663 (1991)
10. Gelenbe, E., Schassberger, R.: Stability of G-Networks. Probability in the Engineering and Informational Sciences 6, 271–276 (1992)
11. Gelenbe, E.: G-networks with instantaneous customer movement. Journal of Applied Probability 30(3), 742–748 (1993)
12. Gelenbe, E.: G-Networks with signals and batch removal. Probability in the Engineering and Informational Sciences 7, 335–342 (1993)
13. Gelenbe, E.: G-networks: An unifying model for queueing networks and neural networks. Annals of Operations Research 48(1-4), 433–461 (1994)
14. Gelenbe, E.: The first decade of G-networks. European J. Op. Res. 126(2), 231–232 (2000)
15. Gelenbe, E.: G-Networks: multiple class of Positive customers, signals and product form results. In: Calzarossa, M.C., Tucci, S. (eds.) Performance 2002. LNCS, vol. 2459, pp. 1–16. Springer, Heidelberg (2002)
16. Gelenbe, E., Lent, R., Xu, Z.: Design and performance of cognitive packet networks. Performance Evaluation 46(2-3), 155–176 (2001)
17. Gelenbe, E., Fourneau, J.M.: G-networks with resets. Performance Evaluation 49(1/4), 179–191 (2002)
18. Gelenbe, E., Fourneau, J.M.: Flow Equivalence and Stochastic Equivalence in G-Networks. Computational Management Science 1(2), 179–192 (2004)
19. Gelenbe, E., Lent, R.: Power-aware ad hoc cognitive packet networks. Ad Hoc Networks 2(3), 205–216 (2004)
20. Gelenbe, E.: Keeping Viruses Under Control. In: Yolum, p., Güngör, T., Gürgen, F., Özturan, C. (eds.) ISCIS 2005. LNCS, vol. 3733, pp. 304–311. Springer, Heidelberg (2005)
21. Harrison, P.: Compositional reversed markov processes, with applications to g-networks. Performance Evaluation 57(3), 379–408 (2004)
22. Jackson, J.R.: Jobshop-like queueing systems. Management Science 10(1), 131–142 (1963)
23. Mohamed, S., Rubino, G., Varela, M.: Performance evaluation of real-time speech through a packet network: a random neural networks-based approach. Performance Evaluation 57(2), 141–161 (2004)
24. Pinedo, M., Chao, X., Miyazawa, M.: Queueing Networks: Customers, Signals and Product Form Solutions. J. Wiley, Chichester (1999)

25. Rubino, G., Tirilly, P., Varela, M.: Evaluating Users' Satisfaction in Packet Networks Using Random Neural Networks. In: Kollias, S., Stafylopatis, A., Duch, W., Oja, E. (eds.) ICANN 2006. LNCS, vol. 4131, pp. 303–312. Springer, Heidelberg (2006)

26. Wang, Y.: G-Networks and the Modeling of Adversarial Agents. In: Kollias, S., Stafylopatis, A., Duch, W., Oja, E. (eds.) ICANN 2006. LNCS, vol. 4131, pp. 330–339. Springer, Heidelberg (2006)

A Appendix: Proof of the Theorem

Consider again the Chapman Kolmogorov equation for steady-state:

$$p(\boldsymbol{x})\sum_{i=1}^{N}\left[\sum_{k=1}^{K}(\lambda_i^{(k)}+M_i^{(k)}(\boldsymbol{x}_i)\mathbb{1}_{\{x_i^{(k)}>0\}})\right]=$$

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)}p(\boldsymbol{x}-e_i^{(k)})\mathbb{1}_{\{x_i^{(k)}>0\}} \quad [1]$$

$$+\sum_{i=1}^{N}\sum_{k=1}^{K}M_i^{(k)}(\boldsymbol{x}_i+e^{(k)})d_i^{(k)}p(\boldsymbol{x}+e_i^{(k)}) \quad [2]$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\sum_{l=1}^{K}M_i^{(k)}(\boldsymbol{x}_i+e^{(k)})P_{i,j}^{+(k,l)}p(\boldsymbol{x}+e_i^{(k)}-e_j^{(l)})\mathbb{1}_{\{x_j^{(l)}>0\}} \quad [3]$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\sum_{l=1}^{K}M_i^{(k)}(\boldsymbol{x}_i+e^{(k)})P_{i,j}^{-(k)}p(\boldsymbol{x}+e_i^{(k)}+e_j^{(l)})\frac{x_j^{(l)}+1}{||\boldsymbol{x}_j||+1} \quad [4]$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}M_i^{(k)}(\boldsymbol{x}_i+e^{(k)})P_{i,j}^{-(k)}p\left((\boldsymbol{x}+e_i^{(k)})\oplus(j,0)\right)\tau_j(\boldsymbol{x}_j) \quad [5].$$

Divide both sides of the equation by $p(\boldsymbol{x})$ and make the following simplifications:

$$\frac{p(\boldsymbol{x}-e_i^{(k)})}{p(\boldsymbol{x})}=\frac{x_i^{(k)}}{\rho_i^{(k)}||\boldsymbol{x}_i||},\quad and\quad \frac{p(\boldsymbol{x})}{p(\boldsymbol{x}+e_i^{(k)})}=\rho_i^{(k)}\frac{||\boldsymbol{x}_i||+1}{x_i^{(k)}+1},$$

$$\frac{p\left((\boldsymbol{x}+e_i^{(k)})\oplus(j,0)\right)}{p(\boldsymbol{x})}=\rho_i^{(k)}\frac{||\boldsymbol{x}_i||+1}{x_i^{(k)}+1}\frac{g_j(\mathbf{0})}{g_j(\boldsymbol{x}_j)}.$$

$$\sum_{i=1}^{N}\left[\sum_{k=1}^{K}(\lambda_i^{(k)}+M_i^{(k)}(\boldsymbol{x}_i)\mathbb{1}_{\{x_i^{(k)}>0\}})\right]=$$

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)}\frac{x_i^{(k)}}{\rho_i^{(k)}||\boldsymbol{x}_i||}\mathbb{1}_{\{x_i^{(k)}>0\}} \quad [1]$$

$$+\sum_{i=1}^{N}\sum_{k=1}^{K}M_i^{(k)}(\boldsymbol{x}_i+e^{(k)})d_i^{(k)}\rho_i^{(k)}\frac{||\boldsymbol{x}_i||+1}{x_i^{(k)}+1} \quad [2]$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\sum_{l=1}^{K}M_i^{(k)}(\boldsymbol{x}_i+e^{(k)})P_{i,j}^{+(k,l)}\rho_i^{(k)}\frac{||\boldsymbol{x}_i||+1}{x_i^{(k)}+1}\frac{x_j^{(l)}}{\rho_j^{(l)}||\boldsymbol{x}_j||}\mathbb{1}_{\{x_j^{(l)}>0\}} \quad [3]$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\sum_{l=1}^{K}M_i^{(k)}(\boldsymbol{x}_i+e^{(k)})P_{i,j}^{-(k)}\rho_i^{(k)}\frac{||\boldsymbol{x}_i||+1}{x_i^{(k)}+1}\rho_j^{(l)} \quad [4]$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}M_i^{(k)}(\boldsymbol{x}_i+e^{(k)})P_{i,j}^{-(k)}\rho_i^{(k)}\frac{||\boldsymbol{x}_i||+1}{x_i^{(k)}+1}\frac{g_j(\mathbf{0})}{g_j(\boldsymbol{x}_j)}\tau_j(\boldsymbol{x}_j) \quad [5].$$

Now remember that $M_i^{(k)}(\boldsymbol{x}_i)=\mu_i^{(k)}\frac{x_i^{(k)}}{||\boldsymbol{x}_i||}\mathbb{1}_{\{||\boldsymbol{x}_i||>0\}}$. Therefore:

$$M_i^{(k)}(\boldsymbol{x}_i+e^{(k)})=\mu_i^{(k)}\frac{x_i^{(k)}+1}{||\boldsymbol{x}_i||+1}.$$

After substitution we get:

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)}+\sum_{i=1}^{N}\sum_{k=1}^{K}\mu_i^{(k)}\frac{x_i^{(k)}}{||\boldsymbol{x}_i||}\mathbb{1}_{\{||\boldsymbol{x}_i||>0\}}=$$

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)}\frac{x_i^{(k)}}{\rho_i^{(k)}||\boldsymbol{x}_i||}\mathbb{1}_{\{x_i^{(k)}>0\}} \quad [1]$$

$$+\sum_{i=1}^{N}\sum_{k=1}^{K}\mu_i^{(k)}\rho_i^{(k)}d_i^{(k)} \quad [2]$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\sum_{l=1}^{K}\mu_i^{(k)}\rho_i^{(k)}P_{i,j}^{+(k,l)}\frac{x_j^{(l)}}{\rho_j^{(l)}||\boldsymbol{x}_j||}\mathbb{1}_{\{x_j^{(l)}>0\}} \quad [3]$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\sum_{l=1}^{K}\mu_i^{(k)}\rho_i^{(k)}P_{i,j}^{-(k)}\rho_j^{(l)} \quad [4]$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\mu_i^{(k)}\rho_i^{(k)}P_{i,j}^{-(k)}\frac{g_j(\mathbf{0})}{g_j(\boldsymbol{x}_j)}\tau_j(\boldsymbol{x}_j) \quad [5].$$

Consider now the last term of the right hand side. Because of assumptions on the distribution $\tau_i()$, the fixed point system 3 and the definition of the marginal probabilities in equation 6, we obtain:

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\mu_i^{(k)}\rho_i^{(k)}P_{i,j}^{-(k)}\frac{g_j(\mathbf{0})}{g_j(\boldsymbol{x}_j)}\tau_j(\boldsymbol{x}_j)=\sum_{j=1}^{N}\sum_{l=1}^{K}\Lambda_j^{-}(1-\rho_j)\alpha_j^{(l)}\frac{x_j^{(l)}}{||\boldsymbol{x}_j||}\frac{1}{\rho_j^{(l)}}\mathbb{1}_{\{||\boldsymbol{x}_j||>0\}}.$$

Clearly we can also substitute indices j and l with i and k in this last term. We perform the same substitution in the third term of the CK equation. After substitution in the global equation, we reorganize the last term and move the negative part on the left side of the equation:

$$\begin{aligned}
\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)} + \sum_{i=1}^{N}\sum_{k=1}^{K}\left(\mu_i^{(k)} + \frac{\Lambda_i^- \alpha_i^{(k)}\rho_i}{\rho_i^{(k)}}\right)\frac{x_i^{(k)}}{||\boldsymbol{x}_i||}\mathbb{1}_{\{||\boldsymbol{x}_i||>0\}} = \\
\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)}\frac{x_i^{(k)}}{\rho_i^{(k)}||\boldsymbol{x}_i||}\mathbb{1}_{\{x_i^{(k)}>0\}} \quad [1] \\
+ \sum_{i=1}^{N}\sum_{k=1}^{K}\mu_i^{(k)}\rho_i^{(k)}d_i^{(k)} \quad [2] \\
+ \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\sum_{l=1}^{K}\mu_j^{(l)}\rho_j^{(l)}P_{j,i}^{+(l,k)}\frac{x_i^{(k)}}{\rho_i^{(k)}||\boldsymbol{x}_i||}\mathbb{1}_{\{x_i^{(k)}>0\}} \quad [3] \\
+ \sum_{j=1}^{N}\sum_{l=1}^{K}\Lambda_j^-\rho_j^{(l)} \quad [4] \\
+ \sum_{i=1}^{N}\sum_{k=1}^{K}\Lambda_i^-\alpha_i^{(k)}\frac{x_i^{(k)}}{||\boldsymbol{x}_i||}\frac{1}{\rho_i^{(k)}}\mathbb{1}_{\{||\boldsymbol{x}_i||>0\}} \quad [5].
\end{aligned}$$

We notice that $\frac{\alpha_i^{(k)}\rho_i}{\rho_i^{(k)}} = 1$ and after substitution we reorganize the terms to factorize them:

$$\begin{aligned}
&\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)} + \sum_{i=1}^{N}\sum_{k=1}^{K}\left(\mu_i^{(k)} + \Lambda_i^-\right)\frac{x_i^{(k)}}{||\boldsymbol{x}_i||}\mathbb{1}_{\{||\boldsymbol{x}_i||>0\}} \\
&\quad= \sum_{i=1}^{N}\sum_{k=1}^{K}\mu_i^{(k)}\rho_i^{(k)}d_i^{(k)} \\
&\quad+ \sum_{j=1}^{N}\sum_{l=1}^{K}\Lambda_j^-\rho_j^{(l)} \\
&\quad+ \sum_{i=1}^{N}\sum_{k=1}^{K}\frac{x_i^{(k)}}{\rho_i^{(k)}||\boldsymbol{x}_i||}\mathbb{1}_{\{x_i^{(k)}>0\}}\left(\lambda_i^{(k)} + \sum_{j=1}^{N}\sum_{l=1}^{K}\mu_j^{(l)}\rho_j^{(l)}P_{j,i}^{+(l,k)} + \Lambda_i^-\alpha_i^{(k)}\right).
\end{aligned}$$

Remark that $\frac{x_i^{(k)}}{||\boldsymbol{x}_i||}\mathbb{1}_{\{||\boldsymbol{x}_i||>0\}} = \frac{x_i^{(k)}}{||\boldsymbol{x}_i||}\mathbb{1}_{\{x_i^{(k)}>0\}}$. Thus, due to the definition of $\rho_i^{(k)}$, the functional terms canceled and we obtain the flow equation of the system:

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_i^{(k)} = \sum_{i=1}^{N}\sum_{k=1}^{K}\mu_i^{(k)}\rho_i^{(k)}d_i^{(k)} + \sum_{j=1}^{N}\sum_{l=1}^{K}\Lambda_j^-\rho_j^{(l)}.$$

And we use Lemma 1 to complete the proof.

Simulation of a Peer to Peer Market for Grid Computing

Uli Harder and Fernando Martínez Ortuño

Department of Computing, Imperial College London
Huxley Building, 180 Queens Gate, London SW7 2RH, UK
{uh,fermaror}@doc.ic.ac.uk
http://aesop.doc.ic.ac.uk/

Abstract. In this paper we investigate the market economy of a Peer-to-Peer network for Grid Computing. We present a simulation of P2P network where nodes either require or offer resources, which can be thought as CPU time for example. We examine the market behaviour of the P2P network for Erdős-Rényi and Barabási-Albert networks types of different sizes ranging from 4,096 nodes to 1,048,576 nodes. We find that utilisation and market behaviour depend on the network type but not the size. Similarly, different price update algorithms have little effect on the price development, which is more determined by the network type. We also measure the average buffer size and number of messages in the system. For the Barabási-Albert network, we find that the buffer size of each node has an effect on the price development in the system. The results are useful to guide designers of P2P Grid computing systems, like the Global Open Grid.

1 Introduction

The concept of Grid computing [1] has now become a well established as a paradigm for distributed computing. In recent years it has been suggested to use peer-to-peer (P2P) technology to build the Grid infrastructure [2,3,4]. Together with economic incentives this technology will be used for service discovery and scheduling of Grid resources and jobs. In economics this relates to Hayek's idea of Catallaxy [3,5] as explained in detail in [2].

The idea to use economic mechanisms to manage access to computer systems is fairly old and the first reference is usually credited to Sutherland [6]. He describes an auctioning scheme used to manage access to a PDP-1. Later, more involved approaches were developed for mainframes, see for instance [7,8]. However, most approaches taken in the mainframe days can not be applied to the Grid, as any central service will normally not scale and prevent the system from becoming larger.

Previously centralised auction models have been investigated for the allocation of Grid resources. For example [9] used a three tier market set-up of producers, brokers and end-users. Another centralised allocation model is Tycoon [10]. An example of an existing P2P Grid is the data grid *P-grid* [4], which allows large

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 234–248, 2008.

scale sharing of data securely. A more sophisticated system is the Gobal Open Grid (GOG) described in [2], which also includes an economic framework. In this paper we present results of simulations of a simplified version of the GOG. One of the main motivation for this work is the desire to assist to establish a market economy for Grid Computing similar to that of privatized electricity markets for instance [11], which is a large Multi-Agent-Simulation of potential changes to the electricity market. In the electricity markets, one of the main difficulties for a regulator is to detect collusion of providers in order to force up prices. In [12] the authors describe how equilibrium auction theory can be used to detect collusion. With the simulations of our model, it might be possible to determine whether any such behaviour is actually based on agents decisions or in fact determined by the market rules. In the latter case the regulator could try to change rules and therefore make detection of cartels easier.

One should also mention that in the Physics community relevant related research has for instance been conducted by Rosvall and Sneppen [13], who investigate the information horizon of agents on a network. They investigate the propagation of information on networks with agents that have only local knowledge. There is also a recent paper by Goshal and Newman [14] which describes a method to self-organize a network that is not scale-free to reduce search time. Whilst some P2P networks show signs of a scale-free small world network [15], this is not necessarily a desirable feature when nodes search for information in the network.

In this paper we first summarize some basic graph theory. Then we describe the two P2P models (plain vanilla and MaGoG) that we investigate. In both models buyer and seller nodes try to allocate resources by the exchange of money.

We find that, in a system where resources are held indefinitely and only one resource per buyer is allowed, it depends on the network structure how long it takes the system to satisfy all nodes, if that is possible. If resources are released and re-advertised after a certain time period, we find that the simulation results indicate that the P2P system settles to an equilibrium on all of the network types we tested. Also, the system measures are independent of the network size for networks of the same type.

2 P2P Networks and Graphs

P2P networks are essentially overlay networks which live on top of existing infrastructure. The P2P networks we evaluate here are non-centralised, unstructured networks. For an overview of other existing P2P networks and their details, see [16].

Graph theory can be used to classify networks. A graph (or network) consists of N *vertices (nodes)* and E *edges (links)* between the vertices. Vertices can be classified by their *node degree*, which is the number of in or out going edges. For an undirected those numbers are the same. Graphs are called *connected* when there is path from any node to any other node. Of interest are measures like average node degree or the distribution of the node degree. A network is also

characterized by the *diameter*, which is the longest shortest path between nodes of the network. A measure indicating how connected nodes are is the *clustering coefficient*. It is essentially the ratio of existing links a node and its neighbours have compared to all possible links. An easy way to categorize graphs is to look at their node degree distribution. On the one extreme there is the Erdős-Rényi graphs [17], where the degree distribution tends to a Poisson distribution for large connected graphs

$$n(k) \approx NK^k \frac{e^{-K}}{k!}.$$

K is the average degree distribution. Then there are small world networks described by Watts and Strogartz [18], which have a much higher clustering coefficient compared to a random network with a similar average distance of all shortest paths between all nodes. Barabási and Albert describe in [19] networks that have a scale-free or power law distribution of the node degrees

$$n(k) \approx (k + const.)^{-\alpha}$$

which are generated by a growing network. The gradient α of the power law tends to be between two and three for these networks. For a good review and more details of complex networks, see for instance [20]. It has been shown Freenet is scale-free [15] and also Gnutella is a small-world and scale-free network [21]. Therefore we choose both scale-free and random networks for our investigation.

3 The Plain Vanilla Model

The peer to peer overlay network of our "Grid" is defined by a connected graph. The nodes of the graph are the computing resources connected to the Grid and are either a

- Buyer. This node tries to acquire a resource and bids a maximum price it is willing to pay.
- Seller. A node that sells exclusive access to its resource for a minimum price.

Either node type is un-satisfied when they are looking for or are offering service. Otherwise they are in a satisfied state. In the model, a buyer node sends messages to its nearest neighbours informing them that it wants to purchase a resource unit at a price p_b. It also forwards messages from its neighbours until a preset time to live (TTL) has expired. Seller nodes look at incoming messages and decide whether their own price p_s can satisfy an incoming message. They also forward messages they can not satisfy to their neighbours. If a match can be made both nodes go into the satisfied state and the message is not forwarded any further by the Seller node. Later incoming possible matches are ignored. Satisfied nodes continue to pass on messages.

In the simulation, updates of the nodes happen asynchronously. At each time step a node is picked randomly and the following actions take place. In [22] the authors show how parallel updates can induce system behaviour which is presumably artificial. It is also unrealistic to assume that all nodes have exactly the same time globally.

- For a buyer node, the node checks whether its messages have timed out. If that is the case the node sends out new messages with a slightly increased price $p'_b = (1 + \Delta p)p_b$. The node also forwards all messages that have arrived from neighbouring nodes.
- A seller node checks all its incoming messages and, if a match can be made, it contacts the node the message originated from. If that node is still in Buyer mode, both nodes go into the satisfied state. Otherwise the Seller carries on as usual. After this, all messages are forwarded to its neighbours. If a seller fails to attract buyer nodes, it lowers its price $p'_s = (1 - \Delta p)p_s$.
- Contracts between a buyer and a seller are always made at a price which is the average between the price the buyer is bidding and the price the seller is asking.
- A satisfied buyer or seller node passes all its incoming messages to its nearest neighbours, excluding the one that sent or passed on the message.
- In the dynamic model, a satisfied node returns to its previous mode after it has been picked for updates a certain number of times. This also changes the state of its peer node. We have defined two rules to set the new price the buyer/seller starts asking/bidding again.
  - Method one: both the buyer and the seller, that return to the unsatisfied state, change their previous price to the price at which the contract with its partner was made, that is, the average between their price and their partner's price, $p'_n = (p_n + p_{npartner})/2$.
  - Method two: the buyer nodes lower their price to a fraction of what they have just paid, $p'_b = (1 - \Delta p)p_b$, and seller nodes increase their price in the same way, $p'_s = (1 + \Delta p)p_s$.

Initially, the nodes set into buyer or seller mode have a bidding and asking price that makes deals possible. We try both fixed prices and overlapping uniformly distributed prices. Price changes occur either at the end of partnership or when buyers or sellers fail to find a match. An Epoch consists of N simulation steps for a network with N nodes.

4 Middleware for Activating the Global Open Grid (MaGoG)

The design of the *Middleware for Activating the Global Open Grid* (MaGoG) is outlined in [2]. The Gobal open Grid (GoG) is an uncentralised P2P network which is meant to connect all possible computing devices from mobile phones to supercomputers as nodes using the MaGoG software. The nodes can either be resource providers or consumers. Using a micropayment system, nodes exchange real money for their services. Both users and providers send out messages containing the maximum or minimum price the nodes are willing to accept for a deal (these messages have been dubbed Ask/Bid bees). Nodes only have local knowledge and know how to contact a finite number of other nodes, known as their nearest neighbours.

Messages originate at a consumer or provider node and get sent to the nearest neighbours. There they remain in a *pub* (in effect a buffer with memory) and wait until a matching other message comes along. They are also cloned and sent to nearest neighbours of the current (pub) node. The belief is that the usual problems associated with message flooding do not hamper the GoG due to the idea of providers and consumers advertising their services. Additionally, the pubs are a more clever message buffer as it retains memory of previously received messages for longer. However, this is at the cost of more memory usage, which presumably is not an issue nowadays. In his report [2], the MaGoG market is compared to that of currencies, stock and futures. The underlying philosophy of Catallaxy [23] is thought to provide a stable market in MaGoG. In contrast to the Walrasian [24], no centralised global knowledge is needed for the economic agents to create a stable market.

Modelling the MaGoG technology is done similarly to the description of the naive model earlier. The two differences are that both buyer and seller nodes are now originators of messages. In addition, the buffer mechanism at each node has been changed, so that messages are retained as long as possible and only copies of the messages are sent on to neighbours. This turns buffers into *pubs*. We do not model any of the intricate details relating to the payment etc. For the dynamic model, the prices are updated according to method one or two described in the naive model section.

5 Experimental Results

For the overlay network, we chose graphs that were either Erdős-Rényi (ER) random graphs or Barabási-Albert (BA) graphs created with the *igraph* package [25].

We have made successful simulations with graphs of different sizes. The largest one was formed by 1,048,576 nodes. However, we do not show in this paper the plots resulting from all the graphs that we have used in the simulations, since our results have proved to be independent of the graph size. Furthermore, in some figures, with the objective of presenting clearer plot, the results of some graphs have been omitted.

5.1 Static Model

In a model where none of the nodes ever changes their state after a suitable partner node has been found, we verified the correctness of our simulation. In overlay networks whose diameter is greater than the TTL or which are disconnected, it can happen that a small subset of the nodes will not find a partner. Generally, nodes in the model are satisfied faster on a ER graph than a BA graph of the same size.

Fig. 1 shows the evolution of a plain vanilla model for a static simulation for different TTLs. Fig. 2 represents the same as figure 1, but for a MaGoG model of the system.

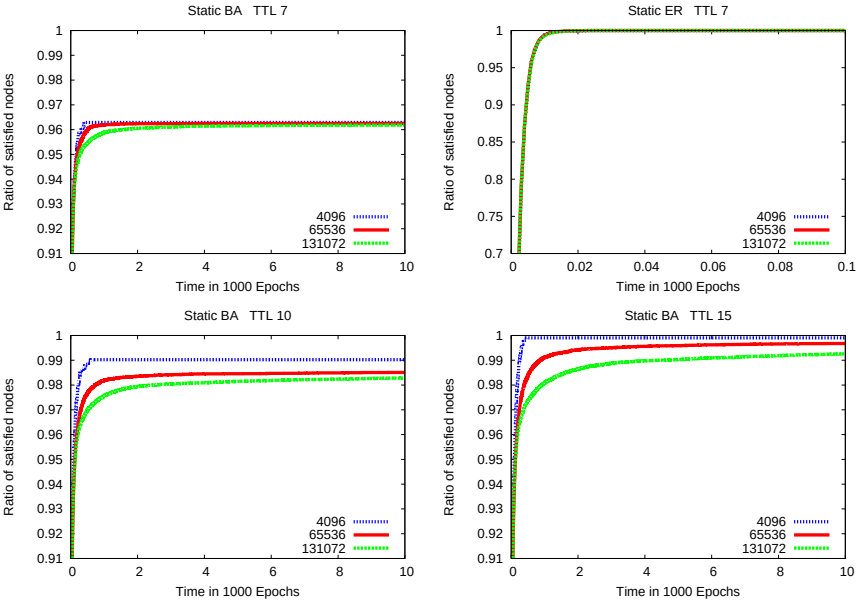


Fig. 1. Demand satisfaction in the plain vanilla static version of the model for different values of TTL and the two kinds of graphs analyzed (BA and ER)

Since the TTL sets the limit in the number of steps a message can walk from the node that issued the message, in those networks whose diameter is bigger than the TTL of the messages, there are nodes that are not able to reach a deal, since their messages vanish before reaching their potential clients further away in the network. This fact explains that the ratio of satisfied nodes in the network is lower than 1 for some of the cases in Fig. 1 and 2. The percentage of satisfied nodes in an ER network is generally higher than in a BA network due to the higher degree of connectivity in the ER graph than in a BA graph. Furthermore, this is the reason why nodes become satisfied faster in ER graphs than in BA graphs, as one can see from Fig. 1 and 2.

On the other hand, the MaGoG system, i.e. the fact that now both buyers and sellers send messages (extending the scope of the potential clients) and that nodes keep a copy of the messages they receive (so that potential future deals can be arranged in their pubs), makes it possible more nodes can become satisfied more quickly, as it is easy to see from Fig. 2. In effect this more than doubles the TTL.

5.2 Dynamic Model

In the dynamic model, satisfied nodes are changed back to the unsatisfied state after being picked a number of times since they became satisfied. In the simulation results shown in Fig. 3, this number was set to 10. The value of other

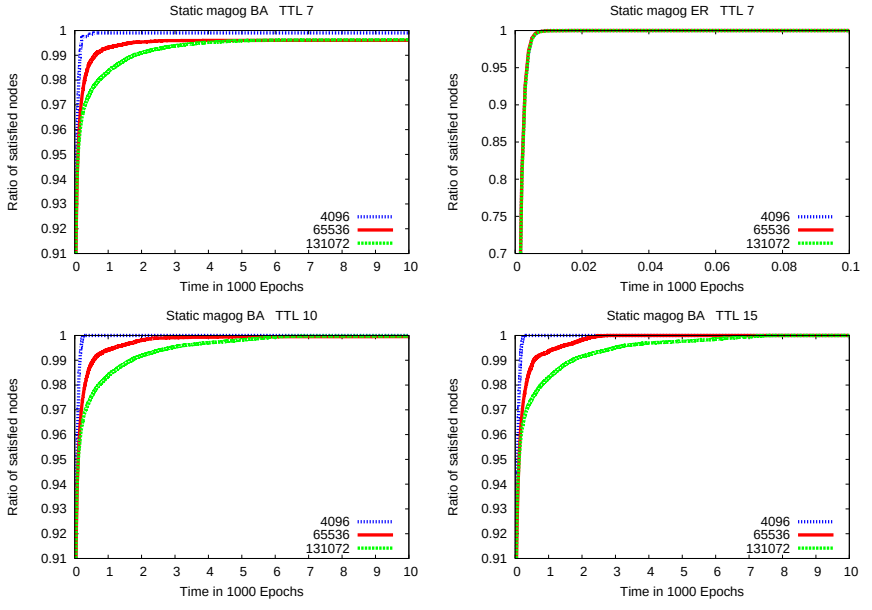


Fig. 2. Demand satisfaction in the MaGoG static version of the model for different values of TTL and the two kinds of graphs analyzed (BA and ER)

parameters in this simulation are 10 hops for the time to live (TTL) of messages and a buffer size of 10 for all nodes.

Despite the fact that the state of the nodes is continuously changing during the simulations, after a certain number of epochs, all the parameters that have been measured (messages per buffer, deals per node and ratio of satisfied nodes) reach an equilibrium in both the plain vanilla version of the system and the MaGoG one. However, the MaGoG version achieves more stable values, although it takes longer to reach the equilibrium state.

Fig. 3 shows the average number of messages in buffers, the number of deals per node and the ratio of satisfied nodes in differently sized BA and ER networks that use the plain vanilla version of the model.

We successfully ran simulations with networks up to 1m nodes for the BA case, and their results were not different from the smaller networks. However, the plots of the results for the 1m-size BA network are omitted in this paper, since we did not have the equivalent size in the ER network in order to compare them.

At the end of each epoch there are roughly 1.4 messages in each node's buffer for a BA graph, but about seven times as many on a random network. The random networks also show variation depending on the network size.

Roughly 3.5% of all nodes on a BA network are involved in a deal per epoch. Again the random network shows a slightly higher number of about 4.7% . But this time there is no noticeable dependence on the network size.

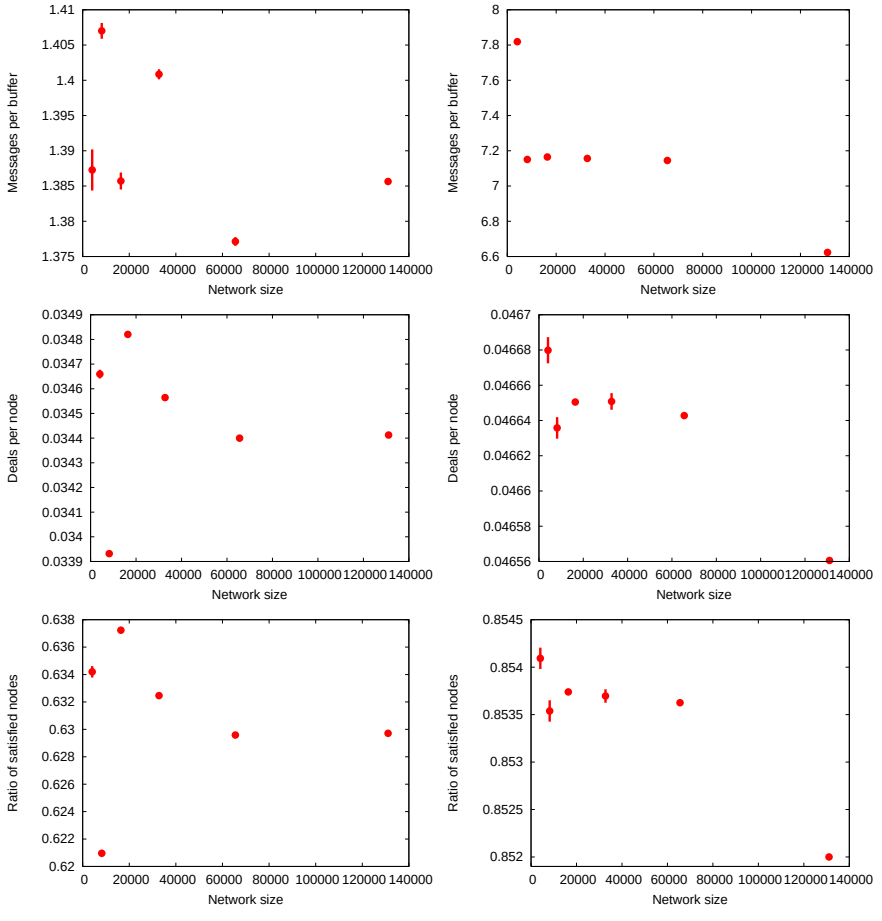


Fig. 3. Various measures for differently sized BA (left) and ER graphs (right). The graph is formed by errorbars, in which the central point of the bars is the average value of the parameter and the vertical line that goes up and down from the central point is the standard deviation of that parameter.

Similarly, the proportion of nodes that are happy per epoch is higher for random networks (roughly 85%) than BA graphs (about 63%). Neither shows much dependence on the network size. This measure can be thought of as the *utilisation* of the system.

We made the same simulations for the MaGoG dynamic version of the system, whose results are shown in Fig. 4. In this case, the average number of messages in the buffers is bigger, mainly due to the fact that, in MaGoG, nodes keep a copy of the messages they receive. More specifically, the number of messages per buffer converges to 6 in a BA graph and to 9.4 (out of a maximum of 10) in an ER graph. These results show little dependence on the network size.

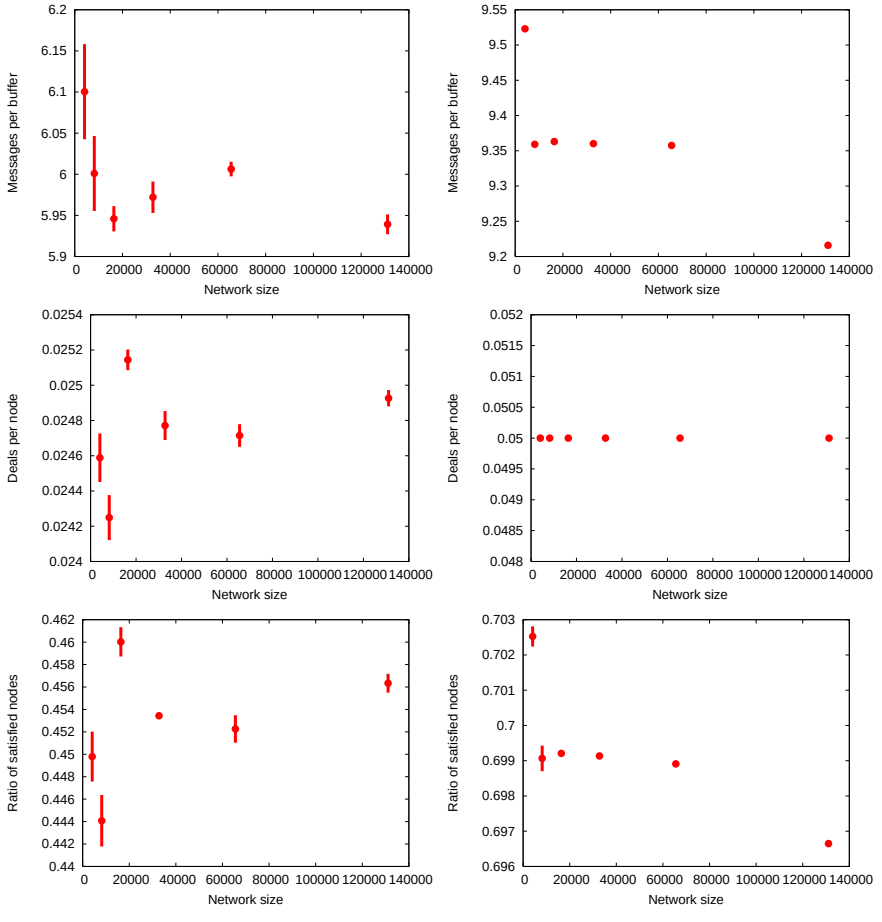


Fig. 4. Various measures for differently sized BA (left) and ER graphs (right). This time the MaGoG algorithm was used. The graph is formed by errorbars, in which the central point of the bars is the average value of the parameter and the vertical line that goes up and down from the central point is the standard deviation of that parameter.

In relation to the volume of contracts, 2.5% of nodes make a deal per epoch in a BA network, whereas this number goes up to 4% in an ER network. These values, compared with the MaGoG dynamic version, are slightly lower for the case of a BA graph, and slightly higher for the case of an ER graph. Once again, there is no dependence on the network size.

On the other hand, the utilization of the system in the MaGoG version is lower compared with the plain vanilla one. Around 45% of the nodes remain happy in a BA graph, and 70% of them do so in an ER graph. Networks of different sizes achieve the same value.

As one can see in Fig. 4, in the MaGoG version with BA graphs, it appears that the values of the observed measures vary more than in the plain vanilla

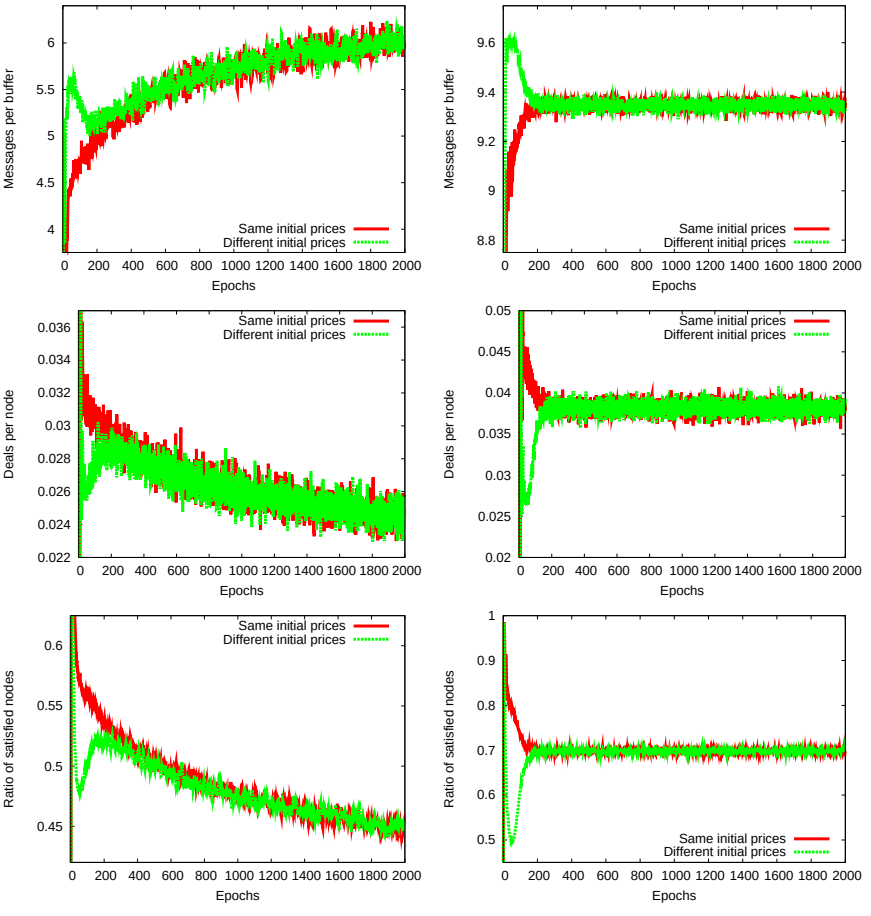


Fig. 5. Comparison between the two different initial setup for prices in the evolution of different parameters in the MaGoG dynamic version of the model. In one of them, all buyers/sellers start bidding/asking the same price. In the other one, buyers/sellers start bidding/asking different prices that have been uniformly picked up from an interval. The three figures on the left column are the results for a BA graph of 65536 nodes, whereas the three figures on the right column are the results for an ER graph of the same amount of nodes.

one. However, this is due to the fact that, in this case, the system takes longer to reach an equilibrium than in the plain vanilla version, and that part of the data that was considered to calculate the average (and standard deviation) in the MaGoG version was taken from the transient state, where the equilibrium was not completely achieved.

5.3 Variation in the Initial Price

With the objective of testing the stability of the system, we changed the values of the initial prices the buyers and the sellers bid/ask. So in this case, instead of

giving all sellers a fixed initial ask price and all buyers a fixed initial bid price, we pick uniformly distributed random numbers from two overlapping intervals (one for the buyers and one for the sellers).

Fig. 5 shows, by overlaying the plots of the two different conditions for the initial prices of the nodes, that after a different start, the system rapidly converges to the same evolution for both cases, which shows the stability of the system and its tendency towards equilibrium.

5.4 Price Evolution in the MaGoG Dynamic Version

We have simulated the price evolution in the MaGoG dynamic version of the system. In this implementation, unsatisfied nodes that have been picked a certain number of times and that have not reached any deal, increase their price by a parameter in the case of the buyers, and decrease it by the same parameter in the case of the sellers. Also, satisfied nodes that are set to be unsatisfied again (and so they re-enter the market) start asking/bidding with a different price from the one they had at the moment they closed their deal. This new price is set according to two different pricing strategies. With the first pricing strategy (the average pricing strategy), the new price is the price at which the contract was made, i.e. the average between the previous price of the seller and the previous price of the buyer. The second strategy (the Δ_p pricing strategy) uses as a new price the price at which the contract was made, but increased (for the seller) or reduced (for the buyer) by a coefficient given in the parameters.

For both pricing strategies, we ran simulations to obtain the evolution of the *ask price* (average price the sellers ask for), the *bid price* (average price the buyers bid) and the *deal price* (average price at which contracts are made). Fig. 6 shows the results of these simulations. In the top two figures the *ask* and *deal price* are almost identical and hard to distinguish.

As one can immediately see from Fig. 6, the kind of network determines a completely different evolution of prices. In an ER graph, prices decrease, whereas in a system defined by a BA graph, prices follow a nearly exponential increase versus time. Changing the pricing strategy introduces a small difference in the price evolution, but it is still the kind of graph the decisive factor that determines the general evolution of prices.

In our pricing strategy, nodes change their prices in two occasions: when they have been in the unsatisfied state for too long and when they re-enter the market after having closed a deal. In the first case, the change tends to increase prices, whereas in the second case prices tend to decrease. The ratio of the number of times that case one occurs and the number of times that case two occurs determines the final price evolution in the system. In a BA graph, the degree of connections is low, which means that nodes need more time to reach an agreement, and when the agreement is not reached after a node has been picked TTL times, nodes change their price according to first case explained above (this fact increases prices). Since, in a BA graph, this action is much more common than closing a deal quickly, prices increase heavily in this kind of networks. On the other hand, in an ER graph, the degree of connections is high, which makes faster for the nodes to

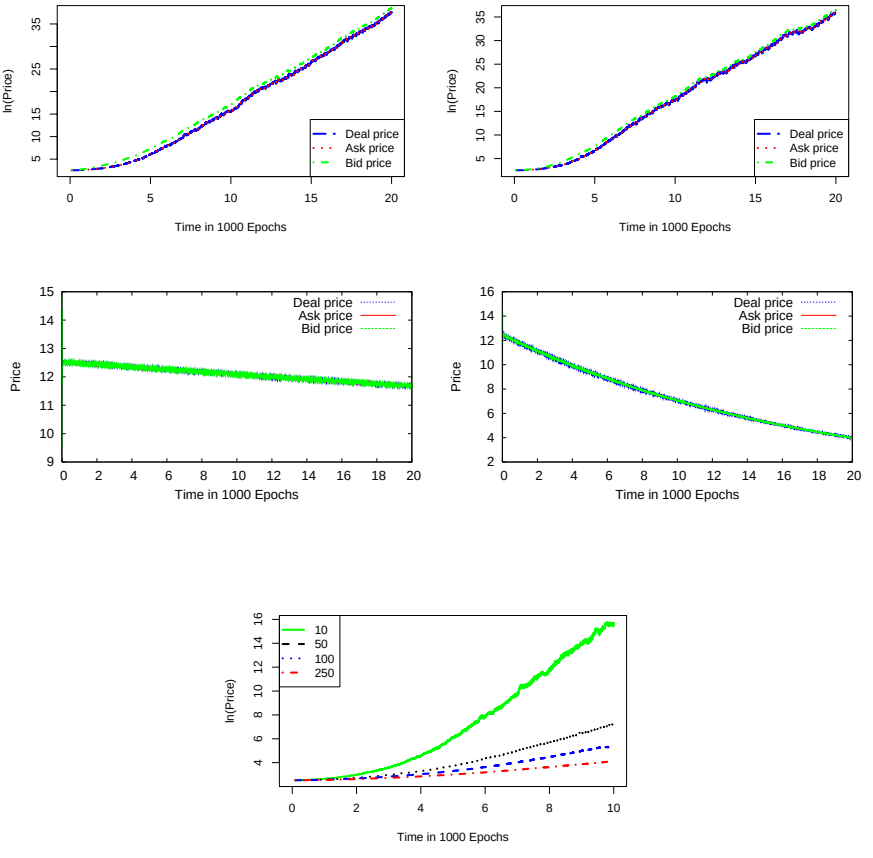


Fig. 6. Price evolution in the MaGoG dynamic version of the system for a graph of 65536 nodes; BA (top) and ER (bottom); average pricing strategy (left) and Δ_p pricing strategy (right). The bottom plot shows the price evolution in the MaGoG dynamic version of the system for different buffer sizes in a BA graph.

close a deal. When the deal is closed, nodes change their prices according to the second case described above (this reduces the prices). Since this is the most common situation in an ER graph, prices tend to decrease in the long run. This is why the evolution of prices is so different depending on the kind of graph.

An additional verification of this explanation is found when one increases the buffer size of the nodes in the BA graph. Since now more messages can meet in the nodes' pubs, and so the probability of closing more deals is higher, the behaviour of the system is more similar to what happens in an ER graph, and prices do not increase so quickly. However, prices still increase because of the lower number of times deals a closed compared with the number of times that the unsatisfied nodes place their new offers after not having reached an agreement since a certain number of epochs. The lower increase in prices in a BA graph

due to a higher buffer size is shown in last plot of Fig. 6 for a graph of 65536 nodes using average pricing strategy and different buffer sizes. Although the simulations carried out with a large size of buffers take much longer to compute, it would be interesting to calculate which size of buffers in the BA graph changes the price evolution to be similar to the one in an ER graph.

Obviously, the exponential growth in prices that takes place in a BA graph is not a realistic behaviour in real markets, since buyers' budgets are limited. In a future work, we plan to add some constraints to this specific issue of our model or change the pricing mechanism in order to achieve a more realistic approximation.

5.5 Scalability

We have analyzed the execution time of the simulations depending on the size and the kind of the network.

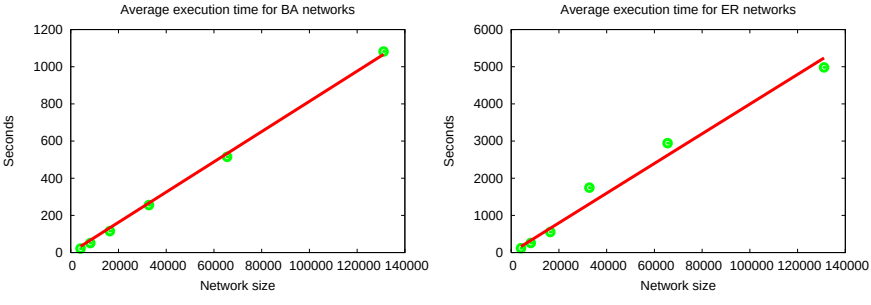


Fig. 7. Average execution time for different sized networks (BA on the left and ER on the right)

Fig. 7 shows these results for differently sized BA and ER graphs. In a BA network, the execution time has a linear increase with the network size; in an ER network, the increase can still be approximated linearly, although is not so precise as in the BA network. On the other hand, the execution time in an ER network is bigger than in a BA network for the same number of nodes. In the Fig. 7, the points are the exact experimental values, and the line is a linear approximation.

6 Conclusions and Future Work

In this paper we have presented a peer-to-peer market model for Grid Computing, in which owners of computing resources of different capacities and potential users will be able to exchange the use of resources for real money. The peer-to-peer model provides the system with the possibility of increasing indefinitely without the need of a central point of service.

We have shown in our simulations that the system actually tends to an 'equilibrium', where the values of utilization of resources, volume of contracts and

messages in buffers get stable, generally independently of the size of the network. The system takes longer to reach equilibrium for the MaGoG version, although is in this version where parameters achieve more stable values once the equilibrium has been reached. Although the time it takes to the system to reach an equilibrium has been investigated, a future analysis will include how much time it takes, for an individual average node, to reach the satisfied state.

We have investigated the effects of using two different kinds of networks (Barabási-Albert and Erdős-Rényi) and networks of different sizes. The simulations have shown that the size of the network does not change the evolution of the system or the observed values achieved at equilibrium, whereas the kind of network does change them, especially the price evolution of the system.

The results achieved in the simulation for the price evolution of the system raise interesting questions.

- Since the system is distributed, does the structure of the network actually determine a different evolution of prices?
- Can we know the future structure of the network?
- Can nodes, then, predict price behaviour according to the network structure?

In relation to this matter and taking into account that nodes exchange real money for the computing resources, one would like to make nodes more intelligent, so that they can act as real traders, and even take advantage of the arbitrage opportunities of the distributed system. In future work we shall also cover the analysis of prices in different parts of the network.

On the other hand, further work will need to define a more precise computing market, where nodes trade over several requests at the same time and take into account the number of 'computational time units' that the buyers demand.

Another obvious change to the system would be to allow nodes to change from buyers to sellers and vice versa, or investigate systems with an uneven distribution of buyers and sellers.

We also hope to be able to validate aspects of the models against real data from either the Global Open Grid or a similar architecture. Unfortunately the Global Open Grid is still in development and can not provide real data yet.

Acknowledgments

UH would like to thank Maya Paczuski, Vishal Sood and Colin Richardson for ideas and discussion.

References

1. Foster, I., Kesselman, C.: Computational Grids. ch. 2. In: The Grid: Blueprint for a New Computing Infrastructure, Morgan-Kaufman, San Francisco (1999), `http://www.globus.org/research/papers/chapter2.pdf`
2. Richardson, C.: Growing the Global Open Grid: Design Brief and Middleware Architecture. Technical report, The Internet Centre, Imperial College, London (2007)

3. Ardaiz, O., Artigas, P., Eymann, T., Freitag, F., Navarro, L., Reinicke, M.: The catallaxy approach for decentralized economic-based allocation in grid resource and service markets. Applied Intelligence 25(2), 131–145 (2006)
4. Aberer, K., Cudré-Mauroux, P., Datta, A., Despotovic, Z., Hauswirth, M., Punceva, M., Schmidt, R.: P-grid: a self-organizing structured p2p system. SIGMOD Rec. 32(3), 29–33 (2003)
5. Eymann, T., Padovan, B., Schoder, D.: The catallaxy as a new paradigm for the design of information systems. In: Proceedings of the 16th IFIP World Computer Congress, Conference on Intelligent Information Processing (2000)
6. Sutherland, I.E.: A futures market in computer time. Commun. ACM 11(6), 449–451 (1968)
7. Cotton, I.W.: Microeconomics and the Market for Computer Services. Computing Surveys 7(2), 95–111 (1975)
8. Nielsen, N.R.: The Allocation of Computing Resources – Is Pricing the Answer? Communications of the ACM 13(8), 467–474 (1970)
9. Harder, U., Harrison, P., Paczuski, M., Shah, T.: A dynamic model of a GRID market. In: Proceedings of Invited poster at ACM/IEEE Mascots 2004 (October 2004) ISBN 0-7695-2251-3
10. Lai, K., Rasmusson, L., Adar, E., Zhang, L., Huberman, B.A.: Tycoon: An implementation of a distributed, market-based resource allocation system. Multiagent Grid Syst. 1(3), 169–182 (2005)
11. Bagnall, A., Smith, G.: A multiagent model of the UK market in electricity generation. Evolutionary Computation, IEEE Transactions 9(5), 522–536 (2005)
12. Harbord, D., Fabra, N., von der Fehr, N.H.: Modeling electricity auctions. The Electricity Journal 15(7), 72–81 (2002)
13. Rosvall, M., Sneppen, K.: Self-assembly of information in networks. Europhysics Letters 74, 1109 (2006)
14. Ghoshal, G., Newman, M.E.J.: Growing distributed networks with arbitrary degree distributions. The European Physical Journal B 59, 75 (2007)
15. Oram, A. (ed.): 14 (Performance) by Theodore Hong. In: Peer-to-Peer: Harnessing the Power of Disruptive Technologies. O'Reilly and Associates, Sebastopol (2001)
16. Lua, K., Crowcroft, J., Pias, M., Sharma, R., Lim, S.: A survey and comparison of peer-to-peer overlay network schemes. IEEE Communications Surveys & Tutorials, 72–93 (2005)
17. Erdős, P., Rényi, A.: On random graphs I. Publ. Math (Debrecen) 6, 290–297 (1959)
18. Watts, D.J., Strogatz, S.H.: Collective dynamics of small-world networks. Nature 393, 440 (1998)
19. Barabasi, A.-L., Albert, R.: Emergence of scaling in random networks. Science 286, 173 (1999)
20. Evans, T.: Complex networks. Contemporary Physics 45, 455–474 (2004)
21. Wang, F., Moreno, Y., Sun, Y.: Structure of peer-to-peer social networks. Physical Review E 73, 36123 (2006)
22. Huberman, B.A., Glance, N.S.: Evolutionary Games and Computer Simulations. In: Proc. Natl. Acad. Sci. USA, August 1993, pp. 7716–7718 (1993)
23. Caldwell, B. (ed.): The Collected Works of F. A. Hayek. University of Chicago Press, Chicago (2007)
24. Walras, L.: Elements of Pure Economics. George Allen and Unwin, London (1954)
25. Csàrdi, G., Nepusz, T.: The igraph software package for complex network research. InterJournal Complex Systems, 1695 (2006)

Perfect Simulation of Stochastic Automata Networks*

Paulo Fernandes[1], Jean-Marc Vincent[2], and Thais Webber[3,**]

[1] PUCRS–CNPq
Paulo.Fernandes@pucrs.br
[2] LIG–MESCAL
Jean-Marc.Vincent@imag.fr
[3] PUCRS–CAPES
twebber@inf.pucrs.br

Abstract. The solution of continuous and discrete-time Markovian models is still challenging mainly when we model large complex systems, for example, to obtain performance indexes of parallel and distributed systems. However, iterative numerical algorithms, even well-fitted to a multidimensional structured representation of Markov chains, still face the state space explosion problem. Discrete-event simulations can estimate the stationary distribution based on long run trajectories and are also alternative methods to estimate performance indexes of models. Perfect simulation algorithms directly build steady-state samples avoiding the warm-up period and the initial state bias of forward simulations. This paper introduces the concepts of backward coupling and the advantages of monotonicity properties and component-wise characteristics to simulate Stochastic Automata Networks (SAN). The main contribution is a novel technique to solve SAN descriptions originally unsolvable by iterative methods due to large state spaces. This method is extremely efficient when the state space is large and the model has dynamic monotonicity because it is possible to contract the reachable state space in a smaller set of maximal states. Component-wise characteristics also contribute to the state space reduction extracting extremal states of the model underlying chain. The efficiency of this technique applied to sample generation using perfect simulation is compared to the overall efficiency of using an iterative numerical method to predict performance indexes of SAN models.

1 Introduction

The solution of Discrete and Continuous-Time Markov Chains (MC) [1] is still challenging mainly when we model large complex systems, such as parallel and distributed systems. The size of the infinitesimal generator to be stored has limits and also the available numerical algorithms must deal with more huge and complex representations. The steady-state is given by the long-run probability distribution obtained by the solution of the linear system $\boldsymbol{\pi}\mathcal{Q} = 0$, where $\boldsymbol{\pi}$ is the probability vector of size n which is initially distributed as $\boldsymbol{\pi}_0$ where $\boldsymbol{\pi}_0 \geq 0$ and $\sum_{i=0}^{n} \boldsymbol{\pi}_i = 1$. However, even with algorithms well-fitted to a multidimensional structured representation of MC, the state space explosion is still a problem when solving models.

* The order of the authors is merely alphabetical.
** Corresponding author.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 249–263, 2008.

Stochastic Automata Networks (SAN) is considered a high-level formalism [2] to represent structured MC. The available numerical solutions take advantage of all structural information in the original model description to obtain a compact format to store and manipulate the descriptor numerically [3,4,5]. One of the major problems with structured representations is the insertion of unreachable states in the product state space, but to cope with that there are very efficient approaches to generate the reachable set [6,7]. It remains an open problem the efficient solution of large and complex models where all, or almost all, states are reachable.

Simulation approaches are alternative methods to estimate indexes of performance models when the numerical solution is no longer sufficient. Based on discrete-event simulation or on Markov properties, simulations estimate the stationary distribution $\boldsymbol{\pi}$ based on long run trajectories. The first approaches to simulate a SAN, or any other structured formalism, focus on the concept of events [8]. Such event-driven dynamics implements a hierarchy of events inside the automata structure starting from a predefined initial global state. Despite of this event-driven choice, the problem of how long one have to run the simulation, *i.e.*, the *burn-in time* period, still remains open in forward simulations [9]. The system is simulated until it is considered that reached the stationary regime. After this time, the simulation is no more dependent of the initial state chosen due to the stationary assumption.

Propp and Wilson [10] proposed a backward coupling simulation method where the problem of biased samples is completely solved. Perfect simulation algorithms directly build steady-state samples avoiding the warm-up period and the initial state bias. The method proposes the running of trajectories in parallel, starting from all possible states, and their coupling guarantees the samples confidence. This method is extremely efficient when the state space is large and the model has dynamic monotonicity because this will determine the number of trajectories in parallel needed to run.

This paper introduces the concepts of backward coupling and the advantages of monotonicity properties to simulate SAN models. The structural information in the original SAN description can be used to contract even more the state space, analysing component-wise characteristics for example. The main contribution is the adaptation of a new simulation technique to SAN models originally unsolvable by iterative methods due state space explosion. Monotone backward coupling methods can run with a reduced state space since models have a monotonic behavior. The efficiency of sample generation using perfect simulation is compared to the overall efficiency of using an iterative numerical method to predict performance indexes of SAN models.

2 Stochastic Automata Networks

The Stochastic Automata Networks formalism (SAN) is an analytical method to obtain performance indexes of systems. It is proposed by Plateau [2] and its basic idea is to represent a whole system by a collection of K subsystems or chains described as K stochastic automata $\mathcal{A}^{(k)}$, with $k \in [1..K]$. In each of these automata the transitions among states are labeled by events. Each event includes probabilistic and timing information, and the network of automata has a set ξ of all possible events in the model. This framework defines a modular way to describe continuous and discrete-time Markovian

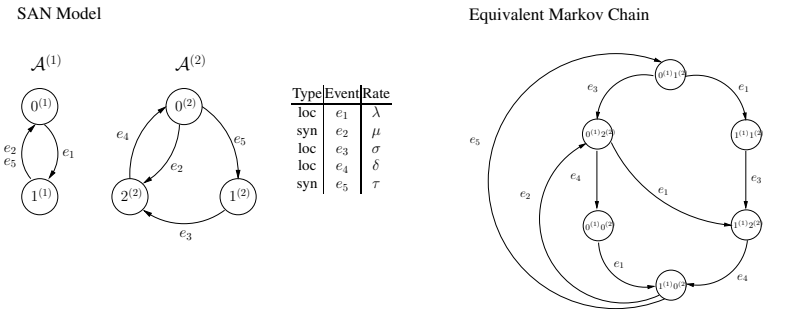


Fig. 1. Example of a SAN model and equivalent MC

models [11]. The SAN formalism has exactly the same application scope as the Markov Chain formalism [12,1].

Definition. Each automaton $\mathcal{A}^{(k)}$ has a set $\delta^{(k)}$ of local states $s^{(i)}$ where $i \in \{1 \dots n_k\}$, interconnected by transitions and their respective events. The constant n_k is the cardinality of $\delta^{(k)}$, *i.e.*, the total number of states in automaton $\mathcal{A}^{(k)}$.

Definition. A global state $\tilde{s}$ of a SAN model with K automata is a vector $\tilde{s} = \{s^{(1)}; \dots; s^{(K)}\}$ where each automaton $\mathcal{A}^{(k)}$ is in the local state $s^{(k)} \in \delta^{(k)}$.

Definition. The set of all global states is called *product state space*. The product state space $\mathcal{X}$ of a SAN model is the Cartesian product of all sets $\delta^{(k)}$.

Considering the product state space $\mathcal{X}$, the system is composed by a set of global states as $\tilde{s}$ and also a finite collection $\xi = \{e_1, \dots, e_P\}$ of P events. Since models with discrete state space can also be described as discrete-event systems [13], the set ξ can be defined with an associated transition function Φ between global states.

Definition. The transition function defined by $\Phi(\tilde{s}, e_p) = \tilde{r}$ ($p \in [1..P]$) is the set of rules that associate to each global state $\tilde{s} \in \mathcal{X}$ a new global state denoted by $\tilde{r} \in \mathcal{X}$, through the firing of the transition labeled by event $e_p \in \xi$.

In each global state $\tilde{s}$ some events are enabled, *i.e.*, they change the global state $\tilde{s}$ into another state $\tilde{r}$. However, not all events may occur from a given global state. In those cases the transition function assigns the permanence in the same global state.

Definition. An event e_p is said to be enabled in the global state $\tilde{s} \in \mathcal{X}$, iff $\Phi(\tilde{s}, e_p) = \tilde{r}$, and $\tilde{s} \neq \tilde{r}$, and $\tilde{r} \in \mathcal{X}$. Analogously, an event is said to be disabled in state $\tilde{s}$, iff $\Phi(\tilde{s}, e_p) = \tilde{r}$, and $\tilde{s} = \tilde{r}$.

The SAN model construction as a Markov process has the rates of each event e_p seen as intensities λ_p of Poisson processes, and they are supposed to be independent.

The SAN description has a table of events extracted from ξ and uniformization techniques are used to introduce the independence between these events. The uniformized

Table 1. Application of $\Phi(\tilde{s}, e_p)$ for the model of Fig. 1

$\tilde{s} \in \mathcal{X}^{\mathcal{R}}$	$\tilde{r} = \Phi(\tilde{s}, e_p), e_p \in \xi$				
	$\Phi(\tilde{s}, e_1)$	$\Phi(\tilde{s}, e_2)$	$\Phi(\tilde{s}, e_3)$	$\Phi(\tilde{s}, e_4)$	$\Phi(\tilde{s}, e_5)$
{0;0}	{**1;0**}	{0;0}	{0;0}	{0;0}	{0;0}
{0;1}	{**1;1**}	{0;1}	{**0;2**}	{0;1}	{0;1}
{0;2}	{**1;2**}	{0;2}	{0;2}	{**0;0**}	{0;2}
{1;0}	{1;0}	{**0,2**}	{1;0}	{1;0}	{**0;1**}
{1;1}	{1;1}	{1;1}	{**1;2**}	{1;1}	{1;1}
{1;2}	{1;2}	{1;2}	{1;2}	{**1;0**}	{1;2}

process is driven by the Poisson process with rate $\Lambda = \sum_{p=1}^{P} \lambda_p$ and generates at each time an event $e_p \in \xi$ according to the distribution $\left(\frac{\lambda_1}{\Lambda}, \ldots, \frac{\lambda_p}{\Lambda}\right)$.

Definition. The dynamic of the system is defined by one initial global state $\tilde{s}_0 \in \mathcal{X}$ and a sequence of events $e = \{e_p\}_{p \in \mathcal{N}}$. The sequence of states $\{\tilde{s}_n\}_{n \in \mathcal{N}}$ is a stochastic recursive sequence typically given by: $\tilde{s}_{n+1} = \Phi(\tilde{s}_n, e_{p+1})$ for $p \geq 0$ and is called a *trajectory*.

The global process execution [14,15] described is related to the underlying uniformized Markov chain. Its transitions are given by Φ applications over $\mathcal{X}$. However, it is common to have global states that are not reachable by any other global state through a transition. Due to this SAN models have established a reachable state space, *i.e.*, the set of global states $\tilde{s} \in \mathcal{X}$ that composes the related MC. The others are considered unreachable global states in the model.

Definition. The reachable state space $\mathcal{X}^R_{\tilde{s}_0}$ (or $\mathcal{X}^R$) is an irreducible component obtained from a given initial global state $\tilde{s}_0 \in \mathcal{X}$ and successive firing of events in ξ. Each global state $\tilde{s}$ reached by any possible combination of events is included in this set.

Note that SAN descriptions must have only one Markovian generator [11], the associated Markov chain contains a set of all global states $\tilde{s} \in \mathcal{X}^R$ that certainly can be reached through the firing of any event. Figure 1 is a SAN model with two automata $A^{(i)}$, and five events ($|\xi| = 5$), and their constant rates represented here by greek letters. The equivalent MC represents the reachable state space $\mathcal{X}^R$ of the model which is a subset of $\mathcal{X}$ ($\mathcal{X}^R \subseteq \mathcal{X}$).

A simple procedure to find reachable states is to apply the notion of stochastic recursive transition function mainly when the reachability function is not explicit in the SAN formal descriptions[1]. Table 1 shows the transition function application for the SAN example in Figure 1, considering all global states $\tilde{s} \in \mathcal{X}^R$ and all events $e_p \in \xi$.

[1] SAN descriptions can define the $\mathcal{X}^R$ set through the insertion of a reachability function. The boolean evaluation of this function, when applied to every global state inside $\mathcal{X}$, returns the reachable states in $\mathcal{X}^R$. More details can be found in [16,17].

The resulting global states $\tilde{r} = \Phi(\tilde{s}, e_p)$ are represented, being those corresponding to possible transitions marked in bold face, *i.e.*, those corresponding to enabled events[2].

Definition. A SAN model is called *well-formed* iff the $\mathcal{X}^R$ component is unique and irreducible.

3 SAN Backward Coupling Simulation

The first approaches to simulate SAN models focused on the concept of transitions and events, instead of having a focus on state transition matrices, *i.e.*, the descriptor [8]. Such event-driven dynamics implements a hierarchy of events inside the automata structure starting from a pre-defined initial global state. Despite of this event-driven choice, the problem of how long one should run the simulation still remains open using forward simulation approaches [9]. Moreover, simulation techniques use the *Random* function to establish the activation of an event inside ξ, considering the current global state analysed, then leading to the next global state inside $\mathcal{X}^{\mathcal{R}}$ going forward in time. The system is often simulated until it reaches its stationary regime. The duration of this step is called the *burn-in time* of the simulation and it determines that the process is no more dependent of the initial state chosen, due the stationary assumption. Since the major challenge in these techniques is to fix a *burn-in time* to allow collecting samples, the Perfect simulation technique enables to compute samples exactly distributed according to the stationary distribution of the Markov process. Propp and Wilson [10] proposed a scheme based on backward coupling, *i.e.*, the *Coupling from the Past* (CFTP) method. The problem of fixing initial state present in forward techniques is completely solved since the proposed idea is to start trajectories in parallel from all possible states.

Algorithm 1. SAN Backward coupling simulation

for all $\tilde{s} \in \mathcal{X}^R$ **do**
  $\omega(\tilde{s}) \leftarrow \tilde{s}$ { choice of the initial value of vector ω}
end for
repeat
  e $\leftarrow$ Generate-event( ) { generation of e according the distribution $(\frac{\lambda_1}{\Lambda} \ldots \frac{\lambda_\mathcal{E}}{\Lambda})$}
  $\bar{\omega} \leftarrow \omega$ { copying vector ω to $\bar{\omega}$}
  for all $\tilde{s} \in \mathcal{X}^R$ **do**
    { computing $\omega(\tilde{s})$ at time 0 of trajectory issued from $\tilde{s}$ at time $-\tau^*$}
    $\omega(\tilde{s}) \leftarrow \bar{\omega}(\Phi(\tilde{s}, e))$
  end for
until All $\omega(\tilde{s})$ are equal
Return $\omega(\tilde{s})$

The coupling of trajectories guarantees the generation of unbiased samples and it ended the *burn-in time* problem. The number of steps (or events applied) to couple all

[2] It is important to observe that the transition function Φ is a theoretical definition that is not necessarily used in current SAN solvers implementation. However, algorithms can be implemented to take advantage of transition functions identifying also the reachability set $\mathcal{X}^R$.

trajectories we denote *coupling time* τ. Figure 2 illustrates the backward coupling, all trajectories issued from all global states of the SAN example (Figure 1) at time -8 coupled in a state at time 0. Since the coupling time τ^* of the backward scheme is almost surely finite, the scheme provides a sample distributed according the steady-state distribution. Given the set of reachable states $\mathcal{X}^R$, a set $\mathcal{E}$ of randomly generated events and the transition function $\Phi : \mathcal{X}^R \times \mathcal{E} \to \mathcal{X}^R$: issuing from all global states of $\mathcal{X}^R$, going backward in time, the set of trajectories will couple for a given sequence of events $\{e_n\}_{n\in\mathcal{N}}$ at time 0, *i.e.*, $|\Phi(\mathcal{X}^R, \{e_n\}_{n\in\mathcal{N}})| = 1$.

SAN models have an underlying Markov chain so perfect simulation principles can be applied to obtain the global states probabilities in the stationary distribution. For perfect simulation execution, it is mandatory a *well-formed* SAN description, *i.e.*, the model must produce valid global states as input for the simulation algorithm. For backward simulations the set of trajectories running in parallel can be at least the $\mathcal{X}^R$ set, when of course the $\mathcal{X}^R$ set is an unordered set of global states. Algorithm 1 initializes the vector ω with all global states $\tilde{s} \in \mathcal{X}^R$ at simulation time $-\tau^*$, supposing a well-formed SAN model. At each simulation iteration one event is generated through the call of a *Random* function and the related transition functions are applied to each position of ω. Each new state generated indexes the vector $\tilde{\omega}$ which has the last version of ω stored. This process is called backward coupling because we compute $\omega(\tilde{s})$ at time 0 of trajectory issued from $\tilde{s}$ at time $-\tau^*$. This procedure will be repeated until all positions of vector ω have the same resulting state $\tilde{s}$, *i.e.*, all trajectories running in parallel have coupled. The sample of each iteration is then collected for statistical analysis.

3.1 Monotonicity Properties

The size of $\mathcal{X}^R$ can be exponential in the size of the model and it can be difficult to generate and really huge to deal, so it becomes a limitation for backward coupling methods. As pointed out in Propp and Wilson [10], CFTP methods are much easier to implement when the state space $\mathcal{X}$ is ordered and the underlying Markov chain has the monotonicity property. A known partial order of $\mathcal{X}$ is favorable to the use of monotonic functions since it allows the identification of maximal states and a considerable coupling time reduction. Models with an underlying Markov chain having this property can be optimized to run a monotone backward coupling procedure with less initial states.

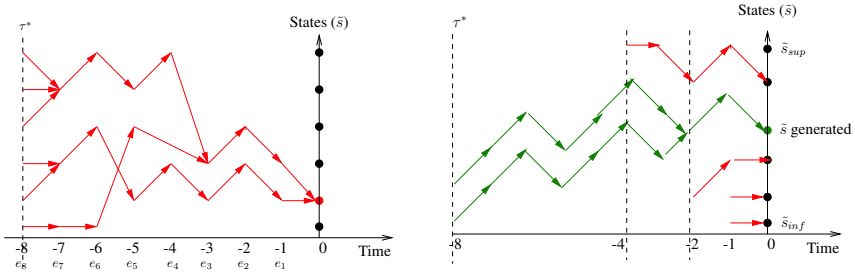


Fig. 2. Illustration of Backward and Monotone Backward coupling

Definition. An event $e_p \in \xi$ is said to be monotone if it preserves the partial ordering ($<$ order) on $\mathcal{X}$. That is $\forall(\tilde{s}, \tilde{s}') \in \mathcal{X} \quad \tilde{s} < \tilde{s}' \implies \Phi(\tilde{s}, e_p) < \Phi(\tilde{s}', e_p)$.

If all events are monotone, the global system is said to be monotone.

The monotonicity property of events guarantees the existence of a set of maximal states $\mathcal{X}^{\max}$ and a set of minimal states $\mathcal{X}^{\min}$. These sets are composed of states which there is no greater (or lower) state than itself in the chain. So the transitions fired from maximal states do not create states greater than these ones (or transitions fired from minimal states do not create states lower than minimal ones).

Definition. Suppose the global states $\tilde{s}_1, \tilde{s}_2 \in \mathcal{X}$, a state $\tilde{s}_1$ is minimal if there exists a state $\tilde{s}_2$ such that $\tilde{s}_2 \leq \tilde{s}_1$ then $\tilde{s}_2 = \tilde{s}_1$. Then $\tilde{s}_1 \in \mathcal{X}^{\min}$. Analogously, given states $\tilde{s}_3, \tilde{s}_4 \in \mathcal{X}$, a state $\tilde{s}_3$ is maximal if there exists a state $\tilde{s}_4$ such that $\tilde{s}_4 \geq \tilde{s}_3$ then $\tilde{s}_4 = \tilde{s}_3$. Then $\tilde{s}_3 \in \mathcal{X}^{\max}$.

Algorithm 2. SAN Monotone backward coupling simulation

```
1: n = 1
2: E[1] ← Generate-event( ) { array E stores the backward sequence of events}
3: repeat
4:    n ← 2n {doubling scheme}
5:    for each s̃ ∈ X^M do
6:       ω(s̃) ← s̃ { initial states at time −n}
7:    end for
8:    for i = n downto (n/2 + 1) do
9:       E[i] ← Generate-event( ) { generate events from (−n/2 + 1) to −n, events from −1 to
         (−n/2 + 1) have been generated in a previous loop}
10:   end for
11:   for i = n downto 1 do
12:      for each s̃ ∈ X^M do
13:         ω(s̃) ← Φ(ω(s̃), E[i]) { ω(s̃) is the state at time (−i − 1) of the trajectories issued
            from s̃ at time −n}
14:      end for
15:   end for
16: until All ω(s̃) are equal
17: Return ω(s̃)
```

Definition. Suppose a given partial order of $\mathcal{X}$ and consequently a maximal set $\mathcal{X}^M$, if all trajectories issued from $\mathcal{X}^M$ coupled at time 0, then they will also coalesce for all states in $\mathcal{X}^R$.

Since $\mathcal{X}$ is finite and the events are monotone, the number of trajectories in parallel can be reduced running simulation only over the $\mathcal{X}^M = \mathcal{X}^{\max} \cup \mathcal{X}^{\min}$ set. Starting trajectories and going from the past from $\mathcal{X}^M$ maximal global states, when all trajectories collapsed, we also obtain a sample of the stationary regime. Figure 2 illustrates the monotone backward coupling, where there are only one *infimum* state $\tilde{s}_{inf}$ and one *supremum* state $\tilde{s}_{sup}$ in the state space $\mathcal{X}^R$. All trajectories issued from $\tilde{s}_{inf}$ and

$\tilde{s}_{sup}$ are computed from time -2^k to 0 until trajectories collapsed at time 0. If we assume $\mathcal{X}^R$ as a *lattice*, then every state $\tilde{s}_i$ is between the state $\tilde{s}_{inf}$ and the state $\tilde{s}_{sup}$, $\tilde{s}_{inf} \leq \tilde{s}_i \leq \tilde{s}_{sup}$. Considering the SAN context, if we know the global states $\tilde{s}_{inf}$ and $\tilde{s}_{sup}$ (or a $\mathcal{X}$ subset of maximal states, *i.e.*, $\mathcal{X}^M$) we can run a monotone version. It uses a coupling vector ω of $|\mathcal{X}^M|$ positions running these trajectories in parallel. Also, it needs to store the events generated of the whole trajectory, because it uses a *doubling scheme* structure [10] to generate and apply events in each trajectory. At each step in the past, the *coupling time* τ^* needed (*i.e.*, the length of the step) is multiplied by 2 (Algorithm 2, line 4).

Canonical Component-Wise Ordering in SAN. Many models are naturally ordered as markovian queueing networks [18,19,20] due the natural order on integer. The partial order of the product state space can be established using for example component-wise ordering concepts. SAN descriptions derived from monotone queueing networks can be simulated taking advantage of having only the canonical minimum (all queues empty) and maximum (all queues full) states. Then only two paths need to simulate in parallel with the monotone algorithm version.

The canonical component-wise ordering means that the underlying Markov chain structure of the model can be viewed as a *lattice*, *i.e.*, all global states have the same *supremum* and *infimum* states. Given two arbitrary global states $\tilde{s}_1, \tilde{s}_2 \in \mathcal{X}$, and verifying $\tilde{s}_1 \leq \tilde{s}_2$, it will often possible to say which is the largest state [21]. The extremal states are given by the first and the last state of $\mathcal{X}$ considering it is ordered lexicographically. The complexity to solve these families of models is then constant (two trajectories in parallel) and the simulation is no more limited by the size of $\mathcal{X}$ but only by τ.

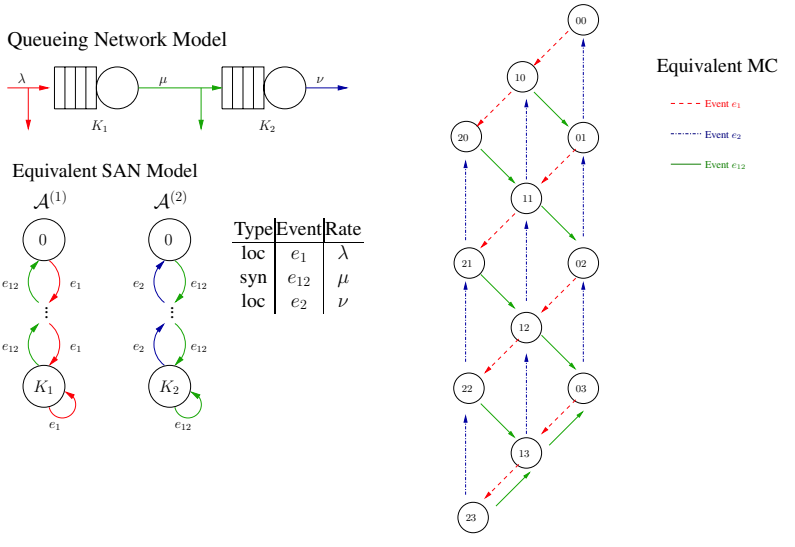


Fig. 3. QN conversion to a SAN model and the ordered $\mathcal{X}^R$

Supposing the queueing system of two queues in Figure 3 with capacities K_1 and K_2 respectively. The $\mathcal{X}$ size of this network is given by the Cartesian product $(K_1 + 1) \times (K_2 + 1)$ and all global states $\tilde{s} \in \mathcal{X}^R$ (equivalent MC) are reachable ($\mathcal{X}^R \cong \mathcal{X}$). The SAN model has two automata $A^{(1)}$ and $A^{(2)}$ respectively, and three events $e_p \in \xi$ (since two are local events and one is a synchronizing event in the model) with their rates. The events e_1, e_{12} and e_2 are monotone according canonical component-wise ordering of $\mathcal{X}$, *i.e.*, there is no event in the model changing the partial order of states in $\mathcal{X}$. The application of the transition function $\Phi(\tilde{s}, e_p)$, for each event $e_p \in \xi$, considering each state $\tilde{s} \in \mathcal{X}^R$, is dependant of the min and max functions[3] evaluations for each global state (in this example the global state to be evaluated has only two local states to observe $\tilde{s} = \{s_1; s_2\}$).

The minimum and maximum global states are extracted from the underlying Markov chain, but they consider the minimal and maximal local states of each automaton $\mathcal{A}^{(k)}$ defined by natural order on integer. Supposing $K_1 = 2$ and $K_2 = 3$, the maximal set can be considered $\mathcal{X}^M = \{\{0; 0\}, \{2; 3\}\}$. The minimal local state of both automata is the state 0, and the maximal local state is 2 for automaton $A^{(1)}$, and 3 for automaton $A^{(2)}$ respectively. The simulation could run only two trajectories in parallel: all queues empty (minimal local states $\{0; 0\}$) and all queues full (maximal local states $\{K_1; K_2\}$). The assumption of existing one minimum and one maximum local state per automaton which guarantees the exact sampling, can be applied also for huge models following component-wise principle.

Non-Lattice Component-Wise Ordering in SAN. Glasserman and Yao [13] investigated the search for partial (and total) ordering in discrete-event models looking at their own structure, naturally retaining the order in which states in the chain are accessed firing the respective events. This procedure incrementally generates a *feasible set*, until all states are accessed (total ordering), or a given partial ordering is identified. So we can consider the feasible set as an ordered representation of the $\mathcal{X}^R$ set. However, in the absence of a canonical component-wise model formation, for each event in the model, the state space ordering must be constructed firing events in the underlying chain structure. If we have the same subchains ordering for the events, this means that exists a partial order for $\mathcal{X}^R \subseteq \mathcal{X}$ ($<$), when it is possible to compare two states for a given event $e_p \in \mathcal{E}$, independent of event rates.

The ordering construction for the queueing system example when already exists a canonical formation leads us to a *lattice* where there are two maximal global states as seen in Figure 3. But without this characteristic the search for a order could be really unfeasible for huge models and with a high enhanced computational cost. The global states ordering in $\mathcal{X}$ becomes not relevant if we can extract through the transition function applications a smaller set of extremal global states of the Markov chain, *i.e.*, not retaining the order of access of states but verifying if the next state in a transition is greater than the current state. This means that if we walk in the chain applying the transition function successively we can reach and collect the extremal elements (Algorithm 3).

[3] $x \wedge y = min(x, y)$ and $x \vee y = max(x, y)$.

Algorithm 3. SAN extremal states identification in component-wise models

```
1: for each s̃ ∈ X^R do
2:    max ← true;
3:    for each e_p ∈ ξ do
4:       st ← Φ(s̃, e_p); { firing transition }
5:       if (st ≥ s̃)
6:        max ← false; break;
7:    end for
8:    if (max = true)
9:     Add state s̃ in X^M; { array X^M stores the extremal states identified}
10: end for
11: Return array X^M;
```

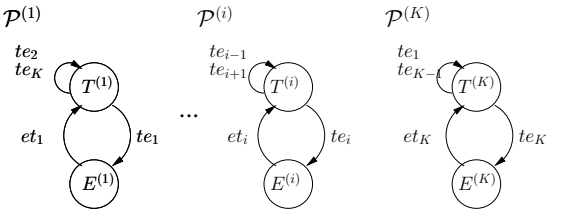
The component-wise ordering supposes that local states have a predefined order, then the Cartesian product of states generates automatically ordered global states. The states will always have transitions to greater or lower global states indexes. So when there is no possible transition to be fired to a greater state, this means we find an extremal state in the chain. In this case, models can have more than two maximal states (according to the $\mathcal{X}^R$ set analysis), *i.e.*, $\mathcal{X}^M$ is now the set of extremal elements formed by global states where the successive transition function application stops when it does not return greater states. Doing this, is not mandatory to know the state space partial ordering but only to identify the subset $\mathcal{X}^M$ of extremal states to run the monotone backward coupling algorithm. The complexity to find extremal global states is given by $|\mathcal{X}^R| \times |\xi|$. The computational cost to run perfect simulation is now dependent of the number of initial global states inside the set $\mathcal{X}^M$.

4 Resource Sharing with Mutual Exclusion

Our case study is the classical model of resource sharing with mutual exclusion and some variations. In this section we show the product state space contraction regarding natural structured formation of these models, and also issues related to the exploitation of monotonicity properties for perfect simulation. All simulation examples were executed on a PC architecture with a 3.2 GHz Intel Xeon processor under Linux operating system, with 1 GByte of memory. The execution times presented consider only usertime estimation running *PEPS* software tool and the perfect simulation module developed (Perfect *PEPS*), *i.e.*, they do not take account of other users in the machine or operating system execution.

4.1 Dining Philosophers without Reservation

The dining philosophers problem is summarized as K philosophers sitting at a table doing one of two things - eating or thinking. The philosophers sit at a circular table. with a large bowl of food in the center. A fork F_k is placed between each philosopher P_k, and as such, each philosopher has one fork to his left and one fork to his right.



Type	Event	Rate
loc	et_k	μ_k
syn	te_k	λ_k

Fig. 4. Philosophers model without reservation

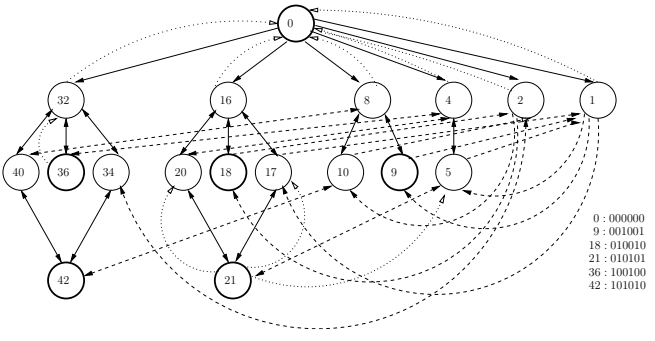


Fig. 5. Ordered $\mathcal{X}^R$ supposing 6 philosophers

The philosopher must have two forks (at the same time) to eat. Figure 4 shows the correspondent SAN model that has K automata $P^{(k)}$ representing the philosophers, each one with two states: $T^{(k)}$ (thinking) and $E^{(k)}$ (eating). The product state space $\mathcal{X}$ is formed by 2^K global states.

$\mathcal{X}^M$ Extraction. Supposing six philosophers in a table (Figure 5) the application of the transition function returns the extremal states for the SAN model. Regarding structural properties of this model all events $et_k, te_k \in \xi$ are monotone since they retain the component-wise ordering of global states in the chain formed by this class of models. Algorithm 3 finds the last states $\tilde{s} \in \mathcal{X}$ that can be accessed, *i.e.*, the extremal states $\tilde{s} \in \mathcal{X}^M$. Then component-wise property allows the feasible set formation based on indexes, because the successive application of events (tracing a trajectory) leads to a state where it is not possible to go on to a greater state, *i.e.*, they are the extremal states of the chain. For these states there are events just to go back in the paths already generated.

Considering the model global states formed by *bits* in the Figure 5, since the state $T^{(k)}$ is represented by O and $E^{(k)}$ represented by 1, we have for example, a set $\mathcal{X}^R = 18$ and $\mathcal{X}^M = 6$ (the marked states are maximal elements) for a model with six philosophers. Table 2 shows the SANmodel with $K = 6 \ldots 26$ and their respective $\mathcal{X}$, $\mathcal{X}^R$ and

Table 2. Resource sharing without reservation

K	$\mathcal{X}$	$\mathcal{X}^R$	$\mathcal{X}^M$	*PEPS* (iteration)	*Perfect PEPS* (sample)
6	64	18	6	0.000004 sec.	0.002711 sec.
8	256	47	11	0.000123 sec.	0.003464 sec.
10	1,024	123	18	0.000542 sec.	0.005682 sec.
12	4,096	322	30	0.002487 sec.	0.012290 sec.
14	16,384	843	52	0.011832 sec.	0.029745 sec.
16	65,536	2,207	91	0.055973 sec.	0.074337 sec.
18	262,144	5,778	159	0.296355 sec.	0.184355 sec.
20	1,048,576	15,127	278	1.394022 sec.	0.457599 sec.
25	33,554,432	167,761	1,131	5.115790 sec.	4.736356 sec.
26	67,108,864	392,836	2,779	—-	13.500322 sec.

the extracted set of extremal states $\mathcal{X}^M$. Since the size of the model grows exponentially the size of maximal set grows slowly comparatively. The size of $\mathcal{X}^M$ is the number of trajectories to run in parallel, *i.e.*, the number of vector positions to store, so it is also the computational cost in memory positions to run perfect simulation. Each position is an integer representing the current global state index in the trajectory. When we have more philosophers in the model the impact of this optimization is more clear mainly verifying the time spent to solve this models using *PEPS* software tool and the perfect simulation module (*Perfect PEPS*).

The actual number of samples to generate depends immensely on the numeric characteristics of the model itself. Different parameters as the actual numeric rates of the events, may change the required number of samples to achieve statistical convergence of the stationary prediction. Analogously, the numbers of iterations to perform the iterative solution methods in the *PEPS* tool also depends on the model numeric characteristics. Therefore in the Table 2 we indicate the amount of time needed to perform one single sample generation with the contracted state space in the *Perfect PEPS* module, and one single iteration in the numerical solution implemented by *PEPS*. The presented values in seconds must be considered with caution, since nothing relates the number of needed iterations in *PEPS* with the number of samples needed in our simulation tool. For example, the first model ($K = 6$) needed 528 iterations to achieve a precision of 10^{-10} in the *PEPS* solver. We obtain the simulation results running a fixed number of $100,000$ samples. However, a smaller number of samples would probably already be enough to achieve (statistically) the required precision for such small example using confidence intervals. The example was extended just beyond the capacity limit of *PEPS*, since the last example ($K = 26$) is already too huge to run on our target machine that holds problems of as many as 64 million states.

4.2 Dining Philosophers with Reservation

We can extend the mutual exclusion in resource sharing models to analyse more deeply the locking of shared resources in systems. But here the goal is to obtain an extensible model where a numerical solution is no longer possible due state space explosion, in order to show the possible product state space contraction also in these cases. Figure 6

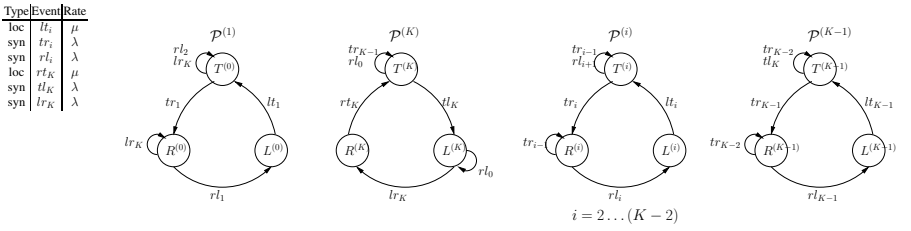


Fig. 6. Philosophers SAN Model with Reservation

Table 3. Resource sharing with reservation

K	$\mathcal{X}$	$\mathcal{X}^R$	$\mathcal{X}^M$	*PEPS* (iteration)	*Perfect PEPS (sample)*
5	243	70	11	0.000130 sec.	0.004547 sec.
6	729	169	17	0.000474 sec.	0.007829 sec.
7	2187	408	27	0.001693 sec.	0.014273 sec.
8	6,561	985	43	0.003185 sec.	0.032354 sec.
10	59,049	5,741	111	0.038100 sec.	0.111365 sec.
12	531,441	33,461	289	0.551290 sec.	0.689674 sec.
14	4,782,969	195,025	755	5.712210 sec.	2.686925 sec.
16	43,046,721	1,136,689	1,975	68.704325 sec.	15.793501 sec.
18	387,420,489	6,625,109	5,169	—-	83.287321 sec.

has K automata $P^{(k)}$ representing the philosophers, each one with three ordered states: $T^{(k)}$ (thinking), $L^{(k)}$ (taking left fork), $R^{(k)}$ (taking right fork). The philosopher can reserve the fork on his immediate left or right waiting for eating with two available forks. To avoid deadlock is established an ordering to get the forks in the table, for each philosopher in the model. The product state space $\mathcal{X}$ is formed by 3^K global states.

$\mathcal{X}^M$ Extraction. Regarding structural properties of this extended model, the monotonicity properties are also maintained for all new events generated lt_i, tr_i, rl_i, rt_k, tl_k and lr_k. The inclusion of a new state in each automaton and new events constraints does not interfere in $\mathcal{X}$ partial ordering. Since the minimal global state 0 (all philosophers thinking) is the initial state to generate the feasible set of the model, the extremal states are naturally the ones with greater indexes than their consequent transitions.

Table 3 shows in its last lines, huge models to solve with *PEPS* software tool mainly because the size of $\mathcal{X}$, and the possible contraction of state space in $\mathcal{X}^M$ to run perfect simulation. The costs in memory are drastically reduced since for monotone versions we used to store just the coupling vector with extremal elements instead of the product state space. The same remarks still apply to the times presented here, specially the fact that this table present times for one iteration in the *PEPS* numerical solution, and one sample generation for *Perfect PEPS*. For the last model ($K = 18$) the *PEPS* solution could not be achieved since it represents a state space of more that 327 million states, which is considerably above the current overall numerical solution limitation that is a little below 100 million states in a 4GBytes memory machine.

5 Conclusions

We show that is possible to design a perfect sampling algorithm for SAN through backward coupling. For the underlying Markovian graphs, the simulation coupling time can be greatly reduced by using extremal initial states to run trajectories in parallel. In fact, this paper not even present the times for sample generation without using the state space contraction because even the average models, *e.g.*, Resource Sharing without reservation $K = 14$, would represent a model much slower than the larger model we were able to solve (Resource Sharing with reservation $K = 18$). However, the study of coupling times considering the maximal set $\mathcal{X}^M$ is a work in progress. Preliminary results shows that for the models which the numerical solution is no longer possible with *PEPS*, perfect simulation seems to be a reasonable alternative. The natural future work in that direction is a study of the effects of our technique in the statistical convergence. Such study could also compare the time of a full stationary solution using perfect simulation and the traditional numerical solution.

The current limitation of *PEPS* software tool is in order of $\mathcal{X} \leq 6 \times 10^7$ states using 1 Gbyte RAM machine because it needs to store probability vectors for the state space $\mathcal{X}$. SAN simulation approaches, on the contrary, will work only with vectors of size related to maximal sets of the models. Even further, in the case we have a particular interest in a given aggregation function of result, *e.g.*, compute the probability of a local state, our solution may avoid to store the probability vectors for the maximal sets, but only compute the aggregation function over the generated samples. This last improvement combined with the monotonicity property identified in models with a component-wise order can help us to solve really huge models. In fact, this approach may, virtually, have no size bound, since not the transition matrix, nor the probability vector can be stored. Such possible applications will only have a time bound to be considered, and since the generation of samples can be performed in parallel, even this time bound can be overcome for models with thousands of millions states. Only the statistical analysis of the generated samples will have to be dealt. Nevertheless, all these reasons let us believe that the perfect simulation of SAN with component-wise ordering allowing state space contraction is a more than worthy solution when the use of numerical methods is just not possible with the current technology.

Acknowledgment

This work is partially supported by the brazilian government (CAPES grants 1341/07-3 and 2265/06-0), FINEP project STGSD grant nb. 4284/05 and ANR SETIN Checkbound and ANR blanche SMS (France). Author ThaisWebber is invited by the Project MESCAL-INRIA (Thais.Webber@imag.fr).

References

1. Stewart, W.J.: Introduction to the numerical solution of Markov chains. Princeton University Press, Princeton (1994)
2. Plateau, B.: On the stochastic structure of parallelism and synchronization models for distributed algorithms. In: ACM Sigmetrics Conference on Measurements and Modeling of Computer Systems, pp. 147–154. ACM Press, New York (1985)

3. Benoit, A., Fernandes, P., Plateau, B., Stewart, W.J.: On the benefits of using functional transitions and Kronecker algebra. Performance Evaluation (in Press, 2004)
4. Buchholz, P., Ciardo, G., Donatelli, S., Kemper, P.: Complexity of memory efficient Kronecker operations with applications to the solution of Markov models. INFORMS Journal on Computing 13(3), 203–222 (2000)
5. Fernandes, P., Plateau, B., Stewart, W.J.: Efficient vector-descriptor multiplication in Stochastic Automata Networks. Journal of the ACM 45(3), 381–414 (1998)
6. Buchholz, P., Kemper, P.: Hierarchical reachability graph generation for Petri nets. Formal Methods in Systems Design 21(3), 281–315 (2002)
7. Miner, A.S., Ciardo, G.: Efficient Reachability Set Generation and Storage Using Decision Diagrams. In: Donatelli, S., Kleijn, J. (eds.) ICATPN 1999. LNCS, vol. 1639, pp. 6–25. Springer, Heidelberg (1999)
8. Jungblut-Hessel, R., Plateau, B., Stewart, W.J., Ycart, B.: Fast simulation for Road Traffic Network. RAIRO Operational Research 35, 229–250 (2001)
9. Häggström, O.: Finite Markov Chains and Algorithmic Applications. Based on lecture notes. Cambridge University Press, Cambridge (2002)
10. Propp, J.G., Wilson, D.B.: Exact Sampling with Coupled Markov Chains and Applications to Statistical Mechanics. Random Structures and Algorithms 9(1-2), 223–252 (1996)
11. Plateau, B., Atif, K.: Stochastic Automata Networks for modelling parallel systems. IEEE Transactions on Software Engineering 17(10), 1093–1108 (1991)
12. Brenner, L., Fernandes, P., Sales, A.: The Need for and the Advantages of Generalized Tensor Algebra for Kronecker Structured Representations. In: 20^{th} Annual UK Performance Engineering Workshop, Bradford, UK, July 2004, pp. 48–60 (2004)
13. Glasserman, P., Yao, D.D.: Monotone structure in discrete-event systems. John Wiley & Sons, Inc., New York (1994)
14. Borovkov, A.A., Foss, S.G.: Two ergodicity criteria for stochastically recursive sequences. ACTA Applicande Math. 34, 125–134 (1994)
15. Stenflo, O.: Ergodic Theorems for Markov chains represented by Iterated Function Systems. Bull. Polish Acad. Sci. Math. 49(1), 27–43 (2001)
16. Benoit, A., Brenner, L., Fernandes, P., Plateau, B., Stewart, W.J.: The PEPS Software Tool. In: Kemper, P., Sanders, W.H. (eds.) TOOLS 2003. LNCS, vol. 2794, pp. 98–115. Springer, Heidelberg (2003)
17. Brenner, L., Fernandes, P., Plateau, B., Sbeity, I.: PEPS2007 - Stochastic Automata Networks Software Tool. In: QEST 2007, pp. 163–164. IEEE Press, Los Alamitos (2007)
18. Vincent, J.M.: Perfect simulation of monotone systems for rare event probability estimation. In: Winter Simulation Conference, pp. 528–537. ACM, New York (2005)
19. Vincent, J.M.: Perfect Simulation of Queueing Networks with Blocking and Rejection. In: International Symposium on Applications and the Internet Workshops: SAINT Workshops, pp. 268–271. IEEE Computer Society, Los Alamitos (2005)
20. Vincent, J.M., Vienne, J.: Perfect simulation of index based routing queueing networks. SIGMETRICS Performance Evaluation Review 34(2), 24–25 (2006)
21. Dimakos, X.K.: A Guide to Exact Simulation. International Statistical Review 69(1), 27–48 (2001)

Model Checking of Infinite State Space Markov Chains by Stochastic Bounds*

Mouad Ben Mamoun[1] and Nihal Pekergin[2]

[1] Département Mathématiques et Informatique, Université Mohammed V, B.P 1014, Rabat, Maroc
ben_mamoun@fsr.ac.ma
[2] LACL, Université Paris Est, 61 av. du General de Gaulle 94010 Créteil, France
nihal.pekergin@univ-paris12.fr

Abstract. In this paper, we discuss how to check Probablistic Computation Tree Logic (PCTL) logic operators over infinite state Discrete Time Markov Chains (DTMC). Probabilistic model checking has been largely applied over finite state space Markov models. Recently infinite state models have been considered when underlying infinite Markov models have special structures. We propose to consider finite state models providing bounds on transient and the stationary distributions in the sense of the $\preceq_{st}$ stochastic order to check infinite state models. The operators of the PCTL logic are then checked by considering these finite bounding models.

1 Introduction

Model checking is a method to automatically check if complex performability guarantees expressed by using formal logics are satisfied or not. Stochastic model checking is a recent extension of traditional model-checking techniques for the integrated analysis of both qualitative and quantitative system properties. Model checking for different classes of stochastic processes and specification logics have been developed [4,12,8] and have been also implemented in different model checkers [16,13]. However in almost all works, the state space size is considered to be finite. To perform model checking by numerical analysis we need to compute transient-state or steady-state distribution of the underlying Markov chain [5]. The numerical methods exist only for finite state models, however for special structured chains like QBD (Quasi Birth Death) models despite the infinite state spaces efficient numerical algorithms called matrix-geometric solutions exist. In [19], Continuous Stochastic Logic (CSL) over Continuous Time Markov Chains (CTMC) model checking has been extended to infinite space QBD models, and in [20] models with product-form solutions have been considered.

In this paper we propose to consider model checking of general infinite Markov chains with the stochastic comparison techniques. These techniques have been

* This work is supported by French research project CheckBound, ANR06-SETIN-002.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 264–278, 2008.

largely applied in different areas of applied probability as well as in reliability, performance evaluation, dependability applications [17]. Intuitively speaking, this method consists in computing bounding distributions rather than the exact distributions by analysing "simpler" bounding chains.

Bounding methods can be applied in model checking context since one needs to verify if some thresholds are satisfied or not without computing the exact values. In [2], the bounds on state reachability probabilities of Markov decision processes are computed by abstraction of the underlying model defined on smaller state spaces. If the verification of the considered property can not be concluded, the abstract model is refined until a verdict to the property can be deduced from the computations. The stochastic comparison techniques have been applied in [18,7] to overcome state space explosion problem in the model checking context. In [18], PRCTL [3] state formulas are considered by using stochastic bounding techniques. In [7], a method to simplify the checking of CSL operators by means of class $\mathcal{C}$ bounding Markov chains having closed-form solutions for transient and the steady-state distribution is given.

Our approach in this work is based on the truncation of the underlying infinite chains which are intractable and the computation of finite stochastic matrices providing, via stochastic comparison, bounds on the relevant probability quantities for the model checking. The idea of truncation is very natural and seems necessary to be able to deal numerically with general infinite DTMCs. It has been proposed to compute approximations of stationary distributions of infinite Markov chains [21,14]. However approximations are not useful for model checking. Moreover we need to compute bounds also on transient distributions in order to check path formulas and transient operators.

In the model checking context we must sum the probabilities of a set of states from a distribution of the underlying model. This set of states depends on the considered formula and the distribution is the steady-state distribution for the stationary operator while we must consider a transient distribution for a path formula. The stochastic comparison has the advantage of providing bounds on the steady-state as well as transient distributions. Moreover the $\preceq_{st}$ stochastic order considered in this work allows to deduce bounds on the partial sums of distributions. In this paper, since we must establish the stochastic comparison of distributions, one having finite size and the other having the infinite size, we apply the stochastic comparison of the images of these distributions on a common space. We present this method and discuss its application to check different formulas depending also on the comparison operator $\leq$ or $\geq$.

The remaining of the paper is organised as follows: In section 2, we give a brief introduction on the stochastic comparison approach and the Probabilistic real time Computation Tree Logic (PCTL) for Discrete Time Markov Chains (DTMC)s. Section 3 is devoted to the bounding of infinite DTMCs by finite DTMCs. We explain in section 4, how these bounds can be used to check PCTL operators over infinite DTMCs.

2 Preliminaries

2.1 Stochastic Comparison

Stochastic comparison is an useful tool to compare random variables and stochastic processes when studying stochastic systems. First, we give the basic definitions and theorems letting to compare random variables and DTMCs defined on the *same state space* with respect to the usual stochastic order $\preceq_{st}$. Secondly we present an interesting extension, called fg-comparison that we apply in this work. This extension introduced by Doisy [9] allows to compare random variables and DTMCs which are not defined on the same state space by means of *state functions* f and g. For further informations on the stochastic comparison we refer to Stoyan's book [17] as the main reference in the domain and to the works [9], [10] and [22] for the fg-comparison.

Comparison of DTMCs on the Same State Space

Definition 1. *Let X and Y be two random variables (r.v) taking values on a totally ordered space E, and $\mathcal{F}_{st}$ the class of all increasing real functions on E.*
$X \preceq_{st} Y \Longleftrightarrow Ef(X) \leq Ef(Y), \quad \forall f \in \mathcal{F}_{st}$ *whenever the expectations exist.*

Property 1. For two r.v X and Y taking values on a totally ordered space E
$X \preceq_{st} Y \iff Prob(X > a) \leq Prob(Y > a), \forall a \in E$

In the case of finite state space $\{0, 1, \ldots, N\}$, the $\preceq_{st}$-comparison of random variables can be characterised through the following probability inequalities.

Property 2. Let X and Y be two r.v taking values on $E = \{0, 1, \ldots, N\}$, and $p = [p_0, \ldots, p_N]$, $q = [q_0, \ldots, q_N]$ be probability distributions of X and Y.

$$X \preceq_{st} Y \Leftrightarrow \sum_{k=i}^{N} p_k \leq \sum_{k=i}^{N} q_k \quad for \quad i = N, N-1, \cdots, 0. \tag{1}$$

$$X \preceq_{st} Y \Leftrightarrow \sum_{k=0}^{j} p_k \geq \sum_{k=0}^{j} q_k \quad for \quad j = 0, 1, \cdots, N. \tag{2}$$

Let us remark that in the sequel we interchangeably use the notations $X \preceq_{st} Y$ and $p \preceq_{st} q$. We apply the following definition to compare Markov chains.

Definition 2. *Let $\{X(n)\}$ (resp. $\{Y(n)\}$) be a DTMC. We say $\{X(n)\} \preceq_{st} \{Y(n)\}$, if $X(n) \preceq_{st} Y(n), \ \forall n$.*

In the case of time-homogeneous DTMC chains, the monotonicity and the comparability of transition matrices yield sufficient conditions to compare stochastically the underlying chains [17, p.186].

Theorem 1. *Let* $\mathbf{P}$ *(resp.* $\mathbf{Q}$*) be the transition matrix of the time-homogeneous Markov chain* $\{X(n)\}$ *(resp.* $\{Y(n)\}$*). The comparison of Markov chains is established (*$\{X(n)\} \preceq_{st} \{Y(n)\}$*), if the following conditions are satisfied :*

i- $X(0) \preceq_{st} Y(0)$,
ii- *at least one of the probability transition matrices is monotone, that is, either* $\mathbf{P}$ *or* $\mathbf{Q}$ *is* $\preceq_{st}$ *monotone :*
$\forall i, j$ *such that* $i \leq j$, *either* $\mathbf{P}[i,*] \preceq_{st} \mathbf{P}[j,*]$ *or* $\mathbf{Q}[i,*] \preceq_{st} \mathbf{Q}[j,*]$
iii- *the transition matrices are comparable in the sense of the* $\preceq_{st}$ *order :*

$$\mathbf{P} \preceq_{st} \mathbf{Q} \Longleftrightarrow \mathbf{P}[i,*] \preceq_{st} \mathbf{Q}[i,*], \quad \forall i \in E$$

where $\mathbf{P}[i,*]$ *denotes the* i*th row of matrix* $\mathbf{P}$.

In the following property, we give the comparison of Discrete-Time Markov chains (DTMCs) in terms of distributions for the sake of readability.

Property 3. Let $\{X(n)\}$ (resp. $\{Y(n)\}$) be a DTMC, $\mathbf{\Pi_X^n}$ (resp. $\mathbf{\Pi_Y^n}$) its transient distribution at time n, and $\mathbf{\Pi_X}$ (resp. $\mathbf{\Pi_Y}$) its steady-state distribution (if it exists). If $\{X(n)\} \preceq_{st} \{Y(n)\}$ then $\mathbf{\Pi_X^n} \preceq_{st} \mathbf{\Pi_Y^n}$, $\forall n$ and $\mathbf{\Pi_X} \preceq_{st} \mathbf{\Pi_Y}$.

***fg*-Comparison of DTMCs.** We now define the fg-comparison between two probability measures p and q which are not defined on the same state space. Let p (resp. q) be defined on the state space E (resp. F); f (resp. g) be a surjective function from E (resp. F) into a state space G; $\{X(n)\}$ (resp. $\{Y(n)\}$) be a time-homogeneous DTMC defined on the discrete space E (resp. F) with transition matrix P (resp. Q). The fg-comparison between p and q is defined as follows:

Definition 3

$$p \preceq_{st}^{fg} q \iff \tilde{p} \preceq_{st} \tilde{q}$$

where $\tilde{p} = fp$ *is the image measure of* p *by* f $(\forall i \in G, \tilde{p}_i = \sum_{j \in E, f(j)=i} p_j)$.

The following example illustrates this type of comparison: $E = \{1,2,3,4\}$, $F = \{1,2,3\}$ and $G = F$. The function $f : E \to F$ is defined as $f(1) = 1$, $f(2) = f(3) = 2$ and $f(4) = 4$ and g is the identity function. Hence, $\tilde{p} = (\tilde{p}_1, \tilde{p}_2, \tilde{p}_3) = (p_1, p_2 + p_3, p_4)$ and $(p_1, p_2, p_3, p_4) \preceq_{st}^{fg} (q_1, q_2, q_3) \Longleftrightarrow (\tilde{p}_1, \tilde{p}_2, \tilde{p}_3) \preceq_{st} (q_1, q_2, q_3) \Longleftrightarrow \sum_{k=i}^{3} \tilde{p}_k \leq \sum_{k=i}^{3} q_k, i = 3, 2, 1$. For instance, $(0.4, 0.2, 0.3, 0.1) \preceq_{st}^{fg} (0.35, 0.45, 0.2)$.

Definition 4. *The DTMC* $\{X(n)\}$ *is said to be less than the DTMC* $\{Y(n)\}$ *with respect to the order* $\preceq_{st}^{fg}$, *if* $X(n) \preceq_{st}^{fg} Y(n)$, $\forall n$.

Definition 5. $\mathbf{P}$ *is* $\preceq_{st}^{f}$*-monotone if and only if,* $\mathbf{P}[x,*] \preceq_{st}^{f} \mathbf{P}[y,*]$, $\forall x, y \in E$, *such that* $f(x) \leq f(y)$

Definition 6. $\mathbf{P} \preceq_{st}^{fg} \mathbf{Q}$ *if and only if,* $\forall x \in E, \forall y \in F$ *such that* $f(x) = g(y)$, $\mathbf{P}[x,*] \preceq_{st}^{fg} \mathbf{Q}[y,*]$

Theorem 2. *$\{X(n)\} \preceq_{st}^{fg} \{Y(n)\}$ if the following conditions are satisfied : $X(0) \preceq_{st}^{fg} Y(0)$, $\mathbf{P} \preceq_{st}^{fg} \mathbf{Q}$ and $\mathbf{P}$ is $\preceq_{st}^{f}$-monotone or $\mathbf{Q}$ is $\preceq_{st}^{g}$-monotone.*

We have the $\preceq_{st}^{fg}$ comparison between transient distributions and the state-state distribution, if the underlying chains are comparable in this sense:

Property 4. Let $\{X(n)\}$ (resp. $\{Y(n)\}$) be a DTMC, $\mathbf{\Pi_X^n}$ (resp. $\mathbf{\Pi_Y^n}$) its transient distribution at time n, and $\mathbf{\Pi_X}$ (resp. $\mathbf{\Pi_Y}$) its steady-state distribution (if it exists). If $\{X(n)\} \preceq_{st}^{fg} \{Y(n)\}$ then $\mathbf{\Pi_X^n} \preceq_{st}^{fg} \mathbf{\Pi_Y^n}$, $\forall n$ and $\mathbf{\Pi_X} \preceq_{st}^{fg} \mathbf{\Pi_Y}$.

2.2 Model Checking DTMC

In this subsection, we briefly present the logic called *Probabilistic real time Computation Tree Logic* (PCTL) [12] which allows to express formulas over discrete time Markov chains.

DTMC and Notations. Throughout this paper, the considered DTMCs may be finite or infinite with a countable state space. A labelled finite (resp. infinite) DTMC $\mathcal{M}$ is a 3-tuple $(S, \mathbf{P}, L)$ where S is a finite (resp. infinite countable) set of *states*, $\mathbf{P} : S \times S \to \mathcal{R}^+$ is the *transition matrix* and $L : S \to 2^{AP}$ is a *labelling* function which assigns to each state $s \in S$, the set $L(s)$ of atomic propositions $a \in AP$ that are valid in s, AP denotes the set of atomic propositions.

For a DTMC, there are two types of state probabilities : transient probabilities where the system is considered at time n and steady-state probabilities when the system reaches an equilibrium if it exists. In the sequel, $\mathbf{\Pi}_\alpha^{\mathcal{M}}(s', n)$ denotes the probability to be in state s' at time n with initial distribution α. $\mathbf{\Pi}_\alpha^{\mathcal{M}}(s') = \lim_{n\to\infty} \mathbf{\Pi}_\alpha^{\mathcal{M}}(s', n)$ is the steady-state probability to be in state s'. If $\mathcal{M}$ is ergodic, $\mathbf{\Pi}_\alpha^{\mathcal{M}}(s')$ exists and it is independent of the initial distribution, so we will denote it by $\mathbf{\Pi}^{\mathcal{M}}(s')$. For Markov chain $\mathcal{M}$ we denote by $\mathbf{\Pi}_\alpha^{\mathcal{M}}(n)$ the transient distribution at time n when the initial distribution is α and by $\mathbf{\Pi}^{\mathcal{M}}$ the steady-state distribution. For $S' \subseteq S$, we denote by $\mathbf{\Pi}_\alpha^{\mathcal{M}}(S', n)$ (resp. $\mathbf{\Pi}^{\mathcal{M}}(S')$) the transient probability to be in states of S', $\mathbf{\Pi}_\alpha^{\mathcal{M}}(S', n) = \sum_{s'\in S'} \mathbf{\Pi}_\alpha^{\mathcal{M}}(s', n)$ (the steady-state probability to be in states of S', $\mathbf{\Pi}^{\mathcal{M}}(S') = \sum_{s'\in S'} \mathbf{\Pi}^{\mathcal{M}}(s')$). In the case of an unique initial state s (i.e. $\alpha(s) = 1$ and $\alpha(s') = 0$ for $s \neq s'$), we simply write $\mathbf{\Pi}_\alpha^{\mathcal{M}}(n)$ by $\mathbf{\Pi}_s^{\mathcal{M}}(n)$.

A path through a DTMC $\mathcal{M}$ can be finite or infinite. For instance a finite path σ of length k is a sequence of states $\sigma = s_0, s_1, \cdots, s_k$ with $s_i \in S$ and $\mathbf{P}(s_i, s_{i+1}) > 0$ $\forall i$. We denote by $paths_s$ the set of all paths starting from state s and by $\sigma[i]$ the i^{th} state s_i of the path σ.

Syntax of PCTL. We give here the syntax of PCTL as defined in [12] and its extension by a steady-state operator that has been proposed in [3]. Let n be an integer, p a probability and $\triangleleft$ a comparison operator $\in \{\leq, \geq\}$. In the sequel, we denote by S_ϕ or ϕ-states the set of states that satisfy ϕ and by $\models$ the satisfaction relation. The syntax of PCTL is:

$$\phi ::= true \mid a \mid \phi \wedge \phi \mid \neg\phi \mid \mathcal{P}_{\lhd p}(\phi\ \mathcal{U}^{\leq n}\phi) \mid \mathcal{S}_{\lhd p}(\phi)$$

In this paper, for the sake of simplicity, we do not consider the next state operator and the other Boolean connectives (false, $\vee$, $\Rightarrow$) that are derived in the usual way. Let us present the semantics of these formulas as defined in [12]:

$s \models true$	for all $s \in S$
$s \models a$	iff $a \in L(s)$
$s \models \neg\phi$	iff $s \not\models \phi$
$s \models \phi_1 \wedge \phi_2$	iff $s \models \phi_1 \wedge s \models \phi_2$
$s \models \mathcal{P}_{\lhd p}(\phi_1\ \mathcal{U}^{\leq n}\phi_2)$	iff $Prob^{\mathcal{M}}(s, \phi_1 \mathcal{U}^{\leq n}\phi_2) \lhd p$
$s \models \mathcal{S}_{\lhd p}(\phi)$	iff $\mathbf{\Pi}_s^{\mathcal{M}}(S_\phi) \lhd p$

$Prob^{\mathcal{M}}(s, \phi_1 \mathcal{U}^{\leq n}\phi_2)$ denotes the probability measure of the paths σ starting in s ($\sigma \in paths_s$) satisfying $\phi_1\ \mathcal{U}^{\leq n}\phi_2$ i.e. $Prob^{\mathcal{M}}(s, \phi_1 \mathcal{U}^{\leq n}\phi_2) = Prob\{\sigma \in paths_s \mid \sigma \models \phi_1\ \mathcal{U}^{\leq n}\phi_2\}$. $\mathcal{P}_{\lhd p}(\phi_1\ \mathcal{U}^I\phi_2)$ asserts that the probability measure of paths satisfying $\phi_1\ \mathcal{U}^{\leq n}\phi_2$ meets the bound given by $\lhd p$. The path formula $\phi_1\ \mathcal{U}^{\leq n}\phi_2$ asserts that ϕ_2 will be satisfied within n time units and that all preceding states satisfy ϕ_1, i.e.

$$\sigma \models \phi_1\ \mathcal{U}^{\leq n}\phi_2 \text{ iff } \exists i \leq n \text{ such that } \sigma[i] \models \phi_2 \text{ and } \forall j < i, \sigma[j] \models \phi_1$$

$\mathcal{S}_{\lhd p}(\phi)$ asserts that the steady-state probability for ϕ-states meets the bound $\lhd p$. Similar to the steady-state operator $\mathcal{S}_{\lhd p}(\phi)$, we define a transient-state operator $\mathcal{T}_{\lhd p}^{@n}(\phi)$ such that: $s \models \mathcal{T}_{\lhd p}^{@n}(\phi)$ iff $\mathbf{\Pi}_s^{\mathcal{M}}(S_\phi, n) \lhd p$.

Checking PCTL Operators. In [12], a methodology has been proposed to check bounded until operator, $\mathcal{P}_{\lhd p}(\phi_1 \mathcal{U}^{\leq n}\phi_2)$. Let us consider the following partition of S into three subsets:

- the success states, are labelled with ϕ_2
- the failures states, are states which are not labelled with ϕ_1 nor ϕ_2
- the inconclusive states, are states labelled with ϕ_1 but not with ϕ_2

Let $\mathcal{M}[\phi]$ be the DTMC defined from $\mathcal{M} = (S, \mathbf{P}, L)$, by making all ϕ-states (states satisfying ϕ) in $\mathcal{M}$ absorbing, i.e. $\mathcal{M}' = (S, \mathbf{P}', L)$ where $\mathbf{P}'(s, s') = \mathbf{P}(s, s'), \forall s' \in S$ if $s \not\models \phi$ and if $s \models \phi$ then $\mathbf{P}'(s, s) = 1$ and $\mathbf{P}'(s, s') = 0$. It has been shown that $Prob^{\mathcal{M}}(s, \phi_1\ \mathcal{U}^{\leq n}\phi_2)$ can be computed by means of transient distributions of DTMC $\mathcal{M}'$ which is obtained from $\mathcal{M}$ by making success states and failures states absorbing. In fact once a success state is reached before n time units, $\phi_1 \mathcal{U}^{\leq n}\phi_2$ is satisfied regardless of which states will be visited in the future. On the other hand, $\phi_1 \mathcal{U}^{\leq n}\phi_2$ is violated once a failure state is visited. Formally, $\mathcal{M}' = \mathcal{M}[\neg\phi_1 \wedge \neg\phi_2][\phi 2] = \mathcal{M}[\neg\phi_1 \vee \phi_2]$ and we have the equation $Prob^{\mathcal{M}}(s, \phi_1\ \mathcal{U}^{\leq n}\phi_2) = \sum_{s' \models \phi_2} \mathbf{\Pi}_{\mathbf{s}}^{\mathcal{M}'}(s', n)$ ([15]).

$\mathbf{\Pi}_{\mathbf{s}}^{\mathcal{M}'}(s', n)$ denotes the probability of reaching state s' in n steps in the DTMC $\mathcal{M}'$ when starting in s. Consequently,

$$s \models \mathcal{P}_{\lhd p}(\phi_1\ \mathcal{U}^{\leq n}\phi_2) \quad \text{iff} \quad \mathbf{\Pi}_{\mathbf{s}}^{\mathcal{M}'}(S_{\phi_2}, n) \lhd p \tag{3}$$

To check the steady-state operator $\mathcal{S}_{\lhd p}(\phi)$ (resp. the transient-state operator $\mathcal{T}^{@n}_{\lhd p}(\phi)$) it suffices to verify that the steady state probability (resp. the transient distribution probability at time n) to be in ϕ states, meets the bound $\lhd p$:

$$s \models \mathcal{S}_{\lhd p}(\phi) \quad \text{iff} \quad \mathbf{\Pi}^{\mathcal{M}}_s(S_\phi) \lhd p \tag{4}$$

$$s \models \mathcal{T}^{@n}_{\lhd p}(\phi) \quad \text{iff} \quad \mathbf{\Pi}^{\mathcal{M}}_s(S_\phi, n) \lhd p \tag{5}$$

3 Bounding Infinite DTMCs by Finite DTMCs

In the sequel, $\{Y(n)\}$ denotes an infinite state space, time-homogeneous DTMC taking values in $\{0, 1, 2, \ldots\}$, with transition matrix $R = (r_{i,j})_{i,j\geq 0}$. We want to define a finite DTMC $\{X(n)\}$ such that $\{X(n)\} \preceq^{fg}_{st} \{Y(n)\}$ for some state functions f and g. In this section, we will give two such lower bounding finite DTMCs. The first bound is valid with a monotonicity condition on the transition matrix R while there is no condition for the second bound. Since a time-homogeneous DTMC is completely defined by its transition matrix and its initial distribution, the proposed bounding chains ($\{X(n)\}$) are given in terms of their transition matrices and initial distributions. Let us remark here that the bounding algorithms given in this section are inspired from bounding algorithms [1,11], so we do not give their proofs. Moreover their complexities in the worst-case without any sparse implementation neither optimisation is quadratic. We first give the definition of partial monotonicity which is required for the first bound.

Definition 7. *A transition matrix P is said partially $\preceq_{st}$-monotone from level K, if* $P[i,*] \preceq_{st} P[j,*] \quad \forall i, j \geq K$ *such that* $i < j$

3.1 First Bound

We first construct a finite state-space transition matrix by truncating the underlying infinite state transition matrix, R at state N and by augmenting the probabilities of column N to make the truncated matrix stochastic. By doing so, we do not remove states greater than N but they are aggregated to state N. Let $Q = (q_{i,j})_{0\leq i,j\leq N}$ be the matrix defined by :

$$q_{i,j} = \begin{cases} r_{i,j}, & 0 \leq i \leq N, 0 \leq j \leq N-1 \\ \sum_{k\geq N} r_{i,k}, & 0 \leq i \leq N, j = N \end{cases} \tag{6}$$

In the sequel we call N *the truncation level* and Q *the stochastic truncated matrix of R at level N*. Remark that the quantity $\sum_{k\geq N} r_{i,k}$ can be easily computed since it is equal to $1 - \sum_{k=0}^{N-1} r_{i,k}$.

The second step consists in constructing $P = (p_{i,j})_{0\leq i,j\leq N}$ through Algorithm 1. The input parameters are the truncated matrix Q, and the probability vector q defined from row $N+1$ of R as $q = (r_{N+1,0}, \ldots, r_{N+1,N-1}, \sum_{k\geq N} r_{N+1,k})$. Therefore P is a stochastic matrix which is $\preceq_{st}$-monotone, lower bound in the

Algorithm 1.

Input : stochastic matrix A; probability vector p.
Output : B such that 1)stochastic matrix, 2)$\preceq_{st}$-monotone, 3)$B \leq_{st} A$, 4)$B[N,*] \preceq_{st} p$.

$b_{N,0} = \max(a_{N,0}, p_0)$
for $i = N-1$ **downto** 0 **do**
  $b_{i,0} = \max(a_{i,0}, b_{i+1,0})$
end
for $j = 1$ to N **do**
  $b_{N,j} = \max(\sum_{k=0}^{j} a_{N,k}, \sum_{k=0}^{j} p_k) - \sum_{k=0}^{j-1} b_{N,k}$
  for $i = N-1$ **downto** 0 **do**
    $b_{i,j} = max(\sum_{k=0}^{j} a_{i,k}, \sum_{k=0}^{j} b_{i+1,k}) - \sum_{k=0}^{j-1} b_{i,k}$
  end
end

sense of $\preceq_{st}$ of Q ($P \preceq_{st} Q$) and the Nth row of P is less than vector q in the sense of $\preceq_{st}$ order ($P[N,*] \preceq_{st} q$). Let $\nu = (\nu_0, \nu_1, \nu_2, \ldots)$ be the initial distribution of the Markov chain $\{Y(n)\}$, i.e., the distribution of $Y(0)$. Probability vector $\mu = (\mu_0, \mu_1, \ldots, \mu_N)$ is defined on $\{0, 1, \ldots, N\}$ such that $\mu_i = \nu_i$, if $i < N$ and $\mu_N = \sum_{k \geq N} \nu_k$. Let define the state spaces $E = \{0, 1, \ldots, N\}$ and $F = \{0, 1, 2, \ldots\}$. f is the identity function on E ($f(i) = i, \forall i \in E$) and function $g : F \to E$ is defined as: $g(i) = i$ if $i < N$ and $g(i) = N$ if $i \geq N$. We now demonstrate that the truncated finite state transition matrix constructed as explained above provides a lower bound on the infinite state transition matrix.

Proposition 1. *Let $\{Y(n)\}$ be an infinite DTMC with state space $\{0, 1, 2, \ldots\}$ and a transition matrix R which is partially $\preceq_{st}$-monotone from level $N+1$. If $\{X(n)\}$ is a finite DTMC with state space $E = \{0, 1, \ldots, N\}$ defined by the initial distribution μ and the transition matrix P as given above, then $\{X(n)\} \preceq_{st}^{fg} \{Y(n)\}$.*

Proof. Let m (resp. n) be a probability measure defined on E (resp. F). Using definition 3, $m \preceq_{st}^{fg} n \iff \tilde{m} \preceq_{st} \tilde{n}$. The image measure $\tilde{m}$ of m by f is equal to m ($\tilde{m}_i = m_i, \forall i \in E$) since f is the identity function. $\tilde{n} = gn$ the image measure of n by g is defined as $\tilde{n}_i = \sum_{k \in F, g(k)=i} n_k, \forall i \in E$. Hence, $\tilde{n}_i = n_i$ if $i < N$ and $\tilde{n}_N = \sum_{k \geq N} n_k$ and

$$m \preceq_{st}^{fg} n \iff (m_0, \ldots, m_{N-1}, m_N) \preceq_{st} (n_0, \ldots, n_{N-1}, \sum_{k \geq N} n_k) \tag{7}$$

Since $\mu = (\nu_0, \ldots, \nu_{N-1}, \sum_{k \geq N} \nu_k)$, it is obvious from the last equation that $\mu \preceq_{st}^{fg} \nu$ then $X(0) \preceq_{st}^{fg} Y(0)$. P is constructed by Algorithm 1, thus it is $\preceq_{st}$-monotone : $P[x,*] \preceq_{st} P[y,*], \forall x, y \in E$ such that $x \leq y$ (see ii- of theorem 1). f is the identity function, so $P[x,*] \preceq_{st}^{f} P[y,*]$, $\forall x, y \in E$, such that $f(x) \leq f(y)$. It follows from definition 5 that P is $\preceq_{st}^{f}$-monotone.

It remains to prove that $P \preceq_{st}^{fg} R$. From definition 6, we must show that $\forall x \in E, \forall y \in F$ such that $x = g(y)$, $P[x,*] \preceq_{st}^{fg} R[y,*]$. This is equivalent to show that $\forall i < N, P[i,*] \preceq_{st}^{fg} R[i,*]$ and $P[N,*] \preceq_{st}^{fg} R[k,*], \forall k \geq N$. By construction of P, we have $P \preceq_{st} Q$, $(\forall i \leq N, P[i,*] \preceq_{st} Q[i,*])$. By definition of matrix Q (equation 6), $\forall i \leq N, Q[i,*] = (r_{i,0}, \ldots, r_{i,N-1}, \sum_{k \geq N} r_{i,k})$. Hence, $\forall i \leq N, P[i,*] \preceq_{st} (r_{i,0}, \ldots, r_{i,N-1}, \sum_{k \geq N} r_{i,k})$ and we conclude from equation 7 that

$$\forall i \leq N, P[i,*] \preceq_{st}^{fg} R[i,*]$$

On the other hand, R is supposed to be partially $\preceq_{st}$-monotone from level $N+1$. Thus, $R[N+1,*] \preceq_{st} R[k,*], \forall k > N$. By property 1, we deduce that $\sum_{j \geq N} r_{N+1,j} \leq \sum_{j \geq N} r_{k,j}, \forall k > N$ and from property 2 we have $q = (r_{N+1,0}, \ldots, r_{N+1,N-1}, \sum_{j \geq N} r_{N+1,j}) \preceq_{st} (r_{k,0}, \ldots, r_{k,N-1}, \sum_{j \geq N} r_{k,j}), \forall k > N$. By construction of matrix P with Algorithm 1, $P[N,*] \preceq_{st} q$, ie. $P[N,*] \preceq_{st} (r_{k,0}, \ldots, r_{k,N-1}, \sum_{j \geq N} r_{k,j}), \forall k > N$. Thus $P[N,*] \preceq_{st}^{fg} R[k,*], \forall k > N$, and it follows from theorem 2 that $\{X(n)\} \preceq_{st}^{fg} \{Y(n)\}$.

3.2 Second Bound

The second bound is constructed by truncating the infinite state space at state N and by augmenting the probabilities of column 0 to make the truncated matrix stochastic. Let $S = (s_{i,j})_{0 \leq i,j \leq N}$ be the truncated matrix defined as :

$$s_{i,j} = \begin{cases} r_{i,j}, & 0 \leq i \leq N, 1 \leq j \leq N \\ r_{i,0} + \sum_{k>N} r_{i,k}, & 0 \leq i \leq N, j = 0 \end{cases} \tag{8}$$

Then we construct a monotone lower bounding matrix $\underline{S}$ for S through Algorithm 2. Thus $\underline{S}$ is $\preceq_{st}$-monotone and $\underline{S} \preceq_{st} S$. Matrix $T = (t_{i,j})_{0 \leq i,j \leq N}$ is obtained from $\underline{S}$ by replacing its first row by the probability distribution $(1, 0, \ldots, 0)$. Obviously, T is a stochastic matrix which is $\preceq_{st}$-monotone, and $T \preceq_{st} S$.

Algorithm 2.

Input : stochastic matrix A.
Output : B such that 1) stochastic matrix, 2) $\preceq_{st}$-monotone, 3)$B \preceq_{st} A$.

1 $b_{N,0} = a_{N,0}$
for $i = N-1$ **downto** 0 **do**
2 $b_{i,0} = \max(a_{i,0}, b_{i+1,0})$
end
for $j = 1$ to N **do**
3 $b_{N,j} = \sum_{k=0}^{j} a_{N,k} - \sum_{k=0}^{j-1} b_{N,k}$
for $i = N-1$ **downto** 0 **do**
4 $b_{i,j} = \max(\sum_{k=0}^{j} a_{i,k}, \sum_{k=0}^{j} b_{i+1,k}) - \sum_{k=0}^{j-1} b_{i,k}$
end
end

Let $\nu = (\nu_0, \nu_1, \nu_2, \ldots)$ be the initial distribution of the Markov chain $\{Y(n)\}$ ($Y(0)$). The probability vector $u = (u_0, u_1, \ldots, u_N)$ is defined on $\{0, 1, \ldots, N\}$ as $u_0 = \nu_0 + \sum_{k>N} \nu_k$ and $u_i = \nu_i$ if $1 \leq i \leq N$. Let define state spaces $E = \{0, 1, \ldots, N\}$ and $F = \{0, 1, 2, \ldots\}$. f is the identity function on E ($f(i) = i, \forall i \in E$) and $h : F \to E$ is defined as: $h(i) = i$ if $i \leq N$ and $h(i) = 0$ if $i > N$.

Proposition 2. *Let $\{Y(n)\}$ be an infinite DTMC with a transition matrix $R = (r_{i,j})_{i,j\geq 0}$. If $\{X(n)\}$ is a finite DTMC with state space $E = \{0, 1, \ldots, N\}$ defined by initial distribution u and transition matrix T as given above, then $\{X(n)\} \preceq_{st}^{fh} \{Y(n)\}$.*

Proof. Let m (resp. n) be a probability measure defined on E (resp. F). $m \preceq_{st}^{fh} n \iff \tilde{m} \preceq_{st} \tilde{n}$ (definition 3). $\tilde{m} = m$ and $\tilde{n} = hn$ the image measure of n by h is defined as $\tilde{n}_i = \sum_{k\in F, h(k)=i} n_k, \forall i \in E$. Hence, $\tilde{n}_i = n_i$ if $i \in \{1, \ldots, N\}$, $\tilde{n}_0 = n_0 + \sum_{k>N} n_k$ and

$$m \preceq_{st}^{fh} n \iff (m_0, m_1, \ldots, m_N) \preceq_{st} (n_0 + \sum_{k>N} n_k, n_1 \ldots, n_N) \tag{9}$$

From this equation it is clear that $u \preceq_{st}^{fh} \nu$, i.e $X(0) \preceq_{st}^{fh} Y(0)$. Similar to the proof of proposition 1, T is $\preceq_{st}^{f}$-monotone. To prove $T \preceq_{st}^{fh} R$, we must show that $\forall x \in E, \forall y \in F$ such that $x = h(y)$, $T[x, *] \preceq_{st}^{fh} R[y, *]$. This is equivalent to show that $T[0, *] \preceq_{st}^{fh} R[0, *]$, $T[0, *] \preceq_{st}^{fh} R[i, *], \forall i > N$ and that $\forall i \in \{1, \ldots, N\}, T[i, *] \preceq_{st}^{fh} R[i, *]$.

We have $T[0, *] = (1, 0, \ldots, 0) \preceq_{st} (r_{i,0} + \sum_{k>N} r_{i,k}, r_{i,1}, \ldots, r_{i,N}), \forall i > N$ (property 2). It follows from equation 9 that $T[0, *] \preceq_{st}^{fh} R[i, *], \forall i > N$. By construction of T, we have $T \preceq_{st} S$, i.e, $\forall i \leq N, T[i, *] \preceq_{st} S[i, *]$. By definition of matrix S (equation 8), $S[i, *] = (r_{i,0} + \sum_{k>N} r_{i,k}, r_{i,1}, \ldots, r_{i,N}), \forall i \leq N$. Hence, $T[i, *] \preceq_{st} (r_{i,0} + \sum_{k>N} r_{i,k}, r_{i,1}, \ldots, r_{i,N}), \forall i \leq N$. By equation 9, $T[i, *] \preceq_{st}^{fh} R[i, *], \forall i \leq N$. Finally, it follows from theorem 2 that $\{X(n)\} \preceq_{st}^{fh} \{Y(n)\}$.

4 Checking Infinite DTMCs by Stochastic Comparison

In this section we propose to check PCTL operators over infinite discrete-time Markov chains using the stochastic lower bounds given in the previous section. Throughout this section, $\{Y(n)\}$ is the underlying infinite DTMC for which we want to check PCTL operators. Before introducing the checking procedures, we first give the following proposition for the monotonicity properties of the transition matrix when some states are permuted.

Property 5. Let N be a given integer. If the transition matrix R is $\preceq_{st}$-monotone, then the infinite transition matrix R_N obtained by permuting some states larger than N is partially $\preceq_{st}$-monotone from level $N + 1$.

In fact by permuting some states of the matrix R we may loose the monotonicity property and R_N may not be monotone. However, since the permutations concern only states which are not larger than N, it is obvious that R_N is partially $\preceq_{st}$-monotone from level $N + 1$.

It can be seen from equations 3,4 and 5 that to check the formula $Fr = \{\mathcal{P}_{\lhd p}(\phi_1\ \mathcal{U}^{\leq n}\phi_2),\ \mathcal{S}_{\lhd p}(\phi),\ \mathcal{T}^{@n}_{\lhd p}(\phi)\}$ we have to sum the probabilities of a set of states. We denote by S_Σ this set of states and by $P_{Fr}(S_\Sigma)$ the probability of S_Σ states for the considered formula Fr. For instance, for $Fr = \mathcal{P}_{\lhd p}(\phi_1 \mathcal{U}^{\leq n}\phi_2)$, S_Σ is the set of ϕ_2 states ($S_\Sigma = S_{\phi_2}$) and $P_{Fr}(S_\Sigma) = \mathbf{\Pi}^{\mathcal{M}'}_{\mathbf{s}}(S_{\phi_2}, n)$ (see equation 3).

In general $P_{Fr}(S_\Sigma)$ is intractable because of the infinite state space except the cases where the the underlying DTMC has some special structures. We propose to compute bounds on $P_{Fr}(S_\Sigma)$ by considering finite state DTMCs. Let $P^{low}_{Fr}(S_\Sigma) \leq P_{Fr}(S_\Sigma) \leq P^{up}_{Fr}(S_\Sigma)$. Depending on the comparison operator $\lhd$, we can deduce if the underlying formula is checked or not through these bounds :

- $\lhd = \leq$: If $P^{up}_{Fr}(S_\Sigma) \leq p$, we can deduce that the underlying formula is checked.
- $\lhd = \geq$: If $P^{low}_{Fr}(S_\Sigma) \geq p$, we can deduce that the underlying formula is checked.
- In the other cases, it is not possible to decide the satisfaction or not and we must refine the bounds by increasing the truncation level.

We apply the stochastic bounding approach to derive the bounding values $P^{up}_{Fr}(S_\Sigma)$, $P^{low}_{Fr}(S_\Sigma)$. Indeed the $\preceq_{st}$ comparison of probability vectors allows to establish the inequalities between the partial sum of probabilities (see equations 1, 2). Thus the set S_Σ must be reordered at the beginning or at the end of the state space depending whether we want to obtain an upper or a lower bound on the partial sum.

The finite state space lower bounding DTMC are constructed by truncating the underlying infinite state space DTMC as explained in section 3. However it is not possible to take into account all of the possible cases depending on the comparison operator and the finiteness or not of the set S_ϕ. In the following subsections, we discuss how we can check by this method the operators $\mathcal{P}_{\lhd p}(\phi_1\ \mathcal{U}^{\leq n}\phi_2)$, $\mathcal{S}_{\lhd p}(\phi)$, $\mathcal{T}^{@n}_{\lhd p}(\phi)$. Let us remark here that the same notations as in section 3 are used in the sequel.

4.1 Checking $\mathcal{P}_{\lhd p}(\phi_1\ \mathcal{U}^{\leq n}\phi_2)$

To check $\mathcal{P}_{\lhd p}(\phi_1\ \mathcal{U}^{\leq n}\phi_2)$, we first make the success states and the failures states absorbing to obtain the DTMC $\mathcal{M}'$ (see subsection 2.2). In addition, as it has been proposed in [12] we aggregate the success states and the failures states into two representative macro-states $s_{success}$ and $s_{failures}$ which are absorbing. Let $\mathcal{M}''$ be the DTMC obtained after these transformations. From equation 3 we deduce that

$$s \models \mathcal{P}_{\lhd p}(\phi_1\ \mathcal{U}^{\leq n}\phi_2) \quad \text{iff} \quad \mathbf{\Pi}^{\mathcal{M}''}_{\mathbf{s}}(s_{success}, n) \lhd p \tag{10}$$

We distinguish in the following the cases when the set of inconclusive states is finite and infinite. The case when the set of inconclusive states is finite is interesting. Indeed we can compute $\mathbf{\Pi}^{\mathcal{M}''}_{\mathbf{s}}(s_{success}, n)$ exactly since the DTMC $\mathcal{M}''$ is finite. If the set of inconclusive states is infinite, we first choose an integer N sufficiently large to take into account the macro-states $s_{success}$ and $s_{failures}$

and a maximum of inconclusive states and also the initial state. In the following $\{Y(n), n \geq 0\}$ is the infinite DTMC corresponding to $\mathcal{M}''$ with transition matrix R. We distinguish the cases of the comparison operators $\lhd = \geq$ and $\lhd = \leq$.

- $\lhd = \geq$: We permute the macro state $s_{success}$ with state N. Let R_N be the matrix obtained after permutation. Let S be the truncated matrix of R_N at level N (equation 8). By proposition 2, $\{X(n)\} \preceq_{st}^{fh} \{Y(n)\}$ and $\pi_u^X(n) \preceq_{st}^{fh} \pi_\nu^Y(n)$. By definition of functions f and h and equation 9, this is equivalent to $(\pi_0^X(n), \pi_1^X(n), \ldots, \pi_N^X(n)) \preceq_{st} (\pi_0^Y(n) + \sum_{k>N} \pi_k^Y(n), \pi_1^Y(n), \ldots, \pi_N^Y(n))$. The success states are aggregated in state N, so we deduce from equation 1 that $\pi_N^X(n) \leq \pi_N^Y(n) = \mathbf{\Pi}_{\mathbf{s}}^{\mathcal{M}''}(s_{success}, n)$. Thus if $\pi_N^X(n) \geq p$ then $\mathbf{\Pi}_{\mathbf{s}}^{\mathcal{M}''}(s_{success}, n) \geq p$ and $\mathcal{P}_{\geq p}(\phi_1 \, \mathcal{U}^{\leq n} \phi_2)$ is satisfied, otherwise we cannot conclude.
- $\lhd = \leq$: We permute the macro state $s_{success}$ with state 1. Let R_N be the matrix obtained after permutation. Using the second bound we have $\{X(n)\} \preceq_{st}^{fh} \{Y(n)\}$ and $\pi_u^X(n) \preceq_{st}^{fh} \pi_\nu^Y(n)$. By definition of functions f and h and equation 9, this is equivalent to $(\pi_0^X(n), \pi_1^X(n), \ldots, \pi_N^X(n)) \preceq_{st} (\pi_0^Y(n) + \sum_{k>N} \pi_k^Y(n), \pi_1^Y(n), \ldots, \pi_N^Y(n))$. We deduce from equation 2 that $\pi_0^X(n) + \pi_1^X(n) \geq \pi_0^Y(n) + \sum_{k>N} \pi_k^Y(n) + \pi_1^Y(n) \geq \pi_1^Y(n)$. Recall that with the considered permutation the success states are aggregated in state 1, so $\pi_0^X(n) + \pi_1^X(n) \geq \mathbf{\Pi}_{\mathbf{s}}^{\mathcal{M}''}(s_{success}, n)$. Thus if $\sum_{k=0}^{1} \pi_k^X(n) \leq p$ then $\mathbf{\Pi}_{\mathbf{s}}^{\mathcal{M}''}(s_{success}, n) \leq p$ and $\mathcal{P}_{\leq p}(\phi_1 \, \mathcal{U}^{\leq n} \phi_2)$ is satisfied, otherwise we cannot conclude.

4.2 Checking $\mathcal{T}_{\lhd p}^{@n}(\phi)$

We distinguish the cases of the comparison operators $\lhd = \geq$ and $\lhd = \leq$ and the cases when the set of ϕ-states, S_ϕ is finite and infinite:

- $\lhd = \geq$ and S_ϕ is finite : In this case we first choose a truncation level, N sufficiently large to take into account all ϕ-states. Let m be the cardinal of S_ϕ. We first permute the ϕ-states with states from $N-m+1$ to N. Let R_N be the matrix obtained after permutation and S be the stochastic truncated matrix of R_N at level N (equation 8). We consider the same notations as in subsection 3.2 for T, u, f, h and the finite DTMC $\{X(n), n \geq 0\}$. By proposition 2, $\{X(n)\} \preceq_{st}^{fh} \{Y(n)\}$ and $\pi_u^X(n) \preceq_{st}^{fh} \pi_\nu^Y(n)$. By definition of functions f and h and equation 9, this is equivalent to $(\pi_0^X(n), \pi_1^X(n), \ldots, \pi_N^X(n)) \preceq_{st} (\pi_0^Y(n) + \sum_{k>N} \pi_k^Y(n), \pi_1^Y(n), \ldots, \pi_N^Y(n))$. The ϕ-states are states from $N - m + 1$ to N, so we deduce from equation 1 that $\mathbf{\Pi}_u^X(S_\phi, n) = \sum_{k=N-m+1}^{N} \pi_k^X(n) \leq \sum_{k=N-m+1}^{N} \pi_k^Y(n) = \mathbf{\Pi}_\nu^Y(S_\phi, n)$. Thus if $\mathbf{\Pi}_u^X(S_\phi, n) \geq p$ then $\mathbf{\Pi}_\nu^Y(S_\phi, n) \geq p$ and $T_{\geq p}^{@n}(\phi)$ is satisfied, otherwise we cannot conclude.
- $\lhd = \geq$ and S_ϕ is infinite : In this case we are obliged to truncate also the ϕ-states. We choose the truncation level, N sufficiently large in order to take more ϕ-states. Let S'_ϕ be the subset of ϕ-states of cardinal m which are less than N ($S'_\phi = \{s, \, 0 \leq s \leq N, s \models \phi\}$). In the same way as the previous case, we can show that $\mathbf{\Pi}_u^X(S'_\phi, n) = \sum_{k=N-m+1}^{N} \pi_k^X(n) \leq \sum_{k=N-m+1}^{N} \pi_k^Y(n) =$

$\mathbf{\Pi}^Y_\nu(S'_\phi, n) \leq \mathbf{\Pi}^Y_\nu(S_\phi, n)$. Thus if $\mathbf{\Pi}^X_u(S'_\phi, n) \geq p$ then $\mathbf{\Pi}^Y_\nu(S_\phi, n) \geq p$ and $\mathcal{T}^{@n}_{\geq p}(\phi)$ is satisfied, otherwise we cannot conclude.

- $\lhd = \leq$ and S_ϕ is finite : In this case we first choose an integer N sufficiently large so that all states satisfying ϕ are less than N. Let m be the cardinal of S_ϕ. We permute the ϕ-states with states from 1 to m. Let R_N be the matrix obtained after permutation. Let S be the stochastic truncated matrix of R_N at level N (equation 8). By proposition 2, $\{X(n)\} \preceq^{fh}_{st} \{Y(n)\}$ and $\pi^X_u(n) \preceq^{fh}_{st} \pi^Y_\nu(n)$. By definition of functions f and h and equation 9, this is equivalent to $(\pi^X_0(n), \pi^X_1(n), \ldots, \pi^X_N(n)) \preceq_{st} (\pi^Y_0(n) + \sum_{k>N} \pi^Y_k(n), \pi^Y_1(n), \ldots, \pi^Y_N(n))$. We deduce from equation 2 that $\sum_{k=0}^m \pi^X_k(n) \geq \pi^Y_0(n) + \sum_{k>N} \pi^Y_k(n) + \sum_{k=1}^m \pi^Y_k(n) \geq \sum_{k=1}^m \pi^Y_k(n)$. Recall that with the considered permutation the ϕ-states are states from 1 to m, so $\sum_{k=0}^m \pi^X_k(n) \geq \mathbf{\Pi}^Y_\nu(S_\phi, n)$. Thus if $\sum_{k=0}^m \pi^X_k(n) \leq p$ then $\mathbf{\Pi}^Y_\nu(S_\phi, n) \leq p$ and $\mathcal{T}^{@n}_{\leq p}(\phi)$ is satisfied, otherwise we cannot conclude. In the case when transition matrix R is in addition $\preceq_{st}$-monotone, we can construct an upper bound for $\mathbf{\Pi}^Y_\nu(S_\phi, n)$ using the first bound exactly in the same way as for the steady state operator with the comparison operator $\lhd = \leq$ (see section 4.3). Thus we can use two bounds for checking $\mathcal{T}^{@n}_{\leq p}(\phi)$ in this case.
- $\lhd = \leq$ and S_ϕ is infinite: If $S_{\neg\phi}$ is finite, we can check $\mathcal{T}^{@n}_{\leq p}(\phi)$. Let m be the cardinal of $S_{\neg\phi}$. We permute the m states not-satisfying ϕ with states from 1 to m. We take the level of truncation N equal to m. Let R_N be the matrix obtained after permutation. Let S be the stochastic truncated matrix of R_N at level N (equation 8). By proposition 2, $\{X(n)\} \preceq^{fh}_{st} \{Y(n)\}$ and $\pi^X_u(n) \preceq^{fh}_{st} \pi^Y_\nu(n)$. By definition of functions f and h and equation 9, this is equivalent to $(\pi^X_0(n), \pi^X_1(n), \ldots, \pi^X_N(n)) \preceq_{st} (\pi^Y_0(n) + \sum_{k>N} \pi^Y_k(n), \pi^Y_1(n), \ldots, \pi^Y_N(n))$. We deduce that $\pi^X_0(n) \geq \pi^Y_0(n) + \sum_{k>N} \pi^Y_k(n)$. Remark that with the considered permutation the set S_ϕ is constituted by state 0 and all states greater than N. Thus $\pi^X_0(n) \geq \mathbf{\Pi}^Y_\nu(S_\phi, n)$ and if $\pi^X_0(n) \leq p$ then $\mathbf{\Pi}^Y_\nu(S_\phi, n) \leq p$ and $\mathcal{T}^{@n}_{\leq p}(\phi)$ is satisfied, otherwise we cannot conclude.

4.3 Checking $\mathcal{S}_{\lhd p}(\phi)$

By this methodology, this operator can be checked only when R is stochastically monotone, the set of ϕ-states is finite and the comparison operator $\lhd = \leq$. Let us remark that the second bound is not interesting for the steady-state case, since we make the first state absorbing.

We first choose an integer N sufficiently large so that all states satisfying ϕ are less than N. Let m be the cardinal of S_ϕ. We put the m ϕ-states at the beginning of the state space, i.e, we permute the ϕ-states with states from 0 to $m-1$. Let R_N be the matrix obtained after permutation. This matrix is partially $\preceq_{st}$-monotone from level $N+1$ (proposition 5). Let Q be the stochastic truncated matrix of R_N at level N (equation 6). By proposition 1, $\{X(n)\} \preceq^{fg}_{st} \{Y(n)\}$ and $\pi^X \preceq^{fg}_{st} \pi^Y$ By definition of functions f and g and equation 7, this is equivalent to $(\pi^X_0, \ldots, \pi^X_{N-1}, \pi^X_N) \preceq_{st} (\pi^Y_0, \ldots, \pi^Y_{N-1}, \sum_{k\geq N} \pi^Y_k)$. When the first m ϕ-states are in the beginning, we deduce from equation 2 that $\sum_{k=0}^{m-1} \pi^X_k \geq$

$\sum_{k=0}^{m-1} \pi_k^Y = \mathbf{\Pi}^Y(S_\phi)$. Thus if $\sum_{k=0}^{m-1} \pi_k^X \leq p$ then $\mathbf{\Pi}^Y(S_\phi) \leq p$ and $\mathcal{S}_{\leq p}(\phi)$ is satisfied, otherwise we cannot conclude.

5 Conclusions

In this paper we propose an approach based on the stochastic comparison to check PCTL operators over infinite DTMCs. We present two algorithms to construct finite bounding matrices from the original Markov chain and show that these bounding matrices provide useful inequalities for checking PCTL formulas. The first proposed bound requires a monotonicity condition on the underlying matrix but it is used only for the steady state operator.

The stochastic comparison approach has been applied in general to simplify the analyse of complex systems. In this work, this approach is may be the unique alternative to deal with general infinite DTMCs which are intractable. However, the proposed method can be also used to simplify model checking of large finite DTMCs by considering smaller ones. The stochastic comparison approach has interesting potentials to perform model checking for infinite state models. We envisage to consider some case studies to illustrate the feasibility of the proposed approach and to study the tightness of the bounds. Also, we will investigate other bounding schemes based on this approach.

References

1. Abu-Amsha, O., Vincent, J.-M.: An algorithm to bound functionals of Markov chains with large state space. In: 4th INFORMS Conference on Telecommunications, Boca Raton, Florida (1998)
2. D'Argenio, P.R., Jeannet, B., Jensen, H.E., Larsen, K.G.: Reduction and Refinement Strategies for Probabilistic Analysis. In: Process Algebra and Probabilistic Methods Performance Modeling and Verification, Springer, Heidelberg (2001)
3. Andova, S., Hermanns, H., Katoen, J.P.: Discrete-time rewards model Checked. In: Formal Modelling and Analysis of timed Systems (FORMATS 2003), France (2003)
4. Aziz, A., Sanwal, K., Singhal, V., Brayton, R.: Model Checking Continuous Time Markov Chains. ACM Trans. on Comp. Logic 1(1), 162–170 (2000)
5. Baier, C., Haverkort, B., Hermanns, H., Katoen, J.P.: Model-Checking Algorithms for Continuous-Time Markov Chains. IEEE Trans. Software Eng. 29(6), 524–541 (2003)
6. Baier, C., Haverkort, B., Hermanns, H., Katoen, J.P.: Automated performance and dependability evaluation using Model Checking. In: Calzarossa, M.C., Tucci, S. (eds.) Performance 2002. LNCS, vol. 2459, pp. 261–289. Springer, Heidelberg (2002)
7. Ben Mamoun, M., Pekergin, N., Younès, S.: Model Checking Continuous-Time Markov Chains by Closed-Form Bounding Distributions. In: QEST, International Conference on the Quantitative Evaluation of Systems, Riverside, pp. 189–198 (2006)
8. Bianco, A., de Alfaro, L.: Model checking of probabilistic and nondeterministic systems. In: Thiagarajan, P.S. (ed.) FSTTCS 1995. LNCS, vol. 1026, Springer, Heidelberg (1995)

9. Doisy, M.: A coupling technique for stochastic comparison of functions of Markov processes. Journal of applied Mathematics Decision Sciences 4(1), 39–64 (2000)
10. Doisy, M.: Comparaison de processus Markoviens, PhD thesis, Université de Pau et des pays de de l'Adour (1992)
11. Fourneau, J.M., Pekergin, N.: An algorithmic approach to stochastic bounds. In: Calzarossa, M.C., Tucci, S. (eds.) Performance 2002. LNCS, vol. 2459, pp. 64–88. Springer, Heidelberg (2002)
12. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Form. Asp. of Comp. 6, 512–535 (1994)
13. Hermanns, H., Katoen, J.P., Meyer-Kayser, J., Siegle, M.: A tool for model-checking Markov chains. International Journal on Software Tools for Technology Transfer 4(2), 153–172 (2003)
14. Heyman, P.D.: Approximating the stationary distribution of an infinite stochastic matrix. Jour. of Applied Probability 28(1), 96–108 (1991)
15. Katoen, J.-P., Kwiatkowska, M., Norman, G., Parker, D.: Faster and Symbolic CTMC Model Checking. In: de Luca, L., Gilmore, S. (eds.) PROBMIV 2001, PAPM-PROBMIV 2001, and PAPM 2001. LNCS, vol. 2165, pp. 23–38. Springer, Heidelberg (2001)
16. Kwiatkowska, M., Norman, G., Parker, D.: Prism: Probabilistic symbolic model checker. In: Proceedings of PAPM/PROBMIV 2001 Tools Session (2001)
17. Muller, A., Stoyan, D.: Comparison Methods for Stochastic Models and Risks. Wiley, New York (2002)
18. Pekergin, N., Younès, S.: Stochastic Model Checking with Stochastic Comparison. In: Bravetti, M., Kloul, L., Zavattaro, G. (eds.) EPEW/WS-EM 2005. LNCS, vol. 3670, pp. 109–123. Springer, Heidelberg (2005)
19. Remke, A., Haverkort, B.R., Cloth, L.: Model Checking Infinite-State Markov chains. In: Halbwachs, N., Zuck, L.D. (eds.) TACAS 2005. LNCS, vol. 3440, Springer, Heidelberg (2005)
20. Remke, A., Haverkort, B.R.: CSL model checking algorithms for infinite-state structured Markov chains. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) FORMATS 2007. LNCS, vol. 4763, Springer, Heidelberg (2007)
21. Seneta, E.: Non-negative Matrices and Markov chains, Springer series in statistics, 3rd edn., USA (2006)
22. Taleb, H.: Bornes stochastiques pour l'évaluation des réseaux informatiques, PhD thesis, Université Paris 6 (1996)

Bottleneck Analysis for Two-Hop IEEE 802.11e Ad Hoc Networks

Anne Remke, Boudewijn R. Haverkort, Geert Heijenk, and Lucia Cloth

Design and Analysis of Communication Systems
University of Twente
{anne,brh,heijenk,lucia}@cs.utwente.nl

Abstract. Recently, a quality-of-service (QoS) extension of the IEEE 802.11 standard (known as IEEE 802.11e) for wireless LANs has been proposed. We present a versatile and accurate performance model to study how these new QoS enhancements can be used to improve the performance of wireless nodes competing for bandwidth in a multi-hop ad hoc network. The paper presents the QoS enhancements, and shows how they can be modeled using a simple, yet effective, parameterized quasi-birth-death model. The model is developed hierarchically, in that results at packet level (e.g., as developed by Bianchi and others) are used in our flow-level model, in which a single bottleneck station interacts with a time-varying number of traffic sources. Thus, we are able to study the impact of the QoS enhancements on the flow-level performance. This has not been done before. The results of our analyses are compared with extensive simulations (using OPNET), and show excellent agreement for throughput, mean number of active sources and mean buffer occupancy at the bottleneck station. An important asset of our model is that it allows for very quick evaluations: where simulations require up to an hour per scenario, our model is solved in seconds.

1 Introduction

The availability of cheap yet powerful wireless access technology, most notably IEEE 802.11 ("wireless LAN"), has given an impulse to the development of wireless ad hoc networks. In such networks, the stations (nodes) that are in reach of each other, help each other in obtaining and maintaining connectivity. At the same time they are also competitors, as they all contend for the same resource, i.e., the shared ether as transmission medium. The medium access control of IEEE 802.11 (based on CSMA/CA) is commonly referred to as the distributed coordination function (DCF) [6,14]. Research has shown that, effectively, the DCF tends to equally share the capacity among contending stations [1,8]. Although this appears to be a nice fairness property, this fairness does lead to undesirable situations in case one of the nodes happens to function as a bridge toward either another group of nodes, or to the fixed internet, as illustrated in Figure 1.

Recently, a quality-of-service (QoS)-extension of the IEEE 802.11 standard, the so-called EDCA ("e") version has been released [5]. Roughly speaking, this

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 279–294, 2008.

extension provides mechanisms to provide preferential treatment of certain traffic classes (or nodes) over others. In this paper, we study these extensions in detail, in a way not done before, as follows.

In earlier work [12], we provide a modeling framework for evaluating capacity sharing strategies, using infinite-state Markov reward models and model checking techniques; this paper was on the numerical algorithms and model checking techniques themselves, and illustrated the approach using a number of generic capacity sharing strategies, not at all related to the IEEE 802.11e standard.

In the current paper, we specialize this framework towards the IEEE 802.11e protocol, including the differentiation parameters. We embed Bianchi's model and Engelstad's extensions [1,3] into our model, to accurately describe the effective capacity, depending on the differentiation parameters in use. Both, the specialization of the modeling framework and the embedding of Bianchi's model has never been done before. Furthermore, we conduct detailed simulations that show strong evidence of the correctness of the full IEEE 802.11e model and the employed techniques.

Important to stress is that our model is a flow-level model, with makes it fundamentally different from packet-level models such as [1,3], which have been proposed to compute the share of bandwidth (radio capacity) allocated to a fixed number of sources and a bottleneck node (for various, but not all QoS enhancements). In this paper, we use the packet-level results from [1,3] (as well as extend them) in a (higher-level) flow-level model in which the number of active sources varies in time, depending on how quickly they are being served.

Earlier work on the performance of IEEE 802.11 ad hoc networks considers a variety of scenarios, see, for instance, [15] and the references therein. Also, [8] provides an extensive simulation study of the EDCA standard, investigating the impact of the various QoS mechanism on the performance for single-hop networks. However, these studies do not explicitly address the delays or throughputs in a *multi-hop* ad hoc network. The only paper we are aware of that explicitly addresses the multi-hop case (that is, the two-hop case we also address here) is [15], by using results for a generalized processor sharing model, as studied by Cohen [2]. However, that approach is limited in that it only allows for an equal sharing of transmission capacity between all active stations (including the bottleneck), hence, it does not address the new QoS-enhancements of the IEEE 802.11e standard.

In summary, the contributions of this paper are the following. The paper presents (i) an extension of the models of Bianchi and Engelstad [1,3] to deal with the $\text{TXOP}_{\text{limit}}$ QoS-enhancement; (ii) the embedding of these extended models into our modeling framework (iii) the numerical evaluation of this combined embedded model, and (iv) a comprehensive comparison with detailed (packet-level) simulations using OPNET [10], showing very favorable results. Moreover the analytical techniques are much faster than simulation.

This paper is further organized as follows. We start with a description of the IEEE 802.11 access mechanism and discuss the four QoS extensions in Section 2. A description of the generic model is given in Section 3. We then describe

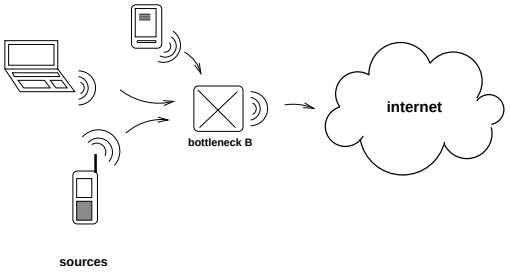


Fig. 1. Bottleneck in a two-hop ad hoc network

precisely how the four IEEE 802.11e QoS-extensions are cast into this model in Section 4. In Section 5 the detailed OPNET simulation setup is described, before we come to a careful comparison of our model's results with the simulation results in Section 6. Section 7 concludes the paper.

2 IEEE 802.11 Ad Hoc Networks

We address a wireless ad hoc network in which the individual nodes communicate with each other through the IEEE 802.11 protocol. In such a network the DCF organizes medium access through a carrier sense multiple access scheme with collision avoidance (CSMA/CA). All stations in such a network contend for the same radio capacity C (measured in packets per second). Whenever a station wants to send a packet, it first senses the medium until the medium is empty for at least a DIFS period (DIFS: DCF inter frame spacing). If the medium is initially found empty, a station that wants medium access immediately starts transmitting after sensing the medium idle for a DIFS period. If the medium is found busy, all stations that want medium access participate in the contention mechanism. After the medium has been idle for at least a DIFS period, each stations draws a random backoff time b. Each station then waits for its chosen backoff time and keeps sensing the medium. If the medium is still idle after DIFS+b time slots, the station may access the medium. As a result, the station with the smallest backoff acquires medium access, if the minimum backoff is unique.

We assume, that whenever two stations draw the same minimum backoff, a collision occurs. Note that in reality this depends on the relative signal strengths at the intended receiver. When the medium is sensed busy during the backoff period, the backoff is suspended and the station continues counting down the backoff after waiting the DIFS period from the moment the medium is sensed idle again. The random backoff is drawn uniformly from the so-called *contention window* (CW), initially set to $[0, \mathrm{CW}_{\min} - 1]$. For up to $r_{\max} - 1$ collisions, the size of the contention window doubles with every unsuccessful transmission and is reset to $\mathrm{CW}_{\min}$ after a successful transmission. Once the contention window has reached size $\mathrm{CW}_{\max} = 2^{r_{\max}} \cdot \mathrm{CW}_{\min}$ it stays unchanged until a successful transmission occurs. Note that in the standard IEEE 802.11 protocol, the values $\mathrm{CW}_{\min}$ and $\mathrm{CW}_{\max}$ are fixed.

The scenario under study, as illustrated in Figure 1, has a varying number N of active nodes, the so-called sources, which are all within reach of each other. Additionally, there is one special node B, referred to as bridge or bottleneck, reachable by all sources. B is the only node that can reach the fixed internet. Thus, all traffic originating from the sources and the traffic passing through the bridge has to share the same wireless transmission capacity. It has been shown that the DCF access mechanism effectively shares the radio capacity equally over all active nodes (a result of the fairness of the access mechanism itself [1,8]). Clearly, this situation benefits the sources as a group, as they can use a relatively large share of radio capacity to send their packets, whereas the bridge becomes a bottleneck: B gets a share as any other individual node, however, it has to support the traffic of *all* other nodes. This leads to a very high buffer occupancy in B, eventually also buffer overflow, and in any case, long delays.

The Enhanced Distributed Channel Access Function (EDCAF) of IEEE 802.11e allows multiple contention instances to be simultaneously active in a single station, each supporting a certain access category (AC). Furthermore, the standard introduces four differentiation parameters (EDCA parameters), as discussed below, which can be set individually for each access category of each individual station to enable QoS provisioning [13].

We facilitate adaptive capacity sharing between stations by letting each station have a single access category, and using the EDCA parameters for differentiating between the source stations and the bottleneck station. In principle the EDCA parameters are meant for service differentiation, while we apply it here for node differentiation. Another relevant scenario for such node level differentiation is the case of UL/DL transfer in an infrastructure-based WLAN, where the access point should get a bigger share of the resources. The considered values for the differentiation parameters per access category are introduced later on in Table 4.

In the remainder of this paper we will analyze the following four scenarios:

0. With standard IEEE 802.11, the medium needs to be idle for at least a DIFS period before stations can start to content for medium access. After winning contention a station is allowed to send exactly one packet.

In the IEEE 802.11e QoS extension, two contention-based methods are proposed to change the above procedure:

1. The initial value of the *contention window* ($\mathrm{CW}_{\min}-1$) and/or the maximum value of the contention window ($\mathrm{CW}_{\max}-1$) are set smaller for a given station, thus, this station draws its backoff from a smaller contention window, hence, has a higher probability to win contention.
2. With so-called *arbitration inter-frame spacing* (AIFS) it is possible to assign different inter-frame spacings for different service classes (or nodes) instead of the fixed DIFS. Thus, high-priority nodes can be assigned shorter AIFS, so that they can start counting off their backoff earlier, hence, have an advantage when contending for medium access.

A way to adapt the capacity sharing that does not alter the actual contention mechanism is the following:

3. The transmission opportunity limit ($\mathrm{TXOP}_{\mathrm{limit}}$) provides a time period during which a station may send packets after having won a contention. Thus, a station with a sufficiently high $\mathrm{TXOP}_{\mathrm{limit}}$ is able to send several packets and will thus be able to grab a larger share of the channel capacity than a station with a smaller $\mathrm{TXOP}_{\mathrm{limit}}$.

The above four parameters ($\mathrm{CW}_{\mathrm{min}}$ and $\mathrm{CW}_{\mathrm{max}}$, AIFS and $\mathrm{TXOP}_{\mathrm{limit}}$) in the IEEE 802.11e standard can be used to reallocate the amount of radio capacity given to the sources and to the bottleneck. These three possibilities will be addressed in detail in the models we discuss next.

3 Overall Capacity Sharing Model

We model the bottleneck B, cf. Figure 1, using an infinite-state stochastic Petri net (iSPN), as given in Figure 2. The left part of this figure contains an unbounded place (double circle) *buffer* that models the (buffer of the) bottleneck of the system. Transition *input* models the total arrival stream of packets from all active sources, whereas transition *output* models the transmission of packets leaving the bottleneck B. Note that the rates of both these transitions depend on the number of active sources and the amount of radio capacity that is allocated to each source and the bottleneck node; we come back on the form this dependency takes below. We limit the maximum number of active sources to some finite number K (taken to be 10 in most evaluations). This is a reasonable restriction, as the number of active sources in an ad hoc network cannot be arbitrarily high. We do not distinguish between individual active sources, so we can model the number of active sources as shown in the right part of the iSPN in Figure 2. To obtain a memoryless behavior needed for Markovian modeling, an inactive source becomes active after a negative exponentially distributed amount of time (with mean $1/\lambda$)and immediately instantiates a flow, which has a geometrically distributed length, measured in packets. The average size of a data packet is assumed to be $E[P] = 1500$ bytes, with exponentially distributed packet length. The duration of a flow does not only depend on its size but also on the radio capacity a source can use to transmit the flow. Note that the duration of a flow implicitly gives the source departure rate, as well.

Following the parametric assumptions made in [15], the expected amount of work put forward per flow (the amount of packets comprising the flow) equals $E[F] = 500$ packets; the other values for the key system parameters are summarized in Table 1. In Table 2 we list the four state-dependent transition rates of the iSPN, where N refers to the current number of active sources (i.e., the number of tokens in place *active sources*), and B to the current number of packets queued in the bottleneck (i.e., the number of tokens in place *buffer*). Note that the transitions *input* and *output* in fact make use of the same medium, hence, they have to share the available capacity; this is exactly what the IEEE

Table 1. Values for the system parameters

parameter	
arrival rate	$\lambda \in [0.1, 0.4]\text{sec}^{-1}$
average flow size	$E[F] = 500$ packets
overall radio capacity	$C = 917$ packets/sec
maximum of active sources	$K = 10$

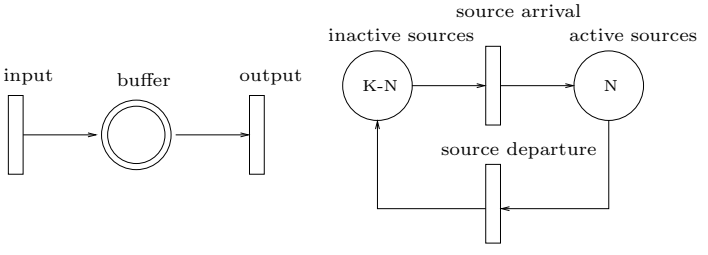


Fig. 2. High-level model as iSPN

802.11[e] access mechanism is for! The functions $S_b(\cdot)$ and $S_s(\cdot)$ (for bottleneck and source) now give the share of capacity that is allocated to the bottleneck and to all sources, respectively. Note that $S_b(\cdot)$ and $S_s(\cdot)$ depend on the number of currently active sources (N), as well as on whether or not the bottleneck has packets queued, or not ($B > 0$).

In Section 4, we will present concrete expressions for the functions $S_b(\cdot)$ and $S_s(\cdot)$; in doing so, we have achieved one generic model at the iSPN level, that can be specialized toward different QoS enhancements, by "plugging in" the appropriate bandwidth sharing functions $S_b(\cdot)$ and $S_s(\cdot)$.

Table 2. State-dependent transition rates for the iSPN

input:	**output:**
if $N = 0$ **then**	**if** $B = 0$ **then**
return 0;	**return** 0;
else	**else**
return $C \cdot S_s(\cdot)$;	**return** $C \cdot S_b(\cdot)$;
end if	**end if**

source departure:
return $C \cdot S_s(\cdot)/E[F]$

source arrival:
return $(K - N)\lambda$

Note that, in practice, the effective capacity C is not fixed, but depends on adaptive modulation, which tunes sending rates to experience link qualities.

4 Modeling the QoS Enhancements

In this section we present explicit expressions for the functions $S_s(\cdot)$ and $S_b(\cdot)$ that express the share of the wireless capacity that sources and the bottleneck

receive, resp., for each of the QoS enhancements. We use the model of Bianchi [1], plus some extensions, to compute these shares.

4.1 Bianchi's Model

Bianchi [1] proposes an analytical evaluation of the saturation throughput, assuming ideal channel conditions for basic IEEE 802.11. This model allows us to accurately compute the throughput for a fixed number of stations under the assumption that they always have a packet to send, as follows. From a single station that is modeled as discrete-time Markov model, two mutually dependent stationary probabilities are obtained: the probability τ that the station transmits a packet in a generic slot time, and the probability p that a transmitted packet encounters a collision. These probabilities are expressed in the form of a system of two non-linear equations [1, Eqn. (7) and (9)].

In a generic slot time three different events can occur: the successful transmission of a packet, a collision, or just an empty slot used for counting down backoff. Thus, the length of a generic slot time depends on the event that occurs. Considering the different durations and probabilities of these events, it is possible to compute the throughput of the system.

4.2 Engelstad's Extended Model

In a recent paper [3], Engelstad extends the model of Bianchi by including the impact of the QoS enhancements on the effectively available capacity in IEEE 802.11e. Furthermore, this extended model allows to compute the throughput for stations from different access categories, also in the non-saturated case, i.e., when the stations not necessarily always have a next packet to send.

In our de-compositional analysis approach, however, we still use the saturated case. An active source sends, on average, 500 packets in a row before becoming inactive; this means that, on average, with probability $\frac{499}{500}$ there is a next packet to be sent. Due to the decomposition, inactive sources are not considered within Engelstad's model. This might be seen as an approximation, but its impact will be small, as we will see later. For our purposes, we use two different access categories, AC_i with $i = b$ for the bottleneck, and $i = s$ for the active sources, where AC_b contains zero or one station, and AC_s contains zero to ten stations.

Engelstad now proceeds to compute similar probabilities as Bianchi does, however, now for each possible access category, i.e., the following two probabilities are computed: $\tau(i)$, the probability that a station of category i transmits, and $p(i)$, the probability that a transmission of category i is successful. As in the Bianchi model, these probabilities are defined through systems of mutually dependent non-linear Equations [3, Eqn. (5) and (12)], which can easily be solved using a tool like Maple. Note that we have to solve these equations once for every combination of number of active nodes in each access category, that is, AC_b and AC_s, to obtain throughputs for every possible combination of active nodes in the high-level model. From the above probabilities $\tau(i)$ and $p(i)$, Engelstad then derives yet another three probabilities: the probability $p_{i,s}$ for a successful

transmission for category i (cf. [3, Eqn. (27)]); the probability p_s for successful transmission for any category (cf. [3, Eqn. (28)]); and, finally, the probability p_b that the medium is busy during a generic time slot (cf. [3, Eqn. (11)]).

4.3 Relative throughputs

Using the probabilities $p_{i,s}$, p_s, and p_b obtained from the model of [3], we can derive the actual throughput for each access category, i.e, the throughput of the bottleneck, and the throughput of all active stations, as follows:

$$S_i = \frac{p_{i,s} \cdot \mathrm{txop}_i \cdot t_{\mathrm{data}}}{(1-p_b) \cdot t_{\mathrm{slot}} + p_s \cdot T_s + (p_b - p_s) \cdot T_c}, \; i \in \{b, s\}, \tag{1}$$

where the denominator states the average duration of a generic time slot, being the sum of the times for an empty slot (t_{slot}), the time for a successful transmission (T_s), and the time for a collision (T_c), weighted by their respective probability. The nominator denotes the part of a time slot that is, on average, used for transmission of data for category i. Here, txop_i denotes the number of packets of length t_{data} sent after winning contention. Instead of modeling $\mathrm{TXOP}_{\mathrm{limit}}$ as a time period, we assume that for a given packet size, a station of category i can send txop_i packets during one TXOP period[1]: Note that Equation (1) provides the input for the state-dependent transition rates in the iSPN model, cf. Table 2.

To actually compute the values, Table 3 specifies the individual durations and the default values for the system parameters that are used to compute the time for a successful transmission and the time for a collision, respectively. Considering the use of Request To Send/Clear To Send (RTS/CTS), the time for a collision is given by

$$T_c = t_{\mathrm{PHY}} + t_{\mathrm{RTS}} + t_{\mathrm{AIFSmin}}. \tag{2}$$

The time for the successful transmission of a packet with RTS/CTS depends on the TXOP values:

$$\begin{aligned} T_s = {} & t_{\mathrm{PHY}} + t_{\mathrm{RTS}} + t_{\mathrm{SIFS}} + t_{\mathrm{PHY}} + t_{\mathrm{CTS}} + \\ & (t_{\mathrm{SIFS}} + t_{\mathrm{PHY}} + t_{\mathrm{MAC}} + t_{\mathrm{data}} + t_{\mathrm{SIFS}} + t_{\mathrm{PHY}} + t_{\mathrm{ACK}}) \cdot \overline{\mathrm{txop}} + t_{\mathrm{AIFSmin}}, \end{aligned} \tag{3}$$

where $\overline{\mathrm{txop}}$ is the average number of packets sent after a successful contention computed as weighted sum:

$$\overline{\mathrm{txop}} = \sum_{i \in \{s,b\}} \mathrm{txop}_i \cdot \frac{p_{i,s}}{p_s}. \tag{4}$$

[1] That is, we model a time-based mechanism by means of a count-based mechanism; a similar modeling approach has been applied successfully, for instance, by Groenendijk [4], to model time-token access mechanisms.

Table 3. Time durations & default values

parameter	value comments	**parameter**	value comment
t_{SIFS}	$10\mu s$	t_{slot}	$20\mu s$
t_{PHY}	$192\mu s$ assuming long preamble	t_{RTS}	$160\mu s$ 20 bytes @ 1 Mbps
t_{CTS}	$112\mu s$ 14 bytes @ 1 Mbps	t_{MAC}	$25\mu s$ 34 bytes @ 11 Mbps
t_{data}	$1091\mu s$ 1500 bytes @ 11 Mbps	t_{ACK}	$112\mu s$ 14 bytes @ 1 Mbps
t_{AIFSmin}	$50\mu s$ $2 \cdot t_{\text{slot}} + t_{\text{SIFS}}$		

The different values of CW_{min}, CW_{max} are immediately taken into account, when computing $\tau(i)$ and $p(i)$, as described in [3]. When modeling access categories with different AIFS, we use the approximation as proposed in [3, Section 3.3] to compute the throughput. This implies that the minimum AIFS over all access categories is used for computing the time for a successful transmission (cf. Eqn. (3)) and the time for a collision (cf. Eqn. (2)). The remaining slots, which stations of lower categories have to wait before starting backoff countdown, are modeled as being distributed uniformly over all slots. This involves rescaling the probability that the chanel is busy, p_b. The differentiation of the AIFS is then taken into account when computing $\tau(i)$ and $p(i)$. Modeling differentiation by means of $\text{TXOP}_{\text{limit}}$ is also incorporated in Equation (1), which is an extension of the throughput as presented in [3].

Table 4 specifies the values for the differentiation parameters for the bottleneck and the sources that are considered in the following. The default values are marked with an asterisk.

Table 4. Considered values for the differentiation parameters per access category

parameter	value for AC_b	value for AC_s
AIFS_b (slots)	2*	2*, 7 or 12
CW_{min} (slots)	32*	32*, 64 or 128
r_{max}	4*	4*
txop (packets)	1*, 2 or 3	1*

4.4 Measures of Interest

The bridge B is the bottleneck of the two-hop ad hoc network. We therefore study the expected number of packets in the buffer of the bottleneck (M1), as well as the throughput of the bottleneck (M2), and the expected number of active sources (M3), for all possible QoS mechanisms and for different source arrival rates. Instead of specifying the performance models of interest manually at the state level, we instead use a high-level specification mechanism. Given the generic iSPN and the expressions for the throughputs, a generation algorithm is used to automatically derive the underlying *Quasi Birth Death* (QBD) process [9,7,11].

Note, that we are not able to compute flow-related measures, as the flow time or flow throughput, as our model does not distinguish between packets of the different sources. However, we think that the throughput at the bottleneck and, possibly, the throughput at the sources is enough to capture the effects of differentiation in this scenario.

5 Simulation Model

The bottleneck scenario has also been modeled and simulated in OPNET, version 11.5. [10]. In this section we explain the simulation setup and the parameter settings for the different QoS parameters.

OPNET is organized hierarchically in levels, where every level adds more detail. At the highest level, we place ten *advanced WLAN stations*, as provided by OPNET, to model the ten sources, and two more to model the bottleneck and the sink, where all the data is sent to. We then adapt the WLAN stations in the node editor once for the sources and once for the bottleneck. For the sources, a new traffic generation process is created that is put inside each source. The traffic generation process can be either active or inactive. As soon as it becomes active it places a geometrically distributed amount of packets into its MAC layer queue. When the transmitter has been able to send all these packets to the bottleneck it gets inactive again. The packet size is set to 1500 bytes. As in the analytical model all sources are independent and becomes active with the global arrival rate λ, when currently inactive.

The WLAN station that models the bottleneck does not need a traffic generation process. Arriving packets from the MAC layer are immediately routed back to the MAC layer and are forwarded to the sink. The size of the data buffer in the MAC layer is set to the highest possible value, 10^8, to match the assumption of the infinite buffer in the analytical model as accurately as possible. Once the buffer limit is reached, data packets will be discarded until the buffer has free space to store new packets. Note that the complete access mechanism, in all its details, is being included in the OPNET simulation. No approximations or further assumptions are being done. The *wireless LAN parameters* in OPNET are set as follows:

- the data rate is set to 11 Mbps,
- regular RTS/CTS is enabled, the RTS threshold is set to 256 bytes,
- EDCA parameters are not supported for basic IEEE 802.11 and supported for IEEE 802.11e and set according to the simulation scenario under study.

The content of the MAC layer queue is sampled over time and its time average corresponds to the measure *expected buffer occupancy* (M1) in the analytical model. The number of currently active sources is also sampled and its time average corresponds to *expected number of active sources* (M3) in the analytical model. Furthermore, the throughput of the bottleneck is sampled over time and its time average is compared to the *throughput* (M3) of the bottleneck in the analytical model. For every QoS parameter we consider seven different loads:

$\lambda \in \{0.01; 0.015; 0.02; 0.025; 0.03; 0.035; 0.04\}$. Every value of λ is simulated in every scenario with ten randomly chosen seeds for two hours, leading to 70 simulation runs per curve. One simulation run takes between 20 and 50 minutes, resulting in an estimated run time per point, including confidence intervals, of 200 to 500 minutes. In the following we show the mean of the simulation results for ten seeds together with the corresponding 95% confidence interval.

6 Comparing Analytical and Simulation Results

In the following sections we compare the analytical and simulation results for the four different scenarios, followed by a discussion of the throughput that is achieved in the bottleneck, depending on the differentiation parameters.

6.1 Basic Scenario

In the basic 802.11 scenario the EDCA parameters are set to the default values, as given in Table 4. Figure 3 shows the expected buffer occupancy in IEEE 802.11 without differentiation and Figure 4 shows the expected number of active sources. In the basic scenario the results from simulation and analysis are very close to each other, the estimated expected buffer occupancy is always inside the confidence intervals. With increasing λ the variance of the simulated buffer occupancy grows as seen from the larger confidence intervals. As could be expected,

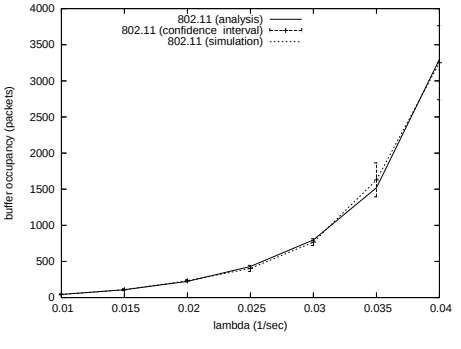


Fig. 3. Mean buffer occupancy for IEEE 802.11 at bottleneck

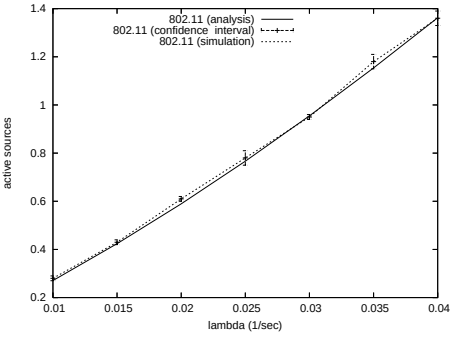


Fig. 4. Mean number of active sources for IEEE 802.11

the buffer occupancy increases exponentially with λ. For λ greater than 0.04 the system soon becomes overloaded. The steady state distribution then does not exist any more and the buffer of the bottleneck in the simulation overflows. The number of active sources grows almost linearly with increasing λ.

6.2 Differentiating with CW_{min}

We compare the results from simulation and analysis for two different settings for the expected buffer occupancy in Figure 5 and in Figure 6 for the expected number of active sources. In this scenario, CW_{min} is extended in the sources to $CW_{min} = 64$ and to $CW_{min} = 128$, whereas the other EDCA parameters are set to the default values as specified in Table 4. Compared to basic IEEE 802.11 the mean buffer occupancy is lower when the sources operate with a larger window size. This is due to the fact that the bottleneck gets a higher capacity share. On the other hand the sources remain active longer (expected number of sources is higher than in the basic scenario).

We observe that in the parameter setting $CW_{min,s} = 64$, simulation and analysis results are close for buffer occupancy and number of active sources, respectively. For the parameter setting $CW_{min,s} = 128$, the analysis overestimates the capacity that is allocated to the bottleneck. Hence, the mean buffer occupancy is underestimated and the mean number of active sources is overestimated by the analysis.

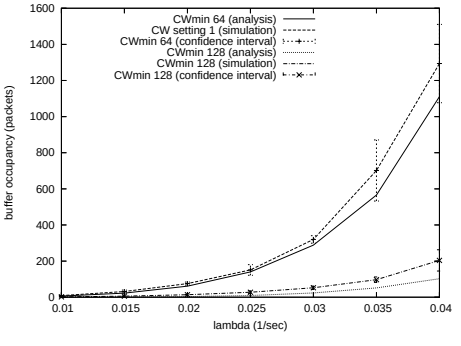


Fig. 5. Mean buffer occupancy for different $CW_{min,s}$ and $CW_{min,b} = 32$

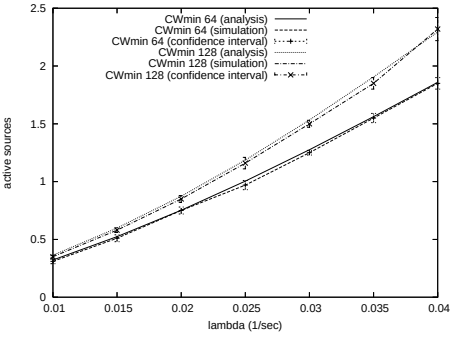


Fig. 6. Mean number of active sources for different $CW_{min,s}$ and $CW_{min,b} = 32$

6.3 Differentiating with AIFS

In this scenario the value of AIFS is changed first to 7 and then to 12 in the sources, while the other EDCA parameters remain set as in the basic scenario. The analytical results and the simulation results for the two different AIFS settings are compared in Figure 8 for the expected number active sources. Simulation shows that our results are highly accurate.

Figure 7 shows the expected buffer occupancy at the bottleneck on a logarithmic scale. The mean buffer occupancy is much lower than in the basic scenario. It is also lower than in the CW scenario, while the mean number of active source is comparable. This indicates that in the CW scenario more capacity is wasted due to longer backoff times.

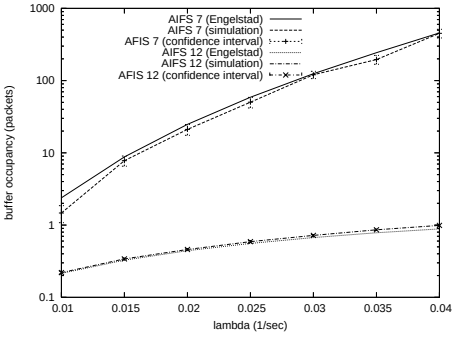


Fig. 7. Mean buffer occupancy for different AIFS_s and $\text{AIFS}_b = 2$

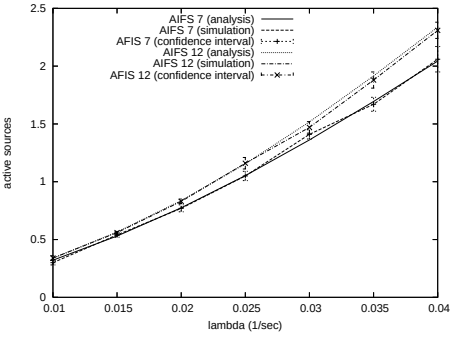


Fig. 8. Mean number of active sources for different AIFS_s and $\text{AIFS}_b = 2$

6.4 Differentiating with $\text{TXOP}_{\text{limit}}$

To study the influence of $\text{TXOP}_{\text{limit}}$ this value is set to 4000 μs, and to 6000 μs in the bottleneck only, while the other EDCA parameters remain unchanged. A $\text{TXOP}_{\text{limit}}$ of 4000 microseconds allows two data packets to be sent, whereas a $\text{TXOP}_{\text{limit}}$ of 6000 microseconds allows three data packets to be sent. The resulting curves from analysis and simulation are compared in Figure 9 for the buffer occupancy and in Figure 10 for the number of active sources, respectively. The mean buffer occupancy is much lower than in all other scenarios. The number of active sources is lower than in the other differentiated scenarios and only slightly higher than in the basic scenario. This is due to the fact that less packets have to undergo contention and thus less collisions occur with leads to a higher effective capacity.

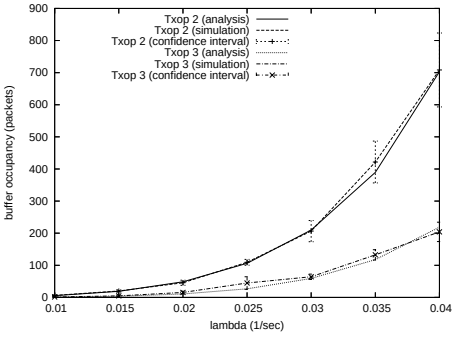


Fig. 9. Mean buffer occupancy for different TXOP_b and $\text{TXOP}_s = 1$

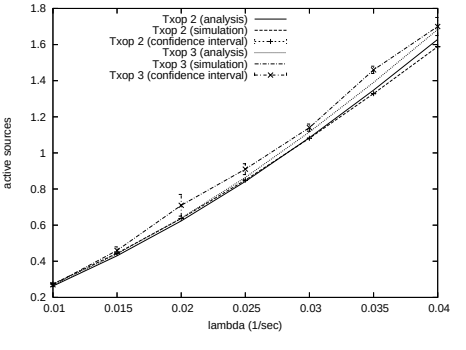


Fig. 10. Mean number of active sources for different TXOP_b and $\text{TXOP}_s = 1$

6.5 Throughput

To further analyze the impact of the differentiation parameters, we compute the throughput of the bottleneck, as a function of the parameter λ. The throughput for the basic IEEE 802.11 is compared with the throughput that is achieved in three differentiated settings: when $\mathrm{TXOP}_{\mathrm{limit}}$ of the bottleneck is set to 3, when $\mathrm{CW}_{\mathrm{min},s}$ is set to 128 and when AIFS_s is set to 12. In Figure 11 we show our highly accurate analytical results for the throughput in these different settings together with the confidence intervals of the corresponding simulation runs.

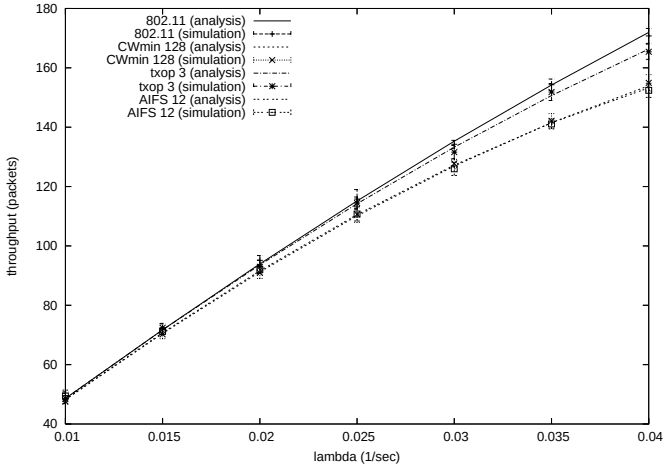


Fig. 11. Throughput of the bottleneck

For all $\lambda \in \{0.01, \ldots 0.04\}$, the largest throughput is achieved in basic IEEE 802.11. The throughput for $\mathrm{txop}_b = 3$ is just a little lower for increasing values of λ. The throughput for differentiated $\mathrm{CW}_{\mathrm{min},s}$ and AIFS_s fall together and this throughput is considerably lower than in the two other cases.

This non-intuitive result is due to the fact, for given λ, the offered load depends on the average number of active sources. Recall, that the number of active sources is higher in the differentiated settings than in basic IEEE 802.11, as part of the capacity has been moved from the sources to the bottleneck. When the sources remain active for a longer time, they become active less frequently, so in total fewer packets are introduced to the bottleneck. Clearly, the bottleneck can only forward packets that have been sent to him from the sources. Since we assume infinite queue lenght, the throughput equals the offered load as long as the bottleneck queue is stable. Note that with differentiation the queue remains stable for much higher λ, compared to the standard 802.11. As a result, the *maximum* throughput using differentiation is expected to be higher than for the standard 802.11.

Finally, when CW_{min} and AIFS are increased to differentiate, this comes at the cost of a decreased effective capacity, as more time slots will pass unused. In contrast, increasing $TXOP_{limit}$ effectively increases capacity, as multiple packets are sent within $TXOP_{limit}$ after just one contention period. However, as can be seen in Figure 11 the increase in effective capacity is not enough to compensate for the slightly higher mean number of active sources for different $TXOP_b$, as shown in Figure 10.

7 Conclusions

In this paper we have presented a new model for analyzing the recently standardized quality-of-service enhancements of the IEEE 802.11e access mechanism in a two-hop ad hoc network. Our high-level model is flow-based, and uses results from packet-based models (such as those proposed by Bianchi and Engelstad [1,3]), and allows for the numerical evaluation of the buffer occupancy at the bottleneck node, the system throughput, as well as provides information on the mean number of active sources. The latter is important, as our model allows for a time-varying number of active sources, as opposed to earlier models that only allow for a fixed number of sources. The model is very easy to use (and extensible) as the basic model structure remains the same for all enhancements; just allocation functions (denotes as $S_b()$ and $S_s()$ in the paper) need to be defined. An efficient numerical solution based on the underlying quasi-birth-death structure of the model is automatically provided (using previously defined algorithms, cf. [12]).

We compare our results with extensive simulations (built using OPNET) and show that our models provide very accurate results at almost negligible cost in comparison to the simulations. No other analytical models that allow for similar evaluations have been proposed so far.

The results show that all differentiation parameters are able to allocate capacity in a more balanced way to the bottleneck and the sources, resulting in a much smaller buffer occupancy. However, the throughput of the bottleneck differs, depending on the differentiation used. In this paper, we only analyzed the throughput of the bottleneck as a function of the source arrival rate λ.

Further work is needed to compare the maximum throughput that can be achieved per differentiation parameter, for arbitrary values of λ.

Acknowledgments

We thank Patrick Goering for his help with the OPNET simulations. Anne Remke is supported through the MC=MC project (612.000.311) and Lucia Cloth is supported through the MOCS project (642.000.505), both financed by the Netherlands Organization for Scientific Research (NWO). We thank the reviewers for the detailed comments on this paper.

References

1. Bianchi, G.: Performance analysis of the IEEE 802.11 Distributed Coordination Function. IEEE Journal on Selected Areas in Communications 18, 535–547 (2000)
2. Cohen, J.W.: The multiple phase service network with Generalized Processor Sharing. Acta informatica 12, 245–284 (1979)
3. Engelstad, P.E., Osterbo, N.: Non-saturation and saturation analysis of IEEE 802.11e EDCA with starvation prediction. In: MSWiM 2005: Proceedings of the 8th ACM International Symposium on Modeling, Analysis and Aimulation of Wireless and Mobile Systems, pp. 224–233. ACM Press, New York (2005)
4. Groenendijk, W.P.: Conservation Laws in Polling Systems. PhD thesis, University of Utrecht (1990)
5. Ieee std 802.11e - 2005, 'part 11: 'Wireless LAN Medium Access Control (MAC) and Physical layer (PHY) specifications. amendment 8: Medium Access Control (MAC) Quality of Service Enhancements. amendment to IEEE 802.11 std. (2005)
6. Kurose, J.F., Ross, K.W.: Computer Networking. Addison-Wesley, Reading (2005)
7. Latouche, G., Ramaswami, V.: Introduction to Matrix Analytic Methods in Stochastic Modeling, ASA-SIAM, Philadelphia (1999)
8. Litjens, R., Roijers, R., van den Berg, J.L., Boucherie, R.J., Fleuren, M.J.: Analysis of flow transfer times in IEEE 802.11 wireless lans. Annals of Telecommunications 59, 1407–1432 (2004)
9. Neuts, M.F.: Matrix Geometric Solutions in Stochastic Models: An Algorithmic Approach. Johns Hopkins University Press (1981)
10. Opnet modeler software, `http://www.opnet.com/products/modeler`
11. Ost, A.: Performance of Communication Systems. Springer, Heidelberg (2001)
12. Remke, A., Haverkort, B.R., Cloth, L.: A versatile infinite-state Markov reward model to study bottlenecks in 2-hop ad hoc networks. In: 3rd International Conference on the Quantitative Evaluation of SysTems (QEST), pp. 63–72 (2006)
13. Roijers, F., van den Berg, J.L., Fan, X., Fleuren, M.: A performance study on service integration in IEEE 802.11E wireless LANs. Computer Communications 29, 2621–2633 (2006)
14. Schiller, J.: Mobile Communications. Addison-Wesley, Reading (2003)
15. van den Berg, J.L., Mandjes, M., Roijers, F.: Performance Modeling of a Bottleneck Node in an IEEE 802.11 Ad-hoc Network. In: Proceedings 5th International Conference on AD-HOC Networks and Wireless, pp. 321–336 (2006)

Contention-Based Polling Efficiency in Broadband Wireless Networks

Sergey D. Andreev[1], Andrey M. Turlikov[1], and Alexey V. Vinel[2]

[1] St. Petersburg State University of Aerospace Instrumentation,
Bolshaya Morskaya street, 67,
190000, St. Petersburg, Russia
{corion,turlikov}@vu.spb.ru
http://suai.ru/

[2] St. Petersburg State University of Information Technologies, Mechanics and Optics,
Vasilievsky Island, Birjevaja Linija, 4,
199034, St. Petersburg, Russia
vinel@ieee.org
http://www.ifmo.ru/eng/

Abstract. This paper addresses the performance of the contention-based polling techniques at the bandwidth reservation stage of IEEE 802.16 standard. A general proposition is proved, which establishes that the grouping of users in the random multiple access system does not change its capacity. Broadcast and multicast polling mechanisms are then considered, for which the throughput and the rate of the truncated binary exponential backoff algorithm are calculated for the lossy and the lossless system types, respectively. It is shown, that subject to proper optimization the performance of the aforementioned algorithm is the same for both system types. The efficiency of the symmetric user grouping is finally studied, which demonstrates that a negligible performance gain may be achieved for the cost of the increased IEEE 802.16 overhead.

1 Introduction and Background

IEEE 802.16 standard [1] defines a high-speed access system supporting multimedia services. In IEEE 802.16 protocol stack the *medium access control* (MAC) layer supports multiple physical (PHY) layer specifications, each of them covering different operational environments. IEEE 802.16 is likely to emerge as an outstanding cost-competitive technology mainly for its longer range and sophisticated *quality-of-service* (QoS) support at the MAC layer.

Many research papers concentrate on the performance evaluation of the various IEEE 802.16 features. In particular, the bandwidth requests transmission by a system user to reserve a portion of the channel resources is frequently addressed. A detailed description of the reservation techniques is known from the fundamental work in [2]. The standard allows a *random multiple access* (RMA) scheme at the reservation stage and implements the truncated *binary exponential backoff* (BEB) algorithm for the purposes of the collision resolution.

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 295–309, 2008.

The asymptotic behavior of the BEB algorithm was substantially addressed in the literature. In [3] it was shown that the BEB algorithm is *unstable* in the infinitely-many users case. By contrast, [4] shows that the BEB is *stable* for any finite number of users, even if it is extremely large, and sufficiently low input rate. These seemingly controversial results demonstrate the two alternative approaches to the analysis of an RMA algorithm [5]. The former is the *infinite population* model, which studies the ultimate performance characteristics of an RMA algorithm. The latter is the *finite population* model that addresses the limits of the practical algorithm operation. An exhaustive description of both models may be found in [6] and [7].

Both finite and infinite models require a framework of additional assumptions, which makes the analysis mathematically tractable. The set of assumptions given by [8], [9] and in Section 3 has nowadays become classical and evolved into a *reference* RMA model. The performance of the BEB algorithm in the framework of the reference model is addressed in [10], which allowed a deeper insight into its operation. In [11] an extremely useful Markovian model to analyze the performance of the BEB algorithm was first introduced.

Together with the analysis of the BEB itself, much attention is paid to its proper usage in IEEE 802.16 standard. In Section 2 we give a brief description of IEEE 802.16 features. It is known that the BEB algorithm may be adopted for both *broadcast* and *multicast* user polling. In case of multicast polling the set of all system users is divided into smaller subsets. The efficiency of broadcast and multicast polling was extensively studied in [12]. Some practical aspects of the BEB application for the delay-sensitive traffic were considered in [13].

The motivation behind this paper is to show that despite the fact that for some scenarios multicast polling results in a slightly better system performance, like it is claimed in [12] and [13], the gain is practically negligible when all the users share the similar QoS requirements. In order to verify this hypothesis, we firstly address the infinite population model in Section 4 and show that the RMA *capacity* (see [14] and [15]), cannot be increased by the grouping of users.

In Section 5 we study the BEB algorithm performance in a practical finite population model. Further, by using various analytical techniques we mathematically express the possible gain from the use of broadcast/multicast polling for the different types of the system. The Conclusion summarizes the paper.

2 Standard Overview

IEEE 802.16 standard specifies PHY and MAC layers and supports two modes of operation: the mandatory point-to-multipoint mode (PMP) and the optional mesh mode. The MAC layer is subdivided into three hierarchical sub-layers. Through the *convergence sub-layer* IP, ATM and Ethernet traffic types are supported. Five levels of QoS are specified within *MAC sub-layer*, which correspond to the QoS classes. MAC data packets can be variable-length with concatenation and fragmentation mechanisms supported. *Privacy sub-layer* performs the encryption of the data packets together with the other cryptographic functions.

The basic IEEE 802.16 architecture assumes that there are one *base station* (BS) and one or more subscriber stations, which are referred to as *users* in what follows. The packet exchange between the BS and the users is assumed to be via separate channels. A *downlink* channel is from the BS to the users and the *uplink* channel is in the reverse direction. Therefore, there is no particular connection associated with the downlink channel, while in the uplink channel all the connections from all the users are multiplexed.

IEEE 802.16 defines two duplexing mechanisms the channels: time division duplex (TDD) and frequency division duplex (FDD). In the TDD mode the frame is separated into the downlink and the uplink parts. The simplified structure of the MAC frame in the TDD mode is shown in Fig. 1. In the FDD mode the users transmit in different sub-bands and do not interfere with each other.

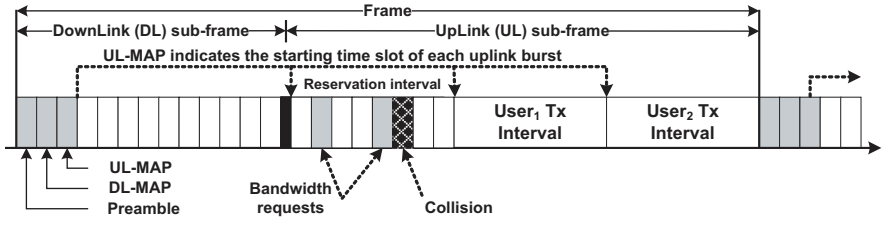


Fig. 1. IEEE 802.16 frame structure in the TDD mode

In the downlink channel the BS is the only station to broadcast packets to all the users. Together with the data packets, the BS also transmits service information about the schedule for each of the users in the uplink channel. This information is incorporated in the UL-MAP message and is used by the users for scheduling their data packets in the uplink channel. To allow the feedback from the users the BS also specifies a portion of channel resources as the *reservation interval*. During this interval the users transmit their *bandwidth requests* (or requests, for short), which are then processed by the BS.

The access procedure of the users to the reservation interval could be either *contention-based* or *contention-free*. The latter is referred to as unicast polling, where the BS assigns a transmission opportunity (which is referred to as *slot* below) to each user for its bandwidth requests. The former comprises two mechanisms, namely, multicast and broadcast polling. During broadcast polling the users send their bandwidth requests by choosing one of all the available slots. In case of multicast polling the users are polled in groups and within a group the rules of broadcast polling apply. During the contention-based access the request collisions may occur, which are resolved following the truncated binary exponential backoff algorithm. The *piggybacking* mode allows a user to attach further bandwidth requests to its data packets, once bandwidth for the first packet has been granted by the BS.

3 Reference System Model

In this section the reference RMA model [8], [9] is discussed that simplifies the below derivations. The system time is divided into adjacent frames of the equal duration. The frames are enumerated with integer and nonnegative numbers. Suppose there are M users in the system. We formulate additional assumptions about the way the requests arrive into the system and are transmitted.

Assumption 1. According to IEEE 802.16 standard each user may potentially establish multiple connections with the BS using different negotiated QoS parameters, and a bandwidth request can be issued on a per-connection or a per-station basis. In what follows we assume that each user has only one connection at a time and all the connections belong to the same QoS class.

Assumption 2. Each frame comprises K equal contention slots for the request transmissions. K is constant throughout the system operation.

Assumption 3. In each slot one of the following situations may occur:
– exactly one user transmits its request (*success*);
– none of the users transmit the request (*empty*);
– two or more users transmit their requests simultaneously, which results in the corruption of all the requests at the BS (*collision*).

Assumption 4. The uplink channel is noise-free. Therefore, the BS faultlessly determines, which situation occurred in a slot. If only one user transmits, then the BS always decodes the bandwidth request successfully.

Assumption 5. No piggybacking is used and for each arrived data packet a separate bandwidth request is generated. As we concentrate on the bandwidth reservation process, we assume the virtual input flow of requests into the system.

Assumption 6. By monitoring user activity in the frame $t-1$ the BS makes a schedule for the uplink sub-frame of the frame t and broadcasts this schedule in the downlink sub-frame of the frame t. A user receives the feedback from the request transmission in the frame $t-1$ by the beginning of the frame t.

According to the standard this is not the case. Feedback information is not explicitly transmitted to a user. A special request timeout is used to wait for the uplink grant from the BS, and only if it is expired, the request transmission is considered corrupted. We make this 'immediate' feedback assumption for the simplicity of the analysis only. All the forthcoming derivations may be generalized for the case of the 'delayed' feedback.

Assumption 7. The downlink channel is noise-free. Therefore, a user faultlessly receives the schedule and the request transmission feedback from the BS.

Assumption 8. Denote the random number of the new request arrivals to the user i in the frame t by $X_i^{(t)}$. For all $t \geq 0$ and $i = 1, \ldots, M$ the random variables $X_i^{(t)}$ are independent and identically distributed (i.i.d.). Assume also that at

most one new request arrives to a user per frame with the probability y. Thus, $E[X_i^{(t)}] = y$ for all $t \geq 0$ and $i = 1, \ldots, M$, as well as $E[\sum_{i=1}^{M} X_i^{(t)}] = My \triangleq \Lambda$. The value of Λ is hereinafter referred to as the cumulative *input rate* and the considered input flow constitutes a Bernoulli flow.

4 Infinite User Population

Following the approach from [5] we allow the number of users in the system M to increase infinitely and the probability of a request arrival y to decrease simultaneously so that their product remains constant, that is $My = const = \Lambda$. Then the limit of the cumulative arrival process given by Assumption 8 is Poisson, i.e. $\lim_{M\to\infty} \Pr\{\sum_{j=1}^{M} X_j^{(t)} = i\} = \frac{\Lambda^i}{i!} e^{-\Lambda}$. Below we make the basic definitions and introduce *lossy* and *lossless* system types as follows.

4.1 Lossy System

Definition 1. The *RMA algorithm* A from the class of algorithms for the lossy system $A \in \mathcal{A}_{lossy}$ is defined as a rule that allows a user with a pending request to determine whether it should transmit this request in the following slot s or *discard* it. If a request is discarded then the corresponding data packet is lost [16].

Definition 2. We introduce a random variable $Z^{(t)}$, which is the number of the successful request transmissions in a frame comprising K slots. Clearly, $Z^{(t)} \in \{0, 1, \ldots, K\}$. Define the random variable $\Psi_A(K, \Lambda, s) \triangleq \frac{\sum_{j=0}^{s} Z^{(t)}}{sK}$. The limit of this expression for s, if it exists, represents the *output rate* per slot of the algorithm A in the lossy system, that is $\Psi_A(K, \Lambda) \triangleq \lim_{s\to\infty} \Psi_A(K, \Lambda, s)$.

Definition 3. The *throughput* of the algorithm A in the lossy system is the maximum achievable output rate for all the input rates, which implies:

$$T_A(K) \triangleq \sup_{\Lambda} \Psi_A(K, \Lambda). \tag{1}$$

Definition 4. The *capacity* of the lossy system is the maximum throughput over the class $\mathcal{A}_{lossy}(K)$ of the RMA algorithms with K slots per frame:

$$\mathcal{C}_{lossy}(K) \triangleq \sup_{A\in\mathcal{A}_{lossy}(K)} T_A(K)\,. \tag{2}$$

Notice, that the throughput value characterizes the behavior of an RMA algorithm, whereas the capacity gives the ultimate performance threshold for the entire lossy system.

4.2 Lossless System

Definition 5. The *RMA algorithm* A from the class of algorithms for the lossless system $A \in \mathcal{A}_{lossless}$ is defined as a rule that allows a user with a pending request to determine whether it should transmit this request in the following slot s. Notice, that no discard rule is specified and, consequently, requests are never lost.

Definition 6. The *request delay* for an RMA algorithm is the time interval from the moment of the request generation to the moment of its successful transmission. The delay $\delta_A(K, \Lambda)$ is a random variable. We inject a new request into the system at the randomly chosen slot s, and denote the delay of this request as $\delta_A^{(s)}(K, \Lambda)$.

Definition 7. The *mean delay* (referred to as virtual mean delay in [7]) is defined as:

$$D_A(K, \Lambda) \triangleq \overline{\lim_{s \to \infty}} E[\delta_A^{(s)}(K, \Lambda)]. \tag{3}$$

Definition 8. The transmission *rate* (tenacity) of the algorithm A in the lossless system is the maximum input rate that can be sustained by the algorithm with finite request delay:

$$R_A(K) \triangleq \sup_{\Lambda} \{\Lambda : D_A(K, \Lambda) < \infty\}. \tag{4}$$

Definition 9. The *capacity* of the lossless system is the maximum possible rate over the class $\mathcal{A}_{lossless}(K)$ of the RMA algorithms with K slots per frame:

$$\mathcal{C}_{lossless}(K) \triangleq \sup_{A \in A_{lossless}(K)} R_A(K) \ . \tag{5}$$

The exact value of the capacity is not yet established. However, the best known upper bound on the capacity $\mathcal{C}_{lossless}(1)$ was established in [14] and is shown to be $\overline{\mathcal{C}}_{lossless}(1) = 0.587$. The best known part-and-try RMA algorithm was proposed in [17] and its rate is $R_{pt} = 0.487$. In subsequent years it was slightly improved, but the core idea of the algorithm remained unchanged.

Notice again, that the rate value characterizes the behavior of an RMA algorithm, whereas the capacity gives the ultimate performance threshold for the entire lossless system.

4.3 User Grouping Analysis

Here we concentrate on showing that the grouping of users does not increase the ultimate measure of the system performance, namely, its capacity. The below arguments may be repeated similarly for both lossy and lossless types of the system. Below we demonstrate the proof for the lossless system, but omit the lower 'lossless' index at $\mathcal{A}$ and $\mathcal{C}$ as redundant.

1. Firstly, consider the RMA system without the framing structure. The system time is divided into equal slots and a user is restricted to start its request transmission in the beginning of a slot. The RMA algorithm A in this system $\mathcal{A}_{slotted}$ may again be defined as a rule that allows a user with a pending request to determine whether it should transmit this request in the following slot s. The feedback of the user transmission is available by the beginning of the next slot $s+1$.
2. Now we additionally divide the system time into frames with each frame comprising some integer and constant number of slots K. However, the feedback is still available after each slot. It is assumed that all the system users monitor the system activity from the start of its operation. Therefore, all the users determine the situation in each slot similarly and the introduction of frames neither improves nor degrades the system performance. The conclusion we draw from this fact is that the set of all the RMA algorithms for this system $\mathcal{A}_{framed}$ coincides with the set of algorithms for the slotted system, that is $\mathcal{A}_{framed} = \mathcal{A}_{slotted} \triangleq \mathcal{A}(1)$. Analogously to the Definition 9 we define the capacity of the framed system as $\mathcal{C}(1) \triangleq \sup_{A \in A(1)} R_A(1)$.
3. We change the feedback availability for the framed system and let a user know the consequences of a request transmission only in the beginning of the next frame, that is, once in K slots. An alternative system with 'delayed' feedback was considered in [18]. We define the RMA algorithm A for this system $\mathcal{A}(K)$ as before and conclude that with the restriction on the feedback availability the set of all possible RMA algorithms is narrowed in comparison to the respective set for the framed system, which yields $\mathcal{A}(K) \subset \mathcal{A}(1)$.

   From the above and the two definitions of capacity $\mathcal{C}(1)$ and $\mathcal{C}(K)$ (5) it immediately follows that $\mathcal{C}(K) \leq \mathcal{C}(1)$.
4. To any algorithm A from the set $\mathcal{A}(1)$ an algorithm A^* may be put into correspondence that belongs to $\mathcal{A}(K)$, such as $R_{A^*} = R_A$. For this it is sufficient to split all the users of the framed system into K equal groups and restrict the slots available for each group to one slot per frame. For instance, group number one monitors and transmits in the first slot of each frame, group number two - in the second, etc. Therefore, for each group the feedback is available at the beginning of the next slot, dedicated to this particular group, which corresponds to the slotted system.
5. From the definition of the capacity and the above (see 3, 4) it follows that $\mathcal{C}(1) = \mathcal{C}(K)$, that is, the capacity does not change for the framed system. Moreover, when all the system users are already split into equal groups with L slots for each of them, the capacity does not change either, i.e. $\mathcal{C}(1) = \mathcal{C}(L) = \mathcal{C}(K)$. We conclude that the grouping of users leaves the system capacity unchanged.

5 Finite User Population

In this section we address the contention-based polling performance in the framework of the model from Section 3 for a practical case of the finite number of users M. We narrow the set of all the RMA algorithms to one algorithm which

is specified by IEEE 802.16 standard. This algorithm is the truncated binary exponential backoff (BEB), which steps may be summarized as follows.

5.1 Truncated Binary Exponential Backoff Algorithm

Rule 1.1. If a new bandwidth request arrives to a user in the frame $t-1$ and this user has no other pending requests, it transmits the request in the frame t (transmission attempt). The slot for the request transmission is sampled uniformly from the number of contention slots dedicated to the group the user belongs to. Notice, that in case of broadcast polling the user may choose between all the contention slots K of the frame t, whereas in case of multicast polling the choice is narrowed to L slots of the respective multicast group.

Rule 1.2. If a request is ready for *retransmission* at the beginning of the frame t at its i-th retransmission attempt ($i > 0$), a user chooses a number (*backoff counter*) in the range $\{0, 1, \ldots, 2^{min(m,i)}W - 1\}$ uniformly, where W and m are the parameters of the BEB algorithm, named *initial contention window* and *maximum backoff stage* respectively and i is the number of collisions this request suffered from so far. The user then defers the request retransmission for the chosen number of slots, accounting only for the slots dedicated to its group.

Rule 2.1. If, after receiving the feedback from the BS, the user determines that its last request collided, it increments the collision counter i for this request. If this counter coincides with the maximum allowable number of retransmission attempts Q, then the request together with the corresponding data packet is discarded and the collision counter is reset to $i = 0$.

Rule 2.2. If, after receiving the feedback from the BS, the user determines that the (re)transmission of the last request was successful, it resets the collision counter to $i = 0$.

5.2 Lossy System

We are interested in the derivation of the BEB algorithm throughput T_{BEB} for the case of minimum possible delay. The motivation for this is the performance evaluation of the delay-critical applications (like VoIP in [13]). In order to minimize the delay for both broadcast and multicast polling the the maximum number of retransmission attempts is set to its minimum value, that is $Q = 0$. Therefore, the corresponding throughput value is denoted as T^1_{BEB}, where 1 stands for the single transmission attempt.

Remember, that according to the Bernoulli input flow (see Assumption 8) the value of y represents the probability of a request arrival to a user in a frame. The standard does not define any relationship between the parameters W, m and K. Notice, for example, that if $W < L$ for multicast polling, then some slots are never used during the first retransmission attempt. For this reason, we set $W = \frac{lK}{G} = lL$, where l is a natural number ($l \geq 1$), in order to distribute the retransmission attempts over the available slots for each multicast group uniformly. In case of no retransmission attempts, $l = 1$ and $m = 0$.

Below we address the throughput T^1_{BEB} per slot, which is achievable by the transmission in the contention slots for both broadcast ($G = 1$) and multicast ($G > 1$) polling. Following the approach from [19] we establish the following:

$$T^1_{BEB}(G, y) = \frac{G}{K} \sum_{k=0}^{N} \binom{N}{k} y^k (1-y)^{N-k} \sum_{i=0}^{\min(k,L)} iP(i, k, L), \tag{6}$$

where $P(r, k, L)$ is the probability that r stations out of k active (with at least one pending request) successfully transmit in a frame that comprises K slots. Denote by $F(r, k, L)$ the total number of ways to put k balls into L boxes, conditioning on the fact that exactly r boxes contain one ball. This number may be computed recursively by the following expressions:

$$F(0,0,L) = 1, F(0,k,0) = 0, F(0,k,L) = L^k - \sum_{i=1}^{\min(i,L)} F(i,k,L), k > 0,$$

$$F(r,k,L) = \binom{k}{r} \binom{L}{r} r! F(0, k-r, L-r), 0 < r \leq \min(k, L). \tag{7}$$

Thus, the conditional probability $P(r, k, L)$ equals to:

$$P(r,k,L) = \frac{F(r,k,L)}{L^k}. \tag{8}$$

Fig. 2 demonstrates the function T^1_{BEB} for different number of groups G, $K = 8$ and $M = 40$. We observe that multicast polling outperforms broadcast polling for small input rates y, whereas the situation reverses for moderate and high input rates. We also notice that the gap between the cases with $G = 1$ and $G = 8$ is the most significant and shows the maximum possible gain/loss from the use of either of polling techniques. We plot the dependence of this maximum gain/loss on the input rate in Fig. 3.

We conclude that despite the fact that the use of multicast or broadcast polling demonstrates a throughput trade-off for different values of request arrival rate, the maximum possible gain/loss is negligible in comparison to the achievable throughput. Therefore, it is not reasonable to split users into multicast groups for the considered minimum delay case ($Q = 0$), as the gain is minor, but IEEE 802.16 overhead increases as the number of groups grows [1].

The result given by (6) may alternatively be obtained with the following approach, one similar to which was first addressed in [20]. In each slot at most one request may be transmitted. We introduce a random variable $Z^{(i)}$ that is equal to 1 in case of *success* in the slot i and is equal to 0 otherwise. Notice, that as the number of users in each group is constant and the users are independent, it is sufficient to obtain the expectation of the sum $Z^{(i)}$ over L for one group only. Clearly, this expected value gives the sought throughput T^1_{BEB}, that is:

$$T^1_{BEB} = \frac{E[\sum_{i=1}^{L} Z^{(i)}]}{L} = E[Z^{(i)}]. \tag{9}$$

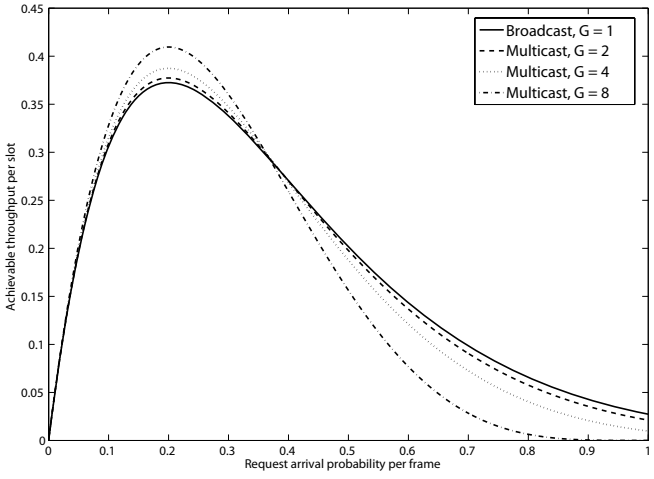


Fig. 2. Contention-based polling efficiency in case of no retransmissions

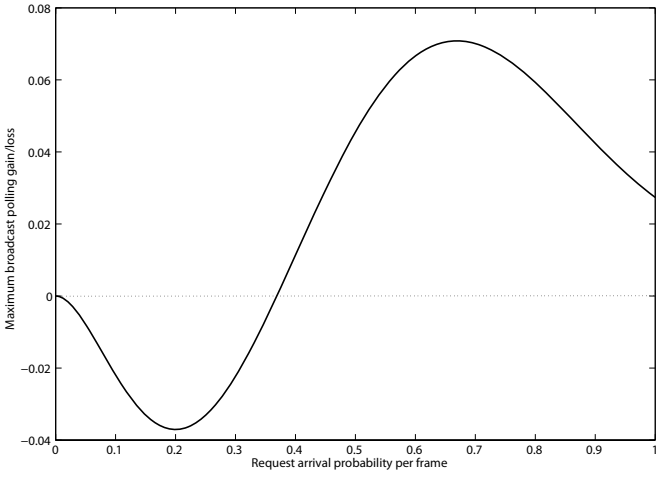


Fig. 3. Maximum gain/loss of the broadcast polling

The expected value of $Z^{(i)}$ represents the probability of the *success* in a slot, which happens *iff* one of N users in a group choses this slot for the request transmission, yielding:

$$T^1_{BEB} = E[Z^{(i)}] = Pr\{Z^{(i)} = 1\} = \frac{yN}{L}(1 - \frac{y}{L})^{N-1}. \tag{10}$$

We notice that the above closed-form expression gives exactly the same result as (6). Additionally, by calculating the first derivative of (10) for y and imposing it equal to 0, we establish the 'optimal' value of the input rate y that results in the maximum throughput value as:

$$y_0 = \frac{L}{N}. \tag{11}$$

The demonstrated approach allows the derivation of a closed-form expression for the maximum broadcast polling gain/loss function (depicted in Fig. 3) as follows:

$$f(y) = \frac{yM}{K}(1 - \frac{y}{K})^{M-1} - \frac{yN}{L}(1 - \frac{y}{L})^{N-1}. \tag{12}$$

5.3 Lossless System

We continue the performance evaluation of the system with finite number of users M under the assumptions given in Section 3. To get a deeper insight into the limitations of the BEB algorithm operation we set the maximum number of request retransmissions Q infinite. This way a request is never discarded and no data packet losses are possible in the system. We are interested in obtaining the rate of the BEB algorithm in the finite population lossless system R_{BEB}.

We introduce the stochastic process $c(s)$ that represents the value of the randomly sampled backoff counter at time s given that the number of collisions suffered by a request so far is $b(s)$. A discrete and integer time scale is also adopted, where s and $s+1$ correspond to the start times of two successive slots. We demonstrate our approach for broadcast polling as the example. All the below derivations may be generalized for the case of multicast polling with N users per group.

We notice, that according to the BEB rules described in Section 5.1 a user after its (re)transmission attempt does not start the backoff process immediately, but rather waits for the beginning of the next frame. Assume, that the (re)transmission attempt occurs in slot s in the frame that consists of K slots. Therefore, the user waits $K - s$ slots before resuming the backoff procedure. At its every retransmission attempt a user may be regarded as choosing the frame to retransmit in first and then choosing one of K slots in this frame. Thus, the number of slots before the (re)transmission in a frame is sampled uniformly in the range $[0, \ldots, K-1]$. Denote the *waiting time counter* as $a(s)$, which accounts for the slots after the (re)transmission attempt by a user and before the start of the next frame.

The considered stochastic process represents a Markov chain analogous to one described in [11] and [21], but with the addition of $K-1$ idle states, which correspond to the possible waiting time counter values. The transition probabilities for these additional states may be computed as follows:

$$\begin{aligned} \Pr\{a(s+1) = k-1 \,|a(s) = k\} &= 1, k = 1, \ldots, K-1, \\ \Pr\{a(s+1) = k \,|b(s) = 0\} &= \frac{1}{K}, k = 1, \ldots, K-1. \end{aligned} \tag{13}$$

Let $b_{i,j} = \lim_{s\to\infty} \Pr\{b(s) = j, c(s) = i\}$, $a_k = \lim_{s\to\infty} \Pr\{a(s) = k\}$, where $i = \{0, \ldots, m\}$, $j = \{0, \ldots, 2^i W - 1\}$ and $k = 1, \ldots, K-1$ is the stationary

distribution of the considered Markov chain. As the probability of a (re) transmission attempt in a slot is equal to $\sum_{i=0}^{m} b_{i,0}$, we establish:

$$a_k = \frac{k}{K-k}\sum_{i=0}^{m} b_{i,0} \Rightarrow \sum_{k=1}^{K-1} a_k = \frac{K-1}{2}\sum_{i=0}^{m} b_{i,0} = \frac{K-1}{2}\cdot\frac{b_{0,0}}{1-p_c}, \tag{14}$$

where p_c is the conditional collision probability, which is equal to the probability that at least one of the remaining $M-1$ users (re)transmits:

$$p_c = 1-(1-p_t)^{M-1}. \tag{15}$$

Accounting for the normalization condition:

$$1 = \sum_{i=0}^{m}\sum_{j=1}^{2^i W} b_{i,j} + \sum_{k=1}^{K} a_k, \tag{16}$$

we notice that the first term is given in [11]. Summarizing, the probability p_t that a user (re)transmits in a randomly chosen slot is readily obtained as:

$$p_t = \sum_{i=0}^{m} b_{i,0} = \frac{2\,(1-2p_c)}{(1-2p_c)\,(W+K) + p_c W\,(1-(2p_c)^m)}. \tag{17}$$

Equations (15) and (17) represent a nonlinear system with two unknowns p_c and p_t, which may be solved numerically. The resulting R_{BEB} value is finally given by the probability of one (re)transmission in a slot:

$$R_{BEB} = Mp_t(1-p_t)^{M-1}. \tag{18}$$

The above approach allows the derivation of the optimal (re)transmission probability value that gives the maximum BEB algorithm rate over all possible pairs of (W, m). It may be shown that this maximum value is reached for $m = 0$. Below we consider the optimal system in more detail.

Substituting $m = 0$ into (17) we obtain that $p_t = \frac{2}{W_0+K}$, where W_0 is the optimal initial contention window value. Notice that (18) closely resembles the expression (10), which is maximized for $\frac{yN}{L} = 1$. Therefore, the expression (18) itself is maximized for $Mp_t = \frac{2M}{W_0+K} = 1$. Finally, W_0 is obtained as $2M-K$, or, accounting for the possible grouping of users:

$$W_0 = 2N - L. \tag{19}$$

It should be emphasized, that the rate of the optimized BEB algorithm with $m = 0$ and W_0 gives precisely the same value as calculated by (10) for the lossy system. However, the usage of the optimal initial contention window W_0 in IEEE 802.16 standard is not straightforward, as it may not be an integer power of two. For this reason we depict the BEB rate for various values of m and different initial contention windows in Fig. 4. We see, that for the example system with $M = 40$, $K = 8$ and broadcast polling, $W_0 = 72$. The BEB rate given by $W = 32$ and $m = 2$ is almost as high as the optimal one. Summarizing, our approach allows the optimization of BEB parameters in terms of the highest achievable rate.

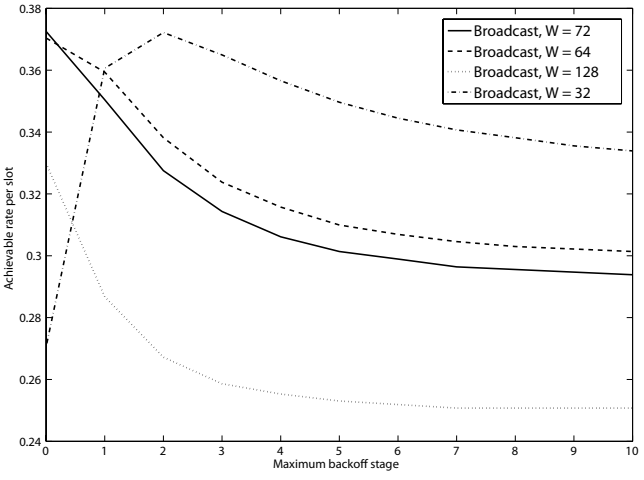


Fig. 4. Broadcast polling efficiency in case of infinite retransmission attempts

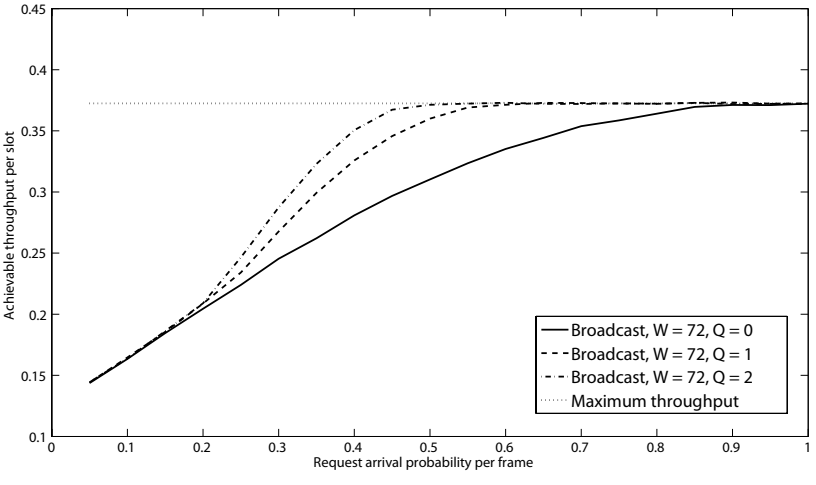


Fig. 5. Broadcast polling lossy throughput performance

5.4 Numerical Results

Here we provide some simulation results that are used to make final conclusions on broadcast and multicast polling efficiency. In Fig. 5 we demonstrate the throughput of the system, where the maximum number of retransmission attempts is set to some natural number, that is, $Q \geq 1$. Therefore, this system represents the intermediate case between those discussed in Section 5.2 and Section 5.3.

We see that for all the values of Q the throughput converges to the value indicated by (18) and (10). However, the convergence is faster for the greater Q value as less requests get discarded. An important conclusion from this is that regardless of the considered system (lossy or lossless) the performance measure of the BEB algorithm is unchangeable, i.e. $T^1_{BEB} = T^{Q+1}_{BEB} \triangleq T_{BEB}$ and $T_{BEB} = R_{BEB}$.

6 Conclusion

In this paper we firstly considered user grouping for the infinite population model and showed that the RMA system capacity remains unchanged. Additionally, we introduced the lossy and lossless system types for which the performance of the BEB algorithm was investigated. Using various analytical techniques, it was demonstrated that the BEB throughput in the lossy system coincides its rate in the lossless one. An important optimization of the BEB parameters was shown and its application in IEEE 802.16 standard was discussed.

The conclusion we make is that multicast polling gain over broadcast polling for the practical scenarios with symmetric grouping, where the groups have equal size and the BEB parameters are the same for each group is minor and decreases as the user population grows. However, to support the QoS requirements, another grouping may be applied, with different BEB parameters and/or unequal group sizes. The contention-based polling performance for these scenarios is subject to a separate investigation, but the demonstrated approaches remain, nevertheless, applicable.

References

1. IEEE Std 802.16e-2005, Piscataway, NJ, USA (December 2005)
2. Rubin, I.: Access-control disciplines for multi-access communication channels: Reservation and tdma schemes. IEEE Transactions on Information Theory 25(5), 516–536 (1979)
3. Aldous, D.: Ultimate instability of exponential back-off protocol for acknowledgment based transmission control of random access communication channels. IEEE Transactions on Information Theory 33(2), 219–233 (1987)
4. Goodman, J., Greenberg, A., Madras, N., March, P.: Stability of binary exponential backoff. Journal of the ACM 35(3), 579–602 (1988)
5. Paterakis, M., Georgiadis, L., Papantoni-Kazakos, P.: On the relation between the finite and the infinite population models for a class of raa's. IEEE Transactions on Communications 35, 1239–1240 (1987)
6. Chlebus, B.: Randomized Communication in Radio Networks. In: Pardalos, P., Rajasekaran, S., Reif, J., Rolim, J.(eds.), Handbook of Randomized Computing, vol. 1, pp. 401–456 (2001)
7. Tsybakov, B.: Survey of ussr contributions to random multiple-access communications. IEEE Transactions on Information Theory 31(2), 143–165 (1985)
8. Tsybakov, B., Mikhailov, V.: Free synchronous packet access in a broadcast channel with feedback. Problems of Information Transmission 14(4), 259–280 (1978)

9. Bertsekas, D., Gallager, R.: Data Networks. Prentice-Hall, Englewood Cliffs (1992)
10. Song, N., Kwak, B., Miller, L.: On the stability of exponential backoff. Journal Research of NIST 108, 289–297 (2003)
11. Bianchi, G.: Performance analysis of the ieee 802.11 distributed coordination function. IEEE Journal on Selected Areas in Communications 18(3), 535–547 (2000)
12. Lin, L., Jia, W., Lu, W.: Performance analysis of ieee 802.16 multicast and broadcast polling based bandwidth request. In: IEEE Wireless Communications and Networking Conference, vol. 1, pp. 1854–1859 (2007)
13. Alanen, O.: Multicast polling and efficient voip connections in ieee 802.16 networks. In: 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems, vol. 1, pp. 289–295 (2007)
14. Tsybakov, B., Likhanov, N.: Upper bound on the capacity of a random multiple-access system. Problems of Information Transmission 23(3), 224–236 (1987)
15. Turlikov, A., Vinel, A.: Capacity estimation of centralized reservation-based random multiple-access system. In: Symposium on Problems of Redundancy in Information and Control Systems, vol. 1, pp. 154–160 (2007)
16. Tsybakov, B.: One stochastic process and its application to multiple access in supercritical region. IEEE Transactions on Information Theory 47(4), 1561–1569 (2001)
17. Tsybakov, B., Mikhailov, V.: Random multiple packet access: Part-and-try algorithm. Problems of Information Transmission 16(4), 305–317 (1980)
18. Tsybakov, B., Berkovskii, M.: Multiple access with reservation. Problems of Information Transmission 16(1), 35–54 (1980)
19. Vinel, A., Zhang, Y., Ni, Q., Lyakhov, A.: Efficient request mechanisms usage in ieee 802.16. In: IEEE Global Telecommunications Conference, vol. 1, pp. 1–5 (2006)
20. Abramson, N.: The throughput of packet broadcasting channels. IEEE Transactions on Communications 25(1), 117–128 (1977)
21. Vinel, A., Zhang, Y., Lott, M., Tiurlikov, A.: Performance analysis of the random access in ieee 802.16. In: IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications, vol. 3, pp. 1596–1600 (2005)

Performance Evaluation of the High Speed Downlink Packet Access in Communications Networks Based on High Altitude Platforms

Tien Van Do[1], Nam H. Do[1], and Ram Chakka[2]

[1] Department of Telecommunications,
Budapest University of Technology and Economics
H-1117, Magyar tudósok körútja 2., Budapest, Hungary
do@hit.bme.hu
[2] Meerut Institute of Engineering and Technology
Meerut, India
ramchakka@yahoo.com

Abstract. In this paper, we provide a performance model and viable evaluation for the Adaptive Modulation and Coding (AMC) used in High Altitude Platforms (HAP) based HSDPA/UMTS telecommunication networks. We incorporate the HAP channel model into a new queueing model which is able to capture most of the features of wireless communications, such as traffic-burstiness, channel fading, channel allocation policy, etc., in an integrated way. In a case study we validate the model with the use of ns-2 simulator and also present numerical results to evaluate the impact of the HAP environment on the performance of HSDPA user terminal categories.

1 Introduction

High Altitude Platforms are defined as airships or aircrafts, which are designed to autonomously fly in the stratosphere at altitudes normally between 17 and 22 km. They can be utilized to provide wireless communications infrastructure. The International Telecommunications Union (ITU) has allocated two bands of mm-wave frequencies (47-48 GHz and 28-31 GHz) for Broadband Fixed Wireless Access (BFWA) services from HAPs.

High Altitude Platforms based communications are very important and highly complementary to satellite systems, current wireless systems and terrestrial networks in the fully converged IP based networks. There have been proposals and attempts (c.f.: the European Union supported projects such as HELINET, CAPANINA[1] and COST 297[2]) to define HAP telecommunications network architectures and protocols [1,2,3]. In these projects, the HAP-based telecommunication architecture is proposed in a such way that it can be harmonized and coexisted with the existing solutions and the evolution of wireless and core network concepts.

[1] http://www.capanina.org
[2] http://www.hapcos.org

K. Al-Begain, A. Heindl, and M. Telek (Eds.): ASMTA 2008, LNCS 5055, pp. 310–322, 2008.

An unprecedented success of the second-generation wireless systems has already been witnessed. Great efforts have been spent by the research community and the standards organisations (ETSI, ITU-T, 3GPP, EU ACTS and IST projects) for the definition of a radio interface and network architectures for the third-generation (UMTS) wireless networks in order to support a wide range of services from voice and low-rate data up to high-rate data services, including multimedia services, and circuit- and packet-oriented services to anyone, anytime, anywhere. The foreseen evolution of wireless and core network concepts would integrate and reuse the network elements of the second-generation wireless networks in order to reduce the cost to the users. Moreover, UMTS as a representative of International Mobile Telecommunications-2000 (IMT-2000 family), is offering wide/broadband services to mobile users. The International Telecommunications Union (ITU) has endorsed the use of High Altitude Platforms (HAPs) to carry base stations for the provision of UMTS wireless services. From the Radio Frequency (RF) perspective, HAP-based communications have been assigned the same spectrum as that of the terrestrial UMTS [4].

High Speed Downlink Packet Access (HSDPA) was introduced by the 3rd Generation Partnership Project (3GPP) to satisfy the demands for high speed data transfer in the downlink direction in UMTS networks. HSDPA is based on the AMC (Adaptive Modulation and Coding) scheme, extensive multicode operation and a retransmission strategy, which aims at an efficient exploitation of the wireless interface. The AMC is motivated by the fact (or assumption) that the stochastic behaviour of the wireless channel cannot be influenced. The technology should adapt to the behavior of the wireless channel in order to get an efficient transport of useful information. Note that there exist previous works to evaluate the performance of AMC in HAP-based networks, but that evaluation was performed using discrete event simulation [2].

In [5], we have provided the first analytical model (a generalized Markovian queue with varying number of servers) for HSDPA terminal in the terrestrial UMTS wireless networks. The analytical model was suitable to the problem that was tackled therein, since (i) traffic correlations and burstiness can be represented by Markov modulation and by the use of Compound Poisson Processes (CPP), (ii) channel conditioning due to fading and the resulting CQI can be represented by a finite-state first-order Markov chain Z, (iii) dynamic channel allocation policy is represented by varying c servers in the queuing model, modulated by an independent Markov process U.

The contributions of this paper include (a) the application of the model to the current problem (i.e.: performance of HSDPA in the HAP environment and investigation of some interesting phenomena related to the HAP environment), (b) the validation of our model with a detailed simulation of the system using *real traffic traces* and fading behaviour, (c) in this respect we also show that the Compound Poisson Processes (CPP) and the simple parameter estimation of the CPP from real traffic trace can serve as input parameters for the performance estimation of HSDPA.

The rest of the paper is organized as follows. In Section 2, an overview of HSDPA operation is given. In Section 3 a new analytical model is proposed in this context. Validation and numerical results are presented in Section 4. Finally, Section 5 concludes the paper.

2 HAP UMTS/HSDPA Operation

The use of HAP for UMTS is straightforward if the the HAP platforms carry and perform the Node-B and/or RNC functionality on board as shown in Figure 1.

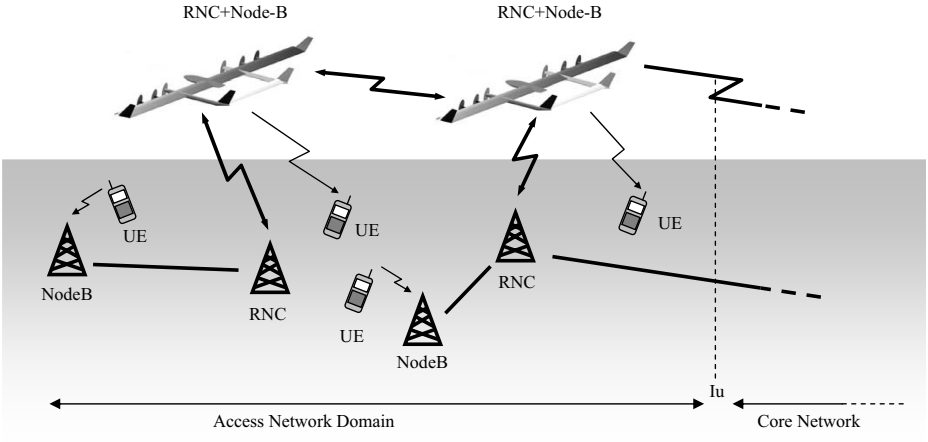


Fig. 1. RNC and Node-B on board, in HAP-based UMTS networks

In the implementation of HSDPA, several channels are introduced. The transport channel carrying the user data, in HSDPA operation, is called the High Speed Downlink Shared Channel (HS-DSCH). The High Speed Shared Control Channel (HS-SCCH), used as the downlink (DL) signaling channel, carries key physical layer control information to support the demodulation of the data on the HS-DSCH.

The uplink (UL) signaling channel, called the High Speed Dedicated Physical Control Channel (HS-DPCCH), conveys the necessary control data in the UL. User Equipment (UE) sends feedback information about the received signal[3] quality on HS-DPCCH. That is, the UE calculates the DL Channel Quality Indicator (CQI) based on the received signal quality measured at the UE. Then, it sends the CQI on the HS-DPCCH channel to indicate which estimated transport block size, modulation type and number of parallel codes (i.e.; physical channels) could be received correctly with a reasonable (less than the maximum permissible) block error rate in the DL. The CQI is integer valued, with a range between 0 and 30 (cf. [6,7]). The higher the CQI is, the better the condition of the channel and more information can be transmitted.

To enable a large dynamic range of the HSDPA link adaptations and to maintain a good spectral efficiency, a user may simultaneously utilize up to 15 codes of SF 16 (physical channels) in parallel. The available code resources are primarily shared in the time domain but it is possible to share the code resources using code multiplexing

[3] In wireless communications, the quality of a received signal depends on a number of factors – the distance between the target and interfering base stations, the path-loss exponent, shadowing, channel-fading and noise.

as well. The relationship between modulation, the number of allocated codes and the maximum throughput is illustrated in Table 1.

Fast scheduling and link adaptation are then performed promptly, depending on the active scheduling algorithm and the user-prioritisation scheme employed by the base station (Node B). In HSDPA, the AMC dynamically changes the Modulation and Coding Schemes (MCS) in subsequent frames, in order to achieve high throughput on fading channels.

Table 1. Modulation and max throughput when 5,10,15 codes are allocated for a specific user

Modulation	Effective code rate	Max. throughput Mbps		
		5 codes	10 codes	15 codes
QPSK	1/4	0.6	1.2	1.8
QPSK	2/4	1.2	2.4	3.6
QPSK	3/4	1.8	3.6	5.4
16 QAM	2/4	2.4	4.8	7.2
16 QAM	3/4	3.6	7.2	10.7

3 Performance Evaluation of HSDPA in the HAP Environment

3.1 An Integrated Analytical Performance Model

We consider the wireless connection with an ideal feedback channel between any specified wireless user and its Node-B located in HAP. We assume the UE categories specified in [7] are to be used in the HAP environment. The features of the proposed queueing model are as follows:

- The packet arrival process is a MMCPP (Markov Modulated Compound Poisson process) process (at the transport block level). The arrival process is thus inherently modulated by a continuous time, irreducible Markov process, X, with N_X states (i.e. phases of modulation). Such a traffic model can accommodate both inter- arrival correlations and traffic-burstiness. Let Q_X be the generator matrix of X. In the modulating phase i, the parameters of the Generalized Exponential (GE)[4] inter-arrival time [8] distribution of the packet arrival stream are (σ_i, θ_i). That is, the inter- arrival time probability distribution function is $1-(1-\theta_i)e^{-\sigma_i t}$, in phase i. Thus, the arrival *point*-process can be seen as batch-Poisson, with batches arriving at each point having geometric size distribution. Specifically, the probability that a batch is of size s is $(1-\theta_i)\theta_i^{s-1}$, in phase i.
- Servers in the queueing model correspond to multicodes allocated for the specific UE that is under consideration, in this study. In HSDPA a user may simultaneously utilize up to 15 codes (physical channels) in parallel. The available code resources

[4] The GE [8,9] is the only distribution that is of least bias, if only the mean and variance are reliably computed from the measurement data. It will be shown that the GE is accurate enough to model Internet traffic (i.e.: GE parameters are estimated from the captured Internet traffic) and to be used for the performance evaluation in telecommunication systems.

are primarily shared in the time domain but it is possible to share the code resources using code multiplexing too. The number of codes allocated to a specific UE depends on the channel condition (signalled by the received CQI value) and also the applied scheduling (channel allocation) algorithm. The number of available servers is thus *varying*. This is represented as the number of servers being modulated jointly by the Markov chains Z and U. U is the Markov chain that represents the channel allocation for the specified user. N_Z and N_U are the number of states, and Q_Z and Q_U are the generator matrices of Z and U, respectively.

– L is the buffer size at Node-B which can be finite or infinite.

The entire system is thus modulated by a single Markov process which is the result of the combination (resultant generator matrix actually obtained by the Kronecker sum of the individual generator matrices) of the three independent Markov modulations: (i) the modulating process (X) of the arrival traffic, (ii) the Markov process (Z) characterizing the fading channel behavior, and (iii) the Markov process (U) representing the channel allocation policy. The entire system now can be modeled by a special variant of the MM $\sum_{k=1}^{K} CPP_k$/GE/c/L G-queue presented in [10]. This variant has the property of varying number of servers. The steady state solution of the system can be obtained using the methodology in [10].

3.2 A HAP Channel Model

Due to operating environment, wireless signals from HAP are affected by environments such as rain attenuation, scattering, etc. In order to characterize the channel behavior in the HAP environment we proceed as follows.

The instantaneous signal-to-noise ratio (SNR) is defined as $\Gamma = \frac{R^2}{BW \times N_0}$, where BW is the bandwidth, N_0 is the one-sided power spectral density of the noise and R denotes the fading amplitude. Without the loss of generality, $BW \times N_0 = 1$ is assumed, then the SNR now is equal to the power of the signal with its average value is $\bar{\Gamma} = E[R^2]$ (cf. [11]). Then the relation between the probability density function of Γ and $f_R(r)$ – the probability density function of R– is given as follows[5]

$$f_\Gamma(\gamma) = \frac{f_R(\sqrt{\gamma})}{2\sqrt{\gamma}}. \tag{2}$$

Following Loo's model [12], the amplitude of the received signal is obtained from the sum of the line-of-sight (LOS) component and the multipath component. The LOS path component follows the lognormal distribution with mean α (in dB relative to LOS) and standard devition Ψ (dB), while the multipath component is characterized by Rayleigh fading with its average power MP (in dB relative to LOS). Therefore, its probability density function is given by [12]:

[5] Ψ is a random variable with the probability density function $f_\Psi(x)$. Define random variable $\Upsilon = g(\Psi)$, where g is an invertible function (i.e: $\Psi = h(\Upsilon)$). Then the probability density function of Υ is

$$f_\Upsilon(y) = h'(y) f_\Psi(h(y)) \tag{1}$$

$$f_R(r) = \frac{r}{b_0\sqrt{2\pi d_0}}\int_0^\infty \frac{1}{z}\exp\left[-\frac{(\ln z-\mu)^2}{2d_0}-\frac{r^2+z^2}{2b_0}\right]I_0\left(\frac{rz}{b_0}\right)dz \quad (3)$$

where μ, d_0 and b_0 are related to α, Ψ and MP as:

$$\alpha = 20\log_{10}(e^{\mu}), \quad \Psi = 20\log_{10}(e^{\sqrt{d_0}}), \quad MP = 10\log_{10}(2b_0) \quad (4)$$

Using (2), the pdf (probability density function) of the SNR is obtained as:

$$f_\Gamma(\gamma) = \frac{1}{2b_0\sqrt{2\pi d_0}}\int_0^\infty \frac{1}{z}\exp\left[-\frac{(\ln z-\mu)^2}{2d_0}-\frac{\gamma+z^2}{2b_0}\right]I_0\left(\frac{\sqrt{\gamma}z}{b_0}\right)dz \quad (5)$$

The Level Crossing Rate (LCR) is defined as the expected rate at which the envelope r crosses a specific level R with positive slope ([13]):

$$\aleph_R(r_{th}) = \int_0^\infty \dot{r}g(r_{th},\dot{r})d\dot{r} \quad (6)$$

where $\dot{r}$ is the time derivative of the signal envelope and $g(r_{th},\dot{r})$ is the joint pdf of the envelope and its derivative $\dot{r}$ computed when r equals the threshold level r_{th}. It is straightforward to show that $\aleph_\Gamma(r_{th}^2) = \aleph_R(r_{th})$.

For the slowly varying LOS, the LCR for Loo's model is derived in [13] using the pdf-based approach:

$$\begin{aligned}\aleph_\Gamma(\gamma_{th}) = {} & \frac{\sqrt{\beta\gamma_{th}}}{\pi^2 b_0\sqrt{d_0}}\exp\left(-\frac{\gamma_{th}}{2b_0}\right)\int_0^\infty \frac{1}{z}\exp\left[-\frac{(\ln z-\mu)^2}{2d_0}\right] \\ & * \exp\left(-\frac{z^2}{2b_0}\right)\int_0^{\pi/2}\cosh\left(\frac{\sqrt{\gamma_{th}}z\cos x}{b_0}\right)\Big\{\exp\left[-(\varepsilon z\sin x)^2\right] \\ & + \sqrt{\pi}\varepsilon z\sin(x)\mathrm{erf}(\varepsilon z\sin x)\Big\}dxdz \quad (7)\end{aligned}$$

where $\cosh(.)$ is the hyperbolic cosine, erf(.) is the error function given by

$$\mathrm{erf}(y) = \frac{2}{\sqrt{\pi}}\int_0^y \exp(-x^2)dx.$$

Others parameters are derived from the material in [13] with some simplifications:

$$\beta = 2b_0 f_d^2(\pi^2-8)$$
$$\varepsilon = \frac{2}{\sqrt{b_0(\pi^2-8)}}$$

where f_d is the mobility-induced Doppler spread.

The popular model for HAP communications divides the channel behavior (between the HAP and ground terminal) into three states [14]. The channel is in the LOS state when the ground terminal and HAP are in the LOS condition. The second state represents the lightly shadowed channel condition, while the third state represents the channel condition of deep shadow. The fading amplitude of the received signal in each state

follows the Loo distribution with different $\{\alpha, \Psi, MP\}$ triplets. Then the instantaneous SNR of the channel can be modeled as independent random variables A, B, C for the three states, respectively. Denote P_A, P_B and P_C as the occurrence probabilities of LOS (A), shadowed (B) and blockage (C) states, respectively. Then the mixed pdf (probability density function) of the SNR can be given as:

$$f_{mix}(\gamma) = P_A f_A(\gamma) + P_B f_B(\gamma) + P_C f_C(\gamma) \tag{8}$$

and the mixed LCR can be obtained as:

$$\aleph_{mix}(\gamma_{th}) = P_A \aleph_A(\gamma_{th}) + P_B \aleph_B(\gamma_{th}) + P_C \aleph_C(\gamma_{th}) \tag{9}$$

Since the CQI integer value sent by UE varies between 0 and 30, we partition the range of the SNR into N_Z = 31 intervals and use a continuous time first-order Markov chain Z of N_Z states to characterise the fading channel (that is, the SNR in state i (denoted by S_i) is associated with $\gamma \in [\gamma_i, \gamma_{i+1})$. Note that $\gamma_1 = 0, \gamma_{N_Z+1} = \infty$ dynamics. Each interval corresponds to a CQI value reported by a specific UE to its Node-B. The CQI corresponding to the fading channel state S_i is $i-1$, for $i = 1, 2, \ldots, N_Z$. These N_Z intervals (partitions) are determined based on the equation between CQI and SNR [15]:

$$CQI = \begin{cases} 0, & \text{SNR} \leq -16 \\ \lfloor \frac{SNR}{1.02} + 16.62 \rfloor, & -16 \leq \text{SNR} \leq 14 \\ 30, & \text{SNR} \geq 14 \end{cases} \tag{10}$$

The elements of the generator matrix, Q_Z, can be determined as follows

$$\begin{aligned} Q_Z(k, k+1) &= \aleph_{k+1}/\pi_k \quad (k = 1, 2, \ldots, 30) \\ Q_Z(k, k-1) &= \aleph_k/\pi_k \quad (k = 2, 3, \ldots, 31) \end{aligned} \tag{11}$$

The level crossing rate ($\aleph_n$) of mode n (the AMC mode n is chosen when the channel is in state S_n) is obtained using equation (9) as $\aleph_n = \aleph_{mix}(\gamma_n)$, n=1, 2,..., 31, and

$$\pi_k = \int_{\gamma_k}^{\gamma_{k+1}} f_{mix}(\gamma)\, d\gamma. \tag{12}$$

4 Numerical Study

We assume that data is transferred from the network to a specified UE (with $N_U = 1$ assumed), which means the number of allocated channels to a UE depends only on the channel state (i.e. the state of process Z). The queuing capacity at the Node-B of this specified UE is assumed to be 150. Traffic is assumed to follow the GE distribution with parameter pair $(\sigma = 151.360101(1/sec), \theta = 0.544786)$, which are estimated from the Auckland traffic trace [16], based on the method of moment matching. It is worth emphasizing that the service parameter for each UE category is determined based on the statistical data of the length of data packets from the trace and the transfer data unit of

each UE specified in [6,7]. The channel parameters in Table 4.1 for the study are based on the result reported in [17], which have been calculated from part of the ESA/ESTEC (European Space Agency) LMS measurement database. Thus, we can calculate the necessary parameters for our analytical evaluation. From Table 4.1 and the traveling speed of UEs (which have impact on the Doppler spread), we can have the summary of the statistics of the reported CQI values from UE in Table 3.

4.1 Validation with Detailed Simulation

To validate our model, we compare the results obtained by the simulation with those obtained by our analytical model. The starting point of our investigation is to modify the ns-2 based EURANE simulator (http://www.ti-wmc.nl/eurane/), which was designed to simulate HSDPA. The first modification in the code was performed to feed the EURANE simulator with the Auckland traffic trace. The second modification was done in order to incorporate the HAP channel model presented in Section 3.2. In Figure 2 we present results related to UE category 10 (similar results are obtained for other categories). It can be observed that our model can provide a good estimate for the performance of the UE categories.

Table 2. Model parameters for different elevations and different channel condition (A: LOS, B: intermediate shadow and C: Deep shadow)

Elevation		Urban, hand-held antenna				Suburban, car roofted antenna			
		α (dB)	Ψ (dB)	MP (dB)	Occurrence prob.	α (dB)	Ψ (dB)	MP (dB)	Occurrence prob.
10	A	-0.7	1.9	-38.3	0.0496	-0.1	0.5	-19.0	0.4389
	B	-18.4	8.6	-14.7	0.1397	-8.7	3.0	-12.0	0.2599
	C	-24.4	9.4	-23.9	0.8107	-12.1	6.0	-25.0	0.3012
20	A	0.7	2.1	-25.5	0.0018	0.0	1.5	-25.0	0.6666
	B	-10.0	4.9	-23.3	0.1058	-6.3	3.5	-20.0	0.1609
	C	-25.3	7.9	-26.5	0.8924	-9.0	5.0	-21.0	0.1725
30	A	0.4	2.5	-34.0	0.1215	-0.5	1.0	-15.0	0.7467
	B	-11.5	5.4	-16.0	0.2726	-4.7	1.5	-19.0	0.1511
	C	-19.2	7.0	-22.0	0.6059	-7.0	3.0	-20.0	0.1022
40	A	-0.2	1.0	-32.9	0.0219	-0.3	1.5	-14.0	0.1626
	B	-8.6	3.8	-16.1	0.1403	-4.5	1.0	-21.0	0.7642
	C	-15.1	2.6	-16.0	0.8378	-7.1	2.0	-21.0	0.0732
50	A	0.0	0.5	-34.5	0.0602	-0.5	1.0	-17.0	0.0275
	B	-6.1	2.7	-17.0	0.2739	-6.5	2.5	-17.0	0.7611
	C	-13.0	4.3	-17.7	0.6660	-14	2.5	-20.0	0.2114
60	A	0.1	1.9	-27.2	0.0669	-1.0	1.0	-15.0	0.0634
	B	-6.9	2.2	-18.6	0.2863	-6.0	2.5	-17.0	0.5337
	C	-13.1	4.2	-19.7	0.6468	-10.2	4.0	-15.0	0.4029
70	A	-0.7	1.8	-25.1	0.0040	-0.2	0.5	-15.0	0.0000
	B	-5.7	1.0	-23.8	0.1680	-6.0	2.1	-17.0	0.8956
	C	-12.7	3.2	-20.2	0.8280	-11.5	2.0	-20.0	0.1044

Table 3. Average of CQIs vs elevation angles

Elevation (degree)	Average CQI	
	Suburban area 5km/h	Urban area 50 km/h
10	14.322561	4.548156
20	15.215819	2.879117
30	19.258761	7.625977
40	17.093137	8.51484
50	16.110073	11.339293
60	13.116639	11.060468
70	13.53116	9.932253

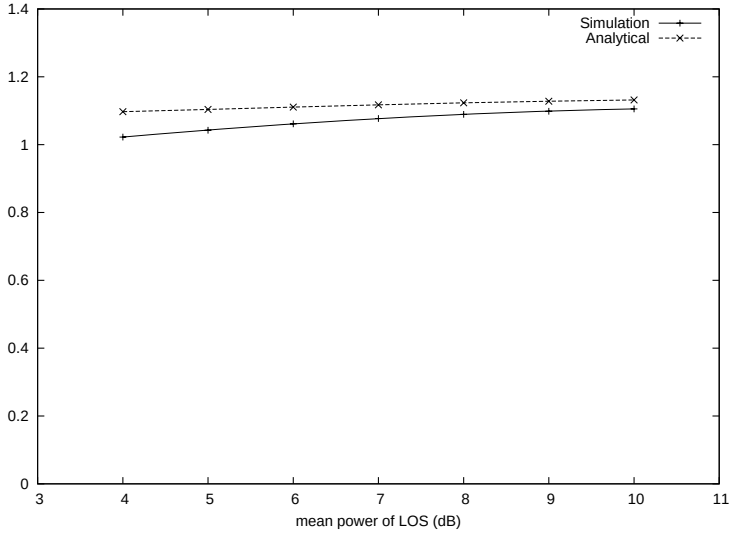


Fig. 2. Validation of analytical results with simulation (throughput in Mbps)

4.2 Impact of the Elevation of the HAP

In what follows the impact of the elevation of the platform on the throughput of the UEs is investigated. In Figure 3, we compare the throughput of different UE categories in a suburban environment with the elevation angle of 20^o, UE traveling speed of 50km/h is assumed and normalized LOS ranges from 4dB to 10dB. It is observed that higher UE categories cannot provide advantage for the subscribers in this environment.

Figure 4 plots the throughput of UE 10 versus the elevation angle for an urban area with traveling speed of 3km/h (pedestrian user) and a suburban area with traveling speed of 50km/h (subscribers traveling in a car). The behavior of the curves can be explained by the characteristic of the reported values of CQI (see Table 3). It worth emphasizing that the intention of Figure 4 is not to compare the throughput of UE traveling at different speeds because the measurement data concerning each traveling speed may not

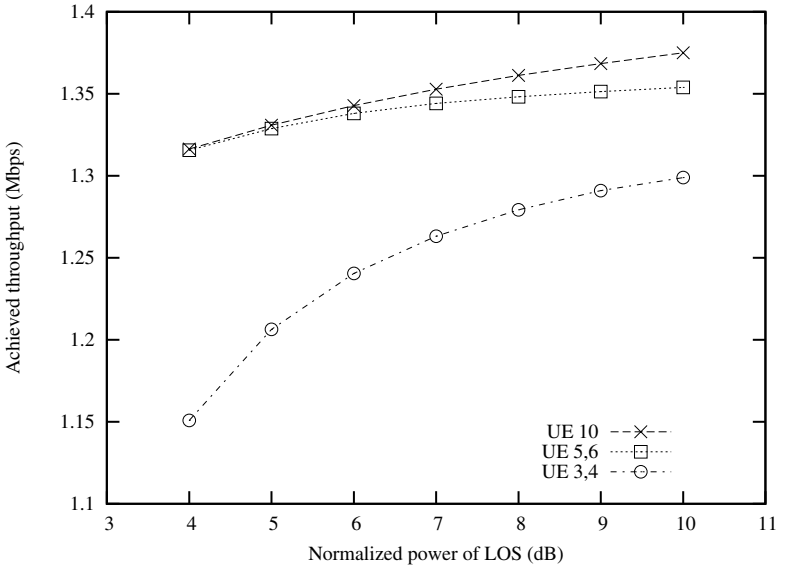


Fig. 3. Comparison of different UE categories

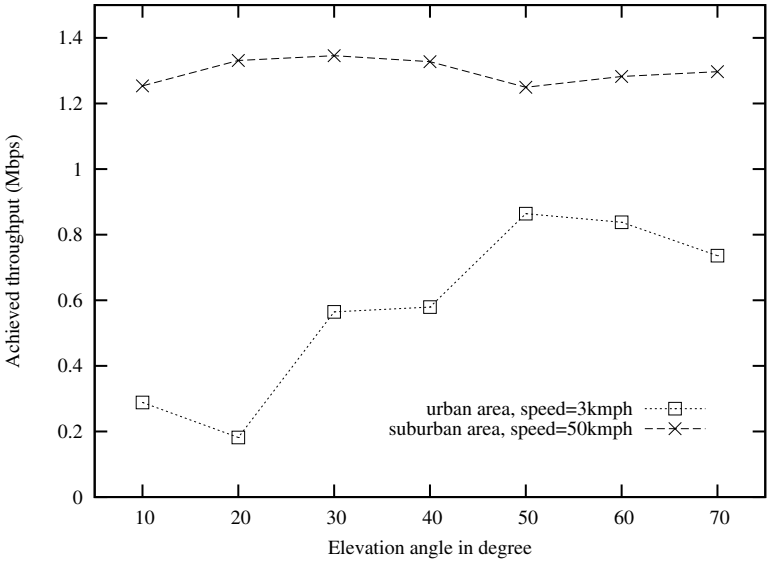


Fig. 4. Throughput versus elevation angles (SNR=5dB, UE 10)

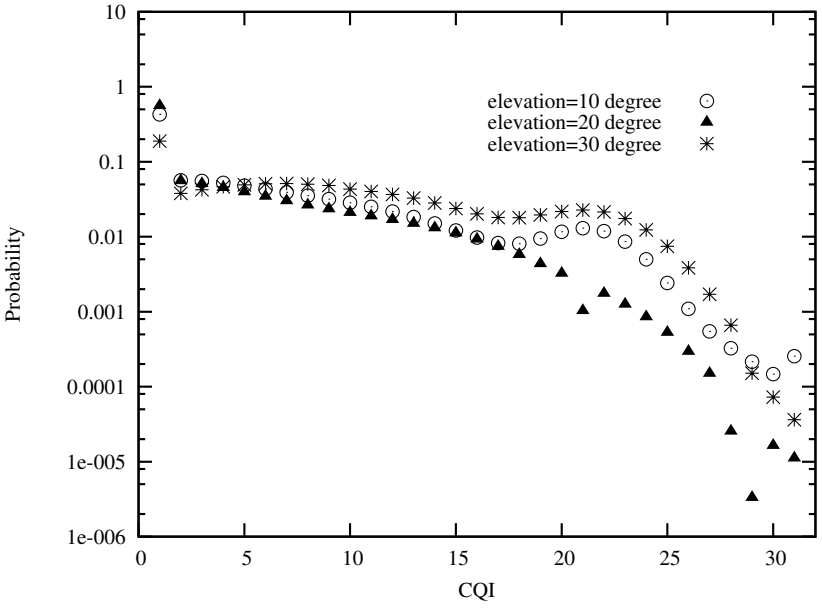


Fig. 5. Probability of CQI values

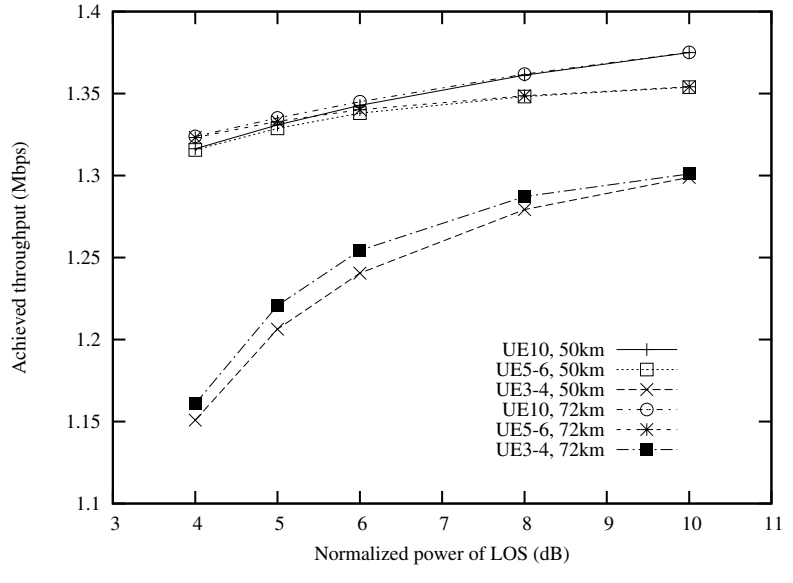


Fig. 6. Comparison of different speeds in suburban with elevation angle is 20^0

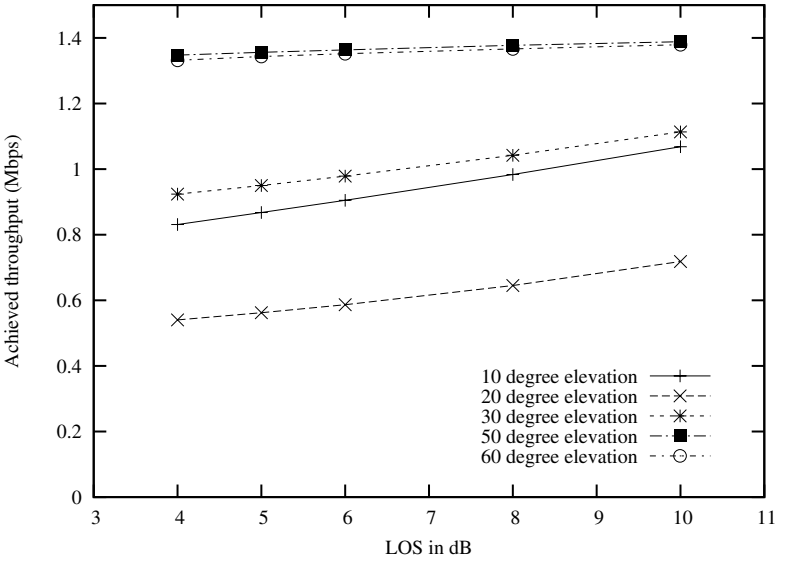


Fig. 7. Throughput via different elevation angles

reflect the same condition (c.f. [17]). It turns out that subscribers traveling in a car get approximately stable throughput performance. It is observed that the throughput of a pedestrian subscriber strongly depends on the elevation angle. For example, when the elevation angle changes from 10^o to 20^o, the performance decreases, which can be interpreted when we plot the probability of the CQI as in Figure 5. The good news for the pedestrian users is that their elevation angle is not changed rapidly due to their low traveling speed (3 kmph assumed).

Now, let us focus on the impact of traveling at higher speed (we fix the elevation angle to be at 20^o) in a suburban area. In Figure 6, we compare the the throughput of UE category 3-4, 5-6, and 10 for the traveling speeds of 50km/h and 72km/h. It can be observed that in low normalized power of LOS, the speed of the 72kmph case provided higher throughput. However, the increment of throughput curves in 50kmph is much faster, and those curves exceed the curves of the 72kmph case as expected, when normalized power of LOS increases.

The throughput with the traveling speed 50 km/h in the urban area is plotted in Figure 7. It can be observed that the curves pertaining to 20^o and 30^0 as well as the curves pertaining to 50^o and 60^o have the same form.

5 Conclusions

We have proposed an analytical model for the HSDPA operation in the High Altitude Platform (HAP) environment. The model is validated using the EURANE simulator fed by the Auckland traffic trace. It has been shown that in the HAP environment the performance experienced by pedestrian users does vary a lot with angle of elevation.

References

1. Batsios, N., Pavlidou, F.N.: Performance of CDMA/PRMA as an Access Technique for Integrated Services in a UMTS High Altitude Platform System. Wirel. Pers. Commun. 32, 319–338 (2005)
2. Javornik, T., Mohorčič, M., Švigelj, A., Ozimek, I., Kandus, G.: Adaptive Coding and Modulation for Mobile Wireless Access Via High Altitude Platforms. Wirel. Pers. Commun. 32, 301–317 (2005)
3. Tozer, T., Grace, D.: High-Altitude Platforms for Wireless Communications. IEE Electronics and Communications Engineering Journal, 127–137 (2001)
4. ITU-R Recommendation M-1456: Minimum performance characteristics and operational conditions for high altitude platform stations providing imt-2000 in the bands 1 885-1 980 Mhz, 2 010-2 025 Mhz and 2 110-2 170 Mhz in regions 1 and 3 and 1 885-1 980 Mhz and 2 110-2 160 Mhz in region 2 (2000)
5. Do, T.V., Chakka, R., Harrison, P.G.: An integrated analytical model for computation and comparison of the throughputs of the umts/hsdpa user equipment categories. In: MSWiM 2007: Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems, pp. 45–51. ACM, New York (2007)
6. 3GPP Technical Report 25.848, version 4.0.0: Physical layer aspects of UTRA High Speed Downlink Packet Access (2001)
7. 3GPP Technical Report 25.214, version 7.0.0: Physical layer procedures (FDD) (2006)
8. Kouvatsos, D.: Entropy maximisation and queueing network models. Annals of Operations Research 48, 63–126 (1994)
9. Skianis, C., Kouvatsos, D.: An Information Theoretic Approach for the Performance Evaluation of Multihop Wireless Ad Hoc Networks. In: Kouvatsos, D.D. (ed.) Proceedings of the Second International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs 2004), Ilkley, UK, pp. 81/1–13 (2004)
10. Chakka, R., Do, T.V.: The MM $\sum_{k=1}^{K} CPP_k/GE/c/L$ G-Queue with Heterogeneous Servers: Steady state solution and an application to performance evaluation. Performance Evaluation 64, 191–209 (2007)
11. Goldsmith, A.: Wireless Communications. Cambridge University Press, Cambridge (2006)
12. Loo, C.: A statistical model for a land mobile satellite link. IEEE Transactions on Vehicular Technology 34, 122–127 (1985)
13. Patzold, M., Li, Y., Laue, F.: A study of a land mobile satellite channel model with asymmetrical doppler power spectrum and lognormally distributed line-of-sight component. IEEE Transactions on Vehicular Technology 47, 297–310 (1998)
14. Pérez Fontán, F., Vázquez-Castro, M., Cabado, C.E., Garcia, J.P., Kubista, E.: Statistical modelling of the lms channel. IEEE Transactions on Vehicular Technology 50, 1549–1567 (2001)
15. Brouwer, F., de Bruin, I., Silva, J.C., Souto, N., Cercas, F., Correia, A.: Usage of Link-Level Performance Indicators for HSDPA Network-Level Simulations in E-UMTS. In: ISSSTA2004, Sydney, Australia (2004)
16. Auckland Internet Traffic Capture: `http://www.wand.net.nz/wand/wits/auck/6/20010612-060000-e1`
17. Pérez Fontán, F., Delgado-Penin, J.A., Palma-Lázgare, I.: A case study: WiMAX HAPS system at S-Band, COST 297 -HAPCOS- Action Document, COST297-0088-WG10-PUB-P01 (Technical report)

Author Index