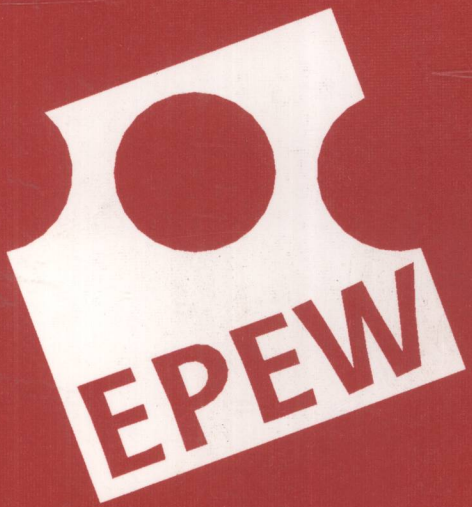
Katinka Wolter (Ed.)

LNCS 4748

Formal Methods and Stochastic Models for Performance Evaluation

Fourth European Performance Engineering Workshop, EPEW 20
Berlin, Germany, September 2007
Proceedings



Springer

Katinka Wolter (Ed.)

Formal Methods and Stochastic Models for Performance Evaluation

Fourth European Performance Engineering Workshop, EPEW 2007
Berlin, Germany, September 2007
Proceedings

Springer

Volume Editor

Katinka Wolter
Humboldt Universität Berlin
Institut für Informatik
Unter den Linden 6, 10099 Berlin, Germany
E-mail: wolter@informatik.hu-berlin.de

Library of Congress Control Number: 2007935135

CR Subject Classification (1998): D.2.4, C.2.4, F.3, D.4, C.4

LNCS Sublibrary: SL 2 – Programming and Software Engineering

ISSN 0302-9743
ISBN-10 3-540-75210-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-75210-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12164042 06/3180 5 4 3 2 1 0

Lecture Notes in Computer Science 4748

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

Editorial Board

Lecture Notes in Computer Science

Sublibrary 2: Programming and Software Engineering

For information about Vols. 1– 4063
please contact your bookseller or Springer

Vol. 4753: E. Duval, R. Klamma, M. Wolpers (Eds.), Creating New Learning Experiences on a Global Scale. XII, 518 pages. 2007.

Vol. 4749: B.J. Krämer, K.-J. Lin, P. Narasimhan (Eds.), Service-Oriented Computing – ICSOC 2007. XIX, 629 pages. 2007.

Vol. 4748: K. Wolter (Ed.), Formal Methods and Stochastic Models for Performance Evaluation. X, 301 pages. 2007.

Vol. 4741: C. Bessière (Ed.), Principles and Practice of Constraint Programming – CP 2007. XV, 890 pages. 2007.

Vol. 4680: F. Saglietti, N. Oster (Eds.), Computer Safety, Reliability, and Security. XV, 548 pages. 2007.

Vol. 4670: V. Dahl, I. Niemelä (Eds.), Logic Programming. XII, 470 pages. 2007.

Vol. 4634: H. Riis Nielson, G. Filé (Eds.), Static Analysis. XI, 469 pages. 2007.

Vol. 4615: R. de Lemos, C. Gacek, A. Romanovsky (Eds.), Architecting Dependable Systems IV. XIV, 435 pages. 2007.

Vol. 4610: B. Xiao, L.T. Yang, J. Ma, C. Muller-Schloer, Y. Hua (Eds.), Autonomic and Trusted Computing. XVIII, 571 pages. 2007.

Vol. 4609: E. Ernst (Ed.), ECOOP 2007 – Object-Oriented Programming. XIII, 625 pages. 2007.

Vol. 4608: H.W. Schmidt, I. Crnković, G.T. Heineman, J.A. Stafford (Eds.), Component-Based Software Engineering. XII, 283 pages. 2007.

Vol. 4591: J. Davies, J. Gibbons (Eds.), Integrated Formal Methods. IX, 660 pages. 2007.

Vol. 4589: J. Münch, P. Abrahamsson (Eds.), Product-Focused Software Process Improvement. XII, 414 pages. 2007.

Vol. 4574: J. Derrick, J. Vain (Eds.), Formal Techniques for Networked and Distributed Systems – FORTE 2007. XI, 375 pages. 2007.

Vol. 4556: C. Stephanidis (Ed.), Universal Access in Human-Computer Interaction, Part III. XXII, 1020 pages. 2007.

Vol. 4555: C. Stephanidis (Ed.), Universal Access in Human-Computer Interaction, Part II. XXII, 1066 pages. 2007.

Vol. 4554: C. Stephanidis (Ed.), Universal Acess in Human Computer Interaction, Part I. XXII, 1054 pages. 2007.

Vol. 4553: J.A. Jacko (Ed.), Human-Computer Interaction, Part IV. XXIV, 1225 pages. 2007.

Vol. 4552: J.A. Jacko (Ed.), Human-Computer Interaction, Part III. XXI, 1038 pages. 2007.

Vol. 4551: J.A. Jacko (Ed.), Human-Computer Interaction, Part II. XXIII, 1253 pages. 2007.

Vol. 4550: J.A. Jacko (Ed.), Human-Computer Interaction, Part I. XXIII, 1240 pages. 2007.

Vol. 4542: P. Sawyer, B. Paech, P. Heymans (Eds.), Requirements Engineering: Foundation for Software Quality. IX, 384 pages. 2007.

Vol. 4536: G. Concas, E. Damiani, M. Scotto, G. Succi (Eds.), Agile Processes in Software Engineering and Extreme Programming. XV, 276 pages. 2007.

Vol. 4530: D.H. Akehurst, R. Vogel, R.F. Paige (Eds.), Model Driven Architecture - Foundations and Applications. X, 219 pages. 2007.

Vol. 4523: Y.-H. Lee, H.-N. Kim, J. Kim, Y.W. Park, L.T. Yang, S.W. Kim (Eds.), Embedded Software and Systems. XIX, 829 pages. 2007.

Vol. 4498: N. Abdennahder, F. Kordon (Eds.), Reliable Software Technologies - Ada-Europe 2007. XII, 247 pages. 2007.

Vol. 4486: M. Bernardo, J. Hillston (Eds.), Formal Methods for Performance Evaluation. VII, 469 pages. 2007.

Vol. 4470: Q. Wang, D. Pfahl, D.M. Raffo (Eds.), Software Process Dynamics and Agility. XI, 346 pages. 2007.

Vol. 4468: M.M. Bonsangue, E.B. Johnsen (Eds.), Formal Methods for Open Object-Based Distributed Systems. X, 317 pages. 2007.

Vol. 4467: A.L. Murphy, J. Vitek (Eds.), Coordination Models and Languages. X, 325 pages. 2007.

Vol. 4454: Y. Gurevich, B. Meyer (Eds.), Tests and Proofs. IX, 217 pages. 2007.

Vol. 4444: T. Reps, M. Sagiv, J. Bauer (Eds.), Program Analysis and Compilation, Theory and Practice. X, 361 pages. 2007.

Vol. 4440: B. Liblit, Cooperative Bug Isolation. XV, 101 pages. 2007.

Vol. 4408: R. Choren, A. Garcia, H. Giese, H.-f. Leung, C. Lucena, A. Romanovsky (Eds.), Software Engineering for Multi-Agent Systems V. XII, 233 pages. 2007.

Vol. 4406: W. De Meuter (Ed.), Advances in Smalltalk. VII, 157 pages. 2007.

Vol. 4405: L. Padgham, F. Zambonelli (Eds.), Agent-Oriented Software Engineering VII. XII, 225 pages. 2007.

Vol. 4401: N. Guelfi, D. Buchs (Eds.), Rapid Integration of Software Engineering Techniques. IX, 177 pages. 2007.

Vol. 4385: K. Coninx, K. Luyten, K.A. Schneider (Eds.), Task Models and Diagrams for Users Interface Design. XI, 355 pages. 2007.

Vol. 4383: E. Bin, A. Ziv, S. Ur (Eds.), Hardware and Software, Verification and Testing. XII, 235 pages. 2007.

Vol. 4379: M. Südholt, C. Consel (Eds.), Object-Oriented Technology. VIII, 157 pages. 2007.

Vol. 4364: T. Kühne (Ed.), Models in Software Engineering. XI, 332 pages. 2007.

Vol. 4355: J. Julliand, O. Kouchnarenko (Eds.), B 2007: Formal Specification and Development in B. XIII, 293 pages. 2006.

Vol. 4354: M. Hanus (Ed.), Practical Aspects of Declarative Languages. X, 335 pages. 2006.

Vol. 4350: M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, C. Talcott, All About Maude - A High-Performance Logical Framework. XXII, 797 pages. 2007.

Vol. 4348: S. Tucker Taft, R.A. Duff, R.L. Brukardt, E. Plödereder, P. Leroy, Ada 2005 Reference Manual. XXII, 765 pages. 2006.

Vol. 4346: L. Brim, B. Haverkort, M. Leucker, J. van de Pol (Eds.), Formal Methods: Applications and Technology. X, 363 pages. 2007.

Vol. 4344: V. Gruhn, F. Oquendo (Eds.), Software Architecture. X, 245 pages. 2006.

Vol. 4340: R. Prodan, T. Fahringer, Grid Computing. XXIII, 317 pages. 2007.

Vol. 4336: V.R. Basili, D. Rombach, K. Schneider, B. Kitchenham, D. Pfahl, R.W. Selby (Eds.), Empirical Software Engineering Issues. XVII, 193 pages. 2007.

Vol. 4326: S. Göbel, R. Malkewitz, I. Iurgel (Eds.), Technologies for Interactive Digital Storytelling and Entertainment. X, 384 pages. 2006.

Vol. 4323: G. Doherty, A. Blandford (Eds.), Interactive Systems. XI, 269 pages. 2007.

Vol. 4322: F. Kordon, J. Sztipanovits (Eds.), Reliable Systems on Unreliable Networked Platforms. XIV, 317 pages. 2007.

Vol. 4309: P. Inverardi, M. Jazayeri (Eds.), Software Engineering Education in the Modern Age. VIII, 207 pages. 2006.

Vol. 4294: A. Dan, W. Lamersdorf (Eds.), Service-Oriented Computing – ICSOC 2006. XIX, 653 pages. 2006.

Vol. 4290: M. van Steen, M. Henning (Eds.), Middleware 2006. XIII, 425 pages. 2006.

Vol. 4279: N. Kobayashi (Ed.), Programming Languages and Systems. XI, 423 pages. 2006.

Vol. 4262: K. Havelund, M. Núñez, G. Roşu, B. Wolff (Eds.), Formal Approaches to Software Testing and Runtime Verification. VIII, 255 pages. 2006.

Vol. 4260: Z. Liu, J. He (Eds.), Formal Methods and Software Engineering. XII, 778 pages. 2006.

Vol. 4257: I. Richardson, P. Runeson, R. Messnarz (Eds.), Software Process Improvement. XI, 219 pages. 2006.

Vol. 4242: A. Rashid, M. Aksit (Eds.), Transactions on Aspect-Oriented Software Development II. IX, 289 pages. 2006.

Vol. 4229: E. Najm, J.-F. Pradat-Peyre, V.V. Donzeau-Gouge (Eds.), Formal Techniques for Networked and Distributed Systems - FORTE 2006. X, 486 pages. 2006.

Vol. 4227: W. Nejdl, K. Tochtermann (Eds.), Innovative Approaches for Learning and Knowledge Sharing. XVII, 721 pages. 2006.

Vol. 4218: S. Graf, W. Zhang (Eds.), Automated Technology for Verification and Analysis. XIV, 540 pages. 2006.

Vol. 4214: C. Hofmeister, I. Crnković, R. Reussner (Eds.), Quality of Software Architectures. X, 215 pages. 2006.

Vol. 4204: F. Benhamou (Ed.), Principles and Practice of Constraint Programming - CP 2006. XVIII, 774 pages. 2006.

Vol. 4199: O. Nierstrasz, J. Whittle, D. Harel, G. Reggio (Eds.), Model Driven Engineering Languages and Systems. XVI, 798 pages. 2006.

Vol. 4192: B. Mohr, J.L. Träff, J. Worringen, J.J. Dongarra (Eds.), Recent Advances in Parallel Virtual Machine and Message Passing Interface. XVI, 414 pages. 2006.

Vol. 4184: M. Bravetti, M. Núñez, G. Zavattaro (Eds.), Web Services and Formal Methods. X, 289 pages. 2006.

Vol. 4166: J. Górski (Ed.), Computer Safety, Reliability, and Security. XIV, 440 pages. 2006.

Vol. 4158: L.T. Yang, H. Jin, J. Ma, T. Ungerer (Eds.), Autonomic and Trusted Computing. XIV, 613 pages. 2006.

Vol. 4157: M. Butler, C. Jones, A. Romanovsky, E. Troubitsyna (Eds.), Rigorous Development of Complex Fault-Tolerant Systems. X, 403 pages. 2006.

Vol. 4143: R. Lämmel, J. Saraiva, J. Visser (Eds.), Generative and Transformational Techniques in Software Engineering. X, 471 pages. 2006.

Vol. 4134: K. Yi (Ed.), Static Analysis. XIII, 443 pages. 2006.

Vol. 4119: C. Dony, J.L. Knudsen, A. Romanovsky, A.R. Tripathi (Eds.), Advanced Topics in Exception Handling Techniques. X, 302 pages. 2006.

Vol. 4111: F.S. de Boer, M.M. Bonsangue, S. Graf, W.-P. de Roever (Eds.), Formal Methods for Components and Objects. VIII, 447 pages. 2006.

Vol. 4089: W. Löwe, M. Südholt (Eds.), Software Composition. X, 339 pages. 2006.

Vol. 4085: J. Misra, T. Nipkow, E. Sekerinski (Eds.), FM 2006: Formal Methods. XV, 620 pages. 2006.

Vol. 4079: S. Etalle, M. Truszczyński (Eds.), Logic Programming. XIV, 474 pages. 2006.

Vol. 4067: D. Thomas (Ed.), ECOOP 2006 – Object-Oriented Programming. XIV, 527 pages. 2006.

Vol. 4066: A. Rensink, J. Warmer (Eds.), Model Driven Architecture – Foundations and Applications. XII, 392 pages. 2006.

¥516.00元

Preface

This volume contains the papers presented at the 4th European Performance Engineering Workshop held during September 27–28 in Berlin.

There were 53 submissions. Each submission was reviewed by at least three Programme Committee members. From these, the committee decided to accept 20 papers.

We were very happy to have Isi Mitrani from Newcastle University give a keynote lecture on his recent work and future challenges in applied queueing theory.

The submitted papers cover all areas of performance engineering. We were able to compose an interesting program in six sessions, including sessions on theoretical work in performance engineering techniques as well as sessions presenting applications of performance engineering techniques. The final workshop program, as well as this volume, comprises the thematic sessions:

- Markov Chains
- Process Algebra
- Wireless Networks
- Queueing Theory and Applications of Queueing
- Benchmarking and Bounding
- Grid and Peer-to-Peer Systems

The volume includes very theoretical papers on topics such as bounds in stochastic ordering, canonical representation of phase-type-distributions and algorithms to solve closed queueing networks. Some papers study properties of numerical solution algorithms, other contributions evaluate hardware or software design and propose benchmarks. On the application side there are, furthermore, evaluations of wireless protocols, simulation studies of distributed systems and performance evaluation of system monitoring tools. We hope that this volume will provide a reference for fundamental work in performance engineering.

The success of the workshop is due to many helping hands. First of all, the members of the Program Committee were very cooperative, spent much time on reading and evaluating the submitted papers and gave advice where needed. Luckily, Miklos Telek passed on his experience after organizing last year's workshop.

Thanks to Levente Bodrog from Budapest the workshop had a professionally designed Web site. The EasyChair conference management software eased the administration of the PC meeting and composition of this volume. We thank the publisher for his support and continuity.

Last, but not least, I would like to thank the local organizers Johannes Zapotoczky and Steffen Tschirpke. Philipp Reinecke derserves special thanks for his help at all times.

July 2007 Katinka Wolter

Conference Organization

Programme Chair

Katinka Wolter

Programme Committee

Jeremy Bradley	Imperial College, London (UK)
Mario Bravetti	University of Bologna (Italy)
Lucy Cherkasova	HP Labs (USA)
Lucia Cloth	University of Twente (Netherlands)
Michel Cukier	University of Maryland (USA)
Tadeusz Czachorski	IITiS PAN, Gliwice (Poland)
Jean-Michel Fourneau	Université de Versailles (France)
Stephen Gilmore	University of Edinburgh (UK)
Armin Heindl	University of Erlangen-Nürnberg (Germany)
András Horváth	University of Torino (Italy)
Carlos Juiz	University Illes Balears (Spain)
Tomáš Kalibera	Charles University in Prague (Czech Republic)
Helen Karatza	Aristotle University of Thessaloniki (Greece)
Leïla Kloul	Université de Versailles (France)
Kim G. Larsen	University of Aalborg (Denmark)
Hermann de Meer	Passau University (Germany)
Aad van Moorsel	Newcastle University (UK)
Manuel Nunez	University Complutense de Madrid (Spain)
Fernando L. Pelayo	University Castilla-La Mancha (Spain)
Brigitte Plateau	Polytechnical Institute of Grenoble, LIG, (France)
Rob Pooley	Heriot-Watt University, Edinburgh (UK)
Marina Ribaudo	University of Genova (Italy)
Marco Scarpa	University of Messina (Italy)
Markus Siegle	University of the Federal Armed Forces, Munich (Germany)
Mark Squillante	IBM T.J. Watson Research Center, NY (USA)
Ann Tai	IA Tech, Inc. (USA)
Miklós Telek	Technical University of Budapest (Hungary)
Nigel Thomas	Newcastle University (UK)
Lisa Wells	University of Aarhus (Denmark)
Katinka Wolter	Humboldt University Berlin (Germany)
Armin Zimmermann	Technical University Berlin (Germany)
Wlodek M. Zuberek	Memorial University (Canada)

Local Organization

Steffen Tschirpke
Philipp Reinecke
Johannes Zapotoczky

External Reviewers

Ashok Argent-Katwala
Leonardo Brenner
Matteo Dell'Amico
Salvatore Distefano
Marco Ferrante
David de Frutos Escrig
Gabor Horvath
Douglas de Jager
Mehdi Khouja
Harini Kulatunga
Luis Llana
Natalia Lopez
Mercedes G. Merayo
Nihal Pekergin
Antonio Puliafito
Philipp Reinecke
Martin Riedl
Afonso Sales
Pere P. Sancho
Johann Schuster
Giuseppe Scionti
Antonio Sola
Valentin Valero
Maria Vigliotti

Table of Contents

Keynote

Markov Chains

Process Algebras and State Machines

Grid and Peer-to-Peer Systems

Queueing Theory and Applications of Queueing

Benchmarking and Bounding

Wireless Networks

Optimization Problems in Service Provisioning Systems

Isi Mitrani

School of Computing Science
Newcastle University
Newcastle upon Tyne
United Kingdom
isi.mitrani@ncl.ac.uk

A service provisioning system typically contains a number of servers which may be distributed, heterogeneous and intermittently unavailable. They are used by the host in order to offer different services to a community of users. There may or may not be Service-Level Agreements involving Quality of Service constraints. In this context, there are several areas where dynamic optimisation problems arise quite naturally. These are (a) Routing and load-balancing: Where should an incoming request be sent for execution? If some queues grow large while others are short, can something be gained by transferring jobs among them? (b) Resource allocation: If different servers are dedicated to different types of service, how many should be assigned to each? When should a server be switched from one type of service to another? (c) Revenue maximisation: How are resource allocation and job admission policies affected by economic considerations? In particular, if service-level agreements specify payments for serving jobs and penalties for failing to provide a given quality of service, how many servers should be assigned to each type of service and when should jobs of that type be accepted?

The talk will describe models that address the above problems and will discuss routing, allocation and admission policies that may be adopted in practical systems.

K. Wolter (Ed.): EPEW 2007, LNCS 4748, p. 1, 2007.

Untold Horrors About Steady-State Probabilities: What Reward-Based Measures Won't Tell About the Equilibrium Distribution*

Alexander Bell and Boudewijn R. Haverkort

University of Twente
Dept. Electrical Engineering, Mathematics and Computer Science
P.O. Box 217, 7500 AE Enschede, the Netherlands
a.bell@math.utwente.nl, brh@cs.utwente.nl

Abstract. These days, parallel and distributed state-space generation algorithms allow us to generate Markov chains with hundreds of millions of states. In order to solve such Markov chains for their steady-state behaviour, we typically use iterative algorithms, either on a single machine, or on a cluster of workstations. When dealing with such huge Markov chains, the accuracy of the computed probability vectors becomes a critical issue.

In this paper we report on experimental studies of, among others, the impact of different iterative solution techniques, erratic and stagnating convergence, the impact of the state-space ordering, the influence of the processor architecture chosen and the accuracy of the measure of interest, in relation to the accuracy of the individual state probabilities.

To say the least, the paper shows that the results from analysing extremely large Markov chains should be "appreciated with care", and, in fact, questions the feasibility of the ambitious "5 nines programs" that some companies have recently started.

1 Introduction

With the advent of high-level description languages for Markovian models, such as those based on stochastic Petri nets or stochastic process algebras, it has become easy to specify extremely large Markovian models. Also, the deployment of structured and symbolic approaches towards state space generation, such as those using Kronecker algebra and those based on, for instance, multi-terminal binary decision diagrams, has made Markovian models with thousands of millions of states a reality. However, describing and generating state spaces is one thing, solving the Markov chains associated with these enormous state spaces is another issue. The largest Markovian models we are aware of that have been solved numerically have close to a billion states [2] (using an explicit state space representation and a disk-based parallel solver). Clearly, currently the solution step is lagging behind.

* The title of this paper has been inspired by [11].

K. Wolter (Ed.): EPEW 2007, LNCS 4748, pp. 2–17, 2007.

In this paper we address the question of 1 how much confidence one actually can have in performance and dependability measures derived from numerical steady-state solutions of such extremely large Markov chains. What can we actually say about the accuracy of the computed probabilities? When we have so many states, can we still compute the state probabilities accurately enough? And how do the numerical algorithms "react" on such very small probabilities? Furthermore, if we employ parallel algorithms for the solution of the steady-state probabilities, does the way in which we distribute the state space over the nodes or the timing of information-exchange between the nodes (non-determinism) affect the accuracy of the measures we compute?

In order to illustrate our thoughts with experimental data, we present results for a generalised stochastic Petri net (GSPN) that has been used by many researchers in the past, the Flexible Manufacturing System (FMS) model [3]. This choice also gives us the ability to compare results computed at four different sites, i.e., at the RWTH Aachen, at the College of William and Mary, at Imperial College, and, most recently, at the University of Twente.

The result of our paper is not so much a recipe for obtaining steady-state probabilities that are always accurate enough. Instead, the aim of the paper merely is to show how difficult it is to actually obtain accurate results, and shows pitfalls and problems that will be all around. In doing so, it actually shows that determining very accurate performance and dependability measures, like needed in the "5 nines programs" of some industrial research laboratories (implying to determine, in a model-based fashion, that the system long-term availability is at least equal to 0.99999, which coincides with a downtime of, roughly, only 5 minutes per year), is far from trivial. In fact, the practical feasibility of such endeavours must be seriously questioned.

The rest of this paper is organised as follows. Section 2 addresses specific issues related to the employed numerical algorithms, and in Section 3 we present experimental results based on our computations and compare them to other published results. Finally, Section 4 concludes the paper with a summary and outlook.

2 Iterative Solvers for Markov Chains

For the solution of very large Markov chains, only iterative solutions can be employed; their background is rehearsed in Section 2.1 (for more details on iterative methods for Markov chains, see [10]). Since these iterative methods only produce approximations of the solution, we discuss in Section 2.2 how to make sure that a certain accuracy has been achieved. Section 2.3 gives background on the usually employed floating point number representation. Because the computed state probabilities often differ by several orders of magnitude, we address the problem caused by summing large numbers of such values in Section 2.4.

2.1 Background

During the computation of the steady-state distribution for a CTMC with generator matrix Q iterative linear equation solvers compute a sequence of

Untold Horrors About Steady-State Probabilities: What Reward-Based Measures Won't Tell About the Equilibrium Distribution*

Alexander Bell and Boudewijn R. Haverkort

University of Twente
Dept. Electrical Engineering, Mathematics and Computer Science
P.O. Box 217, 7500 AE Enschede, the Netherlands
a.bell@math.utwente.nl, brh@cs.utwente.nl

Abstract. These days, parallel and distributed state-space generation algorithms allow us to generate Markov chains with hundreds of millions of states. In order to solve such Markov chains for their steady-state behaviour, we typically use iterative algorithms, either on a single machine, or on a cluster of workstations. When dealing with such huge Markov chains, the accuracy of the computed probability vectors becomes a critical issue.

In this paper we report on experimental studies of, among others, the impact of different iterative solution techniques, erratic and stagnating convergence, the impact of the state-space ordering, the influence of the processor architecture chosen and the accuracy of the measure of interest, in relation to the accuracy of the individual state probabilities.

To say the least, the paper shows that the results from analysing extremely large Markov chains should be "appreciated with care", and, in fact, questions the feasibility of the ambitious "5 nines programs" that some companies have recently started.

1 Introduction

With the advent of high-level description languages for Markovian models, such as those based on stochastic Petri nets or stochastic process algebras, it has become easy to specify extremely large Markovian models. Also, the deployment of structured and symbolic approaches towards state space generation, such as those using Kronecker algebra and those based on, for instance, multi-terminal binary decision diagrams, has made Markovian models with thousands of millions of states a reality. However, describing and generating state spaces is one thing, solving the Markov chains associated with these enormous state spaces is another issue. The largest Markovian models we are aware of that have been solved numerically have close to a billion states [2] (using an explicit state space representation and a disk-based parallel solver). Clearly, currently the solution step is lagging behind.

* The title of this paper has been inspired by [11].

K. Wolter (Ed.): EPEW 2007, LNCS 4748, pp. 2–17, 2007.

of $\pi^{(k)}$ equals 1 or by using the criterion $e^{(k)} = \max_i \left(\frac{|\pi_i^{(k)} - \pi_i^{(k-1)}|}{|\pi_i^{(k)}|} \right) \leq \delta$, which computes the relative error between two successive approximations.

Secondly, it may falsely detect convergence if the iteration process converges very slowly, hence, the difference between two successive approximations is smaller then δ, even though an appropriate solution would require far more iterations. Stewart [10] suggests to check the differences of non-successive approximations resulting in a stopping criterion $e^{(k)} = \max_i \left(\frac{|\pi_i^{(k)} - \pi_i^{(k-m)}|}{|\pi_i^{(k)}|} \right) \leq \delta$, where approximations lying m iterations apart are compared. Note that m is not required to be constant, but may be chosen as a function of the convergence rate or the iteration count. An obvious disadvantage of this criterion is the fact that an additional old approximation has to be stored whereas the comparison of two successive approximations can be done on-the-fly even for a Gauss-Seidel iteration using only a single vector.

The stopping criteria discussed above can only be used if the successive approximations get better during each iteration step. If the method exhibits so-called erratic convergence (see the example in Section 3.3 for the CGS method [9]), then no conclusions about the achieved accuracy can be drawn from the comparison of two successive (or m-step apart) approximations. Hence, stopping criteria based on the residual $r^{(k)} = \pi^{(k)}Q$ should be used in conjunction with the CGS method [2,6]. Of course, these can also be applied in combination with the methods of Jacobi and Gauss-Seidel. The quality of an approximation is better the closer the residual is to zero. Note that the standard definition of the residual of a linear system $Ax = b$ is $r = Ax^{(k)} - b$. As before, the absolute magnitude of the entries in the residual vector can only be interpreted meaningfully if we compare them to the magnitude of the (sought for) elements in the solution vector. Hence, the most common stopping criterion based on the residual is $e^{(k)} = \frac{||r^{(k)}||}{||\pi^{(k)}||} \leq \delta$. Again, any norm will do, but the maximum norm is the most common choice. If we use it and rewrite the stopping criterion as $||r^{(k)}||_\infty \leq \delta ||\pi^{(k)}||_\infty$ we see that the largest entry in the residual, which should be as close to zero as possible, is at most δ times the largest entry in the approximation vector $\pi^{(k)}$. For the rest of this paper, if not mentioned otherwise, we will use this relative residual criterion as the stopping criterion. Note that the residual can be computed at no additional cost during Jacobi and CGS iterations [7].

A stopping criterion not based on the achieved accuracy but on the speed of convergence can be applied to methods like Jacobi and Gauss-Seidel that typically exhibit nearly monotone linear convergence up to a certain accuracy. If this accuracy is achieved, often no further progress will be made. One can observe this point by analysing the fraction of two successive error approximations $\frac{e^{(k-1)}}{e^{(k)}}$. An example where this point is reached will be given in Section 3.2. Note that this criterion can also be used for approximations that are more than just one iteration apart without the need to store the iteration vectors.

approximations $\pi^{(0)}, \pi^{(1)}, \pi^{(2)}, \cdots$ for the solution vector π of the linear system $0 = \pi Q$ (which we may rewrite as $Q^{\mathrm{T}}\pi^{\mathrm{T}} = 0$ to correspond to the more general representation of linear systems $Ax = b$). Any iterative solver computes the next approximation $\pi^{(i+1)}$ by the iteration $\pi^{(i+1)} = H \cdot \pi^{(i)} + c$, where H is called the iteration matrix. Clearly, one iteration step "costs" one matrix-vector product (MVP). The number of iterations k required for an accuracy ϵ can be approximated from the spectral radius ρ of the iteration matrix H as $k = \frac{\log \epsilon}{\log \rho}$ (see: [1,10]). Instead of the spectral radius the magnitude of the sub-dominant eigenvalue can be used. Although this result looks very attractive it is of little use in practice as the computation of the eigenvalues of H requires approximately the same effort as the computation of the steady-state solution. Hence, other methods to detect convergence and hence to limit the number of iterations k have to be used, as will be discussed below.

An important issue to address is the number of solution vectors that needs to be stored at any point in time during the iterative solution process. Using double precision floating point numbers, a single solution vector (which is non-sparse) costs 8 megabyte per 1 million states. On a machine with 1 gigabyte of main memory, roughly speaking, the solution vector for a Markov chain with 100 million states can be stored, provided only a single iteration vector is required, such as is the case for Gauss-Seidel. For the Jacobi method, already two vectors are required, thus limiting the number of states to roughly 50 million. For Conjugate Gradient Squared (CGS), even more vectors are required. In all these cases, it is assumed that the matrix Q is either stored very compactly, recomputed on the fly, or stored on disk. We note that whereas for the serial solution of the steady-state probabilities methods like Gauss-Seidel, SOR, Jacobi and the CGS can be employed, parallel implementations tend to use only the Jacobi and CGS method as they can be parallelised more easily and efficiently.

2.2 Stopping Criteria

The simplest properties that can be used as stopping criteria are either to limit the maximum number of iterations or the time spent computing them. This surely limits the iteration count k but can not guarantee that the remaining error $e^{(k)} = \pi^{(k)} - \pi$ is smaller than some chosen limit. Better, but still traditional stopping criteria are based on the norm of the difference of successive iterates $e^{(k)} = ||\pi^{(k)} - \pi^{(k-1)}||$, cf. [10], where the iteration is stopped if this norm falls below $\delta > 0$. Although any norm will do, the most popular choice is the maximum norm $||x||_\infty = \max_i |x_i|$, as it requires the fewest floating point operations to perform and no underflows or overflows can occur with it. This consideration applies to all norms we will use in this section. This approach has several problems, though. First of all, it does not take into account the magnitude of the (largest) elements of the solution vector, which, indeed, may all be very small if the probability vectors consist of several hundreds of millions of entries. This problem can be overcome by either scaling $\pi^{(k)}$ in a way that the largest element