Nieuwsbrief van de
Nederlandse Vereniging voor Theoretisch Informatica

Susanne van Dam[1] Joost-Pieter Katoen Joost Kok Jaco van de Pol

Inhoudsopgave

[1] CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands. Email: Susanne.van.Dam@cwi.nl

Van de redactie en bestuur

Geachte NVTI-leden!

De NVTI nieuwsbrief van 2008 ligt voor u, of heeft u in de hand. Zoals gebruikelijk vindt u hierin de aankondiging voor de NVTI theoriedag, die dit jaat op vrijdag 14 maart in Utrecht plaatsvindt. Sprekers dit jaar zijn: Christos Papadimitriou (Berkeley), Colin Stirling (University of Edinburgh), Mariëlle Stoelinga (UT) en Ronald de Wolf (CWI). De reputatie van deze sprekers zou voor u meer dan voldoende reden moeten zijn om op 14 maart van de partij te zijn! Daarnaast is de bijeenkomst een uitgelezen mogelijkheid om met uw collega's van gedachten te wisselen.

De wetenschappelijke bijdragen in deze nieuwsbrief zijn van Hans Zantema (TU/e en RU), Bettina Speckmann (TU/e) en Mark van Kreveld (UU), en het kwartet Wan Fokkink (VU), Rena Bakhshi (VU), Francois Bonnet (ENS Cachan/IRISA) en Boudewijn R. Haverkort (UT). Dat de onderzoekscholen IPA en SIKS het afgelopen jaar ook niet hebben stil gezeten blijkt uit hun lijst van activiteiten. Wij danken alle auteurs voor hun bijdrage.

Tenslotte danken wij het CWI, NWO, Elsevier, IPA, SIKS en OzsL voor de sponsoring van de NVTI activiteiten, en Susanne van Dam voor de secretariele ondersteuning en attenderen wij u erop dat onze secretaris het CWI heeft verlaten en als hoogleraar is gestart in het oosten van het land aan de Universiteit Twente.

Namens de redactie en het bestuur van de NVTI,

Joost-Pieter Katoen, hoofdredacteur
Joost Kok, voorzitter
Jaco van de Pol, secretaris

Samenstelling bestuur

Prof. dr. Jos Baeten (TU/e)
Prof. dr. Mark de Berg (TU/e)
Prof. dr. Harry Buhrmann (CWI en UvA)
Prof. dr. ir. Joost-Pieter Katoen (RWTH Aachen en UT)
Prof. dr. Jan-Willem Klop (CWI, RUN en VU)
Prof. dr. Joost Kok (UL), voorzitter
Prof. dr. John-Jules Meyer (UU)
Prof. dr. Jaco van de Pol (UT), secretaris
Dr. Femke van Raamsdonk (VU)
Prof. dr. Gerard Renardel de Lavalette (RUG)
Dr. Leen Torenvliet (UvA)

NVTI Theoriedag 14 maart 2008

We are happy to invite you for the Theory Day 2008 of the NVTI. The Dutch Asssociation for Theoretical Computer Science (NVTI) supports the study of theoretical computer science and its applications.

- Friday March 14, 2008, 9:30-16:45
- Hoog Brabant, Utrecht (close to Central Station)

We have an interesting program with excellent speakers from The Netherlands and abroad, covering important streams in theoretical computer science. Below you will find the abstracts.

The NVTI theory day 2008 is financially or otherwise sponsored by NWO (Netherlands Organisation for Scientific Research), Elseviers Science, CWI (Dutch Center of Mathematics and Computer Science) and the Dutch research schools IPA (Institute for Programming Research and Algorithmics), OzsL (Dutch Graduate school in Logic) and SIKS (Dutch research school for Information and Knowledge Systems).

Scientific Lecturers

- Colin Stirling (University of Edinburgh)
- Marielle Stoelinga (University of Twente)
- Ronald de Wolf (Centrum voor Wiskunde en Informatica)
- Christos Papadimitriou (Berkeley)

Program

9.30-10.00	Arrival with Coffee
10.00-10.10	Opening
10.10-11.00	**Lecture**: Prof. Colin Stirling (U. Edinburgh) **Title**: Higher-Order Matching, Games and Automata
11.00-11.30	Coffee/Tea
11.30-12.20	**Lecture**: Dr. Marielle Stoelinga (U Twente) **Title**: From Quality to Quantity: Logics, approximation and model checking of quantitative system models
12.20-12.40	Dr. Mark Kas (NWO Physical Sciences)
12.40-14.10	Lunch (see below for registration)
14.10-15.00	**Lecture**: Dr. Ronald de Wolf (CWI) **Title**: Fourier analysis of Boolean functions: Some beautiful examples
15.00-15.20	Coffee/Tea
15.20-16.10	**Lecture**: Prof. Christos Papadimitriou (Berkeley) **Title**: Computing Equilibria
16.10-16.20	Henriette Jensenius: Announcement Lorentz Center Leiden
16.20-16.45	Business meeting NVTI

Abstracts

Speaker: Colin Stirling (U. Edinburgh)
Title: Higher-Order Matching, Games and Automata
Abstract: A notable success in theoretical computer science is methods for verifying finite and infinite state systems such as model checking. An active research goal is to transfer these techniques to finite and infinite state systems with binding. In the talk, we report on some recent work in this direction on higher-order schema and on higher-order matching.

Speaker: Marielle Stoelinga (U Twente)
Title: From Quality to Quantity:
Logics, approximation and model checking of quantitative system models
Abstract: Many system models contain quantitative information, such as time, probability, resource consumption, and continuous dynamics. Boolean analysis of such systems (based on a yes/no answer) is unsatisfying, for example because small perturbations in the system model may lead to opposite truth values. This talk presents a framework for quantitative specification, model checking and refinement.
Starting point is the quantitative transition system (QTS) model, where predicates are assigned real numbers in [0,1]. We consider quantitative versions of LTL and CTL, called QLTL and QCTL, assigning to each formula a truth value in [0,1]. We define the linear and branching distances, which are the quantitative analogi of trace equivalence and bisimulation. Finally, we discuss how this framework generalizes to Markov Decision Processes and Stochastic 2-player games.

Speaker: Ronald de Wolf (CWI)
Title: Fourier analysis of Boolean functions: Some beautiful examples
Abstract: Fourier analysis of real-valued functions on the Boolean hypercube has been an extremely versatile tool in theoretical computer science in the last decades. Applications include the analysis of the behavior of functions with noisy inputs, cryptography, machine learning theory, design of probabilistically checkable proofs, threshold phenomena in random graphs, etc. In this talk we will give a brief introduction to this area, illustrated with a number of simple but elegant applications.

Speaker: Christos Papadimitriou (Berkeley)
Title: Computing Equilibria
Abstract: In 1951 Nash showed that every game has a mixed equilibrium; his proof is essentially a reduction to Brouwer's fixpoint theorem. Whether such an equilibrium can be found efficiently has been open since that time. This talk surveys some recent results that shed light to this problem, essentially by demonstrating a reduction in the opposite direction. We also discuss the many surrounding problems, such as approximate equilibria, correlated equilibria, and repeated games.

Lunch

It is possible to participate in the organized lunch, for which registration is required. Please register by E-mail (Susanne.van.Dam@cwi.nl) or by phone (020-5924189), no later than one week before the meeting (March 7, 2008). The costs of 15 Euro can be paid at the location. We just mention that in the direct vicinity of the meeting room there are plenty of nice lunch facilities as well.

www.win.tue.nl/ipa/

Institute for Programming research and Algorithmics

The research school IPA (Institute for Programming Research and Algorithmics) educates researchers in the field of programming research and algorithmics. This field encompasses the study and development of formalisms, methods and techniques to design, analyse, and construct software systems and components. IPA has three main research areas: Algorithmics & Complexity, Formal Methods, and Software Technology & Engineering. Researchers from nine universities (Radboud University Nijmegen, Leiden University, Technische Universiteit Eindhoven, University of Twente, Utrecht University, University of Groningen, Vrije Universiteit Amsterdam, University of Amsterdam, and Delft University), the CWI and Philips Research (Eindhoven) participate in IPA.

In 1997, IPA was formally accredited by the Royal Dutch Academy of Sciences (KNAW). This accreditation was extended in 2002, and again last year, this time for a period of six years. In setting the agenda for 2007 - 2012, the IPA community chose five focus areas, where it expects important developments in the near future and wants to stimulate collaboration:

> *Beyond Turing* where we want to explore novel concepts and paradigms of computation.
>
> *Algorithms & models for life sciences* where we wish to apply algorithmic theory and formal models to contribute to the understanding of biological processes, entities, and phenomena.
>
> *Hybrid systems* where we want to continue to contribute to the confluence of systems and control theory and computer science in integrated methods for modelling, simulation, analysis, and design of such systems.
>
> *Model-driven software engineering* where we want to study various fundamental aspects of the model-driven approach to software engineering.
>
> *Software analysis* where we want to make progress in the extraction of facts from source code and their analysis, to obtain instruments for measuring the various quality attributes of software.

Activities in 2007

IPA has two multi-day events per year which focus on current topics, the Lentedagen and the Herfstdagen. In 2007, the Lentedagen were on Service-oriented Computing and the Herfstdagen were dedicated to the focus area Beyond Turing. In addition, IPA regularly organizes Basic Courses on each of its three major research fields.

Lentedagen on Service-oriented Computing *April 3 - 5, De Kapellerput, Heeze*
Service-oriented Computing (SoC) is an emerging paradigm for distributed computing that has evolved from object-orientation and component-based software engineering. In Service-oriented Computing, applications are constructed from autonomous computational elements that offer a

"service" over a network (usually internet), promising rapid development of distributed applications in heterogeneous environments, i.e. across organisations and platforms.

The Lentedagen addressed Service-oriented Computing from the perspective of compositionality, obtaining larger systems from smaller ones by means of well-understood composition rules, a leading theme of IPA research in the area of software engineering. Many of the research issues involved in constructing predictable and dependable applications out of services are familiar from previous paradigms for distributed computing. However, they appear in a new guise since in composing services one has no knowledge of (or access to) their internal structure, and no control over the way in which they are provided. In addition there are new questions, because of the emphasis on dynamic (even run-time) composition and the (autonomous) adaptation of applications to changes in requirements or context.

The Lentedagen aimed to provide an overview of research in and around IPA on these issues in Service-oriented Computing. Antonio Brogi (Pisa) was invited speaker, the program was composed by Farhad Arbab (CWI/UL), Wil van der Aalst (TU/e) and Mike Papazoglou (UvT). Abstracts, hand-outs and papers are still available through the archive on the IPA-website:
`www.win.tue.nl/ipa/archive/springdays2007/`.

Herfstdagen "Beyond Turing" *November 26 - 30, Willibrordhaeghe, Deurne*
The future challenges of computing require new concepts that are no longer adequately modeled by the classical Turing machine paradigm. Systems become increasingly interactive with their environment and learn and adapt, consist increasingly of autonomous and even mobile components that self-configure and operate by their own mechanisms, control rather than compute, and are always on, and they must deal with ever more complex contexts and masses of data. Moreover, the notion of "system" is widening: in biology as well as other sciences, processes are increasingly understood in novel computational terms. The new forms of non-classical computing that manifest themselves lead us to explore the limits of what the Turing machine models allows us to do and often lead us beyond it.

The computational models of the future are non-uniform, learn and adapt, and are operating on the basis of algorithmic mechanisms rather than deterministic control. The behavior is emergent from their many parts instead of from a central machine. The underlying network structures fluctuate by link-free connections based on proximity or negotiation. The principles of these adaptive structures and algorithms are much less understood than those of (dynamic) discrete algorithms and use new ideas from many sources, ranging from the theory of evolving systems to notions from economic games and market-oriented programming.

The Herfstdagen aimed to bring researchers from in and around IPA together, for an overview of current research in the focus area. The contributions to the program were clustered in three main themes, each presenting a different shift away from the classical paradigm: alternative computational models, interacting algorithmic systems, and evolving algorithmic systems. Invited speakers were Jiri Wiedermann (Academy of Sciences of the Czech Republic) on amorphous computing, and Jan Vahrenhold (Dortmund) on I/O-efficient algorithms. The program was composed by Emile Aarts (Philips Research, TU/e), Mark de Berg (IPA, TU/e), Harry Buhrman (CWI, UvA), Walter Kosters (UL), Jan van Leeuwen (UU), and Han La Poutré (CWI,TU/e). Abstracts, hand-outs, and papers are available through the IPA website at `www.win.tue.nl/ipa/archive/falldays2007/`.

IPA Basic Course Algorithmics and Complexity *June 25 - 29, TU/e, Eindhoven*
This Basic Course, composed by Mark de Berg (TU/e) focussed on four subjects areas in algorithmics where succesfull research is being conducted by groups in IPA. The final course day addressed an interesting application area for algorithmic techniques: bio informatics.

Topics and teachers were: *Machine Scheduling*, Han Hoogeveen (UU); *Parametrized Complexity*, Hans Bodlaender (UU); *Randomized Geometric Algorithms*, Mark de Berg (TU/e); *Evolutionary Algorithms*, Peter Bosman (CWI); *Bio informatics*, Hendrik Jan Hoogeboom & Walter Kosters (UL).

IPA Ph.D. Defenses in 2007

Juan Guillen Scholten (UL, January 10)
Mobile Channels for Exogenous Coordination of Distributed Systems: Semantics, Implementation and Composition
Promotor: prof.dr. F. Arbab. Co-promotores: dr. F.S. de Boer, dr. M.M. Bonsangue
IPA Dissertation Series 2006-21

Nikolay Kavaldjiev (UT, January 31)
A run-time reconfigurable Network-on-Chip for streaming DSP applications
Promotor: prof.dr. P.H. Hartel. Co-promotor: dr.ir. G.J.M. Smit
IPA Dissertation Series 2007-02

Hayco de Jong (UvA, February 1)
Flexible Heterogeneous Software Systems
Promotor: prof.dr. P. Klint. Co-promotor: prof.dr. M.G.J. van den Brand
IPA Dissertation Series 2007-01

Thuy Duoung Vu (UvA, February 13)
Semantics and Applications of Process and Program Algebra
Promotor: prof.dr. J.A. Bergstra. Co-promotores: dr. I. Bethke, dr. A. Ponse
IPA Dissertation Series 2007-04

Martijn van Veelen (RuG, March 2)
Considerations on Modeling for Early Detection of Abnormalities in Locally Autonomous Distributed Systems
Promotor: prof.dr.ir. L. Spaanenburg. Co-promotor: dr.ir. J.A.G. Nijhuis
IPA Dissertation Series 2007-03

Laura Brandán Briones (UT, March 21)
Theories for Model-based Testing: Real-time and Coverage
Promotor: prof.dr. H. Brinksma
IPA Dissertation Series 2007-05

Micha Streppel (TU/e, May 29)
Multifunctional Geometric Data Structures
Promotor: prof.dr. M.T. de Berg. Co-promotor: dr. H.J. Haverkort
IPA Dissertation Series 2007-07

Richard Brinkman (UT, June 1)
Searching in Encrypted Data
Promotores: prof.dr. W. Jonker, prof.dr. P.H. Hartel
IPA Dissertation Series 2007-09

Iris Loeb (RU, June 20)
Natural Deduction: Sharing by Presentation
Promotor: prof.dr. J.H. Geuvers
IPA Dissertation Series 2007-06

N. Trčka (TU/e, June 28)
Silent Steps in Transition Systems and Markov Chains
Promotores: prof.dr. J.C.M. Baeten, prof.dr.ir. J.E. Rooda. Co-promotor: dr. S.P. Luttik
IPA Dissertation Series 2007-08

Joost Noppen (UT, July 5)
Imperfect Information in Software Development Processes
Promotor: prof.dr.ir. M. Aksit. Co-promotor: dr. P.M. van den Broek
IPA Dissertation Series 2007-11

Roel Boumen (TU/e, August 20)
Integration and Test Plans for Complex Manufacturing Systems
Promotor: Prof.dr.ir. J.E. Rooda. Co-promotor: dr.ir. J.M. van de Mortel-Fronczak
IPA Dissertation Series 2007-12

Dimitri Jarnikov (TU/e, August 27)
QoS framework for Video Streaming in Home Networks
Promotor: prof.dr.ir. P.H.N. de With. Co-promotores: dr. J.J. Lukkien, dr. P.D.V. van der Stok
IPA Dissertation Series 2007-18

Anton Wijs (VUA, October 2)
What to do Next? – Analysing and Optimising System Behaviour in Time
Promotores: prof.dr. W.J. Fokkink, prof.dr. J.C. van de Pol
IPA Dissertation Series 2007-13

Arjen van Weelden (RU, October 17)
Putting types to good use
Promotor: prof.dr.ir. R. Plasmeijer. Co-promotor: dr. S. Smetsers
IPA Dissertation Series 2007-10

Christian Lange (TU/e, October 24)
Assessing and Improving the Quality of Modeling: A Series of Empirical Studies about the UML
Promotores: prof.dr. S. Demeyer, prof.dr. M.G.J. van den Brand. Co-promotor: dr. M. Chaudron
IPA Dissertation Series 2007-14

Mohammad Abam (TU/e, November 13)
New Data Structures and Algorithms for Mobile Data
Promotor: prof.dr. M.T. de Berg. Co-promotor: dr. B. Speckmann
IPA Dissertation Series 2007-19

Aad Mathijssen (TU/e, November 15)
Logical Calculi for Reasoning with Binding
Promotor: prof.dr.ir. J.F. Groote. Co-promotor: dr. M.J. Gabbay
IPA Dissertation Series 2007-17

Tijs van der Storm (UvA, November 20)
Component-based Configuration, Integration and Delivery
Promotores: prof.dr. P. Klint, prof.dr. S. Brinkkemper
IPA Dissertation Series 2007-15

Bas Graaf (TUD, November 27)
Model-Driven Evolution of Software Architectures
Promotor: prof.dr. A. van Deursen
IPA Dissertation Series 2007-16

Activities in 2008

IPA is planning several activities for 2008, which include the Lentedagen, the Herfstdagen (on the focus area Software Analysis), and two Basic Courses (on Formal Methods and Software Technology & Engineering) to be held at the TU Eindhoven. More information on these events will appear on the IPA-website as their dates and locations are confirmed.

Addresses

Visiting address
Technische Universiteit Eindhoven
Main Building HG 7.22
Den Dolech 2
5612 AZ Eindhoven
The Netherlands

Postal address
IPA, Fac. of Math. and Comp. Sci.
Technische Universiteit Eindhoven
P.O. Box 513
5600 MB Eindhoven
The Netherlands

tel. (+31)-40-2474124 (IPA Secretariat)
fax (+31)-40-2475361
e-mail `ipa@tue.nl`
url `www.win.tue.nl/ipa/`

School for Information and Knowledge Systems (SIKS) in 2007

Richard Starmans (UU)

Introduction

SIKS is the Dutch Research School for Information and Knowledge Systems. It was founded in 1996 by researchers in the field of Artificial Intelligence, Databases & Information Systems and Software Engineering. Its main concern is research and education in the field of information and computing sciences, more particular in the area of information and knowledge systems. SIKS is an interuniversity research school that comprises 12 research groups from 10 universities and CWI. Currently, over 400 researchers are active, including nearly 200 Ph.D.-students. The Vrije Universiteit in Amsterdam is SIKS' administrative university, and as off January 1 2006 Prof.dr. R.J. Wieringa (UT) was appointed scientific director. The office of SIKS is located at Utrecht University. SIKS received its first accreditation by KNAW in 1998 and was re-accredited in June 2003 for another period of 6 years.

Activities

We here list the main activities (co-)organized or (co-)financed by SIKS. We distinguish basic courses, advanced courses and other activities (including master classes, workshops, one-day seminars, conferences, summer schools, doctoral consortia and research colloquia)

Basic courses:

“Learning and Reasoning”, May 21-22, 2007, Landgoed Huize Bergen, Vught
Course directors: dr. A. Ten Teije (VU), dr. G.A.W. Vreeswijk (UU)

“Information Retrieval”, May 23-24, 2007, Landgoed Huize Bergen, Vught
Course director: Prof. dr. ir. Th. Van der Weide (RUN)

“Research methods and methodology for IKS", November 14-16, 2007, Vught
Course directors: dr. H. Weigand (UvT), prof. dr. R. Wieringa (UT), prof.dr. H. Akkermans (VU) prof. dr. J-J.Ch. Meyer (UU), dr. R. Starmans (UU)

“Agent Technology”, December 10-11, 2007 , Landgoed Huize Bergen, Vught
Course directors: Prof. dr. C.M. Jonker (TUD), Prof. dr. J.-J.Ch. Meyer (UU),
Prof. dr. B. De Schutter (TUD), Prof. dr. C. Witteveen (TUD).

Advanced courses:

“Computational Intelligence: AI and Probability”, April 16-17 2007, Woudschoten, Zeist
Course director: Dr. T. Heskes (RUN)

“Summer course on Datamining”, July 02-06, 2007, Maastricht
Course directors: Dr. E. Smirnov (UM), Dr. J. Donkers (UM), Prof. dr. E.O. Postma (UM)

“Service-oriented computing”, October 04-05, 2007, Landgoed Huize Bergen, Vught
Course directors: Dr. H. Weigand (UvT), Dr. W.-J. van den Heuvel (UvT)

“Multi Agent Systems: Theory, Technology and Applications”,
December 12-13, 2007, Landgoed Huize Bergen, Vught
Course directors: Prof. dr. C.M. Jonker (TUD), Prof. dr. J.-J.Ch. Meyer (UU),
Prof. dr. B. De Schutter (TUD), Prof. dr. C. Witteveen (TUD).

Other activities:

- Conference: Benelearn 07, May 14-15, 2007, Amsterdam
- Conference: BNAIC 07, November 05-06, 2007, Utrecht
- Conference: Collective Intelligent Agents, September 19-21,2007, Delft
- Conference: Dutch-Belgian Database Day 2007 (DBDBD), Eindhoven
- Conference: DIR 2007, March 28-29, 2007, Leuven (BE)
- Conference: Enterprise Information Systems (EIS), June 26, 2007, Groningen
- Conference: 2nd Int. Conf. on the Pragmatic Web, October 22-23, 2007, Tilburg
- Doctoraal Consortium EIS, June 25, 2007, Groningen
- Masterclass: Norms and Institutions, June 25, 2007,Utrecht
- Masterclass: Towards a Science of the Semantic Web, February 05, 2007, Amsterdam
- NVTI Theory Day 2007, March 09, 2007, Utrecht
- SIKS-day 2007, May 04, 2007, Utrecht
- Seminar on Information Retrieval, November 01, 2007, Amsterdam
- Seminar on the Semantic Web, 11 April 2007, Utrecht
- SIKS-Agent Colloquia (9 times), Utrecht/Delft/Amsterdam
- SIKS-IKAT colloquia (5 times), Maastricht
- SIKS-UU seminars on Cognition (2 times) Utrecht
- Summerschool DECOI, August 20-24, 2007, Amsterdam
- Summerschool EASSS 2007, August, 27-31, 2007, Durham (UK)
- Symposium: Human Computer Interaction, November 23, 2007, Amsterdam
- Symposium: Intelligent Systems, October 12, 2007, Maastricht
- Symposium: Logic and Cognition, June 01, 2007, Groningen
- Symposium: Situated Models of Cognition and Perception, September 19, 2007, Maastricht
- Tutorial Program SIGIR, July 23-27, 2007, Amsterdam
- Winterschool Ten Competence, January 22-26, 2007, Innsbruck
- Workshop on Computer Games, June 15-17, 2007, Amsterdam
- Workshop Machine Learning for Natural Language Processing, May 16, 2007, Amsterdam
- Workshop: Latent Semantic Analysis, March 29-30, 2007, Heerlen
- Workshop: Value Modeling, January 18-19, 2007,Tilburg

Ph.D.-defenses in 2007

In 2007 25 researchers successfully defended their Ph.D.-thesis and published their work in the SIKS-dissertation Series.

2007-01
Kees Leune (UvT)
Access Control and Service-Oriented Architectures
Promotor: Prof.dr.ir. M.P. Papazoglou (UvT)
Co-promotor: Dr. W-J. van den Heuvel (UvT)
Promotion: 28 February 2007

2007-02
Wouter Teepe (RUG)
Reconciling Information Exchange and Confidentiality: A Formal Approach
Promotor: Prof. dr. L.R.B. Schomaker (RUG)
Co-promotor: Dr. L.C. Verbrugge (RUG)
Promotion: 18 January 2007

2007-03
Peter Mika (VU)
Social Networks and the Semantic Web
Promotor: Prof.dr. J.M. Akkermans (VU), prof.dr. T. Elfring (VU)
Co-promotor: Dr. P. Groenewegen (VU)
Promotion: 05 February 2007

2007-04
Jurriaan van Diggelen (UU)
Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach
Promotor: Prof.dr. J.-J. Ch. Meyer (UU)
Co-promotores: Dr. ir. R.-J. Beun (UU), dr. F. P.M. Dignum (UU), dr. R. M. van Eijk (UU)
Promotion: 21 March 2007

2007-05
Bart Schermer (UL)
Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance
Promotor: Prof.dr. H.J. van den Herik (UM/UL)
Promotion: 09 May 2007

2007-06
Gilad Mishne (UVA)
Applied Text Analytics for Blogs
Promotor: Prof.dr. M. de Rijke (UVA)
Promotion: 27 April 2007

2007-07
Natasa Jovanovic' (UT)
To Whom It May Concern - Addressee Identification in Face-to-Face Meetings
Promotor: Prof. dr. ir. A. Nijholt (UT)
Co-promotor: Dr. ir. H.J.A. op den Akker (UT)
Promotion: 14 March 2007

2007-08
Mark Hoogendoorn (VU)
Modeling of Change in Multi-Agent Organizations
Promotores: Prof. dr. J. Treur (VU) , Prof. dr. C.M. Jonker (TUD)
Promotion: 18 June 2007

2007-09
David Mobach (VU)
Agent-Based Mediated Service Negotiation
Promotor: Prof.dr. F.M.T. Brazier (VU)
Co-promotor: Dr. B.J. Overeinder (VU)
Promotion: 21 May 2007

2007-10
Huib Aldewereld (UU)
Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols
Promotor: Prof.dr. J.-J. Ch. Meyer (UU)
Co-promotor: Dr. F.Dignum (UU)
Promotion: 04 Juni 2007

2007-11
Natalia Stash (TUE)
Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System
Promotores: Prof.dr. P.M.E. De Bra (TUE), Prof.dr. L. Hardman (CWI / TUE)
Co-promotor: Dr. A.I. Cristea (University of Warwick, UK)
Promotion: 02 July 2007

2007-12
Marcel van Gerven (RUN)
Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty
Promotor: Prof. dr. ir. Th.P. van der Weide (RUN)
Co-promotor: Dr. P.J.F. Lucas (RUN)
Promotion: 05 September 2007

2007-13
Rutger Rienks (UT)
Meetings in Smart Environments; Implications of Progressing Technology
Promotor: Prof. dr. ir. A. Nijholt (UT)
Co-promotor: Dr. D. Heylen (UT)
Promotion: 11 July 2007

2007-14
Niek Bergboer (UM)
Context-Based Image Analysis
Promotores:Prof. dr. H.J. van den Herik (UM), Prof. dr. E.O. Postma (UM)
Promotion: 10 Oktober 2007

2007-15
Joyca Lacroix (UM)
NIM: a Situated Computational Memory Model
Promotores:Prof. dr. J.M.J. Murre (UM/UvA), Prof. dr. E.O. Postma (UM), Prof. dr. H.J. van den Herik (UM)
Promotion: 20 September 2007

2007-16
Davide Grossi (UU)
Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems
Promotor: Prof.dr. J.-J. Ch. Meyer (UU)
Co-promotor: Dr. F.Dignum (UU)
Promotion: 17 September 2007

2007-17
Theodore Charitos (UU)
Reasoning with Dynamic Networks in Practice
Promotor: Prof.dr.ir L.C. van der Gaag (UU)
Promotion: 17 September 2007

2007-18
Bart Orriens (UvT)
On the development an management of adaptive business collaborations
promotor: Prof. Dr. Ir. M.P. Papazoglou (UvT)
Co-promotor: Dr. J. Yang (UvT)
Promotion: 12 September 2007

2007-19
David Levy (UM)
Intimate relationships with artificial partners
Promotores: Prof. dr. M.J.H. Meijer (UM), Prof. dr. H.J. van den Herik (UM)
Promotion: 11 Oktober 2007

2007-20
Slinger Jansen (UU)
Customer Configuration Updating in a Software Supply Network
Promotores: Prof. dr. S. Brinkkemper (UU), Prof.dr. P. Klint (CWI)
Promotion: 08 Oktober 2007

2007-21
Karianne Vermaas (UU)
Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005
Promotor: Prof. dr. S. Brinkkemper (UU)
Co-promotor: Dr. L. van de Wijngaert (UU)
Promotion: 26 November 2007

2007-22
Zlatko Zlatev (UT)
Goal-oriented design of value and process models from patterns
Promotor: Prof. dr. R.J. Wieringa (UT)
Promotion: 04 Oktober 2007

2007-23
Peter Barna (TUE)
Specification of Application Logic in Web Information Systems
Promotores: prof. dr. P. De Bra (TUE), prof.dr G.-J. Houben (VUB/TUE)
Promotion: 30 Oktober 2007

2007-24
Georgina Ramírez Camps (CWI)
Structural Features in XML Retrieval
Promotor: Prof. dr. M.L. Kersten (CWI / UVA)
Co-promotor: Dr. A.P. de Vries (CWI/ TUD)
Promotion: 02 November 2007

2007-25
Joost Schalken (VU)
Empirical Investigations in Software Process Improvement
Promotor: Prof. dr. J.C. van Vliet (VU), Prof. dr. S. Brinkkemper (UU)
Promotion: 17 December 2007

Placing Diagrams and Symbols on Maps

Marc van Kreveld[*] Bettina Speckmann[†]

Abstract

The most commonly used types of maps are road maps or topographic maps. However, to convey specialized information, such as statistical data associated with various regions, cartographers use special purpose maps. This note describes some algorithmic questions that arise from the automated generation of two types of these maps, namely maps that contain diagrams and proportional symbol maps. We also sketch some solution strategies and show the output generated by our algorithms.

1 Introduction

The field of cartography is concerned with the design of high-quality cartographic products, most often, maps. Obviously, the uprise of computer science has had a large impact on cartography. In the beginning, software that allowed the cartographer to manipulate the map, like choosing the color scheme, the line thickness, and the map projection, facilitated the map design process. Later, software for more complex tasks became available, like text placement on maps, and automatic detail reduction to produce small-scale maps from large-scale data (cartographic generalization).

Although the most commonly used maps are road maps and topographic maps, special purpose maps exist for conveying targeted information. Such maps usually focus on a single theme and visualize such diverse topics as the gross domestic product per country, the effects of pollution on the water quality, or the migration patterns of animals. Maps for these examples can be found, respectively, in newspapers, infrastructural reports, and biological studies.

From the algorithmic perspective, there are three main directions of research in automated cartography:

- Automated text placement on maps.
- Automated map generalization.
- Automated special-purpose map design (e.g., cartograms, flow maps, dot maps).

Automated text placement has been studied extensively the last 30 years. There are several hundreds of papers on the topic, collected in an annotated bibliography [13]. Text needs to be placed with point symbols (cities, measurement locations), with linear objects (roads, rivers), and with areal objects (lakes, countries). The main algorithmic problem arising from text placement is to label the map objects in such a way that the labels do not overlap each other or other important map objects, and that there is no ambiguity as to which label corresponds to which object.

A substantial amount of research is also targeted at automated map generalization. There is a sub-community of researchers who focus on this topic, and there are workshops dedicated purely to this theme. Contrary to label placement, software that can perform map generalization satisfactorily in an automated manner is not (yet) available.

There are several types of special-purpose maps, including, but not limited to, (rectangular) cartograms, proportional symbol maps, dots maps, flow maps, and density maps—see the book by Dent [7] for various examples. Of these types, (rectangular) cartograms have probably received the most algorithmic attention, while there are only few attempts to resolve the algorithmic issues arising from dot maps, flow maps, or density maps.

In this note we focus on some algorithmic problems that arise from the automated generation of two specific types of special purpose maps, namely maps that contain diagrams and proportional

[*] Department for Information and Computing Sciences, Utrecht University. Email: marc@cs.uu.nl

[†] Department of Mathematics and Computer Science, TU Eindhoven. Email: speckman@win.tue.nl

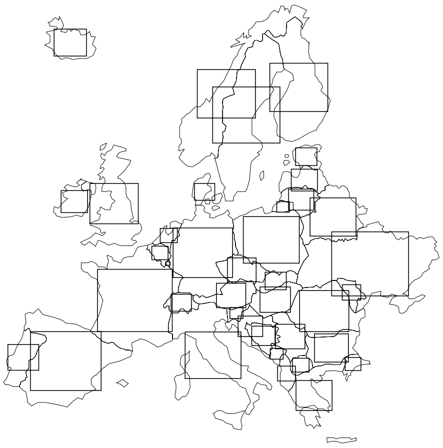


Figure 1: Rectangular diagram placement, only diagram outlines are shown.

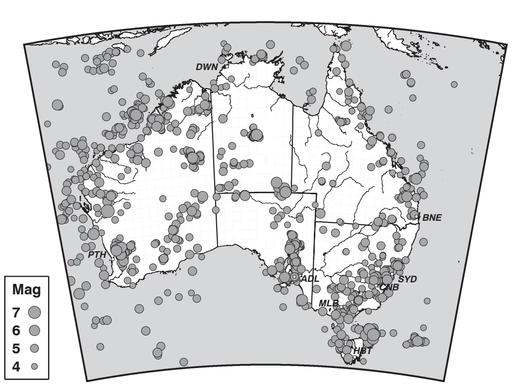


Figure 2: A proportional symbol map depicting Australian earthquakes of size ≥ 4.0 on the Richter scale [12].

symbol maps. Maps that have diagrams associated with their regions are commonly used to display statistical information about these regions—such as age distribution, precipitation per month, or voting behavior—in the form of histograms, graphs, pie charts, or pictograms. These diagrams are then placed in or at their respective regions—see the Dutch Bosatlas [5] for many examples. Proportional symbols maps are used to visualize quantitative data that is associated with specific (point) locations. A symbol, most commonly a disk or a square, is scaled such that its area corresponds to the data value associated with a point and then placed at exactly that point on a geographic map. The spatial distribution of the data can then be observed by studying the spatial distribution of the differently sized symbols. Typical data that are visualized in this way include the magnitude of earthquakes, the production of oil wells, or the temperature at weather stations. Figures 1 and 2 show examples of each type.

Placing diagrams on maps has a flavor similar to text placement, as many of the criteria for high-quality diagram placement are similar to those of text placement. Proportional symbol maps, on the other hand, often contain overlapping symbols and hence the attention is focused on how to make the data still easily readable.

Before algorithms can be developed, it is important that the cartographic problem is abstracted appropriately. A solution to the abstracted version of the problem should give maps that have high cartographic quality. Cartographic quality is of course ill-defined and is influenced by many factors. The challenge is to find abstractions that are simple enough to allow algorithmic solutions of provable quality, and which lead to good maps when applied to real data.

We discuss the algorithmic aspects of diagrams on maps and proportional symbol maps in more detail in the following sections. In both cases we discuss the abstraction of the problem, we present the algorithmic results obtained, and we show some examples of maps that are produced by prototype implementations. This note is based on papers by van Kreveld et al. [11] and by Cabello et al. [3].

2 Diagrams on maps

The input to the diagram placement problem is a map, that is, a set of regions, together with a set of diagrams where each diagram is associated with precisely one region. Note that there can be regions without associated diagram. We model the diagrams simply as circles or rectangles. Although we generally assume that diagrams are opaque we show only their outlines in the figures.

2.1 Abstraction

High-quality diagram placement immediately raises the issue of how the problem should be formalized, that is, which criteria should be taken into consideration and how. The most important criterion is that no two diagrams (belonging to different regions) overlap. If a diagram fits inside its region, then it should be placed at or near the center. If a diagram does not fit inside its region, then it should be placed so that it mostly overlaps with its own region (to obtain a clear association between region and diagram), while avoiding overlap with the region boundaries as much as possible (to keep the regions recognizable).

Even if we consider only the first criterion, namely the problem of placing equal-sized diagrams without overlap in a bounded map area (possibly outside their region), we are faced with a computationally hard problem, namely that of *packing* [10]. However, in practice, the problem is not quite so difficult because the total area taken up by the diagrams is only a small part, say, 10–40%, of the total map area. Hence, we will not attempt to address and solve the problem fully theoretically.

The straightforward solution of centering every diagram at the center of gravity (centroid) of its region is a practical solution that will often give acceptable results. However, it is not good enough to achieve high-quality results without manual post-processing. For example, if two small countries are neighboring, then their diagrams can overlap (e.g., states Vermont and New Hampshire; Figure 3), or a diagram may be unnecessarily much outside its own region (e.g., the country Israel). Nevertheless, the centroid-based solution is used in many geographic information systems.

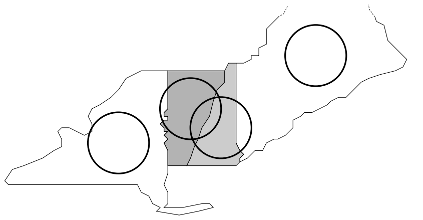


Figure 3: Placement at the center of gravity is not always a good idea.

This note discusses various models for placing a pie chart or a rectangular diagram in association with a region without explicit regard for the placement of other charts and diagrams. We will optimize the area of overlap of the diagram with its region, minimize the length of overlap with boundaries, or minimize the area of overlap with small neighboring regions. Especially the last criterion (implicitly) helps to avoid intersections between diagrams.

We consider the following five models for the diagram placement problem for a subdivision S with m regions $R_1, \ldots, R_m$ to be annotated with a diagram. Each region is assumed to be a simple polygon. The diagram C_i annotating region R_i can be circular or rectangular, and has a pre-specified size. Its center is denoted by c_i.

Centroid: Place the center c_i of the diagram at the center of gravity of R_i.

MaxSelfOverlap: Place the diagram C_i such that the area of overlap with R_i is maximized.

MinAreaOverlap: Place the diagram with $c_i \in R_i$, and such that the area of overlap with other annotated regions is minimized. This is useful to allow partial placement in seas, oceans, and other irrelevant, that is, not annotated, regions. In a formula:

$$\min_{c_i \in R_i} \operatorname{area}(C_i \cap \bigcup_{j \neq i} R_j).$$

MinBorderOverlap: Place the diagram with $c_i \in R_i$, and such that it covers the smallest possible total edge length of subdivision S.

MinMaxOverlapPerc: Place the diagram with $c_i \in R_i$, and such that the percentage that it covers of the area of any other region is minimized. In a formula:

$$\min_{c_i \in R_i} \max_{1 \leq j \leq m,\, j \neq i} \operatorname{area}(C_i \cap R_j)/\operatorname{area}(R_j)$$

The model MinMaxOverlapPerc states, for example, that it is much worse to cover 1 cm^2 of a small region than to cover 1 cm^2 of a larger region, since we measure by percentage of area covered.

All models but Centroid and MaxSelfOverlap specify that the center of the diagram should be inside R_i, otherwise all diagrams could possibly be placed in the sea. We could have resolved this problem in a different way, but we chose this condition because of its simplicity. We could also have included the criterion of avoiding covering the points where three regions meet in the subdivision. In some of the models this condition can easily be included.

In the model MinBorderOverlap we attempt to place the diagram such that the covered length of region boundaries is minimized. We can, however, easily include other linear features that should be avoided as much as possible. This is important in various situations. For example, assume we want to design a map with the German Bundesländer and their highways. We want to show the whole highway network plus various highway statistics for each Bundesland in, say, a rectangular diagram. We want to place the rectangles inside the appropriate Bundesland (for the larger part), while intersecting as little of the highway network as possible. The subdivision into Bundesländer is used to restrict the positions for the rectangles, while the highway network (possibly together with the Bundesland boundaries) is used to find minimum overlap placements.

2.2 Algorithmic results

The best diagram positions in the Centroid model are easily obtained: We can compute all centroids and place the diagrams in $O(n)$ time. For the remaining four models we first analyze how many distinct placements exist for a given diagram and the subdivision S. For ease of explanation we will consider a diagram Q that is a square in the remainder of this section. We consider two placements to be distinct, if the subset of edges of S that Q intersects differs. If we imagine Q translating over S, then a different subset of edges is obtained exactly when one of the following occurs:

- an edge of Q passes over a vertex of S;
- a vertex of Q passes over an edge of S.

If Q translates and neither of these events occurs, then Q keeps intersecting the same subset of edges of S. We will define a configuration plane Π that describes the distinct placements of Q. Let us choose the upper right corner of Q as the reference point q of Q. The configuration plane Π has cells such that q is inside a cell when Q intersects only one subset of the edges of S. Figure 4 shows a subdivision S, a square Q, and the resulting configuration plane Π with its partitioning into cells. The reason for the partitioning is the following. Within one cell of the arrangement, the area of overlap of the square and some region is described by a quadratic function $f(x, y)$. This quadratic function is needed for the optimization, and it is different in every cell.

There are $O(n^2)$ distinct subsets of edges of S intersected by Q for all placements of Q. The arrangement in the configuration plane can be constructed efficiently. It is induced by a set of $O(n)$ line segments and can be constructed by a plane sweep in $O((k + n) \log n)$ time [2, 6], where

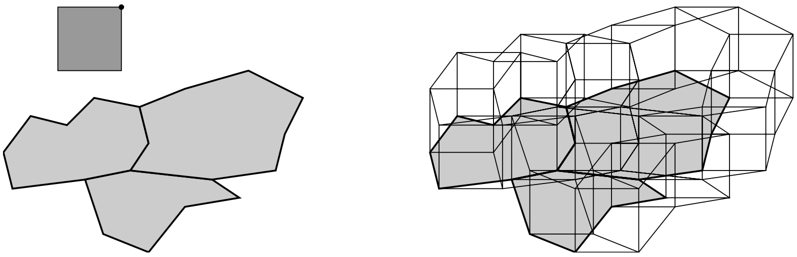


Figure 4: A simple subdivision S with three regions (left), the arrangement Π representing the combinatorially distinct placements of the square (right).

k is the number of intersections. The resulting subdivision has complexity $O(n+k)$, which in practice is less than quadratic. More complex algorithms can construct the arrangement induced by a set of n line segments in optimal $O(k + n \log n)$ time [4, 9].

Theorem 1 *Given a connected subdivision S with n edges and a square Q, the arrangement representing all distinct placements of Q with respect to S can be constructed in $O(n \log n + k)$ time, where $k = O(n^2)$ is the number of distinct placements.*

We can extend this arrangement so that for each cell, if the reference point is in that cell, then the center of the square lies inside only one of the regions. In the models MinAreaOverlap, MinBorderOverlap, and MinMaxOverlapPerc, this implies that for any cell of the partitioning it is uniquely defined which region from $R_1, \ldots, R_m$ can place its square with the reference point in that cell. The extended arrangement still has $O(n^2)$ complexity.

The same approach can be used for circular diagrams. The configuration plane Π is partitioned by a set of $O(n)$ circles and line segments into quadratically many cells. All circles whose reference points lie in the same cell intersect the same subset of the edges of the subdivision S.

The placement of a square for a region in a subdivision is a problem with two degrees of freedom: the x-coordinate and the y-coordinate of the square to be placed. We saw that this leads to a quadratic number of combinatorially distinct placements. For each of these placements, some linear or quadratic function exists that represents the function to be optimized according to the models MaxSelfOverlap, MinAreaOverlap and MinBorderOverlap. This quadratic function can be determined in $O(n^2)$ time overall by traversing the cells of the arrangement and updating in constant time the quadratic function that was valid in the previous cell. Hence we can compute the best placement of the squares in $O(n^2)$ time for the models MaxSelfOverlap, MinAreaOverlap and MinBorderOverlap. For MinMaxOverlapPerc we need to consider up to m quadratic functions for each cell, which makes the solution less efficient. Using so-called *lower-envelope* computation of quadratic surface patches [1, 8] we can determine the best placements of the squares in the MinMaxOverlapPerc model in $O(n^2 m^{1+\epsilon})$ time, where ϵ is an arbitrarily small positive constant.

2.3 Output

In this section we show some sample output of the algorithms. Figure 1 shows diagram placement according to the Centroid model, and Figures 5–8 show diagram placement according to the other four models. Grey rectangles show outlines of other diagrams that are equally good in that model. We have chosen fairly large diagrams so that we can compare the performance in complicated cases. The model MinMaxOverlapPerc performs somewhat better than the other models, where only the diagrams of Sweden and Latvia have significant overlap. The model MinAreaOverlap

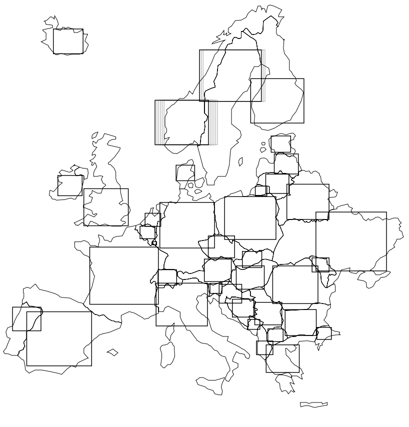


Figure 5: Model MaxSelfOverlap.

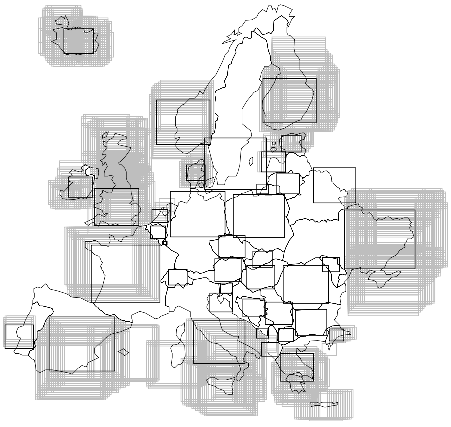


Figure 6: Model MinAreaOverlap.

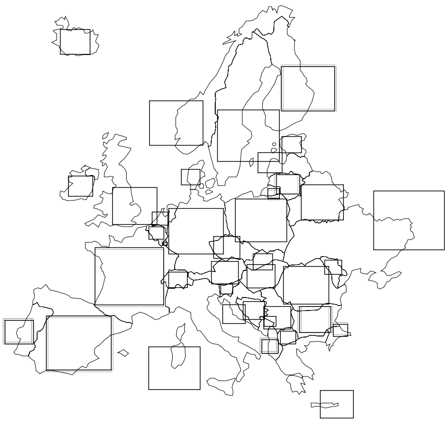


Figure 7: Model MinBorderOverlap.

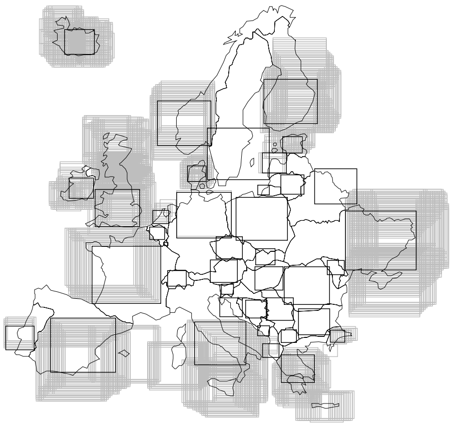


Figure 8: Model MinMaxOverlapPerc.

performs quite well too. The Centroid model (Figure 1) performs poorly in many places. It appears that the more advanced models, requiring more complicated and slower algorithms, do indeed give better quality maps.

3 Proportional symbol maps

The input to this problem is a set of overlapping disks, squares or other convex symbols with fixed locations. Our objective is to overlay these symbols such that all of them can be seen as well as possible.

3.1 Abstraction

Before we can formally state the problem we first need to introduce some definitions and notation. To simplify the presentation we give all definitions for disks, but they naturally extend to opaque squares or other shapes. Let S be a set of n disks $D_1, \ldots, D_n$ in the plane. We denote by $\mathcal{A}$ the arrangement formed by the boundaries of the disks in S. A *drawing* $\mathcal{D}$ of S is a subset of the arcs and vertices of $\mathcal{A}$ which is drawn on top of the filled interiors of the disks in S. A drawing is *bounded* if it includes the boundary of the union of the disks in S.

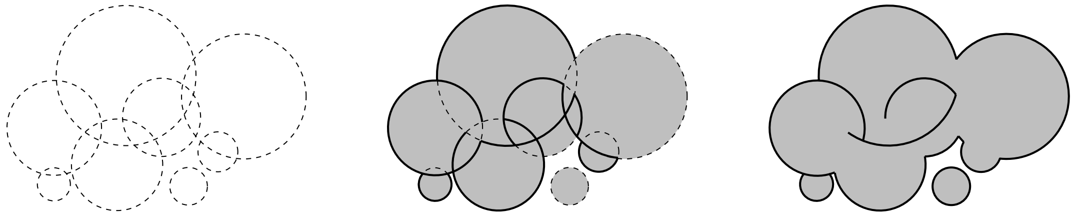


Figure 9: An arrangement $\mathcal{A}$, a drawing with $\mathcal{A}$ visible, a bounded drawing.

Not every drawing is suitable for the use on a proportional symbol map. A suitable drawing needs to be bounded. It should be locally correct at the vertices: a drawing is locally correct at a vertex v, formed by the intersection of the boundaries of two disks D_i and D_j, if locally around v the drawing corresponds to stacking D_i onto D_j or vice versa. Furthermore, a suitable drawing must have only correct faces: a face of the drawing is correct if there is an order in which all disks in S that contain the face can be drawn on top of each other such that the face appears. We call drawings that satisfy these conditions *face correct*. Figure 10 shows that even a face correct

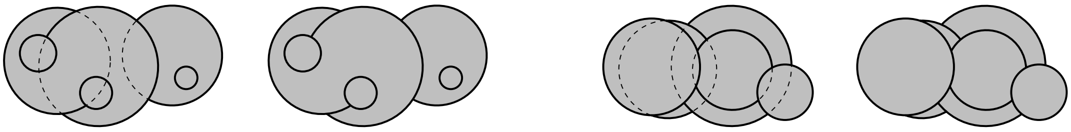


Figure 10: A bounded, vertex correct drawing, which is not face correct, with and without $\mathcal{A}$ visible (left), a bounded, face correct drawing with and without $\mathcal{A}$ visible (right).

drawing can still have an "Escher-like" quality which we would like to avoid on a proportional symbol map. Hence we need to enforce even stronger requirements on what constitutes a proper drawing. We consider two types of drawings.

Physically realizable drawings. A face correct drawing is physically realizable if and only if for every face f of the arrangement $\mathcal{A}$ there exists a total order on the disks in S_f (the disks in S that contain f) such that the topmost disk is visible and the orders associated with any two faces of $\mathcal{A}$ do not conflict. That is, the order in which the disks in S are stacked upon each other is uniquely determined at every face of $\mathcal{A}$ and no two of such orders conflict. In particular, any two or more disks that have a common intersection have a unique ordering. The orders of the disks for all faces of $\mathcal{A}$ immediately imply which arcs of $\mathcal{A}$ are part of the drawing. If we imagine that we are working with actual physical disks then we are allowed to warp them in a "Dali-like" fashion, but we cannot cut them.

Stacking drawings. A stacking drawing is a natural restriction of a physically realizable drawing and also the one most frequently found on proportional symbol maps. A physically realizable drawing $\mathcal{D}$ is a stacking drawing if there exists a total order on the disks in S such that $\mathcal{D}$ is the result of stacking the disks in this order. We call such a total order a *stacking order*.

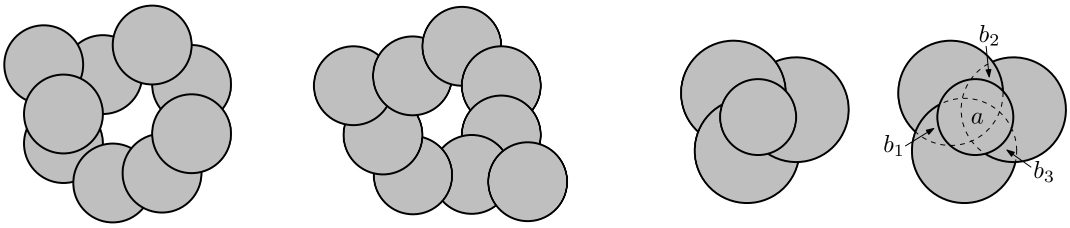


Figure 11: A stacking drawing (left), a physically realizable drawing that is not a stacking drawing (middle), a drawing that may seem physically realizable, but is not—any order for face a will conflict with one of b_1, b_2, or b_3 (right).

Quality of a drawing. Intuitively, a good drawing should enable the viewer to see at least some part of all symbols and to judge their sizes as correctly as possible. The accuracy with which the size of a disk can be judged is proportional to the portion of its boundary that is visible. This leads us to the following two optimization problems. Assume that we are given a set S of n opaque symbols $S_1, \ldots, S_n$.

Max-Min: Find a physically realizable or a stacking drawing that maximizes the minimum visible boundary length of each symbol, that is, $\max \min_{1 \le i \le n}\{\text{visible length of the boundary of } S_i\}$.

Max-Total: Find a physically realizable or a stacking drawing that maximizes the total visible boundary length over all symbols.

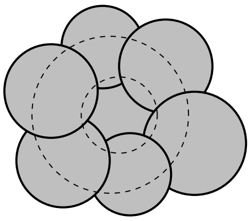


Figure 12: Visible perimeter is more important than visible area.

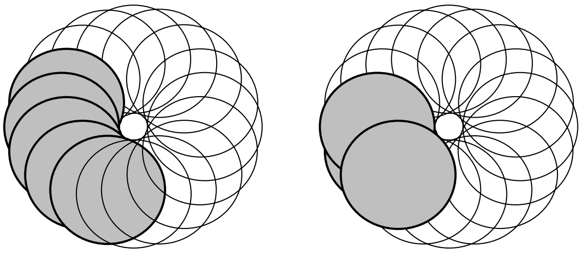


Figure 13: An optimal physically realizable drawing (left), an optimal stacking drawing for the same disks (right).

Figure 12 illustrates why we consider only visible boundary length and not visible area of symbols. The boundary of the center disk is completely covered but a significant part of its area is still visible. It is, however, impossible to judge its size or to determine the location of its center. Figure 13 shows that a stacking drawing can be arbitrarily much worse than a physically realizable drawing with respect to the Max-Min problem. At least half of the boundary of every disk in Figure 13 (left) is visible, whereas the lowest disk in any stacking drawing is covered by its two neighbors and hence has only a very short visible boundary.

Formal problem statement. Assume that we are given a set S of n opaque homothetic disks, squares or other convex symbols that may overlap. Construct a physically realizable drawing or a stacking drawing for the elements of S that either maximizes the minimum visible boundary of each symbol (Max-Min) or maximizes the total visible boundary of all symbols (Max-Total).

3.2 Algorithmic results

For physically realizable drawings both the Max-Min and the Max-Total problems are NP-hard. Furthermore, the status of the Max-Total problem for stacking drawings is open. But we can compute the stacking order that maximizes the minimum of the visible boundary of any symbol in polynomial time. More specifically, the Max-Min problem can be solved in $O(n^2 \log n)$ time for stacking drawings. We describe below a general algorithm for disks, which generalizes naturally to other convex symbols.

The general idea to compute a stacking order of n disks is simple: for each disk, we determine how much boundary would be seen if it were the bottommost disk. We choose the disk with the maximum value, make it the bottommost disk, and then recurse on the $n-1$ remaining disks. To implement this greedy approach efficiently, we maintain for each disk D_i a data structure that represents all of its covered and uncovered boundary intervals. For technical reasons, we consider a disk boundary c_i to be an interval from its topmost point clockwise around. Any other disk D_j intersects c_i in zero, one or two intervals (two if D_j contains the topmost point of c_i). The intersection points on c_i define a set of elementary intervals. In Figure 14, the elementary intervals induced by D_1, D_2, D_3, D_4 are $\alpha_1, \ldots, \alpha_9$. The data structure T_i that stores c_i is a variation of a segment tree that stores the elementary intervals in its leaves; each leaf ν stores this interval in $int(\nu)$. An internal node ν also corresponds to an interval $int(\nu)$, which is the union of elementary intervals below it in T_i. (See De Berg et al. [6] for a detailed description of segment trees.)

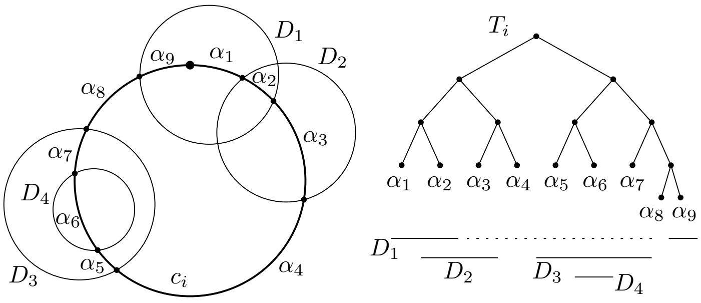


Figure 14: A disk D_i with four disks intersecting it, and the corresponding segment tree T_i.

Every node (internal and leaf) stores the boundary length of $int(\nu)$ and a counter that stores the number of other disks that contain $int(\nu)$, but not $int(\text{parent}(\nu))$. It also stores a value $vis\text{-}int(\nu)$ that is the visible boundary length of $int(\nu)$ that would remain if only the disk intervals of other disks that occur in the subtree rooted at ν would hide parts of $int(\nu)$ from view. Disk intervals at ancestors of ν may still prevent any part of $int(\nu)$ from actually being visible. The root of T_i stores the total visible perimeter length of D_i—if it were placed bottommost—in $vis\text{-}int(\text{root}(T_i))$. The counter at the root of T_i stores the number of disks completely covering D_i.

To initialize, we construct a segment tree T_i for each disk D_i, storing the disk intervals for all disks D_j with $j \neq i$. By inspecting $vis\text{-}int(\text{root})$ and the counter for all trees $T_1, \ldots, T_n$, we determine the one with the largest boundary length if it were bottommost, and select it. When a disk D_j is chosen, we delete the disk interval of D_j from all structures T_i of disks D_i that intersect D_j and were not yet chosen. To this end, we find the canonical nodes of the disk interval of D_j in T_i. For each canonical node ν, we lower the counter. When the counter becomes 0, we also update $vis\text{-}int(\nu)$ by setting it to the sum of the $vis\text{-}int(..)$ values of the two children of ν. By the standard analysis of segment trees and tree augmentation, inserting a disk interval in T_i (at initialization) or deleting a disk interval from T_i (when the disk is chosen) takes $O(\log n)$ time. Therefore, inserting a disk's intervals in all segment trees at initialization takes $O(n \log n)$ time per disk, and the process of choosing a disk to be placed bottommost and updating all trees takes $O(n \log n)$ time per disk as well. Hence we get:

Theorem 2 *Given n disks in the plane, a stacking order maximizing the boundary length of the disk that is least visible can be computed in $O(n^2 \log n)$ time.*

We can improve upon this result in certain special cases. Namely, if no point in the plane is covered by more than $O(1)$ symbols, then the Max-Min problem for stacking drawings can be solved in $O(n \log n)$ time. If the symbols are unit-size squares then it can also be solved in $O(n \log n)$ time, regardless of how many squares overlap in a point.

3.3 Output

We performed experiments to compare the results of different methods that compute a stacking drawing. Proportional symbol maps that are published in books or on the internet do not seem to follow any stacking method consistently. Some appear to be stacked from the left to the right, others appear to be random. For maps with differently sized disks, often the stacking order is from large to small (small on top). Hence we compared the following four stacking methods:

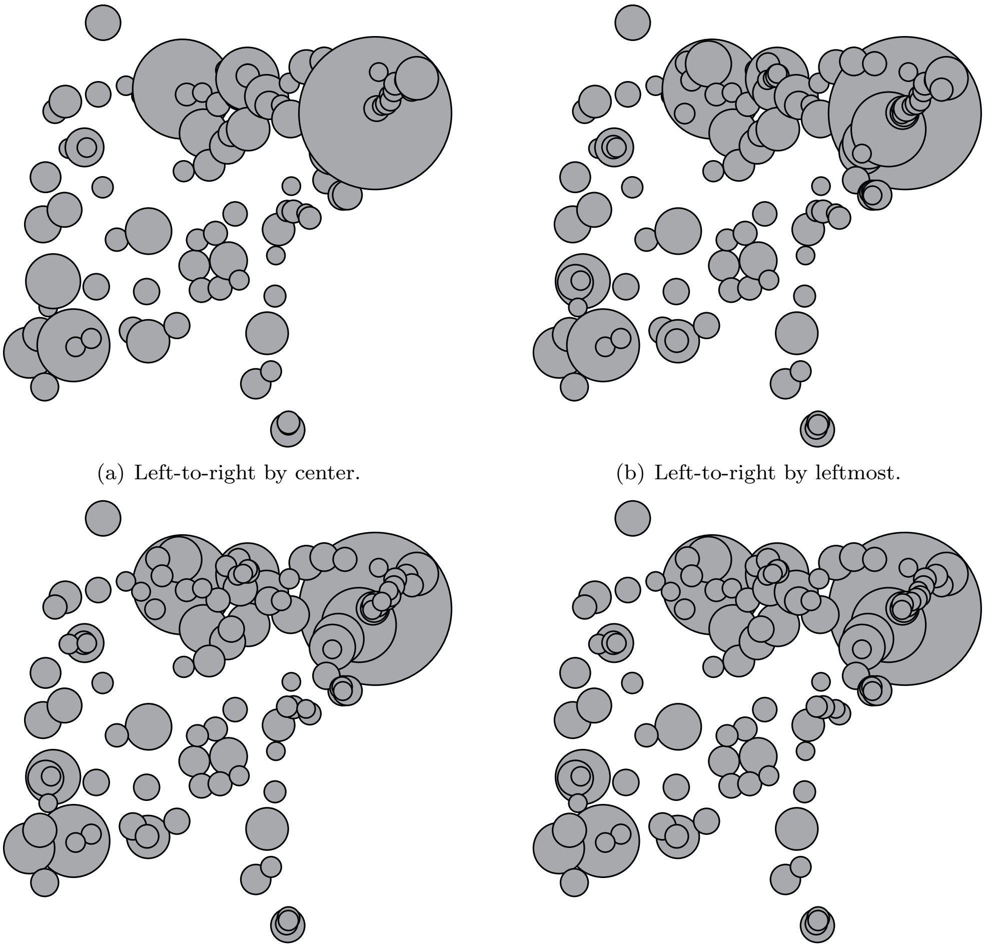
Left-to-right by center: The disk with leftmost center is put at the bottom of the stacking order, and the remaining disks are stacked recursively on top.

Left-to-right by leftmost: The disk with leftmost left extreme is put at the bottom, and the remaining disks are stacked recursively on top.

Large-to-small: The disks are stacked from bottom to top in order of non-increasing radius. If radii are distinct, then the stacking is unique.

Max-Min: The disks are stacked to maximize the visible boundary length of the disk with least visible boundary length, using the algorithm described above.

Figure 15 shows (the eastern part of) the 156 largest cities of the USA with disk areas proportional to the population stacked by the four methods described above (the differences can be seen most clearly in the upper right corners of the maps).



(a) Left-to-right by center.

(b) Left-to-right by leftmost.

(c) Large-to-small.

(d) Max-Min.

Figure 15: USA, 156 biggest cities, showing only the right half of the map.

4 Discussion

This note describes formalizations and algorithms for two cartographic tasks related to special-purpose maps. The appropriate formalization of the tasks is very important in this context for two different reasons. Firstly, the formal problem needs to be amenable to algorithmic design techniques to facilitate efficient algorithmic solutions. Secondly, these algorithmic solutions should yield high-quality output when applied real world data. Concerning diagram placement, we observed that different formalizations, which require different aspects to be optimized, lead to algorithms with running times that differ by one or two orders of magnitude. However, the formalizations that require more time-consuming solutions appear to produce better maps. Concerning proportional symbol maps, we observed that different formalizations make the difference between being solvable in polynomial time and being NP-hard. In our implementation we compared a few strategies for optimization in one model and again observed that the more time-consuming, optimal strategy appears to give slightly better maps. Hence, research on the best formalization and the most efficient algorithm to to solve the resulting problem it is worthwhile and relevant.

References

[1] P. K. Agarwal, A. Efrat, and M. Sharir. Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. *SIAM J. Comput.*, 29:912–953, 1999.

[2] J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Trans. Comput.*, C-28(9):643–647, September 1979.

[3] S. Cabello, H. Haverkort, M. van Kreveld, and B. Speckmann. Algorithmic aspects of proportional symbol maps. In *Proc. 14th Europ. Symp. on Algorithms*, number 4168 in Lect. Notes in Comp. Science, pages 720–731, Berlin, 2006. Springer.

[4] B. Chazelle and H. Edelsbrunner. An optimal algorithm for intersecting line segments in the plane. *J. ACM*, 39(1):1–54, 1992.

[5] *De Grote Bosatlas*. Wolters-Noordhoff, Groningen, 52nd edition, 2001.

[6] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, Berlin, Germany, 2nd edition, 2000.

[7] B. D. Dent. *Cartography: Thematic Map Design*. McGraw-Hill, 5th edition, 1999.

[8] D. Halperin. Arrangements. In J.E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 24, pages 529–562. Chapman & Hall/CRC, Boca Raton, 2nd edition, 2004.

[9] K. Mulmuley. *Computational Geometry: An Introduction Through Randomized Algorithms*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[10] G. Fejes Tóth. Packing and covering. In J.E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 2, pages 25–52. Chapman & Hall/CRC, Boca Raton, 2nd edition, 2004.

[11] M. van Kreveld, E. Schramm, and A. Wolff. Algorithms for the placement of square and pie charts on maps. In *GIS 2004: Proc. of the 12th ACM Sympos. on Advances in GIS*, pages 222–231, 2004.

[12] Queensland University Advanced Centre for Earthquake Studies. `http://www.quakes.uq.edu.au`.

[13] A. Wolff. The map-labeling bibliography, 1996. http://i11www.informatik.unikarlsruhe.de/map-labeling/bibliography.

Proving termination of rewriting automatically

Hans Zantema
Department of Computer Science, TU Eindhoven, P.O. Box 513,
5600 MB Eindhoven, The Netherlands, email: H.Zantema@tue.nl
and
Institute for Computing and Information Sciences, Radboud University
Nijmegen, P.O. Box 9010, 6500 GL Nijmegen, The Netherlands

Abstract

In this paper we give an introduction to term rewriting and termination. Moreover, we sketch some developments in proving termination of rewriting automatically, in particular in using satisfiability for finding suitable interpretations.

1 Term rewriting and termination

Term rewriting is a natural and basic framework to describe computations. As an example consider natural numbers being terms composed from the constant 0 and the unary symbol s. So the numbers $0, 1, 2, 3, \ldots$ are represented by

$$0,\ s(0),\ s(s(0)),\ s(s(s(0))),\ \ldots,$$

respectively. On these natural numbers we want to define the well-known binary operations $+$ and $*$ representing addition and multiplication. This means that if we apply $+$ or $*$ on natural numbers in their 0-s-representation, then we want to be able to compute the result, again being a natural number in 0-s-representation. A stepwise approach to reach this goal is the following. For every ground term (a term without variables) containing at least one of the symbols $+$ and $*$, we want that some computation rule is applicable replacing a term by another one having the same value. Then by repeatedly applying such computation rules we may hope to reach a term on which no computation rule is applicable. That is, a term purely composed from s and 0, so a natural number in our representation. Since the value does not change during the computation, this final term can be seen as the outcome of the computation.

The computation rules are called *rewrite rules*. A set of rewrite rules is called a *term rewriting system*, abbreviated to TRS.

As a first rewrite rule in the natural numbers example we choose

$$0 + x \ \to\ x,$$

expressing that 0 plus 'something' is always 'something'. More precisely, if somewhere in the term there is a +-symbol having 0 as its left argument, we may remove this + and 0. For the right argument of + there is no requirement, that's why we write the variable x in the left hand side $0 + x$ of the rule: for such a variable we may substitute anything we like.

The general pattern of a rewrite rule is $\ell \to r$, where ℓ is called the left hand side and r is called the right hand side of the rule. Both ℓ and and r are terms, possibly containing variables. A *substitution* is defined to be a map from variables to terms; for a term t and a substitution σ we write $t\sigma$ for the term obtained from t by replacing every variable x in t by $\sigma(x)$. The application of a rule is now as follows: if some subterm of a term can be written as $\ell\sigma$ for the left hand side of a rule $\ell \to r$ and a substitution σ, then this subterm may be replaced by $r\sigma$. If the original term is t and the resulting term is u, then we say there is a *rewrite step* from t to u, written as $t \to u$.

In this abstract setting it may look complicated, but it expresses our usual way of computation and only makes explicit what was left implicit in elementary school arithmetic. There we wrote =; here we prefer to write → to express the direction of the computation. Variables in rewrite rules mean that we may plug in what we like: this is expressed by applying substitutions. Moreover, rewrite rules are allowed to be applied on any subterm of the term we consider. For our natural numbers let's add three more rules, arriving at the TRS consisting of the following four rules:

$$\begin{array}{lrcl}
(1) & 0 + x & \to & x \\
(2) & s(x) + y & \to & s(x + y) \\
(3) & 0 * x & \to & 0 \\
(4) & s(x) * y & \to & y + (x * y)
\end{array}$$

The second rule states that if the left argument of + starts by s, this s can be moved outside the +. This does not change the value of the number, since $(x + 1) + y = (x + y) + 1$.

The third rule simply expresses that $0 * x = 0$: if the left argument of $*$ is 0 then the whole subterm may be replaced by 0. Finally, the fourth rule expresses what to do if the left argument is the s of something, using the property $(x + 1) * y = y + (x * y)$.

Using these rules we are able to compute $2 * 3 = 6$, where in the first step we indicate what to choose for $\sigma(x)$ and $\sigma(y)$ in rule 4:

$$s(\underbrace{s(0)}_{x}) * \underbrace{s(s(s(0)))}_{y}$$

$$\begin{array}{lll}
\to & s(s(s(0))) + (s(0) * s(s(s(0)))) & \text{using (4)} \\
\to & s(s(s(0))) + (s(s(s(0))) + 0 * s(s(s(0)))) & \text{using (4)} \\
\to & s(s(s(0))) + (s(s(s(0))) + 0) & \text{using (3)} \\
\to & s(s(s(0 + s(s(s(0 + 0)))))) & \text{using (2), 6 times} \\
\to & s(s(s(0 + s(s(s(0)))))) & \text{using (1)} \\
\to & s(s(s(s(s(s(0)))))) & \text{using (1)}
\end{array}$$

by which indeed the representation of 6 is obtained.

A term on which no rewriting is possible is called a *normal form.* Applying the rules of a TRS as long as possible is called *rewriting to normal form.* The above example shows that using our TRS indeed rewriting to normal form of the term representing $2 + 3$ yields the term representing 6. Much stronger, we can state the following:

> Rewriting to normal form of any ground term composed from $0, s, +, *$ always yields the desired result: $s^n(0)$ for n being the value of the initial ground term when $0, s, +, *$ are given their usual meaning.

Why is this the case? Validity of this claim follows from the following three properties:

1. All rules are *sound*: the meaning of the left hand side is equal to the meaning of the right hand side for every valuation of the variables. As a consequence, by rewriting the meaning does not change.

2. For every ground term containing $*$ or $+$ a rule is applicable. So if no rule is applicable then the term only consists of 0 and s, and is of the desired shape $s^n(0)$.

3. No *infinite* computations are possible.

This last property is called *termination*, and that's what we want to prove automatically. More precisely: a TRS is called *terminating* if no infinite sequence of terms $t_1, t_2, t_3, \ldots$ exists such that $t_i \to t_{i+1}$ for every $i = 1, 2, 3, \ldots$.

These observations are applicable in several other areas, including lists, trees, binary encoded numbers and process algebra. As soon as we have a data structure (in our case the natural numbers) of which the elements can be represented uniquely as ground terms over some *constructor symbols* (in our case 0 and s), and we have some *defined symbols* (in our case $+$ and $*$), together with a semantics over constructor ground terms, and we have a TRS such that

1. all rules are sound,

2. for every ground term containing a defined symbol a rule is applicable, and

3. the TRS is terminating,

then rewriting any ground term to normal form always yields the unique constructor normal form having the same semantics.

Let's investigate these three properties in some more detail. Property 1 is by construction: in designing a TRS we only choose rules that replace a term by another term having the same meaning. Also Property 2 is by construction: we design as many rules that for any term which is not yet of the right shape at least one rule is applicable. We conclude that termination is the only required property in this setting that does not hold by construction.

These observations give a very rough sketch of a particular view on term rewriting, and the corresponding desire to be able to prove termination of TRSs. In the rest of this paper we give an overview of some developments in automatically proving termination of a TRS.

2 Proving termination of a TRS

In the past decades several techniques have been developed to prove termination of TRSs. Roughly speaking we can distinguish the following three families of techniques:

- *Semantical methods*: interpret in some well-founded domain (like natural numbers) in which value decreases at every rewrite step.
- *Syntactical methods* based on syntactically defined reduction orders: if $\ell > r$ for all rules $\ell \to r$ for such an order $>$, then the system is terminating.
- *Transformational methods*: transform the TRS to another one in such a way that termination of the transformed system implies termination of the original system. This is not a full method in itself; the goal is to find such a transformation such that basic semantical or syntactic methods apply on the transformed TRS.

In the eighties most emphasis was on syntactical methods started by *path orders* [4, 8, 5]: in that time proving termination was practically seen to coincide with finding the right order. The basic idea of semantical methods is even older [9]; the semantical approach specified to using polynomials was worked out in [3]. Transformational methods include the transformation ordering [2] and semantic labelling [11], but by far the most powerful transformational method is the *dependency pairs* method [1].

In the late nineties the emphasis in this research shifted towards automation: for a new technique the final goal was not any more to use it by hand for proving termination of a particular system, but to implement it in such a way that termination proofs could be found fully automatically. Since around 2000 several tools were developed for this goal. In 2003 the idea came up to organize a competition on these tool by developing an extensive set of termination problems called TPDB (termination problem data base), and run the tools on them and compare the results. The main objectives for such a competition were and are:

- stimulate research in this area, shifting emphasis towards automation, and
- provide a standard to compare termination techniques.

Since then the competition was held every year. Some characteristics of the 2007 competition are:

- 12 participating tools,
- over 2000 rewrite systems, logic and functional programs,

- subdivided in 7 categories, including certified proofs,
- time limit: 2 minutes per tool per system,
- total running time: around one week,
- strongest tool: AProVE (Giesl et al, Aachen) [7], written in JAVA.

More details on the termination competition including past editions, rules and all results are found on

`http://www.lri.fr/~marche/termination-competition/`

During the years of the competition strong improvements have been made: every year there were several TRSs for which termination could be proved fully automatically while the year before none of the tools was able to do so.

As reasons for recent improvements we mention:

- Encode search in large search spaces into propositional *SATisfiability problem*, and call state-of-the-art SAT solver to solve it.

  This approach was introduced since 2006 in several tools for several search problems, while in 2005 no tool had it.
- Improvements in the dependency pair approach, in particular to split up large problems into smaller ones.
- Particular new techniques, like the *matrix method* [6].

Due to limited space we will not elaborate on all of these issues, but focus on the matrix method and the way how SAT solving is used to find a corresponding proof. As a preparation for the matrix method first we present polynomial interpretations.

3 Polynomial interpretations

This is one of the most basic instances of a semantical method to prove termination. Choose an interpretation of every symbol by a strictly monotone polynomial over the naturals such that by every rewrite rule the interpretation strictly decreases. Then we may conclude termination. This is an instance of *monotone algebras*; for details we refer to [10, 12]. As an example we recall our TRS describing natural number computation:

$$\begin{array}{rcl} 0+x & \rightarrow & x \\ s(x)+y & \rightarrow & s(x+y) \\ 0*x & \rightarrow & 0 \\ s(x)*y & \rightarrow & y+(x*y) \end{array}$$

Termination of this TRS is proved by the following polynomial interpretation $[\cdot]$:

$$[0]=1,\ [s]x=x+1,\ x[+]y=2x+y,\ x[*]y=3xy$$

Indeed then we have

$$\begin{array}{rcl}
[0][+]x = 2 + x & > & x \\
([s]x)[+]y = 2(x+1) + y & > & 2x + y + 1 = [s](x[+]y) \\
[0][*]x = 3x & > & 1 = [0] \\
([s]x)[*]y = 3(x+1) * y & > & 2y + 3xy = y[+](x[*]y)
\end{array}$$

for all $x, y > 0$, proving termination.

For finding such interpretations automatically the old approach was: check the requirements for all (or a great number of randomly chosen) interpretations for which typically

- every constant is either 1, 2 or 8,
- every unary symbol is either identity, successor, or multiply by 2,
- every binary symbol is either addition, multiplication, or $\cdots$

Important: per symbol only a few options should be allowed, otherwise the search space is intractable.

Based on SAT, there is an alternative approach for finding such interpretations, where first we restrict to linear polynomials:

- choose for constants: A
- choose for unary symbols $Ax + B$
- choose for binary symbols $Ax + By + C$
- $\cdots$

where A, B, C are unknown numbers represented in binary notation in n bits. The idea now is to transform the requirements on these n-tuples of bits to a satisfiability problem on these bits, and then apply a SAT solver on the result. If this is successful then the resulting satisfying assignment is transformed back to the desired values $A, B, C, \ldots$.

For this we need to be able to represent basic binary arithmetic using $+, *, >$ in propositional logic. This can be done straightforwardly. For instance, the requirement $a + b = d$ can be expressed simply by remembering how we learned addition in elementary school. As an example consider the addition $7 + 21 = 28$ in binary notation:

$c \rightarrow$	0	0	1	1	1	0
$a = 7 \rightarrow$		0	0	1	1	1
$b = 21 \rightarrow$		1	0	1	0	1
$d = 28 \rightarrow$		1	1	1	0	0

Here we have boolean vectors to represent a, b, d and an auxiliary vector of boolean values c_i to represent the carries. The addition is correct if and only if $a_i \leftrightarrow b_i \leftrightarrow c_i \leftrightarrow d_i$ and $c_{i-1} \leftrightarrow ((a_i \wedge b_i) \vee (a_i \wedge c_i) \vee (b_i \wedge c_i))$ for $i = 1, \ldots, n$, and $\neg c_0 \wedge \neg c_n$.

Multiplication can be expressed by repeated addition and duplication. One way to do this is by the following program

```
$r := 0$;
for $i := 1$ to $n$ do
        begin
        $s := 2 * r$;
        if $b_i$ then $r := s + a$ else $r := s$
        end
```

Executing this program yields $r = a * b$. All ingredients of this program are easily expressed in propositional logic, by introducing several fresh variables for representing intermediate values for r, s. So also multiplication can be expressed in propositional logic.

Finally checking $>$ on binary numbers corresponds to lexicographic comparison which is also easily expressed in propositional logic.

So all ingredients of polynomial interpretations transform to propositional logic. The final formula is a conjunction of several small formulas, on which a SAT solver is applied. If the formula is found to be satisfiable, then the corresponding satisfying assignment represents the corresponding values of the coefficients of the polynomials. In this way for many TRSs a corresponding termination proof can be found.

This approach can be extended to non-linear polynomials. Comparison of linear polynomials can be done in a straightforward way; for comparison of non-linear polynomials an approximation is required.

4 The matrix method

This is a next instance of a semantical method. Essentially it follows the same idea as polynomials, except that now the interpretations are in *vectors* over natural numbers rather than in natural numbers.

As the well-founded order on such vectors we choose:

$$(v_1, \ldots, v_d) > (u_1, \ldots, u_d) \iff v_1 > u_1 \wedge v_i \geq u_i \text{ for } i = 2, 3, \ldots, d$$

As the interpretation for a symbol of arity k we choose:

$$[f](\vec{x}_1, \ldots, \vec{x}_k) \; = \; A_1\vec{x}_1 + \cdots + A_k\vec{x}_k + \vec{v}$$

where $A_1, \ldots, A_k$ are $d \times d$ matrices and $\vec{v}$ is a vector, all over n-bits integers. It turns out that the interpretation of a left hand side or a right hand side of a rule can be expressed by repeated matrix multiplication. All ingredients of matrix multiplication and vector comparison can be expressed in the basic arithmetic $+, *, >$ for binary natural numbers, reusing the implementation discussed in the last section. In this way also the search for a valid matrix interpretation can be expressed as a SAT problem. Typically we choose $d \approx 3$ and $n \approx 3$. For small rewrite systems this easily yields formulas in 10,000 variables and over 100,000 clauses, but for SAT solvers like ZChaff, Minisat, SatElite this is no problem.

We conclude by an example of a termination proof in this style of the TRS consisting of the single rule

$$h(g(s(x),y),g(z,u)) \to h(g(u,s(z)),g(s(y),x))$$

for which other techniques fail to prove termination. The interpretation found in the way described above satisfying

$$[\text{left hand side}] \;>\; [\text{right hand side}]$$

and hence proving termination is the following:

$$\begin{aligned}
[h](\vec{x_0},\vec{x_1}) &= \begin{pmatrix} 3 & 1 \\ 1 & 0 \end{pmatrix} \cdot \vec{x_0} + \begin{pmatrix} 1 & 3 \\ 0 & 1 \end{pmatrix} \cdot \vec{x_1} + \begin{pmatrix} 0 \\ 2 \end{pmatrix} \\
[g](\vec{x_0},\vec{x_1}) &= \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \cdot \vec{x_0} + \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \cdot \vec{x_1} \\
[s](\vec{x_0}) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cdot \vec{x_0} + \begin{pmatrix} 0 \\ 2 \end{pmatrix}
\end{aligned}$$

5 Concluding remarks

Typical proofs found by SAT solving (like matrix interpretations)

- are easy to check by hand, but
- do not provide any understanding or intuition why the proved property holds

Computer found proofs seem to differ from human found proofs just like computer chess differs from human chess: the reasoning is guided by brute force rather than by intuition.

To reach the full power of the present termination tools it is essential to combine these techniques with lots of other techniques. In this way long and complicated termination proofs are found fully automatically. Nevertheless, there are still very small TRSs for which termination is open: all tools fail to prove termination or non-termination, and also no human proof is known. An example of this shape given by Johannes Waldmann is the TRS consisting of the two rules

$$a(a(b(x))) \to b(a(b(a(a(x))))), \quad b(b(x)) \to x.$$

References

[1] T. Arts and J. Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236:133–178, 2000.

[2] F. Bellegarde and P. Lescanne. Termination by completion. *Applicable Algebra in Engineering, Communication and Computing*, 1(2):79–96, 1990.

[3] A. Ben-Cherifa and P. Lescanne. Termination of rewriting systems by polynomial interpretations and its implementation. *Science of Computing Programming*, 9(2):137–159, 1987.

[4] N. Dershowitz. A note on simplification orderings. *Information Processing Letters*, 9(5):212–215, 1979.

[5] N. Dershowitz. Orderings for term rewriting systems. *Theoretical Computer Science*, 17(3):279–301, 1982.

[6] J. Endrullis, J. Waldmann, and H. Zantema. Matrix interpretations for proving termination of term rewriting. In *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR '06)*, volume 4130 of *Lecture Notes in Computer Science*, pages 574–588. Springer, 2006.

[7] J. Giesl et al. Automated program verification environment (AProVE). Available at `http://aprove.informatik.rwth-aachen.de/`.

[8] S. Kamin and J. J. Lévy. Two generalizations of the recursive path ordering. University of Illinois, 1980.

[9] Z. Manna and S. Ness. On the termination of Markov algorithms. In *Proceedings of the Third Hawaii International Conference on System Science*, pages 789–792, Honolulu, 1970.

[10] H. Zantema. Termination of term rewriting: interpretation and type elimination. *Journal of Symbolic Computation*, 17:23–50, 1994.

[11] H. Zantema. Termination of term rewriting by semantic labelling. *Fundamenta Informaticae*, 24:89–105, 1995.

[12] H. Zantema. Termination. In *Term Rewriting Systems, by Terese*, pages 181–259. Cambridge University Press, 2003.

Formal Analysis Techniques for Gossiping Protocols*

Rena Bakhshi
Vrije Universiteit Amsterdam
Amsterdam, Netherlands
rbakhshi@few.vu.nl

François Bonnet
ENS Cachan/IRISA
Rennes, France
fbonnet@irisa.fr

Wan Fokkink
Vrije Universiteit Amsterdam/CWI
Amsterdam, Netherlands
wanf@cs.vu.nl

Boudewijn Haverkort
University of Twente
Enschede, Netherlands
brh@cs.utwente.nl

Abstract

We give a survey of formal verification techniques that can be used to corroborate existing experimental results for gossiping protocols in a rigorous manner. We present properties of interest for gossiping protocols and discuss how various formal evaluation techniques can be employed to predict them.

1 Introduction

The emergence of the Internet as a computing platform asks for new classes of algorithms that combine massive distributed processing and inherent decentralization. These algorithms should be able to execute in an environment that is heterogeneous, changes almost continuously, and consists of millions of nodes. Massive parallel computing on the Internet also demands a degree of self-organization; the amount of devices and software is simply too large to be managed by humans.

Gossiping protocols have shown to be a sensible paradigm for developing stable and reliable communication mechanisms that scale up to massively parallel environments. In a gossiping (also called epidemic) protocol, nodes exchange data similar to the way a contagious disease spreads. That is, a participating peer can select, according to some probability distribution, other peers to exchange information with. Gossiping protocols were originally applied in database replication [26], but more recently also for failure detection [70], and resource monitoring [69]. They are employed in wired as well as wireless environments. In a massively parallel setting, the gossip mechanism should be used at very high speeds, yielding a new generation of protocols that have an unusual style of probabilistic reliability guarantees, regarding scalability, performance, and stability of throughput. Surveys [30, 32, 47] provide an introduction to the field.

Gossiping protocols tend to contain several design parameters that can influence the non-functional properties of these protocols, e.g., performance, robustness and fault tolerance. Values of these parameters are usually determined empirically, without a proper understanding why the protocol performs well for these values, and without any certainty that these values are close to optimal or robust choices. Thorough experimental analysis in [49] has shown that the emergent behaviour of gossiping protocols may vary substantially by changing only a few design parameters.

When a large number of programs interact in a con-

*Original publication appeared in: *ACM SIGOPS Operating Systems Review*, volume 41, Issue 5, October, 2007, pages 28–36 nected environment, various phenomena occur that are not explicable in terms of the behaviour of any single agent. It is necessary to understand these phenomena in order to keep the overall systems both stable and efficient. Distributed algorithms and protocols that run steadily and reliably in small-scale settings, tend to lose those properties as numbers of users, the size of the network and transaction processing rates all increase. Typical problems are disruptive overloads and routing changes, periods of poor connectivity and throughput instability. Failures rise in frequency simply because the numbers of participating components are larger [71].

In practice, properties of gossiping protocols are usually diagnosed by emulating such systems. However, in principle, owing to their often relatively simple structure, gossiping protocols lend themselves very well to formal analysis, in order to predict their behaviour with high confidence. A complication in the analysis of gossiping protocols is that they are meant to work in very large networks, and for ad hoc wireless networks even with lossy channels. In this paper we give an overview of the different approaches that can be taken to formally analyse gossiping protocols, and which properties of such protocols can be verified with which formal verification techniques.

This paper is structured as follows. Section 2 gives an overview of the different types of requirements for gossiping protocols. Section 3 presents the available spectrum of analysis techniques. Finally, Section 4 contains some conclusions.

2 Requirements

Requirements for gossiping protocols can be divided into three classes: general, functional and non-functional requirements. These will be discussed in the current section. We use terminology from [49, 72].

2.1 General Requirements

Gossiping protocols satisfy the following general requirements:

- *Simplicity:* The protocol is simple and easy to deploy. For example, in a wireless network, a node should be able to join the system without executing a complex procedure ("plug-and-play").
- *Scalability:* Each node continues to perform its operations at almost the same rate irrespective of the network size. For example, the local knowledge (neighbours list) of a node does not increase with the network size.
- *Symmetry:* In a large-scale network, all nodes play identical roles. Hence, there is no single point of failure. Randomization, e.g., random peer selection, tends to fit into this requirement, because each node typically runs the same algorithm.

2.2 Functional Requirements

Functional requirements describe properties of the outputs of a system, and how a certain input is transformed into an output. We classify several functional requirements on gossiping protocols. We distinguish between global and local properties.

1. Global properties of the system:
    - *Connectivity:* This can, for example, be expressed as a minimum number of links between nodes, whose removal will result in the partitioning of the network graph. The connectivity of a graph is an important measure of its robustness, because partitioning of a network graph creates difficulties for information dissemination.
    - *Convergence:* One can distinguish between convergence of the system parameters to some values (e.g., achieving a certain accuracy in the estimates of the aggregate function values) and convergence of the system structure to some particular type of graph.
2. Local properties of nodes:
    - *Degree distribution:* The degree of a node is the number of its neighbours in the network graph. This concept is interesting because of its relationship to robustness of a graph

in the presence of node failures, its effect on patterns of epidemic spread, and its importance in the distribution of resource usage of nodes.

- *Clustering coefficient:* The clustering coefficient of a node expresses a ratio of the number of links between the node's neighbours to the number of all possible links between them. Intuitively, it shows how many neighbours of a node are neighbours among themselves. Analysis of this property is interesting because a high clustering coefficient affects information dissemination, as the number of redundant messages increases. Also, it affects the self-healing capacity, by strengthening the connectivity within a cluster, thus decreasing the probability of partitioning.
- *Shortest path length:* The shortest path length between two nodes is the minimum number of edges that must be traversed to go from one node to the other. The average path length is the average of all shortest path lengths between any two nodes in a graph. The shortest and average path length give information on the time and communication costs to reach a node.

2.3 Non-Functional Requirements

Non-functional requirements regard the quality (e.g., performance, maintainability, fault-tolerance) and economics (e.g., timing, cost) of system behaviour. The following high-level non-functional requirements can be identified for gossiping protocols:

- *Time Complexity:* The number of time units it takes (at worst or on average) for a gossiping protocol to "infect" every node in the network, e.g., for data delivery to all nodes, or for computation of an aggregation function output.
- *Message Complexity:* The total number of gossiping messages (at worst or on average) exchanged over the network during an execution.
- *Information dissemination:* There should be a high probability that a piece of information is shared with all processes within a given time.
- *Robustness:* The ability of a gossiping protocol to maintain correct system operation in the face of massive node crashes and node churn.
- *Graceful degradation:* Large numbers of node failures in the system may affect its operation. However, performance, functionality and reliability of gossiping protocols should not drop rapidly as the number of failures increases.
- *Elasticity:* Robustness of the well-operation of the systems in face of largely varying node capabilities in terms of memory, bandwidth and connectivity.
- *Self-organization:* The nodes should be able to organize themselves in unpredictable circumstances without external interventions. For example, in gossiping protocols a network graph forms overlays that are adaptable to network and environmental changes.

Specifically in wireless networks, nodes communicate through error-prone radio channels and typically also have limited computational capabilities. Special design issues then include energy use, mobility, transmission power, memory usage and latency. Network properties such as message reliability and node reachability may in that case influence the behaviour of the protocol.

3 Formal Analysis Techniques

The aims when analysing a system are in general to obtain a better understanding of and gain further confidence in the system's behaviour, to detect possible errors, and to improve its design. A complication in the analysis of gossiping protocols is that they are meant to work in very large networks. Properties of such protocols are generally diagnosed by emulating such networks.

The formal specification of systems helps to make explicit the underlying assumptions (like the synchronization primitives), which tend to remain hidden in implementations or simulation exercises. Also such a specification can be analysed using (semi-)automated formal verification techniques. There is a rich history of the use of such techniques for verifying a variety of desirable properties for a wide range of systems. The efficiency of a particular formal analysis technique depends on the system under study.

Gossiping protocols in general contain several design parameters that heavily influence their behaviour. For example, the number of protocol cycles, message forwarding strategies, message delays, or cache usage and size. Formal analysis techniques can help in the search for optimal values of such parameters.

Rigorous formal analysis techniques for gossiping protocols have so far hardly been applied in large-scale settings, and need to be developed further for this purpose. The main aim of this paper is to investigate which formal analysis techniques can in principle be employed efficiently in the analysis of gossiping protocols, for wired as well as wireless systems. We will provide an overview of the existing analysis methods, their use and limitations, as well as a comparison of related work on the formal verification of gossiping protocols. Our aim is to create a better understanding for selecting a suitable approach for such an analysis.

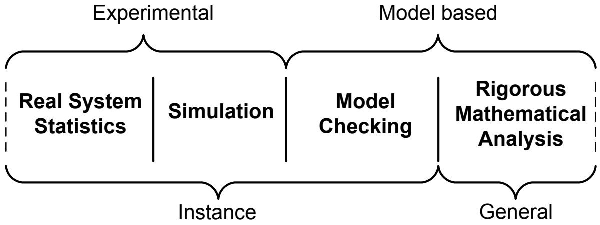


Figure 1: Spectrum of validation

Figure 1 depicts the spectrum of analysis techniques, ranging from experimental work with a real system implementation up to rigorous mathematical analysis. Real system statistics and simulation techniques are based on experiments performed on the system and on collecting data statistics either from the running system through monitoring it at real time or from a discrete-event simulation of the system. Usually, these approaches are used to study the behaviour of a particular implementation (instance) of the system. The other approaches require a formal modelling of the system. Typically, they are used to verify specific properties in a more general context. Although these methods often require particular assumptions to be made, their advantage is that they can be used before a system is being implemented, and that in principle they are not costly (in comparison to full-scale experiments on a real system).

In the following subsections, we present pros and cons of the use of experimental and model-based formal analysis techniques for gossiping protocols.

3.1 Experimental Evaluation

In practice, properties of gossiping protocols are often diagnosed by emulation, and through performing simulations (see, e.g., [36]). Commonly used discrete-event simulators are ns-2, Opnet and Glomosim; but often a customized simulator is built in for example Java or Matlab.

Experimentation is the major source of gaining first insight. The reason is that in reality performance of gossiping protocols depends on many factors: characteristics of the network, certain distributions (capacity, node-degree, etc), usage scenarios, user models, incentives, etc.

However, different simulators can produce vastly different results, even for simple systems (see, e.g., [18]). The reason is that simulators employ different models for the medium access control and physical layers. The results of the simulators say as much about the simulated system as about the particular lower-level implementation of the simulator. Also the employed random number generators have an (unpredictable) impact; for instance, [2] questions the credibility of this type of simulations. Moreover, different simulation analyses of gossiping protocols make different assumptions about the underlying model, which makes comparison of results difficult. For example, [18] and [19] both evaluate the flooding protocol, but sending and receiving is perfectly synchronized in [18], while [19] assumes a random waiting period between sending and receiving.

Surprisingly, few attempts have been made to implement systems that use one of the existing gossiping protocols; we are only aware of Astrolabe [69], Tribler [64] and ARRG [28]. Notably, in [28], it is shown that a gossiping protocol that behaves well in an emulated environment, may not behave well when truly implemented, especially in an environment where nodes can crash.

3.2 Model-Based Analysis Techniques

The formal specification of a system helps to obtain not only a better (more modular) description, but also a clear understanding and an abstract view of the system. Formal analysis techniques, typically referred to as formal verification, are supported by (semi-)automated tools. They can detect errors in the design that are not so easily found using emulation or testing, and can be used to establish the correctness of the design. The most effective way to apply formal methods is actually during the design of a system, rather than after-the-fact, as is, unfortunately, often the case.

Formal models need to be realistic. An experimental evaluation can help to obtain a first insight in the behaviour of a system, and to identify which characteristics need to be included into the model.

There are two main approaches to formal verification. The first approach involves a rigorous mathematical analysis of the properties of the system, using results from for instance calculus and probability theory. Such an analysis can be supported semi-automatically by means of Matlab or a theorem prover. While Matlab is an easy to use but imprecise mathematical tool, a theorem prover requires a lot of effort from the user but supports precise mathematical reasoning about the system. Important theorem proving tools are Isabelle/HOL [61], PVS [63] and Coq [11].

The second approach is model checking, which consists of a systematic and fully automatic exploration of the state-space of the system specification. The explorative nature of the approach in principle requires that the state space is finite. However, recent work also addresses symbolic model checking techniques for infinite-state models.

3.2.1 Rigorous Mathematical Analysis

Rigorous analysis techniques for gossiping protocols are built on sound mathematical foundations, and draw inspiration from the mathematical theory of epidemics [30, 6]. These analysis techniques are in general used to verify specific properties of a gossiping protocol. Therefore, such a study is usually done on a simplified system model of the actual protocol: one has to decide which characteristics of the protocol should be studied (see Section 2), and which parameters of the protocol should be modelled in order to study these characteristics.

Gossiping protocols are intrinsically probabilistic. For instance, a node may randomly selects a “gossip” partner or a data item for the exchange with another node. Or it may be the case that when a node receives a message, then with some probability p it forwards the message to all (or some) of its neighbours, while with probability $1 - p$ the message is purged. A key property is that if the probability p is sufficiently large, and the network sufficiently dense, then the probability of successful information spread remains close to 1, while the number of sent messages is relatively small compared to flooding.

Thus, the mathematical foundations underlying the modelling and verification of gossiping protocols can be found in probability theory.

Hand-crafted Markov chains

Markov chains can be used for modelling a variety of aspects of gossiping protocols. Markov chains allow to capture the evolution of gossip-based systems; each state of the Markov chain describes one state of the system. However, a state of the Markov chain does not represent a state of the whole system, but only a state of the system limited to the list of parameters that are modelled. From one state of the Markov chain to another, there are probabilistic transitions corresponding to the probabilistic evolution of the system. There are two types of Markov chain, distinguished by the transitions occurring at any time (continuous-time Markov chain) or only for defined steps (discrete-time Markov chain). By analysing the different possible sequences (and their probabilities), it is possible to obtain a global insight in the operation of the protocol. The Markov chain describing the system evolution converges to a useful distribution over all possible system states, from which interesting protocol properties can be concluded. For more information, we refer to [42, 66].

To exemplify the approach in the context of gossiping protocols, we provide the description of two case studies: the first one concerns the degree distribution of nodes, and the second the connectivity of network overlays. In both examples, the results from the mathematical analysis have been compared to the results of simulations to confirm their validity.

Bonnet [13] studied the evolution of the in-degree distribution of nodes during the execution of the Cyclon protocol [73]. Markov chains model this distribution, that is, the probability of being in state i of the Markov chain equals the probability for a given node to have i in-edges. From the designed Markov chain it is possible to determine the stationary distribution of the in-degrees, i.e., the distribution to which the protocol converges, as well as to calculate bounds on convergence time.

Allavena *et al.* [1] proposed a scalable gossip-based algorithm for local view maintenance. They counted the number of links between two parts of the system (say A and B) and studied the evolution of these numbers; the states of the associated Markov chain are the numbers of links from A to B and from B to A. From the designed Markov chain the expected time until a network partition occurs was calculated. This case study also included a model of the system under churn.

As other examples of the use of Markov chains for gossiping and related protocols, we refer to studies on gossip-based membership management [1, 7, 13], gossip-based distributed aggregation [15, 16, 60, 27, 53], gossip-based information dissemination [23, 29] and network topology change [34]. Schnoebelen [67] surveyed several proposals for modelling probabilistic lossy channel systems with Markov chains and the verification techniques they support. It would be interesting to see whether these ideas can be of use in both quantitative and qualitative analyses of gossiping protocols in wireless networks. Probability theory has further been applied to analyse gossip-based information dissemination [35, 50, 12, 58] and gossip-based resource location [54, 55].

Mathematical analysis can be combined with simulations to validate the results and understand the system behaviour. A strong point of mathematical analysis is that often it scales well with respect to the size of a network. However, it requires a lot of effort, it can only be used to analyse a limited class of properties, and the assumptions that are invariably made to simplify the analysis often affect the accuracy of the results [14]. For example, the analyses in [50, 55, 54] rely on the assumption of full knowledge of group membership, ignoring its practical infeasibility.

3.2.2 Model Checking

Model checking is an exhaustive state space exploration technique that is used to validate formally specified system requirements with respect to a formal system description [21]. Such a system is verified for a fixed configuration; so in most cases, no general system correctness can be obtained.

Using some high-level formal modelling language, automatically an underlying state space can be derived, be it implicitly or symbolically. The system requirements are specified using some logical language, like LTL, CTL or extensions thereof [48]. Well-known and widely applied model checking tools are SPIN [46], Uppaal [8] (for timed systems), and PRISM [45] (for probabilistic systems). The system specification language can, e.g., be based on process algebra, automata or Petri nets.

Model checking suffers from the so-called state explosion problem, meaning that the state space of a specified system grows exponentially with respect to its number of components. The main challenge for model checking lies in modelling large-scale dynamic systems. To overcome the state explosion problem and to speed up the verification process, various state space reduction techniques have been proposed. For instance, combinations of symbolic verification techniques with explicit state space exploration (symbolic model checking), verification of properties on a smaller abstract model of the system under scrutiny, possibly obtained after bisimulation reduction, parallelisation of verification algorithms, partial exploration of the state space (bounded model checking, on-the-fly model checking, partial order reduction), and efficient state representation (bitstate hashing), have been proposed to make model checking practically feasible.

Initial model checking approaches used as underlying mathematical model a finite-state automaton, i.e., a model with neither explicit time nor probabilities. Recently, model checking techniques have been proposed for system models including both time and probabilities, possibly in combination with nondeterminism. In view of the probabilistic features in gossiping protocols, we focus on model checking of probabilistic models.

Probabilistic and Stochastic Model Checking

In probabilistic and stochastic model checking, the underlying system model is not represented by an automaton, but instead as a stochastic process of some sort, mostly a finite-state Markov chain (discrete- or continuous-time). Often, these Markov chains are extended with state labels and transition labels (so-called action names). These Markov chains are mostly specified using some high-level formalism, like stochastic process algebra [20] or stochastic Petri nets.

Gossiping protocols may require models which, in addition to pure probabilistic choices, also allow for non-deterministic choices. That is, it is possible in a given state of a system to non-deterministically move to another state with some probability. Here, Markov decision processes can be applied [65]. The key idea to a Markov decision process is to allow a set of probability distributions in each state instead of a single distribution as in Markov chains. The choice between these distributions is made externally and non-deterministically, either by a scheduler that decides which sequential subprocess takes the next step (as in e.g., concurrent Markov chains), or by an adversary that influences or affects the system. Probabilistic choices are internal to the process and made according to the selected distribution.

The system requirements of interest are again specified through logical expressions, over the paths that can be taken through the model. For that purpose, the logics are extended to include a notion of time and probability. Prominent examples of such logics are CSL [5, 4], CSRL [41] and asCSL [3] for continuous-time models, and pCTL [40] for discrete-time models.

Where traditional model checking algorithms lean heavily on determining reachability of certain states or state groups (or the non-reachability), in probabilistic and stochastic model checking also the time until some states are reached (or avoided) plays a major role. Furthermore, reachability of states is expressed in terms of a probability (no mutually exclusive yes or no) and a time-bound; as an example, certain states might be highly probably reached for short time periods, but not for longer time periods. Stochastic model checking relies on algorithms for reachability analysis, as well as on numerical algorithms for determining long-term and transient behaviour in Markov chains. Although such algorithms are well understood, their implementation requires care, especially if very large models are to be addressed.

Stochastic and probabilistic model checking has been applied in a wide variety of case studies, ranging from workstation cluster availability [43] to the evaluation of power-saving methods [62] and the analysis of wireless (sensor) networks [59].

We feel that the success of model checking approaches, and especially stochastic and probabilistic model checking approaches, to a wide variety of case studies, is promising. This observation is also fuelled by the fact that gossiping protocols with their probabilistic and asynchronous behaviour fit well to the model classes supported by the known model checking techniques. This is not to say that we do not expect difficulties. On the contrary, the key to successfully verifying gossip-based systems lies in coping with their scale. This implies that an analysis or verification technique should be able to deal with large networks somehow, be it through smart abstractions or approximations (thus avoiding large state spaces), or through smart storage techniques or brute force distributed verification algorithms.

Some of the optimization techniques for general modelling techniques, as described earlier, have been adapted to probabilistic model checking, in particular: abstraction [52, 56, 57, 17], distribution [9, 10, 38] and Markovian bisimulation [51].

Approximate and Statistical Probabilistic Model Checking

An alternative approach to cope with state explosion for probabilistic systems is found in approximate probabilistic model checking. The main idea of this approach is to apply Monte-Carlo sampling techniques [39, 33]; the resulting probabilities are accurate only with respect to some accuracy criterion.

Approximate probabilistic model checking [44, 38] is an approximation method for the logic restricted to time-bounded safety properties ("positive" LTL). Monte Carlo model checking [37] is based on a randomized algorithm for probabilistic model checking of safety properties for general LTL model checking; Monte Carlo model checking uses the optimal approximation algorithm of [22].

In so-called statistical probabilistic methods (e.g., [74]), statistical hypothesis testing is used instead of randomized approximation schemes. The approach of [75] describes a model-independent procedure for verifying properties of discrete-event systems based on Monte-Carlo simulation and statistical hypothesis testing. The procedure uses a refinement technique to build statistical tests for the satisfaction probability of CSL formulas. The statistical method of [68] concentrates on model checking of black-box probabilistic systems against specifications given in a sublogic of CSL.

Similar to the idea of approximation-based probabilistic model checking, [31] combines probabilistic model checking with Monte Carlo simulations for the performance analysis of probabilistic broadcast protocols in a wireless network. In particular, this study shows results for reliability and reachability properties under different assumptions, such as message collision, lossy channels and unreliable timing, and their impact on the results.

Case study [24] presents the modelling of a sensor network using approximate probabilistic model checking. Another case study [17] presents the results of an analysis of the MAC protocol for sensor networks using approximate probabilistic model checking. eXtended Reactive Modules (XRM) [25] have been proposed for modelling wireless sensor networks to generate RM models suitable for PRISM and approximate probabilistic model checking.

4 Conclusions

Concluding, the formal analysis of gossiping protocols is a rather unexplored research field, with many challenges and open problems ahead. A more insightful and systematic methodology should be developed, that targets gossiping protocols. The assumptions made for simplifying such an analysis should be restricted as much as possible, as otherwise the analysis itself becomes unrealistic.

Markov chains give a precise mathematical description, but the analysis is time-consuming and can only be used for a restricted class of properties. It would be worthwhile to use theorem proving tools in order to support such a mathematical analysis.

Probabilistic model checking techniques are convenient to use, as they are based on verification algorithms. But formally modelling a gossiping protocol still requires considerable effort, and can introduce mistakes by itself (if the model deviates from the actual protocol). Also the verification algorithms are very much under development, and probabilistic model checking is, even more than standard model checking, suffering from the state explosion problem. This complicates the analysis of gossiping protocols considerably, as they are supposed to be applied in large-scale networks. The use of optimisation techniques, like abstraction and distributed verification, will form important ingredients for model checking to become practically of interest for the evaluation of gossiping protocols.

Approximate probabilistic model checking could serve as a good compromise between probabilistic model checking and simulation. They do not provide an exhaustive search to verify a given property, and as a result they do not suffer from the state explosion problem that much. Still in practice they can provide rather accurate probabilistic estimates. But approximate probabilistic model checking is still coming off age, and needs to be further developed.

References

[1] A. Allavena, A. Demers, and J. E. Hopcroft. Correctness of a gossip based membership protocol. In *Proc. 24th Annu. ACM Symp. on Principles of Distributed Computing (PODC'05)*, pages 292–301. ACM Press, 2005.

[2] T. Andel and A. Yasinsac. On the credibility of MANET simulations. *IEEE Computer*, 37(7):48–54, 2006.

[3] C. Baier, L. Cloth, B. R. Haverkort, M. Kuntz, and M. Siegle. Model checking markov chains with actions and state labels. *IEEE Trans. Software Eng.*, 33(4):209–224, 2007.

[4] C. Baier, B. R. Haverkort, H. Hermanns, and J.-P. Katoen. Model Checking Continuous-Time Markov Chains by Transient Analysis. In *Proc. 12th Int. Conf. on Computer Aided Verification (CAV'00)*, volume 1855 of *LNCS*, pages 358–372. Springer, 2000.

[5] C. Baier, B. R. Haverkort, H. Hermanns, and J.-P. Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE Trans. Software Eng.*, 29(6):524–541, 2003.

[6] N. T. Bailey. *Mathematical Theory of Infectious Diseases and Its Applications*. Griffin, London, second edition, 1975.

[7] Z. Bar-Yossef, R. Friedman, and G. Kliot. RaWMS: random walk based lightweight membership service for wireless ad hoc network. In *Proc. 7th ACM Int. Symp. on Mobile ad hoc networking and computing (MobiHoc'06)*, pages 238–249. ACM Press, 2006.

[8] G. Behrmann, A. David, and K. G. Larsen. A tutorial on UPPAAL. In *Formal Methods for the Design of Real-Time Systems: Proc. 4th Int. School on Formal Methods for the Design of Comput., Commun. and Software Syst. (SFM-RT 2004)*, number 3185 in LNCS, pages 200–236. Springer, 2004.

[9] A. Bell and B. R. Haverkort. Sequential and distributed model checking of Petri nets. *Software Tools for Technology Transfer (STTT)*, 7(1):43–60, 2005.

[10] A. Bell and B. R. Haverkort. Distributed disk-based algorithms for model checking very large Markov chains. *Formal Methods in System Design*, 29(2):177–196, 2006.

[11] Y. Bertot and P. Castéran. *Interactive Theorem Proving and Program Development Coq'Art: The Calculus of Inductive Constructions*, volume XXV of *Texts in Theoretical Computer Science: An EATCS Series*. Springer, 2004.

[12] K. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Trans. on Comput. Syst.*, 17(2):41–88, 1999.

[13] F. Bonnet. Performance analysis of Cyclon, an inexpensive membership management for unstructured p2p overlays. Master thesis, ENS Cachan Bretagne, University of Rennes, IRISA, 2006.

[14] J. Bowen and M. Hinchey. Ten Commandments of Formal Methods ...Ten Years Later. *Computer*, 39(1):40–48, 2006.

[15] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Gossip algorithms: Design, analysis and applications. In *Proc. 24th Conf. of the IEEE Comput. and Commun. Soc. (INFOCOM 2005)*, volume 3, pages 1653–1664. IEEE Press, 2005.

[16] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Mixing times for random walks on geometric random graphs. In *Proc. 7th Workshop on Algorithm Eng. and Experiments and 2nd Workshop on Analytic Algorithmics and Combinatorics (ALENEX/ANALCO 2005)*, pages 240–249. SIAM, 2005.

[17] M. Cadilhac, T. Hérault, R. Lassaigne, S. Peyronnet, and S. Tixeuil. Evaluating complex MAC protocols for sensor networks with APMC. In *Proc. 6th Int. Workshop on Automated Verification of Critical Syst (AVoCS'06)*, ENTCS. Elsevier, 2006. To appear.

[18] R. Cardell-Oliver. Why flooding is unreliable in multi-hop, wireless networks. Technical Report UWA-CSSE-04-001, University of Western Australia, 2004.

[19] D. Cavin, Y. Sasson, and A. Schiper. On the accuracy of MANET simulators. In *Proc. 2nd ACM Int. Workshop on Principles of Mobile Computing (POMC'02)*, pages 38–43. ACM Press, 2002.

[20] A. Clark, S. Gilmore, J. Hillston, and M. Tribastone. Stochastic process algebra. In *Formal Methods for the Performance Evaluation: Proc. 7th Int. School on Formal Methods for the Design of Comput., Commun. and Software Syst. (SFM-PE 2007)*, number 4486 in LNCS, pages 132–179. Springer, 2007.

[21] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking.* MIT Press, 2000.

[22] P. Dagum, R. Karp, M. Luby, and S. Ross. An optimal algorithm for monte carlo estimation. *SIAM J. Comput.*, 29(5):1484–1496, 2000.

[23] S. Deb, M. Médard, and C. Choute. Algebraic gossip: a network coding approach to optimal multiple rumor mongering. *IEEE/ACM Trans. Netw.*, 14(SI):2486–2507, 2006.

[24] A. Demaille, S. Peyronnet, and T. Hérault. Probabilistic verification of sensor networks. In *Proc. 4th IEEE Int. Conf. on Comput. Sci., Research, Innovation and Vision for the Future (RIVF'06)*, pages 45–54. IEEE Computer Society, 2006.

[25] A. Demaille, S. Peyronnet, and B. Sigoure. Modeling of sensor networks using XRM. In *Proc. 2nd Int. Symp. on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA'06)*, 2006.

[26] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic algorithms for replicated database maintenance. In *Proc. 6th Annu. ACM Symp. on Principles of Distributed Computing (PODC'87)*, pages 1–12. ACM Press, 1987.

[27] A. G. Dimakis, A. D. Sarwate, and M. J. Wainwright. Geographic gossip: efficient aggregation for sensor networks. In *Proc. 5th Int. Conf. on Inform. processing in sensor networks (IPSN'06)*, pages 69–76. ACM Press, 2006.

[28] N. Drost, E. Ogston, R. van Nieuwpoort, and H. Bal. ARRG: Real-World Gossiping. In *Proc. 15th IEEE Int. Symp. on High Performance Distributed Computing (HPDC-15'06).* IEEE Computer Society, 2007. To appear.

[29] P. Eugster, R. Guerraoui, S. B. Handurukande, P. Kouznetsov, and A.-M. Kermarrec. Lightweight probabilistic broadcast. *ACM Trans. on Comput. Syst.*, 21(4):341–374, 2003.

[30] P. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulié. From epidemics to distributed computing. *IEEE Computer*, 37(5):60–67, 2004.

[31] A. Fehnker and P. Gao. Formal verification and simulation for performance analysis for probabilistic broadcast protocols. In *Proc. 5th Conf. on Ad-Hoc, Mobile, and Wireless Networks (ADHOC-NOW'06)*, volume 4104 of *LNCS*, pages 121–141. Springer, 2006.

[32] U. Feige, D. Peleg, P. Raghavan, and E. Upfal. Randomized broadcast in networks. In *SIGAL Int. Symp. on Algorithms*, volume 450 of *LNCS*, pages 128–137. Springer, 1990.

[33] G. S. Fishman. *Monte Carlo: Concepts, Algorithms, and Applications.* Springer Series in Operations Research. Springer, 1996.

[34] A. Ganesh, L. Massoulie, and D. Towsley. The effect of network topology on the spread of epidemics. In *Proc. 24th of the IEEE Comput. and*

Commun. Soc. (INFOCOM 2005), volume 2, pages 1455–1466. IEEE Computer Society, 2005.

[35] A. J. Ganesh, A.-M. Kermarrec, and L. Massoulie. SCAMP: Peer-to-peer lightweight membership service for large-scale group communication. In *Proc. 3rd Int. Workshop on Networked Group Commun. (NGC'01)*, volume 2233 of *LNCS*, pages 44–55. Springer, 2001.

[36] D. Gavidia, S. Voulgaris, and M. van Steen. A Gossip-based Distributed News Service for Wireless Mesh Networks. In *Proc. 3rd IEEE Conf. on Wireless On demand Network Syst. and Services (WONS'06)*, pages 59–67. IEEE Computer Society, 2006.

[37] R. Grosu and S. A. Smolka. Monte carlo model checking. In *Proc. 11th Int. Conf. on Tools and Algorithms for Construction and Analysis of Syst. (TACAS'05)*, volume 3440 of *LNCS*, pages 271–286. Springer, 2005.

[38] G. Guirado, T. Hérault, R. Lassaigne, and S. Peyronnet. Distribution, approximation and probabilistic model checking. *ENTCS*, 135(2):19–30, 2006.

[39] J. M. Hammersley and K. W. Morton. Poor Man's Monte Carlo. *J. Royal Statistical Soc. Series B (Methodological)*, 16(1):23–38, 1954.

[40] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[41] B. R. Haverkort. Are stochastic process algebras good for performance and dependability evaluation. In *Proc. Int. Workshop on Process Algebra and Performance Modeling*, volume 1853 of *LNCS*, pages 501–510. Springer, 2000.

[42] B. R. Haverkort. Markovian models for performance and dependability evaluation. In *European Educ. Forum: School on Formal Methods and Performance Analysis*, volume 2090 of *LNCS*, pages 38–83. Springer, 2002.

[43] B. R. Haverkort, H. Hermanns, and J.-P. Katoen. On the use of model checking techniques for dependability evaluation. In *Proc. 19th IEEE Symp. on Reliable Distributed Syst. (SRDS'00)*, pages 228–237. IEEE Computer Society, 2000.

[44] T. Hérault, R. Lassaigne, F. Magniette, and S. Peyronnet. Approximate probabilistic model checking. In *Proc. 5th Int. Conf. on Verification, Model Checking and Abstract Interpretation (VMCAI'04)*, volume 2937 of *LNCS*, pages 73–84. Springer, 2004.

[45] A. Hinton, M. Z. Kwiatkowska, G. Norman, and D. Parker. Prism: A tool for automatic verification of probabilistic systems. In *Proc. 2nd Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'06)*, volume 3920 of *LNCS*, pages 441–444. Springer, 2006.

[46] G. Holzmann. *The Spin Model Checker, Primer and Reference Manual.* Addison-Wesley, 2003.

[47] J. Hromkovic, R. Klasing, B. Monien, and R. Peine. Dissemination of information in interconnection networks (Broadcasting & Gossiping). *Combinatorial Network Theory*, pages 125–212, 1996.

[48] M. Huth and M. Ryan. *Logic in Computer Science: Modelling and reasoning about systems.* Cambridge University Press, Cambridge, UK, 2004.

[49] M. Jelasity, R. Guerraoui, A.-M. Kermarrec, and M. van Steen. The peer sampling service: Experimental evaluation of unstructured gossip-based implementations. In *Proc. 5th ACM/IFIP/USENIX Int. Middleware Conf. (Middleware'04)*, volume 3231 of *LNCS*, pages 79–98. Springer, 2004.

[50] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking. Randomized rumor spreading. In *Proc. 41st Annu. IEEE Symp. on Found. of Comput. Sci. (FOCS'00)*, pages 565–574. IEEE Computer Society, 2000.

[51] J.-P. Katoen, T. Kemna, I. Zapreev, and D. Jansen. Bisimulation minimisation mostly speeds up probabilistic model checking. In *Proc. 13th Int. Conf. on Tools and Algorithms for Construction and Analysis of Syst. (TACAS'07)*, volume 4424 of *LNCS*, pages 76–92. Springer, 2007.

[52] J.-P. Katoen, D. Klink, M. Leucker, and V. Wolf. Three-valued abstraction for continuous-time Markov chains. In *Proc. 19th Int. Conf. on Comput. Aided Verification (CAV'07)*, 2007. To appear.

[53] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *Proc. 44th Annu. IEEE Symp. on Found. of Comput. Sci. (FOCS'03)*, pages 482–491. IEEE Computer Society, 2003.

[54] D. Kempe, J. Kleinberg, and A. Demers. Spatial gossip and resource location protocols. In *Proc. 33rd Annu. ACM Symp. on Theory of Computing (STOC'01)*, pages 163–172. ACM Press, 2001.

[55] D. Kempe and J. M. Kleinberg. Protocols and impossibility results for gossip-based communication mechanisms. In *Proc. 43rd Annu. IEEE Symp. on Found. of Comput. Sci. (FOCS'02)*, pages 471–480. IEEE Computer Society, 2002.

[56] M. Kwiatkowska, G. Norman, and D. Parker. Game-based abstraction for markov decision processes. In *Proc. 3rd Int. Conf. on the Quantitative Evaluation of Syst. (QEST'06)*, pages 157–166. IEEE Computer Society, 2006.

[57] S. Laplante, R. Lassaigne, F. Magniez, S. Peyronnet, and M. de Rougemont. Probabilistic abstraction for model checking: An approach based on property testing. In *Proc. 17th IEEE Symp. on Logic in Comput. Sci. (LICS 2002)*, pages 30–39. IEEE Computer Society, 2002.

[58] J. Luo, P. T. Eugster, and J.-P. Hubaux. Route driven gossip: Probabilistic reliable multicast in ad hoc networks. In *Proc. 22nd Conf. of the IEEE Comput. and Commun. Soc. (INFOCOM 2003)*, volume 3, pages 2229–2239. IEEE Computer Society, 2003.

[59] A. McIver and A. Fehnker. Formal techniques for the analysis of wireless networks. In *Proc. 2nd Int. Symp. on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA'06)*, 2006.

[60] D. Mosk-Aoyama and D. Shah. Computing separable functions via gossip. In *Proc. 25th Annu. ACM Symp. on Principles of Distributed Computing (PODC'06)*, pages 113–122. ACM Press, 2006.

[61] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.

[62] G. Norman, D. Parker, M. Z. Kwiatkowska, S. K. Shukla, and R. Gupta. Using probabilistic model checking for dynamic power management. *Formal Aspects of Computing*, 17(2):160–176, 2005.

[63] S. Owre, S. Rajan, J. M. Rushby, N. Shankar, and M. K. Srivas. PVS: Combining Specification, Proof Checking, and Model Checking. In *Proc. 8th Int. Conf. on Comput. Aided Verification (CAV'96)*, volume 1102 of *LNCS*, pages 411–414. Springer, 1996.

[64] J. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D. Epema, M.Reinders, M. van Steen, and H. Sips. Tribler: A social-based peer-to-peer system. *Concurrency and Computation: Practice and Experience*, 19:1–11, 2007.

[65] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc., New York, NY, USA, 1994.

[66] D. Randall. Rapidly Mixing Markov Chains with Applications in Computer Science and Physics. *Computing in Sci. and Eng.*, 8(2):30–41, 2006.

[67] P. Schnoebelen. The verification of probabilistic lossy channel systems. In *Validation of Stochastic Systems*, volume 2925 of *LNCS*, pages 445–466, 2004.

[68] K. Sen, M. Viswanathan, and G. Agha. Statistical model checking of black-box probabilistic systems. In *Proc. 16th Int. Conf. on Comput. Aided Verification (CAV'04)*, volume 3114 of *LNCS*, pages 202–215. Springer, 2004.

[69] R. van Renesse, K. P. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Trans. Comput. Syst.*, 21(2):164–206, 2003.

[70] R. van Renesse, Y. Minsky, and M. Hayden. A gossip-style failure detection service. In *Proc. 1st IFIP Conf. on Distributed Syst. Platforms and Open Distributed Processing (Middleware'98)*, pages 55–70. Springer, 1998.

[71] W. Vogels, R. van Renesse, and K. Birman. The power of epidemics: robust communication for large-scale distributed systems. *ACM SIGCOMM Comput. Commun. Review*, 33(1):131–135, 2003.

[72] S. Voulgaris. *Epidemic-Based Self-Organization in Peer-to-Peer Systems.* Doctoral thesis, Vrije Universiteit Amsterdam, 2006. Appears in Collections: Proefschriften Exacte Wetenschappen.

[73] S. Voulgaris, D. Gavidia, and M. van Steen. Cyclon: Inexpensive membership management for unstructured p2p overlays. *J. Network and Syst. Manage.*, 13(2):197–217, 2005.

[74] H. Younes, M. Kwiatkowska, G. Norman, and D. Parker. Numerical vs. statistical probabilistic model checking. *Software Tools for Technology Transfer (STTT)*, 8(3):216–228, 2006.

[75] H. L. S. Younes and R. G. Simmons. Probabilistic verification of discrete event systems using acceptance sampling. In *Proc. 14th Int. Conf. on Comput. Aided Verification (CAV'02)*, volume 2404 of *LNCS*, pages 223–235. Springer, 2002.

Statuten

Artikel 1.
1. De vereniging draagt de naam: "Nederlandse Vereniging voor Theoretische Informatica".
2. Zij heeft haar zetel te Amsterdam.
3. De vereniging is aangegaan voor onbepaalde tijd.
4. De vereniging stelt zich ten doel de theoretische informatica te bevorderen haar beoefening en haar toepassingen aan te moedigen.

Artikel 2.
De vereniging kent gewone leden en ereleden. Ereleden worden benoemd door het bestuur.

Artikel 3.
De vereniging kan niet worden ontbonden dan met toestemming van tenminste drievierde van het aantal gewone leden.

Artikel 4.
Het verenigingsjaar is het kalenderjaar.

Artikel 5.
De vereniging tracht het doel omschreven in artikel 1 te bereiken door
a. het houden van wetenschappelijke vergaderingen en het organiseren van symposia en congressen;
b. het uitgeven van een of meer tijdschriften, waaronder een nieuwsbrief of vergelijkbaar informatiemedium;
c. en verder door alle zodanige wettige middelen als in enige algemene vergadering goedgevonden zal worden.

Artikel 6.
1. Het bestuur schrijft de in artikel 5.a bedoelde bijeenkomsten uit en stelt het programma van elk van deze bijeenkomsten samen.
2. De redacties der tijdschriften als bedoeld in artikel 5.b worden door het bestuur benoemd.

Artikel 7.
Iedere natuurlijke persoon kan lid van de vereniging worden. Instellingen hebben geen stemrecht.

Artikel 8.
Indien enig lid niet langer als zodanig wenst te worden beschouwd, dient hij de ledenadministratie van de vereniging daarvan kennis te geven.

Artikel 9.
Ieder lid ontvangt een exemplaar der statuten, opgenomen in de nieuwsbrief van de vereniging. Een exemplaar van de statuten kan ook opgevraagd worden bij de secretaris. Ieder lid ontvangt de tijdschriften als bedoeld in artikel 5.b.

Artikel 10.
Het bestuur bestaat uit tenminste zes personen die direct door de jaarvergadering worden gekozen, voor een periode van drie jaar. Het bestuur heeft het recht het precieze aantal bestuursleden te bepalen. Bij de samenstelling van het bestuur dient rekening gehouden te worden met de wenselijkheid dat vertegenwoordigers van de verschillende werkgebieden van de theoretische informatica in Nederland in het bestuur worden opgenomen. Het bestuur kiest uit zijn midden de voorzitter, secretaris en penningmeester.

Artikel 11.
Eens per drie jaar vindt een verkiezing plaats van het bestuur door de jaarvergadering. De door de jaarvergadering gekozen bestuursleden hebben een zittingsduur van maximaal twee maal drie jaar. Na deze periode zijn zij niet terstond herkiesbaar, met uitzondering van secretaris en penningmeester. De voorzitter wordt gekozen voor de tijd van drie jaar en is na afloop van zijn ambtstermijn niet onmiddellijk als zodanig herkiesbaar. In zijn functie als bestuurslid blijft het in de vorige alinea bepaalde van kracht.

Artikel 12.
Het bestuur stelt de kandidaten voor voor eventuele vacatures. Kandidaten kunnen ook voorgesteld worden door gewone leden, minstens een maand voor de jaarvergadering via de secretaris. Dit dient schriftelijk te gebeuren op voordracht van tenminste vijftien leden. In het geval dat het aantal kandidaten gelijk is aan het aantal vacatures worden de gestelde kandidaten door de jaarvergadering in het bestuur gekozen geacht. Indien het aantal kandidaten groter is dan het aantal vacatures wordt op de jaarvergadering door schriftelijke stemming beslist. Ieder aanwezig lid brengt een stem uit op evenveel kandidaten als er vacatures zijn. Van de zo ontstane rangschikking worden de kandidaten met de meeste punten verkozen, tot het aantal vacatures. Hierbij geldt voor de jaarvergadering een quorum van dertig. In het geval dat het aantal aanwezige leden op de jaarvergadering onder het quorum ligt, kiest het zittende bestuur de nieuwe leden. Bij gelijk aantal stemmen geeft de stem van de voorzitter (of indien niet aanwezig, van de secretaris) de doorslag.

Artikel 13.
Het bestuur bepaalt elk jaar het precieze aantal bestuursleden, mits in overeenstemming met artikel 10. In het geval van aftreden of uitbreiding wordt de zo ontstane vacature aangekondigd via mailing of nieuwsbrief, minstens twee maanden voor de eerstvolgende jaarvergadering. Kandidaten voor de ontstane vacatures worden voorgesteld door bestuur en gewone leden zoals bepaald in artikel 12. Bij aftreden van bestuursleden in eerste of tweede jaar van de driejarige cyclus worden de vacatures vervuld op de eerstvolgende jaarvergadering. Bij aftreden in het derde jaar vindt vervulling van de vacatures plaats tegelijk met de algemene driejaarlijkse bestuursverkiezing. Voorts kan het bestuur beslissen om vervanging van een aftredend bestuurslid te laten vervullen tot de eerstvolgende jaarvergadering. Bij uitbreiding van het bestuur in het eerste of tweede jaar van de cyclus worden de vacatures vervuld op de eerstvolgende jaarvergadering. Bij uitbreiding in het derde jaar vindt vervulling van de vacatures plaats tegelijk met de driejaarlijkse bestuursverkiezing. Bij inkrimping stelt het bestuur vast welke leden van het bestuur zullen aftreden.

Artikel 14.
De voorzitter, de secretaris en de penningmeester vormen samen het dagelijks bestuur. De voorzitter leidt alle vergaderingen. Bij afwezigheid wordt hij vervangen door de secretaris en indien ook deze afwezig is door het in jaren oudste aanwezig lid van het bestuur. De secretaris is belast met het houden der notulen van alle huishoudelijke vergaderingen en met het voeren der correspondentie.

Artikel 15.
Het bestuur vergadert zo vaak als de voorzitter dit nodig acht of dit door drie zijner leden wordt gewenst.

Artikel 16.
Minstens eenmaal per jaar wordt door het bestuur een algemene vergadering bijeengeroepen; één van deze vergaderingen wordt expliciet aangeduid met de naam van jaarvergadering; deze vindt plaats op een door het bestuur te bepalen dag en plaats.

Artikel 17.
De jaarvergadering zal steeds gekoppeld zijn aan een wetenschappelijk symposium. De op het algemene gedeelte vaan de jaarvergadering te behandelen onderwerpen zijn

a. Verslag door de secretaris;
b. Rekening en verantwoording van de penningmeester;
c. Verslagen van de redacties der door de vereniging uitgegeven tijdschriften;
d. Eventuele verkiezing van bestuursleden;
e. Wat verder ter tafel komt. Het bestuur is verplicht een bepaald punt op de agenda van een algemene vergadering te plaatsen indien uiterlijk vier weken van te voren tenminste vijftien gewone leden schriftelijk de wens daartoe aan het bestuur te kennen geven.

Artikel 18.
Deze statuten kunnen slechts worden gewijzigd, nadat op een algemene vergadering een commissie voor statutenwijziging is benoemd. Deze commissie doet binnen zes maanden haar voorstellen via het bestuur aan de leden toekomen. Gedurende drie maanden daarna kunnen amendementen schriftelijk worden ingediend bij het bestuur, dat deze ter kennis van de gewone leden brengt, waarna een algemene vergadering de voorstellen en de ingediende amendementen behandelt. Ter vergadering kunnen nieuwe amendementen in behandeling worden genomen, die betrekking hebben op de voorstellen van de commissie of de schriftelijk ingediende amendementen. Eerst wordt over elk der amendementen afzonderlijk gestemd; een amendement kan worden aangenomen met gewone meerderheid van stemmen. Het al dan niet geamendeerde voorstel wordt daarna in zijn geheel in stemming gebracht, tenzij de vergadering met gewone meerderheid van stemmen besluit tot afzonderlijke stemming over bepaalde artikelen, waarna de resterende artikelen in hun geheel in stemming gebracht worden. In beide gevallen kunnen de voorgestelde wijzigingen slechts worden aangenomen met een meerderheid van tweederde van het aantal uitgebrachte stemmen. Aangenomen statutenwijzigingen treden onmiddellijk in werking.

Artikel 19.
Op een vergadering worden besluiten genomen bij gewone meerderheid van stemmen, tenzij deze statuten anders bepalen. Elk aanwezig gewoon lid heeft daarbij het recht een stem uit te brengen. Stemming over zaken geschiedt mondeling of schriftelijk, die over personen met gesloten briefjes. Uitsluitend bij schriftelijke stemmingen worden blanco stemmen gerekend geldig te zijn uitgebracht.

Artikel 20.
a. De jaarvergadering geeft bij huishoudelijk reglement nadere regels omtrent alle onderwerpen, waarvan de regeling door de statuten wordt vereist, of de jaarvergadering gewenst voorkomt.
b. Het huishoudelijk reglement zal geen bepalingen mogen bevatten die afwijken van of die in strijd zijn met de bepalingen van de wet of van de statuten, tenzij de afwijking door de wet of de statuten wordt toegestaan.

Artikel 21.
In gevallen waarin deze statuten niet voorzien, beslist het bestuur.