Software Modeling and Verification

Staff

- Professors:

  Prof. Dr. Ir. Joost–Pieter Katoen
  Prof. em. Dr. Klaus Indermark
  `http://www-i2.informatik.rwth-aachen.de/`

- Secretary:

  Elke Ohlenforst

- Lecturer:

  Akademischer Oberrat Priv.–Doz. Dr. Thomas Noll

- Researchers:

  Dr. Henrik Bohnenkamp (since Sept. 1st)
  Dr. Benedikt Bollig (until Okt. 1st)
  Dr. Michael Weber (until Nov. 1st)
  Tingting Han, B.Sc. (since Nov. 1st, funded by the NWO)
  Dipl.-Inform. Carsten Kern (since Sept. 1st)
  Dipl.-Inform. Martin Neuhäußer (since Nov. 1st, funded by the NWO)
  Dipl.-Inform. Stefan Rieger (since Sept. 1st)
  Dipl.-Inform. Volker Stolz
  Ivan Zapreev, M.Sc. (funded by the NWO)

- Technical Staff:

  Arnd Gehrmann

- Student Researchers:

  Eric Bodden
  Frank Birbacher
  Daniel Neider
  Denise Nimmerrichter
  Gustavo Quiros
  Christina Roeckerath
  Michael Rohrbach
  Markus Schlütter
  Henning Stein
  Evamarie Storch
  René Theisen
  Daniel Willems

- Visiting Scientists:

  Dr. Gerald Lüttgen (Univ. of York, UK)
  Dr. David Jansen (University of Twente, NL)
  Dr. Erika Ábrahám (Universität Freiburg, D)

Overview

2005 has been a rather dynamic and evolving year for LS2! Klaus Indermark received his status as emeritus professor on October 1. Since then the chair is led by Joost-Pieter Katoen. As Joost-Pieter already joined the group in December 2004, this change could be accomplished in a co-operative and step-by-step manner. We thank Klaus Indermark for his enormous enthusiasm and dedication with which he has shaped and chaired LS2 over the last 30 years, and hope to welcome him frequently on our floor in good health!

Several changes in the academic staff of the group have taken place in 2005. Benedikt Bollig received his PhD degree in June, and obtained a permanent research position at the ENS Cachan, France. An updated and extended version of his doctoral dissertation will appear in the prestigious ENTCS series *Texts in Theoretical Computer Science*. Michael Weber left in November to the group of Jaco van de Pol in the well-known Dutch research institute CWI (Centrum voor Wiskunde en Informatica, Amsterdam). He received his PhD in January 2006. New group members are: Henrik Bohnenkamp, Tingting Han, Carsten Kern, Martin Neuhäußer, Stefan Rieger and Ivan Zapreev. Welcome to LS2!

As last drastic change in 2005, we mention the enormous renovation of the chair. The newly furnished and equipped chair provides an excellent place to work. We thank all involved people for their support during the renovation.

In August 2005, the chair has been involved in the organization of the workshop *"Algebraic Process Calculi: The First Twenty Five Years and Beyond"*. This event was a great success, and several pioneers in process algebra such as Robin Milner, Tony Hoare, Matthew Hennessy, Jan-Willem Klop and Stephan Brookes gave their acte-de-presence. It was interesting to listen to their excellent retrospective talks.

The research programme of the *Software Modeling and Verification group (MOVES)*, as the group is called now, is concerned with the study, development and application of *formal methods* to software design in a broad sense. Our group aims at modeling and verifying *thrustworthiness aspects* (such as safety, reliability, performance and survivability) of software systems by applying mathematical theories and methods.

Major research topics of interest are:

- modeling formalisms for concurrent systems (such as process algebra, statecharts, message sequence diagrams and mobile process calculi);
- model checking and quantitative extensions thereof (in particular probabilistic model checking, cost bounds, abstractions, scheduling generation and analysis);
- semantics and analysis of modern programming languages (a.o., semantics of Erlang, heap abstractions and pointer analysis, multi-threading);
- probabilistic models for concurrency (i.e., the theory of models, abstraction, refinement etc.);

- testing and run-time verification.

Our research is conducted in the context of several projects that are funded by the NWO (Dutch Research Council), NWO and DFG, and the European Union. We are looking forward to our participation in the forthcoming research training group on the algorithmic synthesis of reactive systems.

Joost-Pieter Katoen

Research Projects

The MoDeST Tool Environment

H. Bohnenkamp, J.-P. Katoen, H. Hermanns (Univ. d. Saarlandes), P. R. D'Argenio (Univ. Nacional de Córdoba, AR)

The specification language MODEST covers a wide spectrum of modelling concepts, ranging from plain labelled transition systems to stochastic systems like Generalised Semi-Markov Decision Processes. MODEST possesses a rigid, process-algebra style semantics, and yet provides modern and flexible specification constructs. MODEST specifications constitute a coherent starting-point to analyse distinct system characteristics with various techniques, e.g., model checking to assess functional correctness and discrete-event simulation to establish the system's reliability. Analysis results thus refer to the *same* system specification, rather than to different (and potentially incompatible) specifications of system perspectives like in the UML.

The tool MOTOR (MODEST Tool enviRonment) is aimed to provide the means to analyse and evaluate MODEST specifications. The tool is written in the C++ programming language. The tool provides (i) interfacing capabilities for connection to existing tools for specific projected models, and (ii) also means for enhancement by *native* algorithms for analysis of (classes) of MODEST specifications. In earlier work, MOTOR has been connected to MÖBIUS, a performance evaluation tool suite that has been developed at the University of Illinois at Urbana-Champaign, US. Currently we are working on a state-space generator for MODEST and aim at connecting MODEST via MOTOR to the PRISM tool, a model-checker for probabilistic timed systems, developed at the University Birmingham, UK. Furthermore, we plan to connect MODEST to the UPPAAL model checker.

Timed Model-Based Testing

H. Bohnenkamp, A. Belinfante (Univ. Twente, NL), M. Stoelinga (Univ. Twente, NL)

Testing is one of the most natural, intuitive and effective methods to increase the reliability of software. Formal methods have been employed to analyse and systematise the testing idea in general, and to define notions of correctness of implementations with respect to specifications in particular. The IOCO testing theory reasons about black-box conformance testing of software components. The test-case generation and execution algorithms of IOCO have been implemented in TORX, a testing tool developed at the University of Twente.

We work on an extension of TORX to allow testing of real-time properties: *real-time testing*. Real-time testing means that the decisions whether an implementation under test has passed or failed a test is not only based on which outputs are observed, given a certain sequence of inputs, but also on *when* the outputs occur, given a certain sequence of inputs *applied at predefined times*. We use as input models non-deterministic *safety timed automata*.

Dependable Global Computing

H. Bohnenkamp, T. Han, J.-P. Katoen, R. De Nicola (U. Florence, I), D. Latella and M. Massink (CNR-ISTI, I)

(funded by the DAAD and CNR-ISTI)

Due to their enormous size—networks typically consist of thousands or even millions of nodes—and their strong reliance on mobility and interaction, performance and dependability issues are of utmost importance for "network-aware computing". Spontaneous computer crashes may easily lead to failure of remote execution or process movement, while spurious network hick ups may cause loss of code fragments or unpredictable delays. The enormous magnitude of computing devices involved in global computing yields failure rates that no longer can be ignored. The presence of such random phenomena implies that correctness of global computing software and their privacy guarantees are no longer rigid notions like: "either it is safe or it is not" but have a less absolute nature, e.g.: "in 99.7% of the cases, privacy can be ensured". The intrinsic complexity of global computers, though, complicates the assessment of these issues severely. Systematic methods, techniques and tools—all based on solid mathematical foundations i.e., *formal methods*, are therefore needed to establish performance and dependability requirements and guarantees.

This project attempts to make a considerable step into this direction by extending a successful programming and specification formalism for global computing, KLAIM, with random delays, and by developing a novel stochastic spatial temporal logic as property specification language for performance and dependability guarantees.

Automata and Logics for Message Sequence Charts

B. Bollig

A message-passing automaton is an abstract model for the implementation of a distributed system whose components communicate via message exchange and thereby define a collection of communication scenarios called message sequence charts. In this project, we study several variants of message-passing automata in a unifying framework. We classify their expressiveness in terms of state-space properties, synchronization behavior, and acceptance mode and also compare them to algebraic characterizations of sets of message sequence charts, among them the classes of recognizable and rational languages.

We then focus on finite-state devices with global acceptance condition that communicate via a priori unbounded channels. We show them to be exactly as expressive as the existential fragment of monadic second-order logic over message sequence charts and to be strictly weaker than full monadic second-order logic. It turns out that message-passing automata cannot be complemented and that they cannot be determinized in general. Those results rely on a new proof technique, which allows to apply graph acceptors as introduced by Thomas to the framework of message sequence charts.

QUPES: Verification of Quantitative Properties of Embedded Software

T. Han, J.-P. Katoen, M. Neuhäußer, D. Willems

(funded by the NWO)

The research challenge faced by the QUPES project is to adapt and enrich model checking, a successful technique for checking the logical correctness of system designs, to meet the requirements of state-of-the-art embedded software engineering. Embedded software typically executes on devices that, first and foremost, are not computers. This imposes high requirements on performance and economical resource usage. Due to its embedded nature, its robustness is of prime importance, and timely reactions to stimuli from its—mostly physical—environment are essential. This project proposes to assess these "non-functional" aspects (e.g., timeliness and robustness) as an integral part of the embedded software validation phase. The aim is to obtain a single framework that supports both the validation of qualitative (i.e., functional) as well as quantitative aspects of embedded software.

To accomplish this, model-checking techniques will be extended with ample means

to reason about costs (power consumption, memory usage, and the like), efficiency, and robustness. In particular, we aim to develop verification algorithms for real-time systems that exhibit both *(continuous-time) randomness as non-determinism* and extend this approach with *cost* aspects. Furthermore, advances to model checking of stochastic systems will be made by developing aggressive *abstraction* techniques and methods that effectively exploit the (compositional) structure of embedded software specifications for verification purposes. The latter activities are aimed at making stochastic model checking applicable to state spaces that are several orders of magnitude larger than currently can be handled. Our techniques will be tailored to hierarchical design notations (statechart diagrams) for embedded software.

This project takes place in the context of the DFG-NWO bilateral project VOSS2 (Validation of Stochastic Systems). In this project, we cooperate with the Universities of Bonn (Prof. Christel Baier), Saarland (Prof. Holger Hermanns), Nijmegen (Prof. Frits Vaandrager), Twente (Prof. Boudewijn Haverkort), and Federal Armed Forces Munich (Prof. Markus Siegle).

Verifying Safety and Liveness in Concurrent Pointer Programs

J.-P. Katoen, Th. Noll, S. Rieger, A. Rensink (U. Twente, NL) and D. Distefano (Queen Mary College London, UK)

The incorrect use of pointers is one of the most common source of software errors. Concurrency has a similar characteristic. Proving the correctness of concurrent pointer manipulating programs, let alone algorithmically, is a highly non-trivial task. This project attempts to develop automated verification techniques and accompanying tool-support for concurrent programs that manipulate dynamic, linked data structures. Initially, we focus on linked lists. As verification technique, we investigate the use of automata-based model-checking algorithms. First and second-order (monadic) temporal logics are employed for the specification of properties of such concurrent programs. These logics can easily express the dynamic creation and deletion of data. In our approach, we consider abstractions of linked data structures that are tailored to both the property and the program to be analysed. The main challenge is to achieve a fully automated technique to prove the correctness of concurrent pointer programs such as deadlock avoidance protocols, concurrent garbage collection algorithms and so forth.

MC=MC: Model Checking Infinite-State Markov Chains

J.-P. Katoen, I.S. Zapreev, B.R. Haverkort (U. Twente, NL), A. Remke (U. Twente, NL)

(funded by the NWO)

Model-based performance evaluation aims at forecasting system behaviour in a quantitative way, starting from an abstract system model. Due to the ever-increasing size and complexity of modern computer and communication systems, performance models that are directly amenable to a numerical solution are often generated from high-level modelling languages, based, e.g., on stochastic Petri nets or stochastic process algebras. For a significant class of systems, these models turn out to be infinite state, and need to be analysed by specific techniques, such as matrix-geometric methods.

Recently, extensions to temporal logics have been developed to ease the specification of important measures-of-interest (like reponse times, or the probability to reach deadlines) over performance models, and logic-based verification algorithms have been integrated with numerical means to automatically check these properties. This novel approach is, however, still restricted to finite-state systems.

This project aims to establish a cross-fertilization between (i) *performance evaluation techniques for infinite-state systems* and (ii) *logic-based model-checking algorithms for Markov chains*. The goal is to develop algorithms and a prototype software tool for the specification and automated evaluation of performance measures for infinite-state Markov chains, and to apply these to case studies with realistic complexity.

This project also takes place as part of the VOSS2 project.

Verification of Erlang Programs

M. Neuhäußer, Th. Noll, L.H. Haß, P. Tawiah, C.K. Roy (Dublin City Univ., IRL)

Software written for telecommunication applications usually has to meet high quality demands. Due to its complexity and its nondeterministic behaviour validation methods which are purely based on testing are generally not sufficient to ensure that the requirements are met. Therefore formal verification methods are highly desirable.

In this project we are developing and studying verification approaches which are tailored to Erlang, a programming language for implementing open, distributed telecommunication software. The complex dynamic and concurrent behaviour of such systems makes standard, finite–state model–checking techniques inapplicable in this setting.

We are tackling this problem from two different sides:

- To make Erlang systems amenable to automatic model checking techniques, one thread of our research focuses on abstraction techniques which can be employed to reduce the state space of the system under consideration. More concretely we have developed a formal definition of the syntax and semantics of a core version of Erlang in *Rewriting Logic*, a unified semantic framework for concurrency which is founded on conditional term rewriting modulo equational theories. In particular, the term rewriting machinery can be employed to model the operational behaviour of programs in terms of transition systems, and equations allow us to define abstraction mappings on the state space.

  A prototype version of an Erlang evaluator has been implemented in Maude, which is a specification language supporting the Rewriting Logic framework. The results obtained so far are very promising, inviting to further investigate the benefits of equational abstractions for Erlang programs.

- In a second approach we try to benefit from existing work by translating the given Erlang program into a specification language for which analysis and verification methods have been already developed. Due to the dynamic and mobile process and communication structures which are present in many Erlang applications, classical languages such as LOTOS or Promela are not suitable for this purpose. Rather we are using the *π–calculus*, a name–passing process algebra which allows to describe concurrent systems with a dynamically developing communication topology.

Optimization of Straight-Line Code

T. Noll, S. Rieger

We study the effect of an optimizing algorithm for straight-line code which first constructs a directed acyclic graph representing the given program and then generates code from it.

It can be shown that this algorithm produces optimal code with respect to classical optimizing program transformations such as Constant Folding, Common Subexpression Elimination, and Dead Code Elimination. In contrast to the former, the latter are also applicable to iterative code containing loops.

We can show that the graph-based algorithm essentially corresponds to a combination of the three classical optimizations in conjunction with Copy Propagation. Thus, apart from its theoretical importance, this result is relevant for practical compiler design as it allows to exploit the optimization potential of the graph-based algorithm for non–linear code as well.

Runtime Verification

V. Stolz, E. Bodden, R. Theisen

Concurrent programs may contain bugs like deadlocks which are hard to reproduce. Potentially bad behaviour of these programs can be detected through runtime verification where Linear-Time Logic (LTL) formulae are used to express temporal assertions. We implemented LTL-checkers in Haskell and Java, which, driven by annotations in the source program, checks the formulae.

Annotations for Haskell where provided through a wrapper for functions in the standard concurrency library. For Java, we used metaprogramming, that is, treating a program as data and manipulating it, through AspectJ. The temporal properties to verify are extracted via Java 5 Metadata annotations from the source code and stored in a standardised way in the class file. Thus, automatic manipulation of such data no longer depends on pre-processors. We use this scheme to store a combination of LTL formulae and AspectJ expressions.

For checking dynamic systems, a template mechanism allows to instantiate placeholder variables in parameterised formulae which generate new formulae on the fly. If a violation is detected, the trace leading to this situation can be examined.

The results where presented at the Fifth Workshop on Runtime Verification in Edinburgh, UK. Also, two diploma theses where published. For his diploma thesis on the Java version with the title “J-LO - A tool for runtime-checking temporal assertions” Eric Bodden won the first place in the undergraduate category of the Association for Computing Machinery (ACM) Student Research Competition 2005. He and his supervisor Volker Stolz enjoyed a trip to the awards ceremony in San Francisco sponsored by the ACM.

DIVSPIN—A SPIN Compatible Distributed Model Checker

M. Weber, M. Leucker (TU München), L. Brim (Brno, CZ)

The project’s goal is the design and implementation of a parallel and distributed model checker called DIVSPIN.

The benefits of this project are threefold: first, we intend to create a ready-to-use, large-scale distributed state space exploration tool directed at a significant part of the user base of verification tools, as well as providing hardware to run on. Second, the tool enables us to empirically evaluate existing algorithms with regard to their performance and characteristics under controlled conditions, providing useful insight

into their strengths and weaknesses, leading to a more informed way to choose an algorithm depending on a given task. Third, we intend to use the environment as a platform for further development of (and experimentation with) parallel and distributed model checking algorithms.

Further development and evaluation of parallel algorithms deserves special attention, considering recent developments in CPU technology wrt. multi-core "Cell processors", and the rising interest in grid computing, which foreshadow the near end of Moore's Law, and consequently the need for well-scaling parallel and distributed algorithms in future verification tools.

Distributed Algorithms Test Bench

M. Weber, M. Rohrbach, E. Bodden

In parallel to the already mentioned DIVSPIN project, work has begun to develop a benchmark environment for distributed algorithms. Complementary to the DIVSPIN project, which aims at producing a practical and competitive model checker, the focus of this project is to provide a *language-independent prototyping framework* for distributed and parallel algorithms.

The tool is aimed at researchers who want to experiment with algorithms in their programming language of choice, without the distractions of having to implement a distributed data processing framework, distributed termination checks, instrumentation for profiling, data structures, etc.

A Virtual Machine for State Space Generation

M. Weber

We develop formal semantics for a virtual machine which can be used for state space generation.

Common approaches in explicit-state model checking employ modeling languages like CSP, LOTOS or Promela for the description of state spaces. The semantics of these languages are usually non-trivial: besides concepts found in programming languages (scopes, variables, expressions etc.) they often provide features like process abstraction, non-determinism, communication primitives, timers, etc. Implementing the semantics of languages like Promela for use in verification tools is consequently not straight-forward, even more so if the language is described only informally and static and operational semantics are unavailable.

The goal of this on-going work is the definition of formally specified semantics of a small and straight-forwardly implementable byte-code language which incorporates the above mentioned features in a compositional way, so that tool implementors can choose what is needed and leave out the rest.

In a second step, we define faithful translation functions for modeling languages like Promela, π-calculus, and DIVINE language which enables us to compile them into byte-code.

Other Activities

J.-P. Katoen

- Member of the Steering Committee of ETAPS (*European Joint Conferences on Theory and Practice of Software*).
- Member of the Steering Committee of QEST (*Quantitative Evaluation of Systems*).
- Board Member of the Dutch Society on Theoretical Computer Science (NVTI).
- Member of the Program Committee of the following events:
  - Workshop on *Methods for Modalities* (M4M 2005), Berlin.
  - 2nd Int. Conf. on *Quantitative Evaluation of Systems* (QEST 2005), Torino, Italy, 2005.
  - 3rd Int. Conf. on *Formal Modelling and Analysis of Timed Systems* (FORMATS 2005), Uppsala, Sweden, 2005.
  - Conf. on *Component-Oriented Enterprise Applications* (COEA 2005), Erfurt, 2005.
- Member of the IFIP Working Group 1.8 on Concurrency Theory.
- Member of the EPSRC Review College (Engineering and Physical Sciences Research Council).
- Member of the Organization Committee of the Workshop on *Algebraic Process Calculi: The First Twenty Five Years and Beyond*, Bertinoro, August, 2005.
- Member of external PhD committees.

K. Indermark

- Scientific Advisor of the German-Israeli Foundation for Scientific Research and Development (G.I.F.)
- Member of the Editorial Board of
  - *Fundamenta Informaticae*, Annales Societatis Mathematicae Polonae
  - *Aachener Beitr¨age zur Informatik*
- Additional member of RWTH Faculty of Electrical Engineering and Information Technology
- Referee for Deutsche Forschungsgemeinschaft (DFG)
- Administrative tasks:
  - assignment of undergraduate courses to the teaching personnel

  - student statistics
  - head of Committee for Teaching Service

Th. Noll

- Organizer of the *5th Workshop on Language Descriptions, Tools and Applications* (ETAPS/LDTA '05)
- Program committee member of the *21st Annual ACM Symposium on Applied Computing* (SAC '06)
- Member of the examination boards for Computer Science and Computational Material Science
- Student advisor for the following subsidiary subjects within CS: Electrical Engineering, Civil Engineering, and Medicine
- Organization of teaching service of CS Department (`http://www-i2.informatik.rwth-aachen.de/Teaching/Service/`)
- Member of external PhD committees.

B. Bollig

- Reviewer for ICALP 2005, FSTTCS 2005, QEST 2005 and FORMATS 2005

Talks and Publications

Talks

E. Bodden: *Temporal Assertions using AspectJ,* 22. Workshop der GI-Fachgruppe 2.1.4 Programmiersprachen und Rechenkonzepte, Bad Honnef

H. Bohnenkamp: *Timed testing with TorX,* 11th Dutch Testing Day, 11. 11. 2005, Enschede, Netherlands

B. Bollig: *On the Expressiveness of Asynchronous Cellular Automata,* The 15th International Symposium on Fundamentals of Computation Theory (FCT'05), Lübeck, Germany, August 2005

B. Bollig: *On the Expressiveness of Asynchronous Cellular Automata,* Kolloquium/Ringvorlesung des Graduiertenkolleg Wissensrepräsentation, Universität Leipzig, Germany, August 2005

B. Bollig: *On the Expressiveness of Asynchronous Cellular Automata,* TU München, August, 2005

B. Bollig: *Automata and Logics for Message Sequence Charts,* Informatik-Oberseminar, RWTH Aachen University, March 2005

J.-P. Katoen: *Are You Still There? - A Lightweight Algorithm to Monitor Node Presence in Self-Configuring Networks,* AMETIST Project meeting, Enschede, Netherlands, January 14, 2005

J.-P. Katoen: *Modeling and Analysis of Probabilistic Systems,* Twente Embedded Systems Initiative Workshop, February 7, 2005

J.-P. Katoen: *Model Checking Markov Processes,* Politecnico di Milano, Italy, May 19, 2005

J.-P. Katoen: *Are You Still There? - A Lightweight Algorithm to Monitor Node Presence in Self-Configuring Networks,* Int. Conf. on Dependable Systems and Networks (DSN), Yokohama, Japan, June 30, 2005

J.-P. Katoen: *Model Checking Meets Performance Evaluation,* Informatik Kolloquium at the University of Stuttgart, July 12, 2005

J.-P. Katoen: *The Semantic Foundations of Stochastic Statecharts,* Tutorial at the 3rd IEEE Int. Conf. on Software Engineering and Formal Methods (SEFM), Koblenz, September 6, 2005 (together with D.N. Jansen and H. Hermanns)

J.-P. Katoen: *Could it Probably be Cheap?,* Invited Talk at the 5th Int. Workshop on Automated Verification of Critical Systems (AVOCS). University of Warwick, UK, September 13, 2005

J.-P. Katoen: *The Semantic Foundations of Stochastic Statecharts,* Tutorial at the 2nd Int. Conf. on the Quantitative Evaluation of Systems (QEST). Torino, Italy, September 19, 2005 (together with D.N. Jansen)

J.-P. Katoen: *Foundations of Stochastic Model Checking,* Tutorial at the ARTIST2 Summerschool on Component & Modelling, Testing & Verification, and Static Analysis of Embedded Systems, Naesslingen, Sweden, October 2, 2005

J.-P. Katoen: *Applications of Stochastic Model Checking,* Tutorial at the ARTIST2 Summerschool on Component & Modelling, Testing & Verification, and Static Analysis of Embedded Systems, Naesslingen, Sweden, October 2, 2005

J.-P. Katoen: *Verifying Liveness and Safety of Concurrent Heap- Manipulating Programs,* Invited Keynote at the 4th Int. Symp. on Formal Methods for Components and Objects (FMCO). Amsterdam, November 2, 2005

J.-P. Katoen: *Who is Pointing When to Whom? Model Checking Concurrent Heap-Manipulating Programs,* Retirement Colloquium of Prof. K. Indermark, RWTH Aachen, November 4, 2005

J.-P. Katoen: *Modeling and Verification of Software,* Tag der Informatik. RWTH Aachen, December 2, 2005

C. Kern: *Analysis and Implementation of MSC Specifications,* Doctoral Symposium of IFM2005, Technische Universiteit Eindhoven, The Netherlands, November 29th, 2005

Th. Noll: *Modelling Erlang in the $\pi-$Calculus,* Computer Science Colloquium, University of Kent at Canterbury, GB, March 25, 2005

Th. Noll: *Modelling Erlang in the $\pi-$Calculus,* 4th ACM SIGPLAN Erlang Workshop, Tallinn, Estonia, September 25, 2005

V. Stolz: *Temporal Assertions using AspectJ,* RV'05 - Fifth Workshop on Runtime Verification

M. Weber: *Parallel Model Checking, SENVA 2005 workshop, Saint Pierre de Chartreuse (Isre), France, 06/2005,*

M. Weber: *State Space Generation Revisited, SENVA 2005 workshop, Saint Pierre de Chartreuse (Isre), France, 06/2005,*

M. Weber: *A Virtual Machine for State Space Generation, 22. Workshop der GI-Fachgruppe 2.1.4 (Programmiersprachen und Rechenkonzepte), Bad Honnef, Germany, 04/2005,*

M. Weber: *DIVSPIN – A SPIN compatible distributed model checker, PDMC 2005 short presentation, Lisbon, Portugal, July 10,*

Publications

C. Baier, B.R. Haverkort, H. Hermanns, J.-P. Katoen: *Model checking meets performance evaluation,* SIGMETRICS Performance Evaluation Review 32(4): 10-15 (2005)

C. Baier, H. Hermanns, J.-P. Katoen, B. Haverkort: *Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes,* Theoretical Computer Science, vol. 345 no. 1, pages 2-26, Nov. 2005

C. Baier, J.-P. Katoen, H. Hermanns, V. Wolf: *Comparative branching- time semantics for Markov chains,* Information & Computation 200(2): 149-214 (2005)

E. Bodden, V. Stolz: *Temporal Assertions using AspectJ,* RV'05 - Fifth Workshop on Runtime Verification, to be published in ENTCS

H. Bohnenkamp and A. Belinfante.: *Timed testing with TorX.,* In John Fitzgerald, Ian Hayes, and Andrzej Tarlecki, editors, *Formal Methods 2005*, volume 3582 of *LNCS*, pages 173–188, Springer-Verlag, 2005

H.C. Bohnenkamp, J. Gorter, J. Guidi, J.-P. Katoen: *Are You Still There? - A Lightweight Algorithm to Monitor Node Presence in Self-Configuring Networks,* Dependable Systems and Networks (DSN 2005), pp. 704-709, IEEE CS Press (2005)

B. Bollig and M. Leucker: *Message-Passing Automata are expressively equivalent to EMSO Logic,* Theoretical Computer Science. To appear.

B. Bollig: *On the Expressiveness of Asynchronous Cellular Automata,* In Proceedings of the 15th International Symposium on Fundamentals of Computation Theory (FCT'05), Lübeck, Germany, August 2005, LNCS 3623, Springer

B. Bollig and M. Leucker: *A Hierarchy of Implementable MSC Languages,* In Proceedings of 25th IFIP WG6.1 International Conference on Formal Techniques for Networked and Distributed Systems (FORTE'05), Taipei, Taiwan, October 2005, LNCS 3731, Springer

B. Bollig: *Automata and Logics for Message Sequence Charts,* Dissertation, Faculty of Mathematics, Computer Sciences and Natural Sciences, RWTH Aachen, Germany, May 2005

M. Broy, B. Jonsson, J.-P. Katoen, M. Leucker, A. Pretschner (editors): *Model-Based Testing of Reactive Systems,* Advanced Lectures, Lecture Notes in Computer Science 3472, Springer Verlag (2005)

L. Cloth, J.-P. Katoen, M. Khattri, R. Pulungan: *Model Checking Markov Reward Models with Impulse Rewards,* Dependable Systems and Networks (DSN 2005), pp. 722-731, IEEE CS Press (2005)

P.R. D'Argenio and J.-P. Katoen: *A Theory of Stochastic Systems Part I: Stochastic Automata,* Information & Computation 203(1): 1-38 (2005)

P.R. D'Argenio and J.-P. Katoen: *A Theory of Stochastic Systems Part II: Process Algebra,* Information & Computation 203(1): 39-74 (2005)

R. De Nicola, J.-P. Katoen, D. Latella, M. Massink: *Towards a Logic for Performance and Mobility,* 3rd Workshop on Quantitative Aspects of Programming Languages (QAPL), Techical Report LFCS, pp. 132-146 (2005)

B.R. Haverkort, J.-P. Katoen: *Performance and verification,* SIGMETRICS Performance Evaluation Review 32(4): 3 (2005)

K. Indermark, Th. Noll: *Algebraic Correctness Proofs for Compiling Recursive Function Definitions with Strictness Information,* accepted for publication in Acta Informatica

J.-P. Katoen, M. Khattri and I.S. Zapreev: *A Markov reward model checker,* In: Quantitative Evaluation of Systems (QEST), IEEE CS Press, 243-245 (2005)

J.-P. Katoen and I.S. Zapreev: *Safe on-the-fly steady-state detection for time-bounded reachability,* CTIT Technical Report TR-CTIT-05-52, Univ. of Twente (2005)

C. Kern: *Analysis and Implementation of MSC Specifications,* Doctoral symposium of the 5th International Conference on Integrated Formal Methods (IFM'05), Eindhoven, The Netherlands, November 2005, CS-Report 05-29 Technische Universiteit Eindhoven, pp. 55–61

M. Leucker, Th. Noll, P. Stevens, M. Weber: *Functional Programming Languages for Verification Tools: A Comparison of ML and Haskell,* International Journal on Software Tools for Technology Transfer (STTT), 7(2), 2005, pp. 405–420

Th. Noll, S. Rieger: *Optimization of Straight–Line Code Revisited,* Aachener Informatik–Bericht 2005–21, 2005

Th. Noll, C.K. Roy: *Modelling Erlang in the $\pi-$Calculus,* Proc. of the 4th ACM SIGPLAN Erlang Workshop, ACM, 2005, pp. 72–77