Susanne Graf
Laurent Mounier (Eds.)

LNCS 2989

Model Checking Software

11th International SPIN Workshop
Barcelona, Spain, April 2004
Proceedings

Springer

Susanne Graf Laurent Mounier (Eds.)

Model Checking Software

11th International SPIN Workshop
Barcelona, Spain, April 1-3, 2004
Proceedings

Springer

Volume Editors

Susanne Graf
Laurent Mounier
VERIMAG
2, avenue de Vignate, 38610 Grenoble-Gières, France
E-mail: {Susanne.Graf, Laurent.Mounier}@imag.fr

Library of Congress Control Number: 2004102408

CR Subject Classification (1998): F.3, D.2.4, D.3.1, D.2

ISSN 0302-9743
ISBN 3-540-21314-7 Springer-Verlag Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by PTP-Berlin, Protago-TeX-Production GmbH
Printed on acid-free paper SPIN: 10994757 06/3142 5 4 3 2 1 0

Preface

Since 1995, when the SPIN workshop series was instigated, SPIN workshops have been held on an annual basis in Montréal (1995), New Brunswick (1996), Enschede (1997), Paris (1998), Trento (1999), Toulouse (1999), Stanford (2000), Toronto (2001), Grenoble (2002) and Portland (2003). All but the first SPIN workshop were organized as satellite events of larger conferences, in particular of CAV (1996), TACAS (1997), FORTE/PSTV (1998), FLOC (1999), the World Congress on Formal Methods (1999), FMOODS (2000), ICSE (2001, 2003) and ETAPS (2002). This year again, SPIN was held as a satellite event of ETAPS 2004. The co-location of SPIN workshops with conferences has proven to be very successful and has helped to disseminate SPIN model checking technology to wider audiences. Since 1999, the proceedings of the SPIN workshops have appeared in Springer-Verlag's Lecture Notes in Computer Science series.

The history of successful SPIN workshops is evidence for the maturing of model checking technology, not only in the hardware domain, but increasingly also in the software area. While in earlier years algorithms and tool development around the SPIN model checker were the focus of this workshop series, for several years now the scope has been widened to include more general approaches to software model checking techniques and tools as well as applications.

The SPIN workshop has become a forum for all practitioners and researchers interested in model checking based techniques for the validation and analysis of communication protocols and software systems. Techniques based on explicit representations of state spaces, as implemented for example in the SPIN model checker or other tools, or techniques based on combinations of explicit representations with symbolic representations, are the focus of this workshop. It has proven to be particularly suitable for analyzing concurrent asynchronous systems. The workshop topics include theoretical and algorithmic foundations and tools, model derivation from code and code derivation from models, techniques for dealing with large and infinite state spaces, timing and applications. The workshop aims to encourage interactions and exchanges of ideas with all related areas in software engineering.

Papers went through a rigorous reviewing process. Each submitted paper was reviewed by three program committee members. Of 48 submissions, 19 research papers and 3 tool presentations were selected. Papers for which one of the editors was a co-author were handled by a sub-committee chaired by Gerard Holzmann.

In addition to the refereed papers, four invited talks were given; of these three were ETAPS invited speakers: Antti Valmari (Tampere, Finland) on the Rubik's Cube and what it can tell us about data structures, information theory and randomization, Mary-Lou Soffa (Pittsburgh, USA) on the foundations of code optimization, and Robin Milner (Cambridge, UK) on the grand challenge of building a theory for global ubiquitous computing. Finally, the SPIN invited

Organization

SPIN 2004 was the 11th instance of the SPIN workshop on Model Checking of Software. It was held in cooperation with ACM SIGPLAN as a satellite event of ETAPS 2004, the European Joint Conferences on Theory and Practice of Software, which was organized by the Technical University of Catalonia in Barcelona, Spain.

Advisory Committee

Gerard Holzmann
Amir Pnueli

Steering Committee

Thomas Ball
Susanne Graf
Stefan Leue
Moshe Vardi
Pierre Wolper (chair)

Program Committee

Chairs: Susanne Graf (VERIMAG, Grenoble)
Laurent Mounier (VERIMAG, Grenoble)

Bernard Boigelot (Liège, Belgium)
Dragan Bošnački (Eindhoven, Netherlands)
David Dill (Stanford, USA)
Javier Esparza (Stuttgart, Germany)
Patrice Godefroid (Bell Labs, USA)
Susanne Graf (Grenoble, France)
John Hatcliff (Kansas State, USA)
Gerard Holzmann (NASA/JPL, USA)
Stefan Leue (Freiburg, Germany)
Pedro Merino (Malaga, Spain)
Laurent Mounier (Grenoble, France)
Mooly Sagiv (Tel Aviv, Israel)
Scott Stoller (Stony Brook, USA)
Antti Valmari (Tampere, Finland)

Reviewers

Robby
Suzana Andova
Gerd Behrmann
Saddek Bensalem
Marius Bozga
Cas Cremers
Maria del Mar Gallardo
Manuel Diaz
Jürgen Dingel
Jean-Claude Fernandez
Jaco Geldenhuys
Keijo Heljanko
Radu Iosif
Natalia Ioustinova
Rajeev Joshi
Tommi Junttila
Antero Kangas
Timo Karvi
Barbara König
Yassine Lakhnech
Johan Lilius
Jesus Martinez
Richard Mayr
Iulian Ober
Shaham Ohad
Michael Périn
Ilya Shlyakhter
Stavros Tripakis
Jaco van de Pol
Kimmo Varpaaniemi
Wei Wei
Tim Willemse
Eran Yahav
Ping Yang
Greta Yorsh

speaker Reinhard Wilhelm (Saarbrücken, Germany) gave a talk on the analysis of timing models by means of abstract interpretation.

This year we took up an initiative started in 2002 and solicited tutorials that provided opportunities to get detailed insights into some validation tools and the methodologies of their use. Out of 3 submissions, the program committee selected 2 tutorials.

- An "advanced SPIN tutorial" giving an overview of recent extensions of the SPIN model checker as well as some methodological advice for its use. It was mainly addressed to users who want to use SPIN as a modelling and validation environment.
- A tutorial on the IF validation environment providing an overview of the IF modelling language and the main functionalities of the validation toolbox. It was addressed to users who want to use IF as a validation environment by feeding it with models in the IF language, or in SDL or UML, but also to tool developers who would like to interface their tools with the IF environment.

Acknowledgements. The volume editors wish to thank all members of the program committee as well as the external reviewers for their tremendous effort that led to the selection of this year's program. We furthermore wish to thank the organizers of ETAPS 2004 for inviting us to hold SPIN 2004 as a satellite event and for their support and flexibility in accommodating the particular needs of the SPIN workshop. We wish to thank in particular Fernando Orejas and Jordi Cortadella. Finally, we wish to thank Springer-Verlag for providing us with the possibility to use a conference online service free of charge, and the METAFrame team, in particular Martin Karusseit, for their very valuable and reactive support.

January 2004

Susanne Graf
Laurent Mounier

Lecture Notes in Computer Science 2989

Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Hong Kong
London
Milan
Paris
Tokyo

Lecture Notes in Computer Science

For information about Vols. 1–2876

please contact your bookseller or Springer-Verlag

Vol. 2996: V. Diekert, M. Habib (Eds.), STACS 2004. XVI, 658 pages. 2004.

Vol. 2995: C. Jensen, S. Poslad, T. Dimitrakos (Eds.), Trust Management. XIII, 377 pages. 2004.

Vol. 2993: R. Alur, G.J. Pappas (Eds.), Hybrid Systems: Computation and Control. XII, 674 pages. 2004.

Vol. 2992: E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, E. Ferrari (Eds.), Advances in Database Technology - EDBT 2004. XVIII, 877 pages. 2004.

Vol. 2991: R. Alt, A. Frommer, R.B. Kearfott, W. Luther (Eds.), Numerical Software with Result Verification. X, 315 pages. 2004.

Vol. 2989: S. Graf, L. Mounier (Eds.), Model Checking Software. X, 309 pages. 2004.

Vol. 2988: K. Jensen, A. Podelski (Eds.), Tools and Algorithms for the Construction and Analysis of Systems. XIV, 608 pages. 2004.

Vol. 2987: I. Walukiewicz (Ed.), Foundations of Software Science and Computation Structures. XIII, 529 pages. 2004.

Vol. 2985: E. Duesterwald (Ed.), Compiler Construction. X, 313 pages. 2004.

Vol. 2984: M. Wermelinger, T. Margaria-Steffen (Eds.), Fundamental Approaches to Software Engineering. XII, 389 pages. 2004.

Vol. 2983: S. Istrail, M.S. Waterman, A. Clark (Eds.), Computational Methods for SNPs and Haplotype Inference. IX, 153 pages. 2004. (Subseries LNBI).

Vol. 2982: N. Wakamiya, M. Solarski, J. Sterbenz (Eds.), Active Networks. XI, 308 pages. 2004.

Vol. 2981: C. Müller-Schloer, T. Ungerer, B. Bauer (Eds.), Organic and Pervasive Computing – ARCS 2004. XI, 339 pages. 2004.

Vol. 2980: A. Blackwell, K. Marriott, A. Shimojima (Eds.), Diagrammatic Representation and Inference. XV, 448 pages. 2004. (Subseries LNAI).

Vol. 2978: R. Groz, R.M. Hierons (Eds.), Testing of Communicating Systems. XII, 225 pages. 2004.

Vol. 2977: G. Di Marzo Serugendo, A. Karageorgos, O.F. Rana, F. Zambonelli (Eds.), Engineering Self-Organising Systems. X, 299 pages. 2004. (Subseries LNAI).

Vol. 2976: M. Farach-Colton (Ed.), LATIN 2004: Theoretical Informatics. XV, 626 pages. 2004.

Vol. 2973: Y. Lee, J. Li, K.-Y. Whang, D. Lee (Eds.), Database Systems for Advanced Applications. XXIV, 925 pages. 2004.

Vol. 2970: F. Fernández Rivera, M. Bubak, A. Gómez Tato, R. Doallo (Eds.), Grid Computing. XI, 328 pages. 2004.

Vol. 2964: T. Okamoto (Ed.), Topics in Cryptology – CT-RSA 2004. XI, 387 pages. 2004.

Vol. 2963: R. Sharp, Higher Level Hardware Synthesis. XVI, 195 pages. 2004.

Vol. 2962: S. Bistarelli, Semirings for Soft Constraint Solving and Programming. XII, 279 pages. 2004.

Vol. 2961: P. Eklund (Ed.), Concept Lattices. IX, 411 pages. 2004. (Subseries LNAI).

Vol. 2960: P.D. Mosses, CASL Reference Manual. XVII, 528 pages. 2004.

Vol. 2958: L. Rauchwerger (Ed.), Languages and Compilers for Parallel Computing. XI, 556 pages. 2004.

Vol. 2957: P. Langendoerfer, M. Liu, I. Matta, V. Tsaoussidis (Eds.), Wired/Wireless Internet Communications. XI, 307 pages. 2004.

Vol. 2954: F. Crestani, M. Dunlop, S. Mizzaro (Eds.), Mobile and Ubiquitous Information Access. X, 299 pages. 2004.

Vol. 2953: K. Konrad, Model Generation for Natural Language Interpretation and Analysis. XIII, 166 pages. 2004. (Subseries LNAI).

Vol. 2952: N. Guelfi, E. Astesiano, G. Reggio (Eds.), Scientific Engineering of Distributed Java Applications. X, 157 pages. 2004.

Vol. 2951: M. Naor (Ed.), Theory of Cryptography. XI, 523 pages. 2004.

Vol. 2949: R. De Nicola, G. Ferrari, G. Meredith (Eds.), Coordination Models and Languages. X, 323 pages. 2004.

Vol. 2948: G.L. Mullen, A. Poli, H. Stichtenoth (Eds.), Finite Fields and Applications. VIII, 263 pages. 2004.

Vol. 2947: F. Bao, R. Deng, J. Zhou (Eds.), Public Key Cryptography – PKC 2004. XI, 455 pages. 2004.

Vol. 2946: R. Focardi, R. Gorrieri (Eds.), Foundations of Security Analysis and Design II. VII, 267 pages. 2004.

Vol. 2943: J. Chen, J. Reif (Eds.), DNA Computing. X, 225 pages. 2004.

Vol. 2941: M. Wirsing, A. Knapp, S. Balsamo (Eds.), Radical Innovations of Software and Systems Engineering in the Future. X, 359 pages. 2004.

Vol. 2940: C. Lucena, A. Garcia, A. Romanovsky, J. Castro, P.S. Alencar (Eds.), Software Engineering for Multi-Agent Systems II. XII, 279 pages. 2004.

Vol. 2939: T. Kalker, I.J. Cox, Y.M. Ro (Eds.), Digital Watermarking. XII, 602 pages. 2004.

Vol. 2937: B. Steffen, G. Levi (Eds.), Verification, Model Checking, and Abstract Interpretation. XI, 325 pages. 2004.

Vol. 2934: G. Lindemann, D. Moldt, M. Paolucci (Eds.), Regulated Agent-Based Social Systems. X, 301 pages. 2004. (Subseries LNAI).

Vol. 2930: F. Winkler (Ed.), Automated Deduction in Geometry. VII, 231 pages. 2004. (Subseries LNAI).

Vol. 2926: L. van Elst, V. Dignum, A. Abecker (Eds.), Agent-Mediated Knowledge Management. XI, 428 pages. 2004. (Subseries LNAI).

Vol. 2923: V. Lifschitz, I. Niemelä (Eds.), Logic Programming and Nonmonotonic Reasoning. IX, 365 pages. 2004. (Subseries LNAI).

Vol. 2919: E. Giunchiglia, A. Tacchella (Eds.), Theory and Applications of Satisfiability Testing. XI, 530 pages. 2004.

Vol. 2917: E. Quintarelli, Model-Checking Based Data Retrieval. XVI, 134 pages. 2004.

Vol. 2916: C. Palamidessi (Ed.), Logic Programming. XII, 520 pages. 2003.

Vol. 2915: A. Camurri, G. Volpe (Eds.), Gesture-Based Communication in Human-Computer Interaction. XIII, 558 pages. 2004. (Subseries LNAI).

Vol. 2914: P.K. Pandya, J. Radhakrishnan (Eds.), FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science. XIII, 446 pages. 2003.

Vol. 2913: T.M. Pinkston, V.K. Prasanna (Eds.), High Performance Computing - HiPC 2003. XX, 512 pages. 2003. (Subseries LNAI).

Vol. 2911: T.M.T. Sembok, H.B. Zaman, H. Chen, S.R. Urs, S.H. Myaeng (Eds.), Digital Libraries: Technology and Management of Indigenous Knowledge for Global Access. XX, 703 pages. 2003.

Vol. 2910: M.E. Orlowska, S. Weerawarana, M.M.P. Papazoglou, J. Yang (Eds.), Service-Oriented Computing - ICSOC 2003. XIV, 576 pages. 2003.

Vol. 2909: R. Solis-Oba, K. Jansen (Eds.), Approximation and Online Algorithms. VIII, 269 pages. 2004.

Vol. 2909: K. Jansen, R. Solis-Oba (Eds.), Approximation and Online Algorithms. VIII, 269 pages. 2004.

Vol. 2908: K. Chae, M. Yung (Eds.), Information Security Applications. XII, 506 pages. 2004.

Vol. 2907: I. Lirkov, S. Margenov, J. Wasniewski, P. Yalamov (Eds.), Large-Scale Scientific Computing. XI, 490 pages. 2004.

Vol. 2906: T. Ibaraki, N. Katoh, H. Ono (Eds.), Algorithms and Computation. XVII, 748 pages. 2003.

Vol. 2905: A. Sanfeliu, J. Ruiz-Shulcloper (Eds.), Progress in Pattern Recognition, Speech and Image Analysis. XVII, 693 pages. 2003.

Vol. 2904: T. Johansson, S. Maitra (Eds.), Progress in Cryptology - INDOCRYPT 2003. XI, 431 pages. 2003.

Vol. 2903: T.D. Gedeon, L.C.C. Fung (Eds.), AI 2003: Advances in Artificial Intelligence. XVI, 1075 pages. 2003. (Subseries LNAI).

Vol. 2902: F.M. Pires, S.P. Abreu (Eds.), Progress in Artificial Intelligence. XV, 504 pages. 2003. (Subseries LNAI).

Vol. 2901: F. Bry, N. Henze, J. Ma luszyński (Eds.), Principles and Practice of Semantic Web Reasoning. X, 209 pages. 2003.

Vol. 2900: M. Bidoit, P.D. Mosses (Eds.), Casl User Manual. XIII, 240 pages. 2004.

Vol. 2899: G. Ventre, R. Canonico (Eds.), Interactive Multimedia on Next Generation Networks. XIV, 420 pages. 2003.

Vol. 2898: K.G. Paterson (Ed.), Cryptography and Coding. IX, 385 pages. 2003.

Vol. 2897: O. Balet, G. Subsol, P. Torguet (Eds.), Virtual Storytelling. XI, 240 pages. 2003.

Vol. 2896: V.A. Saraswat (Ed.), Advances in Computing Science – ASIAN 2003. VIII, 305 pages. 2003.

Vol. 2895: A. Ohori (Ed.), Programming Languages and Systems. XIII, 427 pages. 2003.

Vol. 2894: C.S. Laih (Ed.), Advances in Cryptology - ASIACRYPT 2003. XIII, 543 pages. 2003.

Vol. 2893: J.-B. Stefani, I. Demeure, D. Hagimont (Eds.), Distributed Applications and Interoperable Systems. XIII, 311 pages. 2003.

Vol. 2892: F. Dau, The Logic System of Concept Graphs with Negation. XI, 213 pages. 2003. (Subseries LNAI).

Vol. 2891: J. Lee, M. Barley (Eds.), Intelligent Agents and Multi-Agent Systems. X, 215 pages. 2003. (Subseries LNAI).

Vol. 2890: M. Broy, A.V. Zamulin (Eds.), Perspectives of System Informatics. XV, 572 pages. 2003.

Vol. 2889: R. Meersman, Z. Tari (Eds.), On The Move to Meaningful Internet Systems 2003: OTM 2003 Workshops. XIX, 1071 pages. 2003.

Vol. 2888: R. Meersman, Z. Tari, D.C. Schmidt (Eds.), On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE. XXI, 1546 pages. 2003.

Vol. 2887: T. Johansson (Ed.), Fast Software Encryption. IX, 397 pages. 2003.

Vol. 2886: I. Nyström, G. Sanniti di Baja, S. Svensson (Eds.), Discrete Geometry for Computer Imagery. XII, 556 pages. 2003.

Vol. 2885: J.S. Dong, J. Woodcock (Eds.), Formal Methods and Software Engineering. XI, 683 pages. 2003.

Vol. 2884: E. Najm, U. Nestmann, P. Stevens (Eds.), Formal Methods for Open Object-Based Distributed Systems. X, 293 pages. 2003.

Vol. 2883: J. Schaeffer, M. Müller, Y. Björnsson (Eds.), Computers and Games. XI, 431 pages. 2003.

Vol. 2882: D. Veit, Matchmaking in Electronic Markets. XV, 180 pages. 2003. (Subseries LNAI).

Vol. 2881: E. Horlait, T. Magedanz, R.H. Glitho (Eds.), Mobile Agents for Telecommunication Applications. IX, 297 pages. 2003.

Vol. 2880: H.L. Bodlaender (Ed.), Graph-Theoretic Concepts in Computer Science. XI, 386 pages. 2003.

Vol. 2879: R.E. Ellis, T.M. Peters (Eds.), Medical Image Computing and Computer-Assisted Intervention - MICCAI 2003. XXXIV, 1003 pages. 2003.

Vol. 2878: R.E. Ellis, T.M. Peters (Eds.), Medical Image Computing and Computer-Assisted Intervention - MICCAI 2003. XXXIII, 819 pages. 2003.

Vol. 2877: T. Böhme, G. Heyer, H. Unger (Eds.), Innovative Internet Community Systems. VIII, 263 pages. 2003.

Table of Contents

Abstraction and Symbolic Methods

Applications

Tutorials

Formal Analysis of Processor Timing Models

Reinhard Wilhelm*

Informatik
Universität des Saarlandes
Saarbrücken

Abstract. Hard real-time systems need methods to determine upper bounds for their execution times, usually called worst-case execution times. This talk gives an introduction into state-of-art Timing-Analysis methods. These use Abstract Interpretation to predict the system's behavior on the underlying processor's components and Integer Linear Programming to determine a worst-case path through the program. The abstract interpretation is based on an abstract processor model that is conservative with respect to the timing behavior of the concrete processor. Ongoing work is reported to analyze abstract processor models for properties that have a strong influence on the expected precision of timing prediction and also on the architecture of the timing-analysis tool. Some of the properties we are interested in can be model checked.

1 WCET Determination

Hard real-time systems need methods to determine upper bounds for their execution times, usually called worst-case execution times, (WCET). Based on these bounds, a schedulability analysis can check whether the underlying hardware is fast enough to execute the system's task such that they all finish before their deadlines. This problem is nontrivial because performance-enhancing architectural features such as caches, pipelines, and branch prediction introduce "local non-determinism" into the processor behavior; local inspection of the program can not determine what the contribution of an instruction to the program's overall execution time is. The execution history determines whether the instruction's memory accesses hit or miss the cache, whether the pipeline units needed by the instruction are occupied or not, and whether branch prediction is correct or not.

2 Tool Architecture

State-of-art Timing-Analysis methods split the task into (at least) two subtasks, the prediction of the task's behavior on the processor components such as caches and pipelines, formerly called "micro-architecture modeling" [HBW94], and the determination of a worst-case path. They use Abstract Interpretation to predict

* Work reported herein is supported by the Transregional Collaborative Research Center AVACS of the Deutsche Forschungsgemeinschaft.

S. Graf and L. Mounier (Eds.): SPIN 2004, LNCS 2989, pp. 1–4, 2004.

the system's behavior on the underlying processor's components and Integer Linear Programming to determine a worst-case path through the program [LMW99]. A typical tool architecture is the one of aiT, the tool developed and marketed by AbsInt Angewandte Informatik in Saarbrücken, cf. Fig. 1.

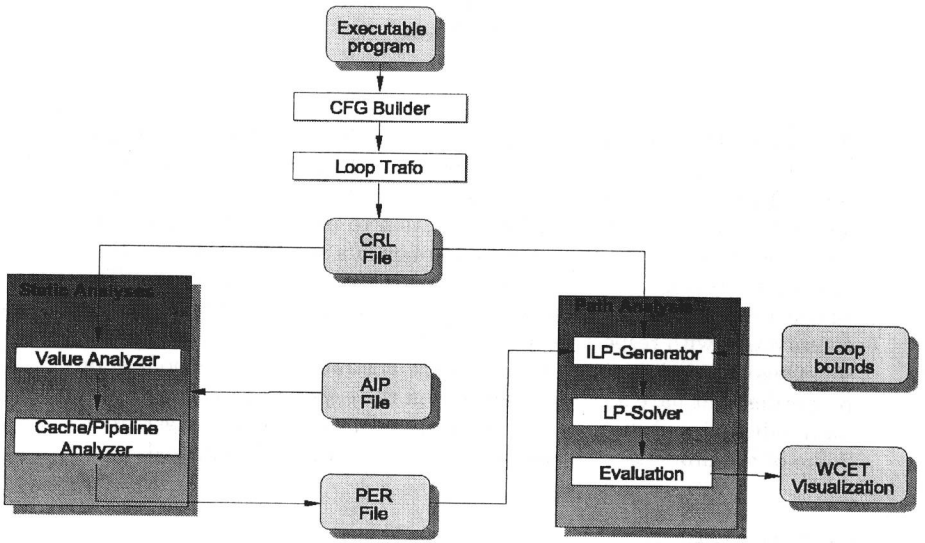

Fig. 1. Architecture of the aiT WCET analysis tool

The articles [FHL+01,LTH02] report on WCET tool developments for complex processor architectures, namely the Motorola ColdFire 5307 and the Motorola PowerPC 755. These were the first fully covered complex processors.

3 Timing Anomalies

The architecture of WCET–tools and the precision of the results of WCET analyses strongly depend on the architecture of the employed processor [HLTW03]. Out-of-order execution and control speculation introduce interferences between processor components, e.g. caches, pipelines, and branch prediction units. These interferences forbid modular designs of WCET tools, which would execute the subtasks of WCET analysis consecutively. Instead, complex integrated designs are needed resulting in high demand for space and analysis time.

In the following, several such properties of processor architectures are described. They cause the processor to display what is called *Timing Anomalies* [Lun02]. Timing anomalies are contra-intuitive influences of the (local) execution time of one instruction on the (global) execution time of the whole program. The interaction of several processor features can interact in such a way

that a locally faster execution of an instruction can lead to a globally longer execution time of the whole program. This is only the first case of a timing anomaly. The general case is the following. Different assumption about the processor's execution state, e.g. the fact that the instruction is or is not in the instruction cache, will result in a difference ΔT_1 of the execution time of the instruction between these two cases. Either assumption may lead to a difference ΔT of the global execution time compared to the other one. We say that a timing anomaly occurs if either

$\Delta T_1 < 0$ i.e., the instruction executes faster, and
- $\Delta T < \Delta T_1$, the overall execution is accelerated by more than the acceleration of the instruction, or
- $\Delta T > 0$, the program runs longer than before.

$\Delta T_1 > 0$ i.e., the instruction takes longer to execute, and
- $\Delta T > \Delta T_1$ i.e., the overall execution is extended by more than the delay of the instruction, or
- $\Delta T < 0$ i.e., the overall execution is the program takes less time to execute than before.

The case $\Delta T_1 < 0 \wedge \Delta T > 0$ is a critical case for WCET analysis. It makes it impossible to use local worst case scenarios for WCET computation. This necessitates a conservative, i.e., upper approximation to the damages potentially caused by all cases or forces the analysis to follow all possible scenarios.

Unfortunately, as [LS99,Lun02] have observed, the worst case penalties imposed by a timing anomaly may not be bounded by an architecture-dependent, but program-independent constant, but may depend on the program size. This is the so-called *Domino Effect*. This domino effect was shown to exist for the Motorola PowerPC 755 in [Sch03].

4 Formal Analysis of Processor Timing Models

The abstract-interpretation-based timing analysis is based on abstract processor models that are conservative with respect to the timing behavior of the concrete processors. To prove this is a major endeavor to be undertaken in the Transregional Collaborative Research Center AVACS. Another line of research is the derivation of processor timing models from formal specifications in VHDL or Verilog.

We are currently applying formal analysis of timing models to check for relevant properties, e.g., use model checking to detect timing anomalies and domino effects or their absence, resp. Bounded model checking can be used to check for the existence of upper bounds on the damage done by one processor component onto the state of another one, e.g. the damage of a branch misprediction to the instruction cache by loading superfluous instructions. The bound can be computed from architectural parameters, such as the depth of the pipeline and the length of prefetch queues.

Acknowledgements. Thanks go to Stephan Thesing for clarifications about timing anomalies.

References

[FHL+01] C. Ferdinand, R. Heckmann, M. Langenbach, F. Martin, M. Schmidt, H. Theiling, S. Thesing, and R. Wilhelm. WCET Determination for a Real-Life Processor. In T.A. Henzinger and C. Kirsch, editors, *Embedded Software*, volume 2211 of *Lecture Notes in Computer Science*, pages 469 – 485. Springer, 2001.

[HBW94] Marion G. Harmon, T.P. Baker, and David B. Whalley. A Retargetable Technique for Predicting Execution Time of Code Segments. *Real-Time Systems*, 7:159–182, 1994.

[HLTW03] Reinhold Heckmann, Marc Langenbach, Stephan Thesing, and Reinhard Wilhelm. The influence of processor architecture an the design and the results of WCET tools. *IEEE Proceedings on Real-Time Systems*, 91(7):1038–1054, July 2003.

[LMW99] Yau-Tsun Steven Li, Sharad Malik, and Andrew Wolfe. Performance estimation of embedded software with instruction cache modeling. *Design Automation of Electronic Systems*, 4(3):257–279, 1999.

[LS99] Thomas Lundqvist and Per Stenström. Timing anomalies in dynamically scheduled microprocessors. In *Proceedings of the 20th IEEE Real-Time Systems Symposium (RTSS'99)*, pages 12–21, December 1999.

[LTH02] M. Langenbach, S. Thesing, and R. Heckmann. Pipeline Modelling for Timing Analysis. In *Static Analysis Symposium*, volume 2274 of *LNCS*, pages 294–309. Springer Verlag, 2002.

[Lun02] Thomas Lundqvist. *A WCET Analysis Method for Pipelined Microprocessors with Cache Memories*. PhD thesis, Chalmers University of Technology, Göteborg, Sweden, 2002.

[Sch03] Joern Schneider. *Combined Schedulability and WCET Analysis for Real-Time Operating Systems*. PhD thesis, Saarland University, 2003.