

G. v. Bochmann D.K. Probst (Eds.)

cc01-663

Computer Aided Verification

Fourth International Workshop, CAV '92
Montreal, Canada, June 29 - July 1, 1992
Proceedings

Springer-Verlag

Berlin Heidelberg New York
London Paris Tokyo
Hong Kong Barcelona
Budapest

Series Editors

Gerhard Goos
 Universität Karlsruhe
 Postfach 69 80
 Vincenz-Priessnitz-Straße 1
 W-7500 Karlsruhe, FRG

Juris Hartmanis
 Cornell University
 Department of Computer Science
 4130 Upson Hall
 Ithaca, NY 14853, USA

Volume Editors

Gregor von Bochmann
 Departement d'IRO, Université de Montreal
 P. O. Box 6128, Station A, Montreal, Quebec, H3C 3J7, Canada

David Karl Probst
 Department of Computer Science, Concordia University
 1455 de Maisonneuve West, Montreal, Quebec, H3G 1M8, Canada

CR Subject Classification (1991): F.3, D.2.4, B.7, C.2.2

622

ISBN 3-540-56496-9 Springer-Verlag Berlin Heidelberg New York
 ISBN 0-387-56496-9 Springer-Verlag New York Berlin Heidelberg

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag. Violations are liable for prosecution under the German Copyright Law.

© Springer-Verlag Berlin Heidelberg 1993
 Printed in Germany

Typesetting: Camera ready by author/editor
 45/3140-543210 - Printed on acid-free paper

Preface

This is the Proceedings of the Fourth Workshop on Computer-Aided Verification (CAV '92), held in Montreal, June 29 - July 1, 1992. The objective of this series of workshops is to bring together researchers and practitioners interested in the development and use of methods, tools and theories for the computer-aided verification of concurrent systems. The workshops provide an opportunity for comparing various verification methods and practical tools that can be used to assist the applications designer. Emphasis is placed on new research results and the application of existing results to real verification problems.

Of the 75 papers that were submitted, 31 were accepted for presentation. Leslie Lamport gave the invited talk on hierarchical structure in proofs. Amir Pnueli was the banquet speaker. There were sessions devoted to Reduction Techniques, Proof Checking, Symbolic Verification, Timing Verification, Partial-Order Approaches, Case Studies, Model and Proof Checking, and Other Approaches.

Financial support was provided by Concordia University, Computer Research Institute of Montreal (CRIM), Bell Northern Research (BNR), the IDACOM-NSERC-CWARC Industrial Research Chair on Communication Protocols, the Institut National de la Recherche Scientifique (INRS-Telecommunications), the Natural Sciences and Engineering Research Council of Canada (NSERC), and the University of Montreal.

The Program Committee reviewed, managed other reviewers, and helped in the establishment of the program. The Steering Committee, consisting of E.M. Clarke (Carnegie Mellon University), R.P. Kurshan (AT&T Bell Laboratories), A. Pnueli (Weizmann Institute), and J. Sifakis (LGI-IMAG), reviewed and offered council at appropriate moments. This year, the Program Committee members were: R. Alur (AT&T Bell Labs), R. Brayton (UC Berkeley), E. Brinksma (U. Twente), E. Cerny (U. Montreal), C. Courcoubetis (U. Crete), R. de Simone (INRIA), D. Dill (Stanford U.), A. Emerson (UT Austin), O. Grumberg (Technion), H. Hiraishi (Kyoto Sangyo U.), G. Holzmann (AT&T Bell Labs), W.A. Hunt Jr. (CLI), K. Larsen (Aalborg U.), P. Loewenstein (Sun), A. Mazurkiewicz (Polish Acad. Sci.), L. Paulson (Cambridge U.), D.K. Probst (Concordia U.), B. Steffen (TU Aachen), D. Taubner (sd&m GmbH) and P. Wolper (U. Liege). The names of additional reviewers are listed on the following page.

Gregor v. Bochmann was General and Program Chair. David K. Probst was Local Arrangements Chair and much more. Lucie Levesque was Registration Chair and resource person. Anindya Das was Treasurer. Stan Swiercz and Daniel Ouimet provided technical support for tool demonstrations. Most of the articles in this volume were typeset using the \LaTeX document preparation system and Springer-Verlag's LNCS style file, slightly modified.

Montreal, January 1993

Gregor v. Bochmann
David K. Probst

Additional Reviewers

P. Attie (UT Austin), A. Aziz (UC Berkeley), W. Baker (UC Berkeley), F. Balarin (UC Berkeley), D. Barnard (TU Munich), H. Baumer (U Twente), R. Bayardo (UT Austin), O. Bernholtz (Technion), A. Borjesson (Aalborg U), B. Botma (U Twente), A. Bouajjani (LGI-IMAG), A. Bouali (INRIA), G. Boudol (INRIA), O. Burkart (RWTH Aachen), I. Castellani (INRIA), A. Claen (RWTH Aachen), H. Eertink (U Twente), P. Eijk (U Twente), U. Engberg (Aarhus U), T. Filkorn (Siemens), N. Francez (Technion), M. Fujita (Fujitsu), H. Garavel (Verilog), P. Godefroid (U Liege), C. Godskesen (Aalborg U), M. Gordon (Cambridge), S. Graf (LGI-IMAG), P. Gutwin (UC Berkeley), N. Halbwachs (LGI-IMAG), K. Hamaguchi (Kyoto U), T. Henzinger (Cornell U), R. Hojati (UC Berkeley), A. Hu (Stanford U), H. Huttel (Aalborg U), C. Jard (IRISA), T. Jeron (Alcatel), C. Jutla (IBM), M. Kaltenbach (UT Austin), T. Kam (UC Berkeley), P. Kars (U Twente), S. Katz (Technion), S. Kimura (Kobe U), A. Kindler (RWTH Aachen), M. Klein (RWTH Aachen), J. Knoop (Kiel), S. Krishnan (UC Berkeley), W. Lam (UC Berkeley), R. Langerak (U Twente), L. Lavagno (UC Berkeley), C. Loiseaux (LGI-IMAG), A. Mader (TU Munich), J. Makowsky (Technion), A. Mendelson (Technion), F. Mignard (INRIA), C. Moon (UC Berkeley), D. Ouimet (U Montreal), R. Rajaraman (UT Austin), C. Rattel (LGI-IMAG), D. Russinoff (CLI), S. Sagiv (Haifa), A. Scholz (Siemens), M. Sekine (UC Berkeley), N. Shankar (SRI), T. Shiple (UC Berkeley), M. Sinderen (U Twente), A. Skou (Aalborg U), P. Stephan (UC Berkeley), J. Tretmans (U Twente), F. Vaandrager (INRIA), J. Vaucher (U Montreal), T. Villa (UC Berkeley), H. Wang (UC Berkeley), C. Weise (RWTH Aachen), G. Whitcomb (UC Berkeley), H. Wong-Toi (Stanford U), W. Yi (Aalborg U), M. Yoeli (Technion), G. York (UC Berkeley), S. Yovine (LGI-IMAG).

Table of Contents

Invited Lecture:

Computer-Hindered Verification (Humans Can Do It Too) <i>Leslie Lamport (DEC SRC)</i>	1
--	---

Session: Reduction Techniques I

Modular Abstractions for Verifying Real-Time Distributed Systems <i>Hana De-Leon and Orna Grumberg (Technion)</i>	2
Layering Techniques for Development of Parallel Systems <i>Mannes Poel and Job Zwiers (U. Twente)</i>	16
Efficient Local Correctness Checking <i>Kim Larsen (Aalborg U.)</i>	30

Session: Proof Checking

Mechanical Verification of Concurrent Systems with TLA <i>Urban Engberg (Aarhus U.), Peter Grønning (TU Denmark) and Leslie Lamport (DEC SRC)</i>	44
Using a Theorem Prover for Reasoning about Concurrent Algorithms <i>Joakim von Wright and Thomas Långbacka (Åbo Akademi U.)</i>	56
Verifying a Logic Synthesis Tool in Nuprl: A Case Study in Software Verification <i>Mark Aagaard and Miriam Leeser (Cornell U.)</i>	69

Session: Symbolic Verification I

Higher-Level Specification and Verification with BDDs <i>Alan Hu, David Dill, Andreas Drexler and Han Yang (Stanford U.)</i>	82
Symbolic Bisimulation Minimisation <i>Amar Bouali (ENSMP) and Robert de Simone (INRIA)</i>	96
Towards a Verification Technique for Large Synchronous Circuits <i>Prabhat Jain, Prabhakar Kudva and Ganesh Gopalakrishnan (U. Utah)</i>	109

Session: Timing Verification I

Verifying Timed Behavior Automata with Nonbinary Delay Constraints <i>David Probst and Hon Li (Concordia U.)</i>	123
Timing Verification by Successive Approximation <i>Rajeev Alur, Alon Itai, Robert Kurshan and Mihalis Yannakakis (AT&T Bell Labs)</i>	137
A Verification Strategy for Timing-Constrained Systems <i>Felice Balarin and Alberto Sangiovanni-Vincentelli (UC Berkeley)</i>	151

Session: Partial-Order Approaches

Using Unfoldings to Avoid the State Explosion Problem in the Verification of Asynchronous Circuits

Kenneth McMillan (Carnegie Mellon U.)..... 164

State-Space Caching Revisited

Patrice Godefroid (U. Liège), Gerard Holzmann (AT&T Bell Labs) and

Didier Pirotin (U. Liège)..... 178

Session: Case Studies

Verification in Process Algebra of the Distributed Control of Track Vehicles - A Case Study (Extended Abstract)

Siegfried Fischer, Andreas Scholz and Dirk Taubner (Siemens AG)..... 192

Design Verification of a Microprocessor Using Branching Time Regular Temporal Logic

Kiyoharu Hamaguchi (Kyoto U.), Hiromi Hiraishi (Kyoto Sangyo U.) and

Shuzo Yajima (Kyoto U.)..... 206

A Case Study in Safety-Critical Design

Glenn Bruns (U. Edinburgh)..... 220

Session: Reduction Techniques II

Automatic Reduction in CTL Compositional Model Checking

Thomas Shiple (UC Berkeley), Massimiliano Chiodo (Magnet Marelli),

Alberto Sangiovanni-Vincentelli and Robert Brayton (UC Berkeley)..... 234

Compositional Model Checking for Linear-Time Temporal Logic

Roope Kaivola (U. Helsinki)..... 248

Property Preserving Simulations

Saddek Bensalem, Ahmed Bouajjani, Claire Loiseaux and Joseph Sifakis

(IMAG-LGI)..... 260

Session: Timing Verification II

Verification with Real-Time COSPAN

Costas Courcoubetis (U. Crete), David Dill (Stanford U.), Magda Chatzaki and

Panagiotis Tzounakis (U. Crete)..... 274

Model-Checking for Real-Time Systems Specified in Lotos

Nathalie Rico, Gregor von Bochmann (U. Montreal) and Omar Cherkaoui

(UQAM)..... 288

Decidability of Bisimulation Equivalences for Parallel Timer Processes

Kārlis Čerāns (U. Latvia)..... 302

Session: Model and Proof Checking**A Proof Assistant for Symbolic Model-Checking***Julian Bradfield (U. Edinburgh)* 316**Tableau Recycling***Angelika Mader (TU Munich)* 330**Crococ: An Integrated Environment for Interactive Verification of SDL Specifications***Dominique Méry and Abdelillah Mokkedem (CRIN and INRIA)* 343**Session: Other Approaches****Verifying General Safety and Liveness Properties with Integer Programming***James Corbett (U. Massachusetts)* 357**Generating Diagnostic Information for Behavioral Preorders***Ufuk Celikkan and Rance Cleaveland (North Carolina State U.)* 370**A Verification Procedure via Invariant for Extended Communicating Finite-State Machines***Masahiro Higuchi, Osamu Shirakawa, Hiroyuki Seki, Mamoru Fujii (Osaka U.) and Tadao Kasami (AIST, Nara)* 384**Session: Symbolic Verification II****Efficient ω -Regular Language Containment***Ramin Hojati (UC Berkeley), Hervé Touati (DEC PRL), Robert Kurshan (AT&T Bell Labs) and Robert Brayton (UC Berkeley)* 396**Faster Model Checking for the Modal Mu-Calculus***Rance Cleaveland (North Carolina State U.), Marion Klein and Bernhard Steffen (RWTH Aachen)* 410

Modular Abstractions for Verifying Real-Time Distributed Systems

Hana De-Leon and Orna Grumberg
 Computer Science Department
 The Technion
 Haifa 32000, Israel
 orna@cs.technion.ac.il

1 Introduction

Temporal logics are widely used as languages for specifying system behaviors. Within temporal logics, qualitative reference to time (e.g., “*eventually*”, “*always*”) is possible. Real-time systems are systems in which the correct behavior depends not only on the actions performed, but also on the time duration of each action. In order to specify the behavior of such systems, quantitative reference to time is necessary. Real-time temporal logics include quantitative reference to time (e.g., “*within 5 time units*”) and therefore are suitable to express properties of real-time systems.

Considerable research has recently been done on the specification and verification of finite-state real-time systems ([ACD90], [AH89], [Alu91], [EMSS89], [Ha88], [HLP90], and others). A number of specification languages have been suggested, models to describe real-time systems have been presented and the problems of satisfiability and model checking have been investigated. A comprehensive survey of recent work appears in [AH91].

In this work, we address the verification of finite-state, real-time distributed systems. The number of states of a distributed system may grow exponentially with the number of its components, thus state explosion may occur. Following solutions that work well with untimed systems, our verification methodology is based on two main ideas, modularity and abstraction. We exploit the modular structure of the system to reduce each of the components by abstracting away from details that are irrelevant for the required specification. The abstract components are then composed to form an abstract system to which a model checking procedure is applied. The abstraction relation and the specification language are chosen to guarantee that if the abstract system satisfies a specification then the original system satisfies it as well.

Since each of the components is abstracted independently, our methodology enables easy and modular changes of a verified distributed system. When replacing a component in an already verified system, it is enough to show that the original component is an abstraction of the new one. This immediately implies that the new system satisfies the specification. Thus, an application of a model checking procedure to the new system is avoided. Moreover, by proving a system correct we actually prove correct a whole family of instantiations of the system.

Similar verification methodologies appear in [Ku90] for the linear-time case and in [SG90] and [GL91] for the branching-time case. However, none of them considers the real-time framework. Real-time models for processes and composed systems are

suggested in [NS90] in the context of process algebra. However, they consider strong bisimulation while our abstraction is a preorder. Also, they do not consider any temporal specification language.

The logic *RTL* is a branching-time version of the linear-time, real-time logic *TPTL*, presented in [AH89]. A formula in the logic is:

$$E \Box x.(p \rightarrow A \Diamond y.(q \wedge y \leq x + 10))$$

It means that for *some* path, it is *always* true (at time x) that if p holds then *along every* path, *eventually* (at time y) q holds and $y \leq x + 10$. In other words, whenever p becomes true, then q becomes true within 10 time units. x and y are variables that *freeze* the value of the clock at certain events. The freezing variables range over the natural numbers. They can be compared by means of \geq and modulo a time constant.

We developed a model checking procedure for *RTL*, which is exponential in the size of the formula and linear in the size of the model. The exponent arises from the time constants, represented in the formula by $\log n$ symbols, but induce n computation steps. *TPTL* is a powerful language ([AH90]) however, model checking for *TPTL* although linear in the size of the model is double exponential in the size of the formula. Thus, we are motivated to choose the branching version of *TPTL* as our specification language. Due to lack of space, we do not present our model checking here. Its details can be found in [DG92].

Other branching-time real-time logics are suggested in [EMSS89], [ACD90] and [Alu91], all solving the model checking problem for global systems. The first one suggests a simplified model of computation in which each transition takes exactly one time unit. The others discuss dense time and it is not clear how to introduce the notions of processes and composition into this framework.

In this work, real-time processes and real-time systems are modeled by state transition graphs in which states are labeled by atomic propositions and transitions are labeled by action names and time duration. We adopt the concept of *two alternating phases* of synchronous and asynchronous behaviors, suggested in [NS90]. All processes synchronize on time progress, while between two time progresses, processes cooperate asynchronously. Processes communicate via *handshaking* message passing.

Four types of actions are introduced: internal actions, message passing actions, interrupt actions and alternative actions. The first two types are self explanatory. Interrupts are communication actions with higher priority, i.e., a process that can receive an interrupt must do so, unless it is involved in another interrupt. Alternative actions are performed by a process only if the communication actions associated with them are not enabled. Their aim is to avoid deadlocks caused by waiting for communications to be enabled. By means of alternative actions a bounded delay may be modeled.

Two notions of abstraction are introduced, each defining a preorder on the model domain. One is an adaptation of the preorder presented in [SG90] to include reference to time duration of actions. The language preserved by this abstraction is a sublanguage of *RTL*, denoted *RTL_A*, in which only universal path quantifiers are allowed. The other is based on the stuttering equivalence for *CTL_{-X}* and *CTL_{-X}**, presented in [BCG88]. This abstraction allows further reduction, since a sequence of events can be abstracted to one event, provided that this sequence is 'unobservable' with respect to the specification. i.e., all states along the sequence are identically labeled, all actions are

internal and the sequence takes zero time. Note that, since our time is discrete, zero time means time duration that is smaller than the chosen time unit. The stuttering preorder preserves the language RTL_{A-next} , which is a sublanguage of RTL_A from which the $next$ operator has been eliminated. Thus, stuttering preorder enables further reduction but preserves a smaller set of properties. We present an example in which stuttering preorder is used to verify a specification expressible in RTL_{A-next} .

The rest of the paper is organized as follows. The logic RTL is described in Section 2. In Section 3 the framework for verifying distributed systems is presented. Sections 3.1, 3.2 and 3.3 describe the structures used to model distributed systems. Section 3.4 describes the abstraction relation and the logic RTL_A . Section 3.5 presents the verification methodology and Section 3.6 includes the stuttering abstraction. In Section 4, an application of the methodology is exemplified on an alternating bit protocol. We conclude in Section 5 with a discussion of our results.

2 The logic RTL

Let AP be a set of atomic propositions and let V be a set of variables. $\forall c \in \mathbb{N}, \forall x \in V, x, c$ and $(x + c)$ are *time expressions*.

Definition : The set of atomic formulas consists of all $p \in AP$. Also, if π_1 and π_2 are time expressions then $\pi_1 \leq \pi_2$ and $\pi_1 \equiv_c \pi_2$ are atomic formulas.

We use the symbol \sim to represent the relation \equiv_c or \leq .

Definition : The logic RTL is the set of formulas inductively defined as follows:

1. If f is an atomic formula then $f \in RTL$.
2. If $f, g \in RTL$ then $\neg f, f \wedge g, A \circ x.f, E x.f U y.g, A x.f U y.g \in RTL$.

We use the following abbreviations (where E/Af denotes the formulas Ef or Af):
 $E \circ x.f = \neg A \circ x.\neg f$ $E/A \Diamond x.f = E/A \text{ true } U x.f$ $E/A \Box x.f = \neg A/E \Diamond x.\neg f$

Definition : A timed Kripke structure $M = (S, R, L)$ is a Kripke structure in which each transition is labeled with a value over the natural numbers, denoting the time duration of the transition. $(s_0, t_0), (s_1, t_1), (s_2, t_2) \dots$ is a path in M from s_0 iff $\forall i \in \mathbb{N} (s_i, t_i, s_{i+1}) \in R$.

Definition : Let T be a variable which represents the "current time" ($T \notin V$). ε is an environment iff ε is a function from the time expressions over $V \cup \{T\}$ to \mathbb{N} which satisfies the following: $\forall x \in V \cup \{T\}, \forall c \in \mathbb{N}, \varepsilon(c) = c$ and $\varepsilon(x + c) = \varepsilon(x) + \varepsilon(c)$.

Definition : Let ε be an environment. $\varepsilon[x_1 := c_1, x_2 := c_2, \dots, x_n := c_n]$ is an environment in which $\forall 1 \leq i \leq n$ the value of x_i is c_i . The values of all other variables are the same as in ε .

The semantics of RTL is defined with respect to a timed Kripke structure $M = (S, R, L)$, $s \in S$ and an environment ε . We use the notation $M, s, \varepsilon \models f$ to denote that f is true in state s , in the structure M , with respect to environment ε . M is omitted when no confusion may occur.

Definition : The satisfaction relation \models is inductively defined as follows:

1. if $a \in AP$ then $s, \varepsilon \models a$ iff $a \in L(s)$.
 $s, \varepsilon \models \neg f$ iff $s, \varepsilon \not\models f$ and $s, \varepsilon \models f \wedge g$ iff $s, \varepsilon \models f$ and $s, \varepsilon \models g$.