Information Societies Technology (IST) Programme

Future and Emerging Technologies - Open Domain

Project acronym: Advance

Project full title:

Advanced Validation Techniques
for Telecommunication Protocols

Contract no: IST-1999-29082

Deliverable 14 & 15

Methods and Tools for Parametric Reasoning
Methods and Tools for Dynamic Networks

Year 3 : October 2002 - January 2004

Date of preparation: January 2004

Contributors: Paris, Liège, Edinburgh, Grenoble, Uppsala, Weizmann Inst.

1 Introduction

We report in this section on work on verification of *parameterized systems* and *dynamic systems*. In a *parameterized system*, the description of the system is parameterized by the number of components inside the systems. For instance, the specification of a protocol may be parametrized by the number of components which may participate in a given session of the protocol. In such a case, it is interesting to verify the correctness of the protocol, regardless of the number of participants in a particular session. A *dynamic-control system* allows creation and deletion of components during the execution of the system. Such systems arise naturally in computer applications, e.g. mobile protocols, security protocols, and multi-threaded programs.

The fundamental challenge in verification of parameterized and dynamic-control systems is to cope with the fact that both types of systems have *unbounded control structures*. This is due to the fact that there is in general no bound on size of the system to be verified; or on the number of processes created during a run of a system.

During the third year of the project, we have considered the following tasks related to verification of parameterized and dynamic systems.

- Extending the paradigm of *regular model checking*, through the design of more efficient and general algorithms.
- Developing powerful methods, based on combining abstraction, proof principles, and model-checking techniques, for verifying both safety and liveness properties of parametrized networks of processes (including parametrized probabilistic algorithms).
- Developing new algorithms for several classes of parameterized and dynamic systems. Examples include multicast protocols, systems where processes compete for an arbitrary number of resources, and *Process Rewriting Systems (PRS)* which can be used to model the behavior multi-threaded programs, and concurrent programs with procedures.

Another class of dynamic system is what we call *dynamic-memory systems*. Such systems allow creation and deletion of memory cells, pointer manipulation, garbage collection, etc. This year, we have considered problems related to the specification and the verification of such systems. Our results concern the definition of specification logics allowing to reason about heap properties (shapes), and efficient verification techniques based on symmetry and partial-order reductions (for the finite-state case).

2 New Regular Model Checking Algorithms

The *Regular model checking* framework was advocated by the paper [KMM$^+$01] as a uniform framework for the analysis of parametrized systems. The idea of regular model checking is to perform symbolic reachability analysis, using regular relations as a symbolic representation for the behavior of systems with unbounded control structures. We have already devoted a lot of effort [Tou01, DLS01, Mai01, AJNd02] in the previous two years to developing techniques which allow computing the transitive closure of regular relations described by *regular transducers*. We have also extended some of these techniques from the standard theory of word transducers into that of tree transducers [AJMd02, BT02].

In the last year of the project we have made the following contributions:

- In [AJNd03] we present several improvements of the earlier developed techniques. The paper presents a simplified theoretical framework, based on commuting simulations, for showing soundness. This leads to large reductions in the size of representations of the transitive closure, which is evident in the speedups obtained by a prototype implementation.
- In [BLW03], we investigate the possibility of using generic techniques to perform *widening* in the iteration transducers. The main idea of the method is to compare successive elements of a sequence of approximations of the iteration in order to detect a so called *increment*. An increment is a portion of a transducer which is added in a specific well-defined manner, when moving from one approximation to the next one. In such a case, the sequence is extrapolated adding arbitrary repetitions of the increment. The approach exploits a number of new conceptual and algorithmic ideas, often induced with the help of experiments. This gives broad scope as well as good performance.
- In [BLW04] we extend regular model-checking to systems whose states are represented by infinite (omega) words. The feasibility and usefulness of this approach is shown through a combination of the necessary theoretical developments, implementation, and experimentation. The iteration technique that is used is adapted from [BLW03] on the iteration of finite-word. Example of an application is a leaking gas burner in which time is modeled by real variables have been handled completely within the automata-theoretic setting.

3 Liveness properties of parametrized networks of processes

The problem of verifying liveness properties of infinite-state systems is hard in general, and particularly for parametrized networks of processes. It requires the combination of several analysis approaches: abstraction techniques, proof principles, and algorithmic verification techniques such as model-checking. Indeed, techniques such as symbolic reachability analysis are not sufficient since they are more adapted to the verification of safety properties. The work developed in ADVANCE concerning the verification of liveness proposes several methods and approaches allowing to tackle automatically several nontrivial examples of parametrized systems.

- The paper [APZ03] studies automatic verification of liveness properties with probability 1 over parameterized programs that include probabilistic transitions, and proposes two novel approaches to the problem. The first approach is based on a Planner that occasionally determines the outcome of a finite sequence of "random" choices, while the other random choices are performed non-deterministically. Using a Planner, a probabilistic protocol can be treated just like a non-probabilistic one and verified as such. The second approach is based on γ-fairness, a notion of fairness that is sound and complete for verifying simple temporal properties (whose only temporal operators are $\Diamond$ and $\Box$) over finite-state systems. The paper presents a symbolic model checker based on γ-fairness. We then show how the network invariant approach can be adapted to accommodate probabilistic protocols. The utility of the Planner approach is demonstrated on a probabilistic mutual exclusion protocol.

The utility of the approach of γ-fairness with network invariants is demonstrated on Lehman and Rabin's Courteous Philosophers algorithm.

- In [PA03], we present our PVS implementation of a linear temporal logic verification system. The system includes a set of theories defining a temporal logic, a number of proof rules for proving soundness and response properties, and strategies which aid in conducting the proofs. In addition to implementing a framework for existing rules, we have also derived new methods which are particularly useful in a deductive LTL system. A distributed rank rule for the verification of response properties in parameterized systems is presented, and a methodology is detailed for reducing compassion requirements to justice requirements (strong fairness to weak fairness). Special attention has been paid to the verification of unbounded systems – systems in which the number of processes is unbounded – and our verification rules are appropriate to such systems.

- In [FPPZ04], we develop a method based on the concept of Invisible Invariants for the verification of liveness properties of parametrized networks. The method of Invisible Invariants was developed originally in order to verify safety properties of parameterized systems fully automatically. Roughly speaking, the method is based on a *small model property* that implies it is sufficient to prove some properties on small instantiations of the system, and on a heuristic that generates candidate invariants. Liveness properties usually require well founded ranking, and do not fall within the scope of the small model theorem. In this paper we develop novel proof rules for liveness properties, all of whose proof obligations are of the correct form to be handled by the small model theorem. We then develop abstraction and generalization techniques that allow for fully automatic verification of liveness properties of parameterized systems. We demonstrate the application of the method on several examples.

4 Verification algorithms for parametrized networks

- In [BHV03], we consider the problem of parametric verification over a class of systems consisting of processes competing for access to shared resources. We suppose the access to be controlled according to a FIFO-based policy with a possibility of distinguishing low-priority and high-priority resource requests. We propose a model based on automata extended with queues. By reducing the parametric verification problems to finite-state model checking, we establish several decidability results for different classes of the considered properties and systems. Moreover, we show that parametric verification against formulae with local process quantification is undecidable in the given context.

- The paper [EM03] deals with multicast protocols on tree structures. It shows that for protocols satisfying certain conditions, it is enough to check the protocol on trees of depth two (sender, one layer of internal processes, and one layer of receivers). Under stronger conditions, it shows that it suffices to check the protocol on one single instantiation. The paper proves that the PGM protocol satisfies the first set of conditions. If we assume that the protocol sends only a finite number of messages, then it also satisfies the second set of conditions.

5 Verification algorithms for concurrent/dynamic-control programs

- In [BT03], we address the problem of analyzing *Process Rewrite Systems* (*PRS* for short). These systems subsume many common (infinite-state) models such as pushdown systems and Petri nets. They can be adopted as formal models of parallel programs (multi-threaded programs) with procedure calls. We develop automata techniques allowing to build finite representations of the forward/backward sets of reachable configurations of PRS modulo various term structural equivalences. We show that, in several cases, these reachability sets can be represented by polynomial size finite bottom-up tree-automata. When associativity and commutativity of the parallel composition is taken into account, non-regular representations based on (a decidable class of) counter tree automata are sometimes needed.

- In [BET03], we present a generic approach to the static analysis of concurrent programs with procedures. We model programs as communicating pushdown systems. It is known that typical dataflow problems for this model are undecidable, through a reduction from the emptiness problem for the intersection of context-free languages. In this paper we propose an algebraic framework for defining abstractions (upper approximations) of context-free languages. We consider two classes of abstractions: finite-chain abstractions, which are abstractions whose domains do not contain any infinite chains, and commutative abstractions corresponding to classes of languages that contain a word if and only if they contain all its permutations. We show how to compute such approximations by combining automata theoretic techniques with algorithms for solving systems of polynomial inequations in Kleene algebras.

6 Specification and verification of dynamic-memory programs

- The paper [BIL03] proposes a logic-based framework for reasoning about programs manipulating dynamic memory. Pioneering work has been done by Jonkers to define a semantics of pointer manipulating programs that is abstract in the sense of ignoring low-level aspects such as dangling pointers and garbage objects. We explore in this paper the principles of such storeless semantics from a logical point of view, first defining a simple logic to completely characterize heap structures up to isomorphism. Second, we extend this language to a full-blown alias logic (AL) that allows to express regular properties of unbounded heap structures. Along the development, we present an operational storeless semantics and give sound and complete total correctness axioms for deterministic programs in the form of Hoare triples, using AL.

- The contribution of the paper [Ios03] is a framework for applying both symmetry and partial order reduction methods to a particular class of software, namely *dynamic concurrent* programs, for which the number of state components (objects, threads) is continuously modified as a result of their ongoing execution. This concept can be used to formalize the semantics of most high-level object-oriented programs, such as the ones written in Java or C++. We show how existing reduction techniques can be tailored to exploit the dynamic nature of software systems in order to achieve more effective verification results.

7 Other related works

Beside parametrized and dynamic networks, we have also produced some relevant work concerning the verification of models such as pushdown systems, and context-free processes with one integer parameter.

7.1 Model-checking LTL with regular valuations for pushdown systems [EKS03]

Recent works have proposed pushdown systems as a tool for analyzing programs with (recursive) procedures, and the model-checking problem for LTL has received special attention. However, all these works impose a strong restriction on the possible valuations of atomic propositions: whether a configuration of the pushdown system satisfies an atomic proposition or not can only depend on the current control state of the pushdown automaton and on its topmost stack symbol. In this paper we consider LTL with regular valuations: the set of configurations satisfying an atomic proposition can be an arbitrary regular language. The model-checking problem is solved via two different techniques, with an eye on efficiency. The resulting algorithms are polynomial in certain measures of the problem which are usually small, but can be exponential in the size of the problem instance. However, we show that this exponential blowup is inevitable. The extension to regular valuations allows to model problems in different areas; for instance, we show an application to the analysis of systems with checkpoints. We claim that our model-checking algorithms provide a general, unifying and efficient framework for solving them.

7.2 Context-free processes with one integer parameter [BHM03]

Context-free processes (BPA) have been used for dataflow-analysis in recursive procedures with applications in optimizing compilers. We introduce a more refined model called BPA(Z) that can model not only recursive dependencies, but also the passing of an integer parameter to subroutines. Moreover, this parameter can be tested against conditions expressible in Presburger-arithmetic. This new and more expressive model can still be analyzed automatically. We define Z-input 1-CM, a new class of one-counter machines that take integer numbers as input, to describe sets of configurations of BPA(Z). We show that the $Post^*$ (the set of successors) of a set of BPA(Z)-configurations described by a Z-input 1-CM can be effectively constructed. The Pre^* (set of predecessors) of a regular set can be effectively constructed as well. However, the Pre^* of a set described by a Z-input 1-CM cannot be represented by a Z-input 1-CM in general and has an undecidable membership problem. Then we develop a new temporal logic based on reversal-bounded counter machines that can be used to describe properties of BPA(Z) and show that the model-checking problem is decidable.

References

[AJMd02] Parosh Aziz Abdulla, Bengt Jonsson, Pritha Mahata, and Julien d'Orso. Regular tree model checking. In *Proc. 14^{th} Int. Conf. on Computer Aided Verification (CAV'02)*, volume 2404 of *Lecture Notes in Computer Science*, 2002.

[AJNd02] Parosh Aziz Abdulla, Bengt Jonsson, Marcus Nilsson, and Julien d'Orso. Regular model checking made simple and efficient. In *Proc. 13^{th} Int. Conf. on Concurrency*

Theory (CONCUR'02), volume 2421 of *Lecture Notes in Computer Science*, pages 116–130, 2002.

[AJNd03] Parosh Aziz Abdulla, Bengt Jonsson, Marcus Nilsson, and Julien d'Orso. Algorithmic improvements in regular model checking. In *Proc. 15^{th} Int. Conf. on Computer Aided Verification (CAV'03)*, volume 2725 of *Lecture Notes in Computer Science*, 2003.

[APZ03] T. Arons, A. Pnueli, and L. Zuck. Parameterized verification by probabilistic abstraction. In *Proc. Intern. Conf. on Foundations of Software Science and Computation Structures (FOSSACS'03)*, volume 2620 of *Lecture Notes in Computer Science*, 2003.

[BET03] Ahmed Bouajjani, Javier Esparza, and Tayssir Touili. A generic approach to the static analysis of concurrent programs with procedures. In *30th Symp. on Principles of Programming Languages (POPL'03)*, pages 72–73. ACM Pub., 2003. Full version in the Intern. Journal of Foundations of Computer Science, volume 14, No 4, August 2003.

[BHM03] Ahmed Bouajjani, Peter Habermehl, and Richard Mayr. Automatic verification of recursive procedures with one integer parameter. *Theoretical Computer Science*, 295:85–106, 2003.

[BHV03] Ahmed Bouajjani, Peter Habermehl, and Tomas Vojnar. Verification of parametric concurrent systems with prioritized fifo resource management. In *Proc. 14^{th} Int. Conf. on Concurrency Theory (CONCUR'03)*, volume 2761 of *Lecture Notes in Computer Science*, pages 174–190, 2003.

[BIL03] M. Bozga, R. Iosif, and Y. Lakhnech. Storeless Semantics and Alias Logic. In *Proceedings of PEPM'03 Workshop - Partial Evaluation and Semantics-Based Program Manipulation*, SIGPLAN. ACM, 2003.

[BLW03] Bernard Boigelot, Axel Legay, and Pierre Wolper. Iterating transducers in the large. In *Proc. 15^{th} Int. Conf. on Computer Aided Verification (CAV'03)*, volume 2725 of *Lecture Notes in Computer Science*, pages 223–235, 2003.

[BLW04] Bernard Boigelot, Axel Legay, and Pierre Wolper. Omega-regular model checking. In *Proc. TACAS '04, 10^{th} Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems*, Lecture Notes in Computer Science, 2004.

[BT02] Ahmed Bouajjani and Tayssir Touili. Extrapolating Tree Transformations. In *Proc. 14^{th} Int. Conf. on Computer Aided Verification (CAV'02)*, volume 2404 of *Lecture Notes in Computer Science*, 2002.

[BT03] Ahmed Bouajjani and Tayssir Touili. Reachability Analysis of Process Rewrite Systems. In *Proc. Intern. Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'03)*, Lecture Notes in Computer Science, Mumbai, India, 2003.

[DLS01] D. Dams, Y. Lakhnech, and M. Steffen. Iterating transducers. In G. Berry, H. Comon, and A. Finkel, editors, *Computer Aided Verification*, volume 2102 of *Lecture Notes in Computer Science*, 2001.

[EKS03] Javier Esparza, Antonín Kučera, and Stefan Schwoon. Model-checking LTL with regular valuations for pushdown systems. *Information and Computation*, 186(2):355–376, November 2003.

[EM03] J. Esparza and M. Maidl. Simple representative instantiations for multicast protocols. In *Proc. TACAS '03, 9^{th} Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems*, volume 2619 of *Lecture Notes in Computer Science*, pages 128–142, 2003.

[FPPZ04] Y. Fang, N. Piterman, A. Pnueli, and L. Zuck. Liveness with invisible ranking. In G. Levi and B. Steffen, editors, *Proc. of the 5^th workshop on Verification, Model Checking, and Abstract Interpretation*, Lecture Notes in Computer Science, 2004. To appear.

[Ios03] Radu Iosif. Symmetry reductions for model checking of concurrent dynamic software. *Software Tools for Technology Transfer, STTT*, 2003.

[KMM+01] Y. Kesten, O. Maler, M. Marcus, A. Pnueli, and E. Shahar. Symbolic model checking with rich assertional languages. *Theoretical Computer Science*, 256:93–112, 2001.

[Mai01] M. Maidl. A unifying model checking approach for safety properties of parameterized systems. In Berry, Comon, and Finkel, editors, *Proc. 13^th Int. Conf. on Computer Aided Verification*, volume 2102 of *Lecture Notes in Computer Science*, pages 324–336, 2001.

[PA03] A. Pnueli and T. Arons. TLPVS: A PVS-based LTL verification system. In Nachum Dershowitz, editor, *Verification–Theory and Practice: Proceedings of an International Symposium in Honor of Zohar Manna's 64th Birthday*, volume 2772 of *Lecture Notes in Computer Science*, pages 601–628, 2003.

[Tou01] T. Touili. Regular Model Checking using Widening Techniques. *Electronic Notes in Theoretical Computer Science*, 50(4), 2001. Proc. Workshop on Verification of Parametrized Systems (VEPAS'01), Crete, July, 2001.