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

Deliverables 7 & 8

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

Year 2 : October 2001 - September 2002

Date of preparation: September 2002

Contributors: Paris, Uppsala, Grenoble, Weizmann

1 Introduction

We report hereafter 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 system. 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 system* allows creation and deletion of components during the execution of the system. Such systems arise naturally in computer applications, e.g. in mobile protocols, security protocols, and multi-threaded programs.

The fundamental (and common) challenge in verification of parameterized and dynamic systems is to cope with *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 the system. 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 more general algorithms.
- Developing new algorithms for checking liveness properties.
- Designing new methods for verification of dynamic systems, in particular multi-threaded programs and data security protocols.
- Case Studies and extension of the IF environment in order to take into account dynamic creation of processes.

2 New Regular Model Checking Algorithms

Regular model checking was advocated by the paper [KMM$^+$01] as a uniform framework for the analysis of parametrized and dynamic systems. The idea of regular model checking is to use regular relations (encoded by *finite-state transducers*) as a symbolic representation for transitions of the system. As we mentioned in the our report last year, several papers [Tou01, DLS01, Mai01] in the project were devoted to developing techniques which allow computing the transitive closure of regular relations.

One drawback with the above algorithms is that they involve heavy (and often complicated) global computations on the produced transducer, which significantly limit their applicability in practical cases. In [AJNd02] we propose a new algorithm, which replaces the global computations with a simple local operation making the algorithm both simpler to present and more efficient to implement. We have implemented a prototype and carried out verification of a number of protocols with improved results compared to the previous algorithms.

All previous work in the context of regular model checking assume a linear topology of the system to be verified. In [AJMd02, BT02] we present approaches to extend the applicability of model checking to systems with tree-like structures. We lift previous algorithms for computing transitive closures of linear transducers into algorithms capable of handling regular tree relations. We have implemented a prototype and performed verification of several protocols with tree-like topologies. Furthermore, we believe that our algorithms reveal an important direction for future work. More precisely, our algorithms rely on a small set of properties of trees which are shared by other data structures. We aim to develop model checking algorithms for systems whose behaviours are captured by relations on general forms of graphs. This would allow us to verify systems with dynamic behaviours such as security protocols.

3 Liveness Properties

Most of our earlier work on systems with unbounded control structures has concentrated on checking safety properties. Recently, we have started extending our methods to liveness properties. The paper [KPSZ02] presents the method of *network invariants* for verifying a wide spectrum of LTL properties, including liveness, of parameterized systems. The effectiveness of the network invariant method is illustrated on several examples, including a deterministic and probabilistic versions of the dining-philosophers problem.

The paper [PXZ02] introduces the (0,1,infinity)-counter abstraction method by which a parameterized system of unbounded size is abstracted into a finite-state system. Assuming that each process in the parameterized system is finite-state, the abstract variables are limited counters which count, for each local state s of a process, the number of processes which currently are in local state s. The counters are saturated at 2, which means that counter(s)=2 whenever 2 or more processes are at state s. The emphasis of the paper is on the derivation of an adequate and sound set of fairness requirements (both weak and strong) which enable proofs of liveness properties of the abstract system, from which we can safely conclude a corresponding liveness property of the original parameterized system. The method is illustrated on a number of parameterized systems, including Szymanski's Algorithm for mutual exclusion. The method is also extended to deal with parameterized systems whose processes may have infinitely many local states, such as the Bakery Algorithm. The extension is based on a choice of few interesting and relevant state assertions and then (0,1,infinity)-counting the number of processes satisfying each of these assertions.

The paper [ZPK03] addresses the the verification problem of parametrized probabilistic systems. It proposes a verification method which is based on a reduction to the construction of abstractions with adequate fairness constraints. The method has been implemented in the tool TLV and has been used for the automatic verification of nontrivial probabilistic parametrized algorithms.

4 Dynamic Systems

We have developed algorithms for verification of two important classes of systems which exhibit dynamic behaviour, namely security protocols and multi-threaded programs.

In [BLP02], we present a correct but, in general, incomplete verification algorithm to prove secrecy without putting any assumption on messages nor on the number of sessions. The method is necessarily incomplete since it is known that reachability is undecidable for cryptographic protocols in the general case, even when a bound is put on the size of messages. Proving secrecy means proving that secrets, which are pre-defined messages, are not revealed to unauthorized agents. The main contribution of this paper is an original method for proving that a secret is not revealed by a set of rules that model how the initial set of messages known by the intruder evolves. In contrast to almost all existing methods, we do not try to compute or approximate the sets of messages that can be known by the intruder.

The paper [BT02] applies tree regular model checking in order to analyze multi-threaded programs. More precisely, it considers programs with recursive calls, dynamic creation of processes, and communication. Program behaviour is described as a tree relation on terms built from process variables, sequential composition, and asynchronous parallel composition. The method is illustrated on an example of a server which has the ability of launching an unbounded number of threads.

5 Implementation Effort and Case Studies

We have extended the IF environment in order to take into account dynamic creation of processes. This includes the extension of the common language, and a substantial amount of work on extending the simulation and (finite-state) model-checking tools to this case [BGM02].

In [BM02] we have carried out a case study on verification of a protocol used in the automotive industry. The protocol, called TTP/C allows communication between several devices (micro-processors) embedded in car, and aims at guaranteeing safe execution of different driving actions. The protocol involves several aspects of dynamic behaviour, e.g. discarding faulty stations. The verification is performed for an arbitrary number of processes and faults which may occur in the system.

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*, 2002.

[BGM02] M. Bozga, S. Graf, and L. Mounier. IF-2.0: A Validation Environment for Component-Based Real-Time Systems. In K.G. Larsen Ed Brinksma, editor, *Proceedings of CAV'02 (Copenhagen, Denmark)*, volume 2404 of *LNCS*, pages 343–348. Springer, July 2002.

[BLP02] L. Bozga, Y. Lakhnech, and M. Périn. Pattern-based Abstraction for Verifying Secrecy in Protocols. Technical report, Verimag, University of Grenoble, 2002.

[BM02] Ahmed Bouajjani and Agathe Merceron. Parametric Verification of a Group Membership Algorithm. In *7th Intern. Symp. on Formal Techniques in Real-Time and Fault Tolerant Systems (FTRTFT'02)*, Oldenburg, Germany, September 2002. Lecture Notes in Computer Science, Springer.

[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.

[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.

[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.

[KPSZ02] Y. Kesten, A. Pnueli, E. Shahar, and L. Zuck. Network invariants in action. In *Proc. 13^{th} Int. Conf. on Concurrency Theory (CONCUR'02)*, volume 2421 of *Lecture Notes in Computer Science*, 2002.

[Mai01] M. Maidl. A unifying model checking approach for safety properties of parameterized systems. In *Proc. 13^{th} Int. Conf. on Computer Aided Verification*, pages 324–336, 2001.

[PXZ02] A. Pnueli, J. Xu, and L. Zuck. Liveness with (0,1,infinity)-counter abstraction. In *Proc. 14^{th} Int. Conf. on Computer Aided Verification (CAV'02)*, volume 2404 of *Lecture Notes in Computer Science*, 2002.

[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.

[ZPK03] L. Zuck, A. Pnueli, and Y. Kesten. Automatic verification of free choice. In *3th Intern. Workshop on Verification, Model Checking, and Abstract Interpretation*, Lecture Notes in Computer Science, 2003.