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

Final Report

October 2000 - January 2004

Date of preparation: January 2004

Edited by: Ahmed Bouajjani, University of Paris 7

1 Introduction

The aim of the ADVANCE project was to develop new verification methods and techniques in order to handle complex systems such as modern telecommunication protocols. A particular effort was devoted to the design of automatic analysis and verification techniques for powerful models which have in general an unbounded number of configurations, and therefore, which cannot be analyzed using standard techniques such as finite-state model-checking. Indeed, complex systems involve several aspects which cannot be captured by finite-state models such as real-time constraints, communication through unbounded channels, data manipulation, unbounded control due to dynamic creation of processes, parameterization, etc. The challenge of our project was to develop a general methodology allowing to deal with such sources of complexity and to push further the capabilities of automatic verification techniques.

Complex systems can only be handled by combining several analysis techniques. The approach developed in the project is based on: (1) the definition of uniform modeling frameworks, (2) the design of techniques allowing to generate automatically (potentially infinite-state) abstract models, and (3) the design of advanced model-checking techniques for infinite-state models, in particular symbolic reachability analysis techniques based on efficient representation structures for infinite sets of configurations.

One main feature of the approach developed in our project is that it is based on the use of automata. Indeed, automata are fundamental objects which appear at different levels in our modeling and verification methodology. They first appear in the definition of several natural models of infinite-state systems, and they also appear as generic symbolic representation structures used in model-checking algorithms of various kinds of systems.

2 Automata-based models

Extended automata are natural models of various kind of systems such as protocols, concurrent recursive programs, multithreaded programs, etc. They consist on machines with a finite control manipulating variables and data structures such as counters, clocks (counting time in a discrete or continuous manner), fifo queues (modeling communication channels), stacks (modeling control stacks in sequential programs with recursive calls), etc.

Other automata-based models of infinite-state systems are finite transducers (or equivalently rewrite systems). Indeed, configurations of systems can be encoded as sequences of symbols, and sometimes as labelled trees (representing for instance terms in some process algebra), and therefore, actions of a systems can be encoded as relations between words or trees. Various kind of systems can be modeled in a uniform way based on this approach. This is obvious for systems manipulating sequential of tree-like structures (pushdown systems, fifo-channel systems, process term algebras, etc), but this is also possible in the case of parameterized networks of identical processes (with linear or tree topologies), counter systems, and timed systems.

3 Abstraction techniques

Abstraction techniques are crucial in the modeling and verification process. They allow to reduce the size of the models in order to get models which can be handled efficiently, and they also allow to reduce the complexity of these models (for instance by reducing the number of different types of the variables) in order to get models which can be analyzed by the existing automatic verification techniques and tools.

3.1 Model reduction techniques

Several works in our project have contributed to the development of new model reduction techniques such as static analysis techniques (slicing, live-variable analysis, dead-code elimination) for extended automata [BFG01, GJ01, BGM02], partial-order techniques based on unfoldings [ES01b, ES01a], and symmetry reduction techniques [Ios03]. These techniques have been implemented in the tools IF-toolset and MC Kit.

3.2 Abstraction for infinite-state systems

One of the main contribution of our project is the design of general methodologies for the verification of infinite-state systems (such as extended automata and parametrized networks of processes) based on the combination of abstraction techniques with finite/infinite-state model-checking techniques. Indeed, the abstraction techniques developed in the project allow to reduce the parametric verification problem of a complex "concrete" infinite-state system to the verification problem of an abstract model in some class of (extended) automata. The abstract model could be a finite-state automaton, but in many interesting cases, it can also be an infinite-state model (e.g., a counter automaton). The obtained abstract model can be analyzed by either finite-state model checking techniques or using (infinite-state) symbolic reachability analysis techniques (see next section). The main feature of the developed methods and techniques is that they allow to verify not only safety properties, but also liveness properties, which is hard problem in the context of infinite-state systems. The main results of the project in this context include:

- general verification methods based on constructing abstractions and refining them by analyzing counter-examples [LBOB01]. These methods combine abstraction techniques with model-checking and deductive techniques.
- verification techniques of liveness properties for extended automata (counter/clock automata) allowing to construct accurate finite abstractions with fairness constraints. These techniques use symbolic reachability analysis to generate finite symbolic graphs, and use symbolic techniques to decide the iterability of loops in the abstract models in order to generate fairness constraints [ABLS01].
- powerful abstraction techniques, combined with proof and algorithmic verification techniques, for the verification of temporal properties, including safety ans liveness properties, of parametrized networks. [BLS01, BSBL01, PRZ01, APR$^+$01, PXZ02, KPSZ02, ZPK03, APZ03, PA03, FPPZ04]. These techniques have been implemented in the tool PAX (of the university of Kiel, in collaboration with the Grenoble team of our project), and in the tool TLV.
- cut-off properties for several classes of (both safety and liveness) properties and classes of parametrized networks of systems (including multicast protocols, shared resource management systems, etc). These results allow to reduce the verification problem for a network of processes with an arbitrary number of components n, to a finite number of finite-state model checking problems, stated for all the instances of the network having less components than some (cut-off) bound k [EM03, BHV03].

4 Symbolic reachability analysis

One of the main approaches for automatic verification of infinite-state systems is based on symbolic reachability analysis, i.e., computing a representation of the set of (backward or forward) reachable configurations.

In order to perform this analysis, we need to develop:

- finite representation structures allowing to manipulate infinite sets of configurations. These structures must satisfy minimal requirements such as closure under some necessary operations, and decidability of the emptiness and inclusion problems. Several symbolic representations can be used depending on the type of the considered models. Among these representations, there are for instance data structures for manipulating specific classes of arithmetical constraints such as convex polyhedra, difference bound matrices, and also finite-state automata representing sets of contents of stacks, queues, etc.

- procedures for computing the reachable configurations. The main problem to face here is that reachability analysis is not guaranteed to terminate in general. This is due to (1) the fact that expressive infinite-state models are often Turing-machines powerful (and hence their reachability problem is undecidable), and also to (2) the fact that even for decidable models, a naive procedure based on the iterative exploration of the reachable configurations will never terminate in general since the state space is infinite. Therefore, the aim of our research is (1) to determine classes of models for which there exist efficient algorithms which terminate and compute the precise set of reachable configurations, and (2) to define generic procedures for reachability analysis involving *acceleration* techniques which can help termination in most encountered cases, or which can always force it (may be by introducing some approximations).

The project has produced an important work on this topic leading to new verification algorithms and techniques which have been implemented in several infinite-state verification tools. Among the main achievements we can mention: several decidability results for significant classes of infinite-state models, powerful techniques for the analysis of counter/clock automata using either parametric difference bound matrices [ABS01] or automata-based symbolic representations [LAS01, BJW01], and a general automata-based framework for the verification of infinite-state systems (and in particular for parametrized networks of processes and dynamic systems) [KMM$^+$01, Bou01, Tou01, Mai01, FP01, DLS01, AJNd03, BT03, BLW03, BLW04].

4.1 Regular Model Checking

One of the main achievements of the project is the design and the development of the so-called *Regular Model Checking* paradigm, an automata-based approach for the modeling and the verification of infinite-state systems [KMM$^+$01, Bou01, Tou01, Mai01, BJW01, AJNd03, BLW03]. Indeed, as mentioned above, configurations of various kinds of infinite-state systems can be encoded as words or trees, and therefore (1) transducers can be used as uniform models for infinite-state systems, (2) finite automata can be naturally used to represent and to manipulate (potentially infinite) sets of such configurations, and (3) reachability analysis can be performed by computing transitive closures of transducers, or images of automata by iterations of transducers. So, in this framework, the idea is to design algorithmic techniques which allow, given a transducer T representing the transition relation of a system and an automaton A representing sets of initial configurations, to compute the set $T^*(A)$ of the reachable configurations. The obtained results in this context include:

- the definition of classes of regular relations (transducers or rewrite systems) and classes of (word/trees) languages for which the precise computation of the set of reachable configurations can always be done. Several decidability results and verification algorithms for specific classes of models have been designed in this manner, including pushdown systems[ES01c, KPV02b, KPV02a, EKS03, BHM03], classes of parameterized networks [BMT01, Mai01], lossy channel systems [AJ01a, ABB01], process rewrite systems (modeling programs with dynamic creation of processes and procedure calls) [BT03, BET03], etc.

- the definition of generic automata-based techniques allowing to perform regular model checking, regardless from the considered relation of set of initial configurations. These techniques are either based on (1) state-space reduction techniques allowing to collapse on-the-fly infinite automata-based representations into finite equivalent ones [DLS01, AJNd02, AJMd02, AJNd03] or (2) widening techniques allowing to guess automatically the limit of the iteration of a transducer from a finite set of approximations (obtained after a finite number of iterations) [Tou01, BT02, BLW03, BLW04].

4.2 Monotonic systems

Another important contribution of our project is a generic theory which allow to characterize a class of models for which the iterative computation of the reachable configurations terminates. This theory concerns the so-called class of monotonic systems with respect to a well-quasi ordering or a better-quasi ordering on the set of their configurations [AJ01b].

This general framework, which was defined before the beginning of ADVANCE by members of our project (Uppsala team), has been significantly developed during the project and has been applied for establishing nontrivial decidability results for significant classes of models such as unbounded timed Petri nets and parameterized networks of (one-clock) timed automata [AN01, AMN02, AN02].

5 A general specification and verification environment

The project has devoted an important effort in developing new tools and integrating them into a common specification and verification environment where they can be used through a common interface. The efforts concerned several aspects we present hereafter.

- **A a common model description language:** The language IF allows to describe various types of extended automata: automata with integer counters, real valued clocks, (lossy) fifo channels, etc. Parametric networks of processes are also defined in this language as automata manipulating (multidimensional) parametric bound arrays. These array automata are compiled into finite transducers (regular relations on words or trees) when data domains are finite.

- **State-space reduction techniques:** Static analysis techniques and state-space based reduction techniques available in IF-toolset and MC Kit allow to reduce the size of the analyzed models.

- **Finite-state verification techniques** Efficient finite-state model checking techniques are available through the tools IF-toolset and MC Kit.

- **Finite/Infinite-state abstraction techniques:** Abstraction techniques in IF-toolset and TLV allow to build automatically (potentially infinite-state) abstract models using principles such as predicate abstraction and counter abstraction.

- **Symbolic reachability analysis techniques:** The tools LASH, RMC, TLV, and TReX, allow to analyze automatically various kinds of infinite-state models (counter/clock automata, fifo-channel systems, parametrized networks of processes) using logic-based or automata-based representation structures.

6 Complex case studies

Significant case studies of industrial interest have been considered in our project. Important results on their automatic verification have been achieved, showing that our methods and tools are powerful enough to deal with many complex aspects which appear in realistic systems, in particular by taking into account several dimensions of parameterization (possible input values, bounds on the domain of some variables, size of data structures, number of components, topologies, etc.) Our results constitute the state-of-the-art concerning parametric verification of such kind of complex systems.

6.1 PGM

This case study is a multicast protocol which involves several sources of complexity: tree-like topology with an arbitrary number of participants, unbounded number of messages, parametric bounds on the sizes of buffers, sender and receivers windows, delays, rate of losses in communication channels, speed of the sender, etc. The teams of our project have been able to establish results on the parametric verification of this protocol using complementary approaches allowing to cover each of the aspects mentioned above. These results have been obtained by applying the tools IF-toolset, LASH, and TReX. In particular, we have considered the full recovery property (i.e., the fact that a no messages are definitively lost). Using LASH we were able to check this property for an unbounded number of messages. Moreover, using TReX we were able to synthesize automatically complex constraints on the various parameters of the protocol which guarantee the satisfiability of the property [BS03, ADV04].

6.2 IEEE1394 Root Contention Protocol

This protocol uses timing constraints in order to elect a leader among a set of processes. The protocol has several parameters such as the transmission delays, the bounds of waiting intervals, etc. The parametric analysis of this complex protocol requires the development of techniques allowing to deal in a precise manner with models mixing integer counters and real-valued clocks. The tools TReX and LASH offer different symbolic reachability analysis techniques which allow, both of them, to perform such kind of complex analysis. They have been applied successfully to synthesize parameter constraints which guarantee the correctness of the protocol [CAS01, ADV04].

6.3 TTP/C group membership algorithm

This algorithm is part of an embedded software used in automotive industry. It consists of a very subtle mechanism which allows to stations which communicate (periodically) through a ring network to realize that they are faulty, to withdraw themselves in such a case from the group, and to join later the group. The main problem is to avoid that the emitting stations form two (or more) disconnected groups (so-called clique avoidance problem).

The verification of this algorithm is a real challenge since it has a very complex and nontrivial behavior which involves many parameterization and infinity sources. We have been able to reduce, through a nontrivial abstraction, the verification problem of this algorithm with an arbitrary number of station to the reachability problem of counter automata. Using this approach, we have been able to verify automatically that if a fault occurs, the algorithm stabilizes after two communication rounds in a correct state (i.e., with only one group of emitting stations) [BM02].

7 Conclusion

The project ADVANCE has significantly contributed to the domain of computer-aided verification in the last three years. The achievements of the project constitute important progress in the domain of verification techniques for complex systems, especially techniques for various kinds of parametric analysis of infinite-state models. The methods, techniques, and tools developed in the projects have shown their relevance in tackling nontrivial problems of industrial interest. The scope of these techniques and tools is not limited to the domain of telecommunication protocols, but it concerns a wide range of complex systems. Indeed, the verification technology (for infinite-state systems) we have developed in the project is extremely relevant in the the analysis of embedded software systems in general.

References

[ABB01] P. Abdulla, L. Boasson, and A. Bouajjani. Effective Lossy Queue Languages. In *Proc. 28th Intern. Coll. on Automata, Languages and Programming (ICALP'01)*, Crete, Greece, July 2001. Lecture Notes in Computer Science 2076, Springer-Verlag.

[ABLS01] A. Annichini, A. Bouajjani, Y. Lakhnech, and M. Sighireanu. Analysis of Fair Extended Automata. In *Proc. 8th Intern. Static Analysis Symposium (SAS'01)*, Paris, France, July 2001. Lecture Notes in Computer Science 2126, Springer-Verlag.

[ABS01] A. Annichini, A. Bouajjani, and M. Sighireanu. TReX: A Tool for Reachability Analysis of Complex Systems. In *Proc. 13th Intern. Conf. on Computer Aided Verification (CAV'01)*, Paris, France, July 2001. Lecture Notes in Computer Science 2102, Springer-Verlag.

[ADV04] ADVANCE. Case studies: PGM protocol, IEEE1394 root contention protocol. Technical report, University of Pari7 and France Telecom R&D, 2004.

[AJ01a] Parosh Aziz Abdulla and Bengt Jonsson. Channel representations in protocol verification. In *Proc. CONCUR 2001, 12th Int. Conf. on Concurrency Theory*, volume 2154 of *Lecture Notes in Computer Science*, pages 1–15, 2001.

[AJ01b] Parosh Aziz Abdulla and Bengt Jonsson. Ensuring completeness of symbolic verification methods for infinite-state systems. *Theoretical Computer Science*, 256:145–167, 2001.

[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.

[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.

[AMN02] P. A. Abdulla, P. Mahata, and A. Nylén. Downward closed language generators. Technical report, Uppsala University, Sweden, 2002.

[AN01] Parosh Aziz Abdulla and Aletta Nylén. Timed Petri nets and BQOs. In *Proc. ICATPN'2001: 22nd Int. Conf. on application and theory of Petri nets*, volume 2075 of *Lecture Notes in Computer Science*, pages 53 –70, 2001.

[AN02] P. A. Abdulla and A. Nylén. Undecidability of ltl for timed petri nets. In *Proceedings of 4th International Workshop on Verification of Infinite-State Systems*, 2002.

[APR+01] T. Arons, A. Pnueli, S. Ruah, J. Xu, and L. Zuck. Parameterized verification with automatically computed inductive assertions. In *CAV'01*, pages 221–234, 2001.

[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.

[BFG01] M. Bozga, J-Cl. Fernandez, and L. Ghirvu. Using static analysis to improve automated test generation. *Software Tools for Tehnology Transfer*, 2001. submitted.

[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.

[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.

[BJW01] B. Boigelot, S. Jodogne, and P. Wolper. On the use of weak automata for deciding linear arithmetic with integer and real variables. In *Proceedings of IJCAR '01*, number 2083 in LNAI, pages 611–625, 2001.

[BLS01] K. Baukus, Y. Lakhnech, and K. Stahl. Verification of parameterized networks. *Journal of Universal Computer Science*, 7(2), 2001.

[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.

[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.

[BMT01] A. Bouajjani, A. Muscholl, and T. Touili. Permutation Rewriting and Algorithmic Verification. In *Proc. 16th Symp. on Logic in Computer Science (LICS'01)*, Boston (MA), USA, 2001. IEEE.

[Bou01] A. Bouajjani. Languages, Rewriting systems, and Verification of Infinte-State Systems. In *Proc. 28th Intern. Coll. on Automata, Languages and Programming (ICALP'01)*, Crete, Greece, July 2001. Lecture Notes in Computer Science 2076, Springer-Verlag. invited paper.

[BS03] Marc Boyer and Mihaela Sighireanu. Synthesis and Verification of Constraints in the PGM Protocol. In *FME'03*, volume 2805 of *Lecture Notes in Computer Science*, 2003.

[BSBL01] K. Baukus, K. Stahl, S. Bensalem, and Y. Lakhnech. Networks of processes with parameterized state space. *Electonic Notes of Theoretical Computer Science*, 50(4), 2001. Proc. Workshop on Verification of Parametrized Systems (VEPAS'01), Crete, July, 2001.

[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.

[CAS01] A Collomb-Annichini and M. Sighireanu. Parameterized Reachability Analysis of the IEEE 1394 Root Contention Protocol using TReX. In *Proc. Workshop on Real-Time Tools (RT-TOOLS'01)*, Aalborg, August 2001. Aalborg University.

[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 *LNCS*, 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.

[ES01a] J. Esparza and C. Schröter. Net reductions for LTL model-checking. In *Proceedings of CHARME '01*, 2001. To appear.

[ES01b] J. Esparza and C. Schröter. Unfolding based algorithms for the reachability problem. *Fundamenta Informaticae*, (34), 2001. To appear.

[ES01c] J. Esparza and S. Schwoon. A BDD-based model checker for recursive programs. In *Proceedings of CAV '01*, number 2102 in LNCS, pages 324–336, 2001.

[FP01] Dana Fisman and Amir Pnueli. Beyond regular model checking. In *Proc. 21th Conference on the Foundations of Software Technology and Theoretical Computer Science*, Lecture Notes in Computer Science, December 2001. To appear.

[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.

[GJ01] S. Graf and G. Jia. Verification experiments on the Mascara protocol. In *Proceedings of the SPIN'01 Workshop*, 2001. 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.

[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.

[KPV02a] O. Kupferman, N. Piterman, and M. Vardi. Model checking linear properties of prefix-recognizable systems. In *Proceedings of 14th International Conference on Computer-Aided Verification*, 2002.

[KPV02b] O. Kupferman, N. Piterman, and M. Vardi. Pushdown specifications. In *Proceedings of 9th International Conference on Logic for Programming Artificial Inteligence and Reasoning*, 2002.

[LAS01] The Liège Automata-based Symbolic Handler (LASH), 2001. Available at `http://www.montefiore.ulg.ac.be/~boigelot/research/lash/`.

[LBOB01] Y. Lakhnech, S. Bensalem, S. Owre, and S. Berezin. Incremental verification by abstraction. In *TACAS 2001*, volume 2031 of *LNCS*, 2001.

[Mai01] M. Maidl. A unifying model checking approach for safety properties of parameterized systems. In *CAV'01*, 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.

[PRZ01] A. Pnueli, S. Ruah, and L. Zuck. Automatic deductive verification with invisible invariants. In *TACAS'01*, volume 2031, pages 82–97, 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.