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 5

Progress Report

Year 1 : October 2000 - September 2001

Date of preparation: October 2001

Contributors: all groups

Edited by: Ahmed Bouajjani, University of Paris 7

PART A

EXECUTIVE SUMMARY

1 Work progress and achievements

After one year, the project ADVANCE has made considerable progress towards its aim of developing advanced techniques and tools for the validation of complex reactive systems such as modern telecommunication protocols.

The main achievements obtained during the first 12 months of the project are the following:

1.1 Abstraction and reduction techniques

1. Abstraction techniques have been developed in order to verify (safety and liveness properties) of classes of infinite-state systems such as (1) systems manipulating data ranging over infinite domains, and (2) parametrized networks with an arbitrary number of components.

2. Abstraction refinement techniques, based on the analysis of counter-examples, have been developed for use in automatic design of accurate abstraction relations.

3. Static analysis techniques, such as live variable analysis and program slicing, have been developed and used as model reduction techniques in the framework of the validation (verification and test case generation) of real industrial case studies.

4. Reduction techniques and efficient state exploration techniques based on Petri nets unfoldings have been developed and used for model-checking.

1.2 Symbolic reachability analysis of extended automata

Algorithmic verification techniques based on symbolic reachability analysis have been developed for infinite-state extended automata. More precisely:

1. New symbolic representation structures have been developed for the analysis of models based on automata extended with integer counters and real valued clocks as well as for models based on timed Petri nets.

2. Efficient symbolic state-space exploration procedures have been developed for various classes of extended automata: pushdown automata (modeling mutually recursive procedures), automata with lossy FIFO queues, counter and clock automata with parametric guards. Classes of decidable extended automata have been determined, and powerful fixpoint acceleration techniques have been designed to help termination of the analysis in the undecidable cases.

1.3 Verification methods and techniques for parametrized systems

Parameterization appears naturally in modeling and verifying protocols. A lot of work have been devoted in the project to the development of new methods, techniques, and tools for parametric reasoning.

1. The symbolic reachability analysis paradigm has been extended to parametrized systems. The decidability of the verification problem (of safety properties) has been established for significant classes of parametrized systems and automatic verification methods and tools have been developed for more general (undecidable) cases.

2. A framework, called *Regular Model Checking*, has been introduced for reasoning about networks of finite-state systems arranged in linear, ring-like, or tree-like topologies. In this framework, sets of configurations of arbitrary lengths are represented by means of regular languages (automata on words or trees), systems are modeled as regular relations (transductions). This framework has also been extended to systems generating some classes of nonregular languages.

3. New approaches for parametric reasoning have been introduced based on abstraction, and on systematic methods for generating and checking inductive invariants.

4. New tools have been developed for the analysis of extended automata with parametric guards. These tools have been applied to the verification of protocols with complex parametric timing constraints.

1.4 Combination of methods and techniques

Several works in our project show how to combine complementary analysis techniques in order to tackle complex validation problems.

1. static analysis techniques such as slicing techniques and live variable analysis are used to generate reduced models (finite-state or extended automata) which are analyzed using model-checking algorithms or symbolic reachability analysis procedures.

2. abstraction techniques are used to generate automatically (abstract) extended automata which can be analyzed using model-checking algorithms or symbolic reachability analysis techniques.

3. conversely, symbolic reachability analysis procedures for infinite-state systems are used to generate automatically finite-state abstract models which are analyzed using model-checking algorithms. This approach was applied to the analysis of protocols with complex relations between their variables and parameters.

4. abstraction techniques and finite-state model-checking are used in deriving candidate induction assertions, and to prove that the candidate assertions are indeed inductive. This methodology has been applied to the verification of nontrivial parametrized systems such as mutual exclusion protocols and cache coherence protocols.

1.5 Development/Integration of tools and Experimentations

1. Important efforts have been devoted to the implementation of the methods and algorithms elaborated within the project. These efforts lead to the development of new and original validation tools.

2. the IF language is adopted as a common language for the description of models. It is used as input language by the tools developed in the project and connections has been made with other validations tool-sets.

3. efforts have been made in order to extend the IF language to new features which appear naturally in modeling protocols, such as dynamic creation of processes. A new version of the IF language has been proposed and is under study. A specification of the PGM protocol has been written using this new version of IF.

4. Significant experimentations have been carried out using the methods and tools developed in the project. These experimentations include the analysis of protocols with complex parametric timing constraints such as the Bounded Retransmission Protocol of Philips, and the IEEE 1394 root contention protocol, as well as industrial case studies (e.g. a protocol for wireless ATM).

2 Dissemination of results

2.1 Publications

The obtained results have been published in the following international conferences and journals:

- Intern. Colloquium on Automata, Languages and Programming *ICALP* (Greece),
- Intern. Symposium on Logic in Computer Science *LICS* (USA),
- Intern. Conference on Computer Aided Verification *CAV* (France),
- Intern. Conference on Tools and Algorithms for the Construction and Analysis of Systems *TACAS* (Italy),
- Intern. Static Analysis Symposium *SAS* (France),
- Intern. Conference on Concurrency Theory *CONCUR* (Denmark),
- Intern. Conference on Mathematical Foundations of Computer Science *MFCS*,
- Intern. Conference on Foundations of Software Technology and Theoretical Computer Science *FST&TCS* (India),
- Intern. Conference on Application and Theory of Petri Nets *ICATPN* (UK),
- Intern. Joint Conference on Automated Reasoning *IJCAR* (Italy),
- Intern. Symposium on Theoretical Aspects of Computer Software *TACS* (Japan),
- Advanced Research Working Conference on Correct Hardware Design and Verification Methods *CHARME* (Scotland),
- Intern. Conference on Implementation and Applications of Automata *CIAA* (South Africa).
- Journal *Theoretical Computer Science*, Elsevier Science Pub.,
- Journal of Universal Computer Science, Springer Pub.,
- Journal Fundamenta Informaticae,
- Electronic Notes in Theoretical Computer Science, Elesevier Science Pub.

All the articles produced in our project can be found at:

`http://verif.liafa.jussieu.fr/~haberm/ADVANCE/main.html`

2.2 Editorial activities and organization of workshops

- A special issue of the journal *Theoretical Computer Science* has been published by A. Bouajjani (**UP7**) on *Algorithmic Verification of Infinite State Systems*, volume 256, April 2001.
- A Workshop on *Verification of Infinite State Systems* has been organized by (**UP7**) in Paris, March 2001. This workshop was attended by around 70 researchers and students.

3 Coordination and Management

3.1 Internal meetings, visits, and cooperations

- Four project meetings have been organized in (6-7) November 2000 in Paris, (3-4) May 2001 in Grenoble, (24) July 2001 in Paris, and (17-18) October 2001 in Liège.
- Thomas Arts and Hans Nilsson from (**Ericsson-UAB**) attended the first project meeting. Hans Nilsson gived a presentation of the SIP protocol at that meeting. Thomas Arts visited (**UP7**) in July 2001.
- working meetings have been organized between (**Ericsson-UAB**) and (**Uppsala**) around the SIP protocol.
- Yassine Lakhnech, Aurore Annichini, and Eugene Asarin from (**UJF**) visited (**UP7**). Ahmed Bouajjani and Mihaela Sighireanu from (**UP7**) visited (**UJF**). The intensive cooperation between the two groups lead to several common publications (4 articles) and to the development of the tool TReX.
- Yves-Marie Quemener from (**F-Telecom**) visited (**UJF**) where he had several working meetings with the group of Grenoble around the PGM protocol and its specification in the new version of the IF language,
- Saddek Bensalem from (**UJF**) visited (**Uppsala**) where he worked on combining abstraction techniques with efficient symbolic verification techniques.
- The group of (**UP7**) and (**Uppsala**) have tight collaboration links. This year, they have produced a common paper on the problem of constructing reachability sets of FIFO channel systems.

3.2 Management of the project

The table below describes the manpower (Paid/Non paid persons-months effort) used by the partners of the project. There was no major deviations from the expected effort defined in the original proposal. Notice that the partner Technical University of Munich (**TUM**) left the project in March 31, 2001, and was substituted by the partned University of Edinburgh (**UEdin**) starting from April 1, 2001. This was due to the fact that Javier Esparza, who was the head of the group of Munich, moved to the University of Edinburgh.

Participant	**WP1**	**WP2**	**WP3**	**Total**
(**UP7**)	0/10	0/6	0/1.5	0/17.5
(**F-Telecom**)	1	2	0	3
(**Liège**)	8.7/1.4	3.3 / 0.7	0/0.3	12/2.4
(**TUM+ UEdin**)	6.5/6.5	3/2	0/0	9.5/8.5
(**Uppsala**)	7/10	4/1	0/1	11/12
(**UJF**)	8	9	1	18
(**Weizmann**)	6/1.5	4.6/1.3	0/0	10.6/2.8
	37.2 /29.4	25.9 / 11	1/2.8	64.1/43.2

The deviations from the proposal have the following explanations:

- The 8 paid months planned by Paris (**UP7**) for this year were not used. The effort was compensated by non paid work (permanent researchers and students paied on public grants). The non used 8 months will be used next year.

- France-Telecom (**F-Telecom**) planned an effort of 4 months for this year (instead of the 3 spent months). This was due to the fact that the expected development of symbolic testing tools have not yet started, but only preliminary studies of 1 man-month instead of 2 have been carried out. The development of such tools will start in the second year of the project.

- Munich and Edinburgh used 9.5 paid man-months instead of 12 due to the process of moving the project between the two universities. This amount was compensated by a non paid work. In fact, the person who is payed currently by the project in Edinburgh was payed by the University of Edinburgh in the 2.5 months transition phase of April to June 15. The non used paid resources will be spent next year.

- Uppsala (**Uppsala**) spent more non-paid man-months than specified in the project proposal. The reason is that members of the group (permanent researcher and student) participate in three national research projects related to *Advance*. On the other hand, they have used 11 instead of 13.2 paied man-months. The non spent resources will be used next year.

PART B

ACHIEVEMENTS PER WORKING PACKAGE

1 WP1: Methods and tools

1.1 Task T1.1: Abstraction

Abstraction techniques are used to construct faithful models which can be analyzed automatically. Abstraction is necessary in order (1) to reduce the validation problem of complex systems to the analysis of manageable (may be infinite-state) models and (2) to tackle the state explosion problem. Abstraction techniques have been developed in our project for various kinds of systems. Theses techniques are combined with other techniques such as symbolic reachability analysis and static analysis techniques (e.g., program slicing).

1. In [ABLS01] (D1,D2), abstraction techniques based on symbolic reachability analysis are introduced in order to verify safety and liveness properties of infinite-state models which are timed counter automata with parametric guards.

2. In [BLS01, BSBL01] (D2), abstraction techniques are introduced in order to analyze classes of parametrized networks with an arbitrary number of components.

3. In [LBOB01] (D1), a methodology for constructing abstractions and refining them by analyzing counter-examples. The paper show also a novel and uniform verification method based on combining abstraction, model-checking and deductive techniques.

4. In [BFG01, GJ01] (D1) abstraction techniques inspired from static analysis techniques, such as live variable analysis and program slicing, are used in the framework of the validation (verification and test case generation) of real industrial case studies.

5. In [PRZ01, APR$^+$01] (D2), a method is presented where techniques based on abstraction and finite state model-checking are used both in deriving automatically candidate induction assertions, and to prove that the candidate assertions are indeed inductive. The efficiency of the method is demonstrated through verification of several types of parametrized systems such as mutual exclusion protocols and cache coherence protocols.

1.2 Task T1.2: Symbolic analysis

Symbolic reachability analysis is based on using data structures which allow to represent and manipulate (infinite) sets of configurations, and powerful configuration-space exploration procedures which allow to construct effectively the set of all reachable configurations. This analysis can be used in several ways, mainly to perform on-the-fly verification of safety properties, and to generate invariants and finite abstract models.

In our project, algorithmic verification techniques for various infinite-state systems have been developed based on symbolic reachability analysis.

1. In [ES01c, EKS01] (D1), efficient symbolic model-checking algorithms are developed for Boolean programs with mutually recursive procedures. In [BHM01] (D1,D2), symbolic analysis techniques are developed for mutually recursive procedures with one integer parameter.

2. In [AJ01b] (D1), a general framework for the verification of infinite-state systems is developed based on backward symbolic reachability analysis. General termination conditions are given from which it is possible to derive the decidability of verification problems for various classes of models.

3. In [AB01] (D1), the power of various computational models is studied under the assumption of perturbation (noise, unreliability, etc). A natural notion of robustness under perturbation is defined, and it is shown that the class of robust systems coincides with the class of decidable ones.

4. In [AN01] (D1), a symbolic algorithm for deciding the coverability problem for (unbounded) Timed Petri Nets is provided. In [AJ01c] (D2) similar techniques are used to solve the verification problem of safety properties (such as mutual exclusion) for a class of parametrized networks of 1-clock timed automata.

5. In [AJ01a, ABB01] (D1), symbolic verification algorithms are provided for automata communicating through unbounded lossy FIFO channels.

6. In [BJW01] (D1), efficient data structures (NDDs and RVAs) based on finite-state automata have been introduced for symbolic representation of infinite sets of integer or real vectors. These representations are shown to have nice algorithmic properties an that they are expressive enough to handle linear arithmetics. These representations have been implemented in the tool LASH [LAS]. In [BL01] (D1), NDDs are used to compute automatically the number of solutions of arithmetical constraints expressed in Presburger arithmetics.

7. In [ABS01] (D1), a new tool called TReX is presented for the verification of extended automata (clock and counter automata) with parametric guards. This tool manipulates data structures based on a matrix-based representation and applies fixpoint acceleration techniques on these representations allowing to generate the reachability sets of protocols with complex parametric constraints [CAS01] (D2).

8. In [KMM+01] (D2), the *regular model checking* framework is advocated as a uniform framework for the analysis of parametrized networks of finite-state systems arranged in linear, ring-like, or tree-like topologies. In this framework, sets of configurations of arbitrary lengths are represented by means of regular languages (automata on words or trees), systems are modeled as regular relations (transductions), and verification problems are reduced to the problem of computing closure of languages under regular relations (which is not possible in general since the transition relation of any Turing machine is a regular transduction). In [Bou01] (D1), a similar framework based on the use of languages and rewriting systems is presented.

9. In [BMT01] (D2), a subclass of regular languages is considered and shown to be a natural class for the analysis of infinite-state systems. In contrast to regular languages, this class is shown to be closed under permutations (semi-commutations) which correspond to common operations in the description of many parametrized systems.

10. In [Tou01] (D2), widening operations are developed for accelerating fixpoint computations in the framework of regular model checking. These techniques are shown to be exact and complete in many cases.

11. In [DLS01] (D2), techniques for computing transitive closures of regular relations (finite transducers) are defined. These techniques are based collapsing states during the computation of the iterated transducers using notions of backward and forward bisimulations.

12. In [FP01] (D2), the framework of regular model checking is extended in order to deal with classes of parametrized systems generating nonregular languages (such as context-free languages).

13. In [Mai01] (D2), an original symbolic analysis technique is introduced in order to deal with parametrized networks. The paper gives a characterization of a subclass of parameterized systems for which the algorithm is always guaranteed to terminate. The subclass is sufficiently wide to cover existing results for token rings and broadcast protocols.

1.3 Efficient state-space exploration

The Edinburgh site is working on the development of the unfolding technique for state space exploration. In this technique a concurrent system, represented as a Petri net, is unfolded into a usually larger acyclic net, equivalent to the first one with respect to a suitable equivalence notion. Different notions are used, depending on the type of property considered. In [ES01b] (D1), the equivalence notion is essentially "to have the same reachable markings". Four solutions to the reachability problem are studied and compared experimentally. The paper [ES01a] (D1) presents a technique to automatically simplify the system before constructing the unfolding. The authors profit from reduction techniques for Petri nets well established in the literature, and show that in their setting these techniques can be sharpened, leading to larger reductions.

2 WP2: Integration of tools and Experimentations

2.1 Task T2.1: Common language and interface

1. The IF language developed at Verimag (**UJF**) is adopted as a common language for model description (D3). This allows the use of SDL as a specification language, through the connection between ObjectGeode and the IF environment. Tools developed in the project use IF as input language.

2. In [BGM01] (D1), the Grenoble group summarizes its experience with the language IF and the IF environment in the specification and the validation of distributed software systems, including telecommunication protocols.

3. In [VER01] (D3), a new version of the language IF-2.0 is defined including dynamic creation of processes. This feature appears in many protocols such as for instance the PGM protocol [VR01] (D1). A compiler and a simulator for this new version have been implemented, and preliminary experiments with the new specification of the PGM have been carried out.

4. An extension of the IF language is under study in order to include parametric descriptions of networks of identical processes.

2.2 Task T2.2: Integration of tools

1. The tools LASH and TReX which are developed in the project are interfaced with the IF environment. These tools can handle extended automata written in IF, more precisely, automata communicating through shared variables or unbounded FIFO queues, equipped with variables ranging over the following types: booleans, enumerate types, clocks, and unbounded integer counters.

2. The group of Edinburgh is developing a specification and verification environment called the Model Checking Kit (MCK). It consists of a collection of programs which allow to model a finite-state system using a variety of modeling languages, and verify it using a variety of checkers, including deadlock-checkers, reachability-checkers, and model-checkers for the temporal logics CTL and LTL. The most interesting feature of the Kit is that, independently of the description language chosen by the user, (almost) all checkers can be applied to the same model. The checkers that are currently available include PEP, PROD, SMV, SPIN, and others.

3. An interface between the MCK and the common description language IF has been implemented. It includes a compiler from IF into safe Petri Nets (the basic model of the Kit), a formula converter which translates properties of the original model into Petri net properties, and a trace interpreter which translates error traces of the Petri net back into traces of the original model. The fragment covered by the compiler excludes the real-time features of the language and is limited to finite data types (since these cannot be modeled in the target language), but includes important features like dynamic process creation or different signal delivery disciplines.

2.3 Task T2.3: Experimentations and case studies

1. The paper [GJ01] (D1) describes an experiment in verifying a real industrial protocol for wireless ATM, called MASCARA. Several tools have been used. SDL has been chosen as the specification language and the commercial tool Object-Geode has been used for creating, maintaining and modifying SDL descriptions. The IF tool-set has been used for generation, minimization and comparison of system models and verification of expected properties. All specification and verification tools are connected via the IF language, which has been defined as an intermediate representation for timed asynchronous systems as well as an open validation environment. Due to the complexity of the protocol, static analysis techniques, such as live variable analysis and program slicing, were the key to the success of this verification experiment. The results obtained give some hints concerning a methodology for the formal verification of real systems.

2. The paper [CAS01] (D2) reports on the reachability analysis of fully parametrized models of the IEEE 1394 root contention protocol. This protocol uses timing constraints in order to elect a leader. The interesting point is that the timing constraints involve parameters (transmission delay, bounds of waiting intervals), and the behavior of the protocol strongly depends on the relation between these parameters. The authors use the symbolic reachability techniques implemented in the TReX tool in order to synthesize automatically the parametric constraints which ensure the correctness of the protocol.

3. In [ABS01] (D1), a parametric version of the Bounded Retransmission Protocol of Philips has been automatically analyzed using the tool TReX. The protocol was modeled by means of automata communicating through lossy FIFO channels, and manipulating integer counters and clocks with parametric guards. The analysis requires the manipulating nonlinear arithmetical constraints expressing complex relations between parameters, counters and clocks.

4. In [VR01] (D1), a specification of the PGM protocol is given using the language IF. This work has influenced the definition of the new version of IF [VER01] (D3). Experiments (e.g., simulations) with this specification have been carried out.

3 WP3: Project management / assessment

3.1 Task T3.1: Management and coordination

See the Executive Summary (Part A of the report).

3.2 Task T3.2: Assessment and quality control

During its first year, *Advance* has made significant progress towards its aim of extending the state of the art in the area of validation techniques.

3.2.1 Results on methodology and validation techniques

The research has produced both theoretical and applied works with high technical level and scientific value. We have published our work in the best conferences related to the topics of the project[1]. These conferences target several different research communities such as computer-aided verification, concurrency theory, automata theory, logic in computer science, static analysis, software validation, etc.

In the second year, we will continue the theoretical work on developing analysis techniques for parametrized systems, and we will also consider the case of systems with dynamical structures. Our work will address in particular the problem of analyzing networks with nonlinear topologies (trees, grids, etc), and networks with non finite-state systems (networks of processes manipulating unbounded or parametrized data structures).

For the next two years we prefer to concentrate on dynamic and parameterized systems. We think that progress regarding mobile systems will not be as feasible in the short term.

3.2.2 Construction of validation tools and their integration

We have developed and extended several tools for abstraction and symbolic verification of extended automata. We consider that the work done this year on this topic is an important step toward our objectives. Further work will be done next year to enhance the capabilities of our tools.

We have also defined a common language (IF) which allows integration of the tools in an SDL environment through a connection to ObjectGeode of Telelogic. The new version of the language (IF-2.0) extend the previous one by several features, mainly the possibility of handling procedures, and dynamic creation and destruction of processes. Extensions of the language have been proposed and are under study in order to be able to describe parametrized networks of processes. In particular, we plan to include quantification in expressions (e.g. guards). This is necessary to write global conditions on networks such as *there exists a process instance such that*, or *all the processes of index greater than i are such that*.

We shall continue the work of integrating the tools through designing interfaces to the IF environment. We expect to make substantial progress by the end of the second year.

We will also consider the integration of analysis packages and representation structures packages developed by different partners. For instance, NDDs developed by the group of Liège (**Liège**) will be used by the group of Edinburgh (**UEdin**) in model-checking tools for recursive procedures with integer parameters.

[1] The paper [ABB01] received the award of best track B paper in ICALP'01.

3.2.3 Applications and experimentations

Several encouraging results have been obtained during the first year. One part of the results concern verifying systems beyond the capabilities of previously existing methods such as parametrized networks of processes, parametrized versions of the Bounded Retransmission Protocol, IEEE root contention protocol, etc. The second part consists of case studies showing that our methods scale up to protocols with substantial sizes, e.g., the protocol MASCARA.

A new specification of the PGM protocol has been produced using the new version of IF (extended with the feature of dynamic creation of processes), and some preliminary experimentation has been carried out.

We have also performed some work on the modeling and verification of the SIP protocol. We shall not pursue the study of SIP further in the project since Ericsson have dropped their interest in the protocol. On the other hand, Ericsson will provide new case studies of interest for the project.

3.3 Task T3.3: Dissemination and transfer activities

See the Executive Summary (Part A of the report).

References

[AB01] E. Asarin and A. Bouajjani. Perturbed Turing Machines and Hybrid Systems. In *Proc. 6th IEEE Symp. on Logic in Computer Science (LICS'01)*, Boston (MA), USA, June 2001. IEEE.

[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.

[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.

[AJ01c] Parosh Aziz Abdulla and Bengt Jonsson. Model checking of systems with many identical timed processes, 2001. To appear in the journal of Theoretical Computer Science.

[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.

[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.

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

[BGM01] M. Bozga, S. Graf, and L. Mounier. Automated validation of distributed software using the IF environment. In *Proceedings of Software Model Checking Workshop*, Electronic Notes in Theoretical Computer Science, Paris, France, July 2001.

[BHM01] A. Bouajjani, P. Habermehl, and R. Mayr. Automatic Verification of Recursive Procedures with one Integer Parameter. In *Proc. 26th Intern. Symp. on Mathematical Foundations of Computer Science (MFCS 2001)*, Marianske Lazne, Czech Republic, August 2001. Lecture Notes in Computer Science, Springer-Verlag.

[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.

[BL01] B. Boigelot and L. Latour. Counting the solutions of Presburger equations without enumerating them. In *Proceedings of CIAA '01*, LNCS, 2001. To appear.

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

[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.

[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.

[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.

[EKS01] J. Esparza, A. Kucera, and Stefan Schwoon. Model-checking LTL with regular valuations for pushdown systems. In *Proceedings of TACS '01*, LNCS, 2001. To appear.

[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.

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

[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.

[LAS] The Liège Automata-based Symbolic Handler (LASH). 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.

[PRZ01] A. Pnueli, S. Ruah, and L. Zuck. Automatic deductive verification with invisible invariants. In *TACAS'01*, volume 2031, pages 82–97, 2001.

[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.

[VER01] VERIMAG. The Language IF-2.0. Technical report, University of Grenoble, September 2001. Internal tech. report. Verimag. Deliverable 3 of project Advance.

[VR01] VERIMAG and France-Telecom R&D. Modeling PGM with IF-2.0. Technical report, University of Grenoble, September 2001. Internal tech. report. Verimag.