Curriculum Vitae

1 Personal Details

Name:	Parosh Aziz Abdulla
Nationality:	Swedish
Date of Birth:	18 June 1961
Languages:	Kurdish: Native; Arabic, English, Swedish: Excellent.
Adress:	Department of Computer Systems Uppsala University P.O.Box 325 751 05 Uppsala Sweden
Tel:	+46 - 18 - 471 31 63
Fax:	+46 - 18 - 55 02 25
Email:	parosh@docs.uu.se
Present Employment:	Associate Professor, Senior Lecturer, Uppsala University.

2 Eduction

- May 1990, *PhD in Computer Science*, Uppsala University. Faculty opponent: Prof. J V Tucker, Swansea University. Supervisor: Prof. Stefan Arnborg, The Royal Institute of Technology, Stockholm.
- June 1982, *BSc. in Electrical Engineering*, University of Salahaddin, Arbil (previously University of Sulaimanyah), Iraq.

3 Appointments

October 1997 - present	Consultant, Prover Technology, Stockholm.
December 1992 - present	Senior Lecturer, Department of Computer Systems, Uppsala University.
June 1997 - present	Associate Professor (Docent), Department of Computer Systems, Uppsala University.
July 1990 - December 1992	Temporary Lecturer, Department of Computer Systems, Uppsala University.
November 1985 - June 1990	PhD Student, Department of Computer Systems, Uppsala University.
July 1986 - June 1990	PhD Student, The Royal Institute of Technology, Stockholm.

4 Other Scientific Activities and Qualifications

- Received the "Bjurzons prize for outstanding theses" ("Bjurzons premium för förtjänstfull avhandling"), November 1990.
- Director of the postgraduate program at the department of computer systems, Uppsala University, since 1995.
- Project leader for: *Methods and Tools for Design and Analysis of Distributed Concurrent Systems.* The project was supported by the Swedish Board for Technical Development (STU), during the period July 1990 - July 1992.
- Involved in the project: *Automated Verification of Distributed Systems.* The project was supported by the Swedish Research Council for Engineering Sciences (TFR), during the period July 1992 - June 1995.
- Involved in the European ESPRIT BRA project REACT, July 1992 - June 1995.

- Involved in the project: *Formal Techniques for Design of Real-time Systems and Hardware*. The project started in July 1993 and is supported by the Swedish Research Council for Engineering Sciences (TFR).
- Project leader for: *Automated Verification of Distributed Systems, Part II*. The project started in July 1995 and is supported by the Swedish Research Council for Engineering Sciences (TFR).

5 Teaching Experience

I have taught the following undergraduate courses:

- Compiler Design, in the computer engineering program (Teknisk Fysik) and as an external course, 10 times. In this course, theoretical and practical aspects of compiler design are studied.
- Parallel Systems, in the computer engineering program (Datavetenskapliga programmet), twice. The course is mainly concerned with teaching students how to use formal methods (such CCS - Calculus of Communicating Systems) to specify and verify distributed systems.
- Theory of Computation, in the computer engineering program and the computer science program (Datavetenskapliga Linjen), three times. The course involves studying and comparing different theoretical tools for the analysis of both sequential and parallel programs. These include the theory of abstract data types, fixed point theory, queueing theory, Petri Nets, etc.
- Computer-aided Verification, in the computer science program, 4 times. The aim of the course is twofold. First, the students learn formal techniques for the specification and verification of hardware circuits. Then, the main ideas behind verification methods using theorem prover systems are introduced . As examples of such systems the *Higher Order Logic* and the *Boyer-Moore* theorem provers are considered. Case studies are given to illustrate how hardware circuits can be specified and verified in the above systems.

- Computer Security, in the computer science program and as an external course, 4 times. I am involved with the part of the course dealing with cryptography and encryption algorithms. Both classical and modern algorithms are described inclusing the Data Encryption Standard (DES) and public-key algorithms.

- Theory of Programming, in the computer science program, 4 times. The course involves formal specification, verification, and synthesis of programs. The students are introduced to the notions of equational reasoning, wekaest precondions, invariants, and program derivation.

Development of Courses I have defined and developed the contents of the courses in computer-aided verification and programming theory. For each of them, I have also written a compendium with numerous examples and exercises.

I have participated in the development of laboratory exercises and projects in compiler design (exercises in using compiler-compiler programs), formal methods for parallel systems (formal specification and verification of distributed systems using Petri nets and calculi of communicating systems), and computer-aided verification (using theorem provers to verify hardware circuits).

6 Research Experience

Current capabilities in computer technology allow larger and larger systems to be implemented. This has led to the design of these systems becoming increasingly complicated. Consequently, the design of verification methods for such systems has a great methodological and practical importance. Most approaches to automated verification model the implementation as a finite-state machine. The objective of my reserach has been to extend such methods in order to be able to deal with systems with infinite state spaces. The results have been applied for the verification of hardware circuits and distributed systems.

Systolic Hardware Circuits

A *systolic circuit* consists of a large number of processors (computation cells), each of which performs a limited amount of computation and contains very little storage. These processors are placed inside the circuit and interconnected in a simple and regular pattern allowing communication between near neighbors. Such architectures allow us to minimize circuit parameters such as circuit area and power consumption. In addition, the regularity of the system and the simplicity of the processors make systematic design, analysis, and reasoning possible. Applications of systolic circuits can be found in basic combinatorial circuits, adders, multipliers, digital signal processing, image processing, pattern matching, linear algebra, graph theory, etc. Prototype systolic circuits are available and are used especially in digital signal processing systems.

Verifying systolic circuits involves dealing with an inifinite state spcace, since a circuit usually operates on an infinite stream of data values, and since one have to verify an infinite number of circuits in one step (for example, one would like to verify an adder for all sizes of the input numbers).

I have constructed methods for automatic verification of several classes of systolic circuits [AA90, Abd90a, Abd92b, Abd91, Abd92a]. This has been done in several steps. A model is provided for the formal description of implementations and specifications of systolic circuits as systems of recurrence equations over a signature. The implementation description of a systolic circuit reflects the mathematical operations performed inside the cells, and the interconnection pattern among the cells of the circuit. The specification is a statement of the desired relation between the output data and the input data of the circuit. Basic ideas from the theories of abstract data types and fixed points are used to provide a semantical definition for the model, and to give a formal meaning of circuit verification. The notion of an *algebra* is used to interpret the operations inside the circuit cells. Fixed point theory is used to relate the equation systems corresponding to the implementation and specification of a circuit. I have constructed methods for automatic verification of three classes of circuits. The cell operations in the three classes are interpreted as the operators of a commutative ring [Abd90b], the operators of a boolean algebra [Abd92a], and uninterpreted function symbols [Abd91] respectively. The above classes of circuits can be shown [Abd91] to be "large", in the sense that simple generalizations to each class make the corresponding verification problem undecidable.

For each class of circuits the verification method has been applied to nontrivial circuits in the class. Examples of circuits which can be verified by my methods include systolic circuits for implementing convolution algorithms, matrix operations, string matching algorithms, graph-theoretical problems, and digital filters.

No methods for automatic verification of nontrivial classes of systolic circuits were known in the literature before the publication of these results.

Lossy Channel Systems

I have considered the class of finite-state systems that communicate via unbounded but *lossy* FIFO channels (called *lossy channel systems*). Lossy channel systems can model many interesting systems, e.g. link protocols such as the Alternating Bit Protocol and HDLC. These protocols and others are designed to operate correctly even in the case that the FIFO channels are faulty and may lose messages. These sytems are infinite-state due to the unboundedness of the channels. I have developed a verification theory for lossy channel systems. In [AJ93, AJ96b], I proved, together with Bengt Jonsson, that several interesting verification problems are decidable for lossy channel systems, namely

- The *reachability problem*: is a set of given states of such a system reachable from some other state of the system.
- *Safety properties*: does a system satisfy a *safety property over traces*, formulated as a *regular set* of allowed finite traces.
- *Eventuality properties*: Do all computations of a system eventually reach a given set of states.

These results were quite surprising in view of the fact that all nontrivial verification problems are undecidable for the class of finite-state systems that communicate via unbounded *perfect* FIFO channels. In fact, our results generalize to other types of sequences that can lose elements such as “lossy stacks”, “lossy tapes” (of Turing machines). It follows e.g. that the halting problem for Turing machines with “lossy tapes” is decidable.

In [AJ94, AJ96a], we investigated the decidability of more general verification problems for lossy channel systems. We were able to prove that most of the interesting verification problems that were not proven decidable in [AJ93] are in fact undecidable. More precisely, we showed that the following problems are undecidable:

- The *model checking* problem in propositional temporal logics such as Propositional Linear Time Logic (PTL) and Computation Tree Logic (CTL), interpreted over lossy channel systems. Model checking is more general than the problem of deciding safety properties, which was shown decidable in [AJ93].
- The problem of deciding *eventuality properties with fair channels*: Do all computations eventually reach a given set of states if the unreliable channels are fair in the sense that they deliver infinitely many messages if infinitely many messages are transmitted. This problem is interesting, since one of its instances models the problem of verifying that a link protocol, such as HDLC, will eventually transfer all messages if the unreliable channel is not permanently broken.

In [AK95] I studied together with Mats Kindahl (a student of mine), simulation and bisimulation relations between lossy channel systems and finite transition systems, and showed the decidability of

- whether a state in a finite transition system *simulates* a state in a lossy channel system, and conversely,
- whether a state in a finite transition system is *bisimilar* to a state in a lossy channel system, and
- whether a state in a finite transition system *weakly simulates* a state in a lossy channel system.

Furthermore, we showed the undecidability of

- whether a state in a lossy channel system *weakly simulates* a state in a finite transition system, and
- whether a state in a finite transition system is *weakly bisimilar* to a state in a lossy channel system.

Mats Kindahl has defended a licentiate thesis and will soon defend a PhD thesis based mainly on the results of this paper and [AKP97, AJKP98].

In [ABJ98, AAB99] we consider symbolic on-the-fly verification methods for lossy channel systems. We propose a novel representation formalism, called *simple regular expressions* (SREs), for representing sets of states of lossy channel systems. We show that the class of languages representable by SREs is exactly the class of downward closed languages that arise in the analysis of such systems. We give methods for

- computing inclusion between SREs,
- an SRE representing the set of states reachable by executing a single transition in a system, and
- an SRE representing the set of states reachable by an arbitrary number of executions of a control loop of a program. All these operations are rather simple and can be carried out in polynomial time.

With these techniques, one can straightforwardly construct an algorithm which explores the set of reachable states of a protocol, in order to check various safety properties. We also show how one can perform model-checking of LTL properties, using a standard automata-theoretic construction. It should be noted that all these methods are by necessity incomplete, even for the class of protocols with lossy channels.

General Theories for Infinite-State Systems

In a joint work with Karlis Čerāns, Bengt Jonsson and Tsay Yih-Kuen [AČJYK96, AČJYK98, AJ98a, AJ97], we are interested in finiding a general theory for deciding properties of infinite state systems. We try to identify common mathematical structures and principles which could serve as sufficient conditions for achieving decidability. Our objective is twofold. We aim on one hand to give a unified explanation of existing decidability results for different classes of infinite-state systems such as timed automata, Petri nets, lossy channels systems, and relational automata, and on the other hand to provide guidelines for discovering similar decidability results for other novel classes of systems.

Parametrized Systems

In [AJ98b] we consider a class of infinite-state systems which are also parametrized by the number of parallel components they contain. More precisely, we present a method for deciding reachability properties of networks of timed processes. Such a network consists of an arbitrary set of identical timed automata, each with a single real-valued clock. Using a standard reduction from safety properties to reachability properties, we can use our algorithm to decide general safety properties of timed networks. To our knowledge, this is the first decidability result concerning verification of systems that are infinite-state in "two dimensions": they contain an arbitrary set of (identical) processes, and they use infinite data-structures, viz. real-valued clocks. We illustrate our method by showing how it can be used to automatically verify Fischer's protocol, a timer-based protocol for enforcing mutual exclusion among an arbitrary number of processes.

In a recent work we have shown the undecidability of temporal logics, such as PTL and CTL, for timed networks. To do that we first prove the undecidability of the recurrent state problem: g iven a state in a timed network, check whether there is a computation of the network visiting the state infinitely often. The undecidability result follows immediately, since the recurrent state problem is expressible in these logics.

Partial Order Methods

In [AKP97] we adopt partial order techniques to improve the algorithm in [AJ96b]. We define a preorder, which we call the better than relation , among the set of transitions of the system. Intuitively a transition is better than another if choosing the first transition instead of the second preserves the reachability of the initial state during the analysis. This relation is weaker than the independence relation, which is an equivalence relation, used in traditional partial order methods, in the sense that two transitions are independent if and only if each of them is better than the other. Consequently, our method gives a better reduction in the number of states considered during the analysis. We demonstrate the efficiency of the approach by a number of experimental results. The idea of using the "better than" relation is general as it is applicable to all systems which have "monotone" behaviour, e.g.

Petri nets, real-time automata, relational automata, etc.

In [AJKP98] we present a general method for applying partial order reductions to improve symbolic verification. The method is equally applicable to the verification of finite-state and infinite-state systems. It considers methods that check safety properties, either by forward reachability analysis or by backward reachability analysis. We base the method on the concept of commutativity (in one direction) between predicate transformers. Since the commutativity relation is not necessarily symmetric, this generalizes those existing approaches to partial order verification which are based on a symmetric dependency relation. We show how our method can be applied to several models of infinite-state systems: systems communicating over unbounded lossy FIFO channels, and unsafe (infinite-state) Petri Nets. We show by a simple example how partial order reduction can significantly speed up symbolic backward analysis of Petri Nets.

Automata Theory

In [AČ98] we We prove that the simulation preorder is decidable for the class of one-counter nets. A one-counter net consists of a finite-state machine operating on a variable (counter) which ranges over the natural numbers. Each transition can increase or decrease the value of the counter. A transition may not be performed if this implies that the value of the counter becomes negative. The class of one-counter nets is computationally equivalent to the class of Petri nets with one unbounded place, and to the class of pushdown automata where the stack alphabet is restricted to one symbol.

References

[AA90] Parosh Aziz Abdulla and Stefan Arnborg. A decision problem in systolic circuit verification. In C. Rattray, editor, *Specification and Verification of Concurrent Systems*, pages 236–245. Springer Verlag, 1990.

[AAB99] Parosh Aziz Abdulla, , Aurore Annichini, and Ahmed Bouajjani. Algorithmic verification of lossy channel systems: An application to the bounded retransmission protocol, 1999. To appear in Proc. TACAS 99.

[Abd90a] Parosh Aziz Abdulla. *Decision Problems in Systolic Circuit Verification*. PhD thesis, Dept. of Computer Systems, Uppsala University, Sweden, Uppsala, Sweden, 1990. Available as report DoCS 90/21.

[Abd90b] Parosh Aziz Abdulla. A method for automatic verification of a class of systolic circuits. In L. J. M. Claesen, editor, *Formal VLSI Specification and Synthesis*, pages 371–390. North-Holland, 1990. Proc. MEC-IFIP International Workshop on Applied Formal Methods for Correct VLSI Design.

[Abd91] Parosh Aziz Abdulla. Decidable and undecidable problems in systolic circuit verification. In *The ACM International Workshop on Formal Methods in VLSI Design*, January 1991.

[Abd92a] Parosh Aziz Abdulla. Automatic verification of a class of data dependent systolic circuits. In L. Augustsson, P. Dybjer, B. Nordström, K.V.S. Prasad, and B. von Sydow, editors, *Proc. of the Workshop on Concurrency*, volume 63 of *The Programming Methodology Group's Report Series*. Department of Computer Sciences, Chalmers University of Technology, Göteborg, Sweden, 1992.

[Abd92b] Parosh Aziz Abdulla. Automatic verification of a class of systolic circuits. *Formal Aspects of Computing*, (4):149–194, 1992.

[ABJ98] Parosh Aziz Abdulla, Ahmed Bouajjani, and Bengt Jonsson. On-the-fly analysis of systems with unbounded, lossy fifo channels. In *Proc.* 10^{th} *Int. Conf. on Computer Aided Verification*, volume 1427 of *Lecture Notes in Computer Science*, pages 305–318, 1998.

[AČ98] Parosh Aziz Abdulla and Karlis Čerāns. Simulation is decidable for one-counter nets. In *Proc. CONCUR '96,* 9^{th} *Int. Conf. on Concurrency Theory*, volume 1466 of *Lecture Notes in Computer Science*, pages 253–268, 1998.

[AČJYK96] Parosh Aziz Abdulla, Karlis Čerāns, Bengt Jonsson, and Tsay Yih-Kuen. General decidability theorems for infinite-state systems. In *Proc. 11^{th} IEEE Int. Symp. on Logic in Computer Science*, pages 313–321, 1996.

[AČJYK98] Parosh Aziz Abdulla, Karlis Čerāns, Bengt Jonsson, and Tsay Yih-Kuen. Algorithmic analysis of programs with well quasi-ordered domains, 1998. To appear in the journal of Information and Computation.

[AJ93] Parosh Aziz Abdulla and Bengt Jonsson. Verifying programs with unreliable channels. In *Proc. 8^{th} IEEE Int. Symp. on Logic in Computer Science*, pages 160–170, 1993.

[AJ94] Parosh Aziz Abdulla and Bengt Jonsson. Undecidable verification problems for programs with unreliable channels. In Abiteboul and Shamir, editors, *Proc. ICALP '94*, volume 820 of *Lecture Notes in Computer Science*, pages 316–327. Springer Verlag, 1994.

[AJ96a] Parosh Aziz Abdulla and Bengt Jonsson. Undecidable verification problems for programs with unreliable channels. *Information and Computation*, 130(1):71–90, 1996.

[AJ96b] Parosh Aziz Abdulla and Bengt Jonsson. Verifying programs with unreliable channels. *Information and Computation*, 127(2):91–101, 1996.

[AJ97] Parosh Aziz Abdulla and Bengt Jonsson. Model checking through constraint solving, 1997. Invited contribution for the Grenoble-Alpe d'Huez European School of Computer Science - Methods and Tools for the Verification of Infinite State Systems.

[AJ98a] Parosh Aziz Abdulla and Bengt Jonsson. Ensuring completeness of symbolic verification methods for infinite-state systems, 1998. To appear in the journal of Theoretical Computer Science.

[AJ98b] Parosh Aziz Abdulla and Bengt Jonsson. Verifying networks of timed processes. In Bernhard Steffen, editor, *Proc. TACAS '98, 7^{th} Int. Conf. on Tools and Algorithms for the Construction and*

Analysis of Systems, volume 1384 of *Lecture Notes in Computer Science*, pages 298–312, 1998.

[AJKP98] Parosh Aziz Abdulla, Bengt Jonsson, Mats Kindahl, and Doron Peled. A general approach to partial order reductions in symbolic verification. In *Proc.* 10^{th} *Int. Conf. on Computer Aided Verification*, volume 1427 of *Lecture Notes in Computer Science*, pages 379–390, 1998.

[AK95] Parosh Aziz Abdulla and Mats Kindahl. Decidability of simulation and bisimulation between lossy channel systems and finite state systems. In Lee and Smolka, editors, *Proc. CONCUR '95,* 6^{th} *Int. Conf. on Concurrency Theory*, volume 962 of *Lecture Notes in Computer Science*, pages 333 – 347. Springer Verlag, 1995.

[AKP97] Parosh Aziz Abdulla, Mats Kindahl, and Doron Peled. An improved search strategy for Lossy Channel Systems. In Tadanori Mizuno, Nori Shiratori, Teruo Hegashino, and Atsushi Togashi, editors, *FORTE X / PSTV XVII '97*, pages 251–264. Chapman and Hall, 1997.