AFRL-SR-BL-TR-98-

0360

REPORT DOCUMENTATION PAGE

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	11 Mar 98	FINAL TECH RPT, 30 SEP 93 TO 29 SEP 97

4. TITLE AND SUBTITLE
Semantic Theories and Automated Tools for Real-Time and Probabilistic Concurrent Systems

5. FUNDING NUMBERS
F49620-93-1-0616

6. AUTHOR(S)
Dr. Amy E. Zwarico

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Dept of Computer Science
Johns Hopkins University
Baltimore MD 21218

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NM
110 Duncan Avenue Suite B115
Bolling AFB DC 20332-8050

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This grant has been used to fund research in process algebras. The process algebra results include a richer semantic framework for expressing real-time processes, the development of an algebra for expressing both pre-emptive and nonpre-emptive processes, axiomatizations of the algebras, implementations of these axiomatizations, and equivalences for Pict, a concurrent object-oriented programming language based on the π-calculus.

19980430 088

14. SUBJECT TERMS
Keywords: process algebras, pre-emptive and nonpre-emptive processes

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

Part I

F49620-93-1-0616

Final Technical Report: Semantic Theories and Automated Tools for Real-Time and Probabilistic Concurrent Systems

Amy E. Zwarico
Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218
amy@cs.jhu.edu

March 11, 1998

1 Results

This grant has been used to fund research in process algebras. The process algebra results include a richer semantic framework for expressing real-time processes, the development of an algebra for expressing both pre-emptive and nonpre-emptive processes, axiomatizations of the algebras, implementations of these axiomatizations, and equivalences for Pict, a concurrent object-oriented programming language based on the π-calculus.

- We extended the class of nonpre-emptive processes first developed in Cleaveland and Zwarico's "A Theory of Testing for Real-Time" (Proceedings of the Sixth IEEE Symposium on Logic in Computer Science, July 1991); revised the alternative characterization of nonpre-emptive processes; defined both real-time and nonpre-emptive algebras; and axiomatized the timed preorders. The extension of the class of nonpre-emptive processes allows us to represent more nondeterministic processes, thus increasing the expressivity of the formalism. The real-time algebra, in conjunction with the testing preorders, allows us to express general timing constraints such as upper and lower bounds. The nonpre-emptive algebra is useful because it can only be used to express nonpre-emptive processes. Furthermore, the nonpre-emptive preorders are precongruences for all operators of the algebra. We have used results from Aceto et al (Aceto, Bloom and Vaandrager "Turning SOS Rules into Equations", Proc. of Seventh Annual IEEE Symposium on Logic in Computer Science, 1992) to develop the axiomatizations for both the general real-time algebra and the nonpre-emptive algebra.
- We currently have a Standard ML implementation of BDD's for the real-time algebras developed. We are using Emerson's quantitative mu-calculus (Emerson, "Real-Time and the Mu-Calculus" (Real-Time: Theory in Practice, Lecture Notes in Computer Science 600, Springer-Verlag 1991). to specify formulas which are to be checked in the model.

PT I

- We are trying to obtain a fully abstract translation to a small process algebraic core language from a larger programming language. We have developed a testing theory for a choice-free variant of the pi-calculus and intend to use this algebra as our core language. The larger language has not been fully determined as yet. This language will likely be object-oriented, so it may use asynchronous communication. Since our core language uses synchronous communication, we are working on a full abstraction result for an encoding of asynchronous communication in the core language.

  We are also trying to develop a variant of the pi-calculus which is more object-oriented in its communication style. By this we mean that if a message is sent on a channel x we know which process (which can be though of as an object) is going to receive that message. In essence, only one process may receive on any given channel. For example, suppose we have a process p which can receive on a channel x. Then in the system p—q, q cannot receive on the channel x.

- RESEARCH IN SCHEDULING THEORY (Summer of 1996, Rome Laboratory) Elizabeth Leonard spent the summer at the Rome Laboratory doing preliminary work for a scheduling project they were preparing to undertake.

  Scheduling can be defined as the allocation of resources to tasks over time. Because most scheduling problems are NP-complete, good heuristics are necessary to efficiently search for solutions. One such heuristic, constraint propagation, involves maintaining the consistency of the domains of uninstantiated variables during the search for solutions. Since scheduling problems are instances of the more general constraint satisfaction problem (CSP), we examined three consistency checking techniques used in searching for solutions to CSPs with binary constraints.

  Aside from the numerous benchmarks for the relatively simplistic Job-Shop Scheduling Problem there is a dearth of available benchmarks for testing scheduling algorithms. There are however numerous benchmarks available for linear programming algorithms. Linear programming problems, like scheduling problems, are a subclass of constraint satisfaction problems. As such, similar techniques can be used to solve linear programming problems. We examined in depth a set of benchmarks for linear programming problems, the Kennington datasets (Carolyn, Hill, Kennington, Niemi, and Wichmann, "An Empirical Evaluation of the KORBX Algorithms for Military Airlift Applications", Operations Research, 38(2):240-248, 1990) which are similar to scheduling problems in their scope.

  We also evaluated ILOG SOLVER and ILOG SCHEDULE, tools for constraint reasoning and scheduling. (ILOG SOLVER and ILOG SCHEDULE are products of Ilog,Inc., 2005 Landings Dr., Mountain View, CA 94043.) SOLVER is a C++ library of constraint reasoning algorithms. SCHEDULE is a C++ library which can be used in conjuction with SOLVER to specify scheduling problems.

2 Papers

- Elizabeth Leonard and Amy Zwarico, "an Algebraic Framework for Developing and Maintaining Real-Time Systems," AMAST '95, Lecture Notes in Computer Science, vol. 936, Springer

Verlag.

3 Supported Students

Elizabeth Leonard

Part II

F49620-93-1-0616

Final Technical Report: Semantic Theories and Automated Tools for Real-Time and Probabilistic Concurrent Systems

Amy E. Zwarico
Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218
amy@cs.jhu.edu

March 11, 1998

1 Results

The results fall into two primary categories: real-time process algebras and type theories for object oriented programs. The process algebra results include a richer semantic framework for expressing real-time processes, the development of an algebra for expressing both pre-emptive and nonpre-emptive processes, axiomatizations of the algebras, implementations of these axiomatizations, and equivalences for Pict, a concurrent object-oriented programming language based on the π-calculus. The OO results incluce the development of constraint-based type systems (including inference algorithms) for object-oriented languages.

- We extended the class of nonpre-emptive processes first developed in [?]; revised the alternative characterization of nonpre-emptive processes; defined both real-time and nonpre-emptive algebras; and axiomatized the timed preorders. The extension of the class of nonpre-emptive processes allows us to represent more nondeterministic processes, thus increasing the expressivity of the formalism. The real-time algebra, in conjunction with the testing preorders, allows us to express general timing constraints such as upper and lower bounds. The nonpre-emptive algebra is useful because it can only be used to express nonpre-emptive processes. Furthermore, the nonpre-emptive preorders are precongruences for all operators of the algebra. We have used results from [?, ?] to develop the axiomatizations for both the general real-time algebra and the nonpre-emptive algebra.
- We currently have a Standard ML implementation of BDD's for the real-time algebras developed. We are using Emerson's quantitative mu-calculus [?]. to specify formulas which are to be checked in the model.

Pt II

- We are developing eqvivalences for Pict, a concurrent object-oriented programming language based on the π-calculus. Pict consists of a core language in which higher-level language can be defined. Proving equivalence of Pict programs consists in first developing an equivalence for core Pict, developing translation rules that map high-level constructs into low level ones, and proving that the translation rules preserve program equivalence. Thus if two high-level programs are equivalent then so are their core-level translations.

- The development of rich type systems for object-oriented programming languages. Our main area of interest is constraint-based type inference for object-oriented languages.

  Type inference, the idea of automatically inferring type information from untyped programs, is originally due to Hindley and Milner. The type inference for object-oriented languages is more difficult than for traditional sequential programs because even simple object-oriented programs are more "advanced" type-theoretically. Object types can be self-referential so some form of recursive type is also needed. This can be captured using a form of operator polymorphism or F-bounded polymorphism.

  We hav defined a polymorphic, constraint-based type inference algorithm for an object-oriented language, I-Loop. This language incorporates standard notions of class, multiple inheritance, object creation, message send, mutable instance variables, and hiding of instance variables. Thus, there is enough of a core that we are reasonably confident the ideas will scale up to an implemented language. We infer a generalized form of type, recursively constrained (rc) types of the form $\tau\ C$, with "reading "where." C is a set of type constraints of the form $\tau <: \$tau'$ possibly containing free type variables. These constraints may be recursive in that a variable t could occur free in both τ and τ'. This form of type is not standard, and generalizes existing notions. The rc types have the advantage of being very expressive, but the disadvantage of being more difficult to read. The recursive constraints generalize recursive types and the rd-polymorphism generalizes F-bounded polymorphism. The constrained type inference approach to objects allows the inference of very detailed rc types. Far more precise than any type a programmer is likely to come up with. And, this will in turn give the programmer a degree of flexibility well beyond what is provided by current typed object-oriented langauges.

2 Papers

- Elizabeth Leonard and Amy Zwarico, "an Algebraic Framework for Developing and Maintaining Real-Time Systems," AMAST '95, Lecture Notes in Computer Science, vol. 936, Springer Verlag.

- Kim B. Bruce, Luca Cardelli, Giuseppe Castagna, Scott F. Smith, Jonathan Eifrig, Valery Trifonov, Tiejun Wang, Gary T. Leavens, and Benjamin Pierce. On Binary Methods. Submitted for publication. Department of Computer Science, Iowa State University, TR95-08¡/A¿, May 1995.

- J. Eifrig, S. Smith, V. Trifonov, "Sound Polymorphic Type Inference for Objects," OOPSLA 1995.

- J. Eifrig, S. Smith, V. Trifonov, "Type Inference for Recursively Constrained Types and its Application to OOP", Mathematical Foundations of Programming Semantics 1995 (Elsevier Electronic Notes in Theoretical Computer Science, volume 1).
- J. Eifrig, S. Smith, V. Trifonov, A. Zwarico, "Application of OOP Type Theory: State, Decidability, Integration", OOPSLA 1994.
- J. Eifrig, S. Smith, V. Trifonov, A. Zwarico, "An Interpretation of Typed OOP in a Language with State", Lisp and Symbolic Computation, 8 (4), 1995.
- J. Eifrig, S. Smith, V. Trifonov, A. Zwarico, "A Simple Interpretation of OOP in a Language with State", Workshop on State in Programming Languages, 1993.

3 Supported Students

Elizabeth Leonard, Jonathan Eifrig, Valery Trifonov