Chicago Journal of Theoretical Computer Science

The MIT Press

Volume 1996, Article 1

9 February 1996

ISSN 1073–0486. MIT Press Journals, 55 Hayward St., Cambridge, MA 02142 USA; (617)253-2889; *journals-orders@mit.edu, journals-info@mit.edu*. Published one article at a time in LaTeX source form on the Internet. Pagination varies from copy to copy. For more information and other articles see:

- *http://www-mitpress.mit.edu/jrnls-catalog/chicago.html*
- *http://www.cs.uchicago.edu/publications/cjtcs/*
- *gopher.mit.edu*
- *gopher.cs.uchicago.edu*
- anonymous *ftp* at *mitpress.mit.edu*
- anonymous *ftp* at *cs.uchicago.edu*

The *Chicago Journal of Theoretical Computer Science* is abstracted or indexed in *Research Alert®, SciSearch®, Current Contents®/Engineering Computing & Technology, CompuMath Citation Index®.*

©1996 The Massachusetts Institute of Technology. Subscribers are licensed to use journal articles in a variety of ways, limited only as required to insure fair attribution to authors and the journal, and to prohibit use in a competing commercial product. See the journal's World Wide Web site for further details. Address inquiries to the Subsidiary Rights Manager, MIT Press Journals; (617)253-2864; *journals-rights@mit.edu.*

The *Chicago Journal of Theoretical Computer Science* is a peer-reviewed scholarly journal in theoretical computer science. The journal is committed to providing a forum for significant results on theoretical aspects of all topics in computer science.

Editor in chief: Janos Simon

Consulting editors: Joseph Halpern, Stuart A. Kurtz, Raimund Seidel

Editors:			
	Martin Abadi	Greg Frederickson	John Mitchell
	Pankaj Agarwal	Andrew Goldberg	Ketan Mulmuley
	Eric Allender	Georg Gottlob	Gil Neiger
	Tetsuo Asano	Vassos Hadzilacos	David Peleg
	Laszló Babai	Juris Hartmanis	Andrew Pitts
	Eric Bach	Maurice Herlihy	James Royer
	Stephen Brookes	Stephen Homer	Alan Selman
	Jin-Yi Cai	Neil Immerman	Nir Shavit
	Anne Condon	†Paris Kanellakis	Eva Tardos
	Cynthia Dwork	Howard Karloff	Sam Toueg
	David Eppstein	Philip Klein	Moshe Vardi
	Ronald Fagin	Phokion Kolaitis	Jennifer Welch
	Lance Fortnow	Stephen Mahaney	Pierre Wolper
	Steven Fortune	Michael Merritt	

Managing editor: Michael J. O'Donnell

Electronic mail: *chicago-journal@cs.uchicago.edu*

Rank Predicates vs. Progress Measures in Concurrent-Program Verification

Moshe Y. Vardi

9 February, 1996

Abstract

Abstract-1 This note describes a direct relationship between rank predicates and progress measures in concurrent-program verification.

1 Introduction

1-1 In [Var87, Var89, Var91], we presented an automata-theoretic framework that unified several trends in the area of concurrent-program verification. At the foundation of that framework is the observation (due to G. Plotkin) that *recursive* ω*-automata* can express all Σ_1^1 sets of computations. Using this observation it was shown how to extend the *helpful-directions* methodology of [GFMdR85, LPS81] to verification with respect to all Σ_1^1 fairness conditions and Π_1^1 correctness conditions. The technical notion underlying this methodology is that of *rank predicate*, which defines some ranking of program states by means of elements of some well-founded sets.

1-2 Another approach to concurrent-program verification was pursued by Klarlund. The intuition behind his approach is described by the following paraphrase of ideas from [KK91, Kla90, Kla91, Kla92, KS93, Kla94]:

> A *progress measure* is a mapping on program states that quantifies how close each state is to satisfying a property about infinite computations. On every program transition the progress measure must change in a way ensuring that the computation converges toward the property.

We show that there is a direct relationship between rank predicates and progress measures.

2 Background

2.1 Languages and Automata

2.1-1 An *ω-word* w is a function $w: \omega \to \omega$. (One can view w as an ω-word over the alphabet $\{ i \mid \exists j \text{ s.t. } w(j) = i \}$.) In this paper, a *language* is a set of ω-words, i.e., a subset of ω^ω. A language L is Σ^1_1 if it is the projection of an arithmetical relation, i.e., there is an arithmetical relation $R \subseteq \omega^\omega \times \omega^\omega$ such that $L = \{ w \mid \exists u \text{ s.t. } R(w, u) \}$. A language L is Π^1_1 if its complement is a Σ^1_1 language. (See [Rog67] for basic concepts in recursion theory.)

2.1-2 A *table* T is a tuple (S, S^0, α), where S is a (possibly countably infinite) set of *states*, $S^0 \subseteq S$ is the set of *starting* states, and $\alpha \subseteq S \times \omega \times S$ is the *transition relation*. T is said to be *recursive* in case S, S^0, and α are recursive. A *run* r of T *on* the word w is a sequence $r: \omega \to S$ such that $r(0) \in S^0$ and $(r(i), w(i), r(i+1)) \in \alpha$ for all $i \geq 0$.

2.1-3 Automata are tables with *acceptance conditions*. A *Wolper automaton* has a vacuous acceptance condition, so it is just a table $T = (S, S^0, \alpha)$. It *accepts* a word w if it has a run on w. A *Büchi automaton* A is a pair (T, F), where $T = (S, S^0, \alpha)$ is a table and $F \subseteq S$. A accepts a word w if there is a run r of T on w such that for infinitely many i's we have $r(i) \in F$. A is recursive if T and F are recursive. The language accepted by an automaton A, consisting of all ω-words accepted by A, is denoted $L_\omega(A)$.

2.1-4 The following theorem, which follows easily from *Kleene's Normal Form Theorem*, asserts that Wolper automata and Büchi automata have the same expressive power: they both can define all Σ^1_1 languages.

Theorem 2.1 ([Var87, Var89, Var91]) *Let L be a language. The following are equivalent:*

- *L is a Σ^1_1 language.*
- *There is a recursive Wolper automaton A such that $L = L_\omega(A)$.*
- *There is a recursive Büchi automaton A such that $L = L_\omega(A)$.*

2.2 Program Verification

2.2-1 Rather than restrict ourselves to a particular programming language, we use here an abstract model for nondeterministic programs (we model concurrency by nondeterminism). A *program* P is a triple (W, I, R), where W is a set of *program states*, $I \subseteq W$ is a set of *initial states*, and $R \subseteq W^2$ is a binary *transition relation* on W. A computation is a sequence σ in W^ω such that $\sigma(0) \in I$ and $(\sigma(i), \sigma(i+1)) \in R$ for all $i \geq 0$.[1] The set of computations of P is denoted by $L_\omega(P)$. Given that programs are supposed to be effective, we require that W, R, and I are recursive sets.

2.2-2 We assume some means of specifying fairness and correctness. The fairness condition is used to specify what computations are considered to be "fair," i.e., the scheduling of nondeterministic choices is not too pathological. Thus, only computations that satisfy the fairness condition need be considered when the program is verified. The correctness condition is used to express the performance required of a computation; in other words, this is what the user demands of the computation. Instead of focusing on concrete specification languages, we can view the fairness and correctness conditions abstractly as sets of ω-words.

2.2-3 Given a fairness condition Φ and a correctness condition Ψ, the program P is *correct with respect to* (Φ, Ψ) if every computation of P that satisfies Φ also satisfies Ψ, that is, if $L_\omega(P) \cap \Phi \subseteq \Psi$. Our approach is applicable when the fairness condition is a Σ_1^1 language and the correctness condition is a Π_1^1 language. In that case, the intersection of fairness and incorrectness, i.e., $\Phi \cap \overline{\Psi}$, is a Σ_1^1 language, and, by Theorem 2.1, can be expressed by a recursive Büchi automaton or by a recursive Wolper automaton. Thus, let $A_{\Phi,\Psi} = (S, S_0, \alpha, F)$ be a recursive automaton such that $L_\omega(A_{\Phi,\Psi}) = \Phi \cap \overline{\Psi}$, then P is correct with respect to (Φ, Ψ) precisely when $L_\omega(P) \cap L_\omega(A_{\Phi,\Psi})$ is empty.

3 Rank Predicates

3-1 The crux of the approach is to define a *rank predicate* on pairs consisting of program states and automata states.

Theorem 3.1 ([Var87, Var89, Var91]) *Let $P = (W, I, R)$ be a recursive program, let Φ be a Σ_1^1 language, and let Ψ be a Π_1^1 language. Let $A_{\Phi,\Psi} = (S, S_0, \alpha, F)$ be a Büchi automaton such that $L_\omega(A_{\Phi,\Psi}) = \Phi \cap \overline{\Psi}$. Then P is correct with respect to (Φ, Ψ) iff there exists an ordinal κ and a rank predicate $\rho \subseteq 2^{W \times S \times \kappa}$ such that the following holds:*

[1] For simplicity we assume that the program has only infinite computations. A terminating computation is assumed to loop forever in its last state.

- *for all $u \in I$ and $p \in S_0$, we have that $\rho(u, p, \kappa)$ holds,*
- *for all $u, v \in W$ and $p, q \in S$, if $\rho(u, p, \mu)$ holds, $(u, v) \in R$, and $(p, u, q) \in \alpha$, then $\rho(v, q, \nu)$ holds for some $\nu \leq \mu$, and*
- *for all $u, v \in W$ and $p, q \in S$, if $\rho(u, p, \mu)$ holds, $(u, v) \in R$, $(p, u, q) \in \alpha$, and $p \in F$, then $\rho(v, q, \nu)$ holds for some $\nu < \mu$.*

3-2 The conditions in the theorem get simpler if we assume that $A_{\Phi,\Psi}$ is a Wolper automaton, i.e., when we take $F = S$.

Corollary 3.2 *Let $P = (W, I, R)$ be a recursive program, let Φ be a Σ_1^1 language, and let Ψ be a Π_1^1 language. Let $A_{\Phi,\Psi} = (S, S_0, \alpha)$ be a Wolper automaton such that $L_\omega(A_{\Phi,\Psi}) = \Phi \cap \overline{\Psi}$. Then P is correct with respect to (Φ, Ψ) iff there exists an ordinal κ and a rank predicate $\rho \subseteq 2^{W \times S \times \kappa}$ such that the following holds:*

- *for all $u \in I$ and $p \in S_0$, we have that $\rho(u, p, \kappa)$ holds, and*
- *for all $u, v \in W$ and $p, q \in S$, if $\rho(u, p, \mu)$ holds, $(u, v) \in R$, and $(p, u, q) \in \alpha$, then $\rho(v, q, \nu)$ holds for some $\nu < \mu$.*

4 Progress Measures

4-1 A *progress measure* labels each program state with an element in an ordered set such that each program transition decreases the rank of the label [KK91, Kla90, Kla91, Kla92, KS93, Kla94]. Intuitively, the assigned label measures the progress that a computation makes toward meeting its specification. The goal is to show that a progress measure exists for P if and only if all computations of P satisfy a given specification.

4-2 We now show that there is a direct connection between rank predicates and progress measures. Let $P = (W, I, R)$ be a recursive program and let $A_{\Phi,\Psi} = (S, S_0, \alpha, F)$ be a Büchi automaton for the intersection of fairness and incorrectness.

4-3 Consider now the set $Z_\kappa = W \times 2^{S \times \kappa}$ for an ordinal κ. Let $A, B \subseteq S \times \kappa$, and let $u, v \in W$. Then $x = (u, A)$ and $y = (v, B)$ are in Z_κ. We say that x *succeeds* y, denoted $x \rhd y$, if the following holds:

- if $(p, \mu) \in A$ and $(p, u, q) \in \alpha$, then $(q, \nu) \in B$, for some $\nu \leq \mu$,
- if $(p, \mu) \in A$, $(p, u, q) \in \alpha$, and $p \in F$, then $(q, \nu) \in B$, for some $\nu < \mu$.

We say that x is *grounded* if for all $p \in S_0$ we have that $(p, \kappa) \in A$.

4-4 A *progress measure* of P *with respect* to $A_{\Phi,\Psi}$ is a mapping $\beta\colon W \to Z_\kappa$ for some ordinal κ such that:

- $\beta(u) = (u, A)$ for some $A \subseteq S \times \kappa$,
- if $(u, v) \in R$ then $\beta(u) \rhd \beta(v)$, and
- if $u \in I$ then $\beta(u)$ is grounded.

Theorem 4.1 *Let $P = (W, I, R)$ be a recursive program, let Φ be a Σ_1^1 language, and let Ψ be a Π_1^1 language. Let $A_{\Phi,\Psi} = (S, S_0, \alpha, F)$ be a Büchi automaton such that $L_\omega(A_{\Phi,\Psi}) = \Phi \cap \overline{\Psi}$. Then P is correct with respect to (Φ, Ψ) iff it has a progress measure with respect to $A_{\Phi,\Psi}$.*

Proof of Theorem 4.1

If: Let $\beta\colon W \to Z_\kappa$ be a progress measure of P with respect to $A_{\Phi,\Psi}$ for some ordinal κ. Define a rank predicate $\rho \subseteq 2^{W\times S\times\kappa}$ as follows: $\rho(u, p, \mu)$ holds iff $\beta(u) = (u, A)$ and $(p, \mu) \in A$. It is easy to verify that ρ satisfies the conditions of Theorem 3.1; it follows that P is correct with respect to (Φ, Ψ).

Only if: Let $\rho \subseteq 2^{W\times S\times\kappa}$ be the rank predicate given by Theorem 3.1. Consider now the function $\beta\colon W \to Z_\kappa$ defined by

$$\beta(u) = (u, \{\, (p, \mu) \mid \rho(u, p, \mu) \text{ holds} \,\})$$

It is easy to see that β is a progress measure of P with respect to $A_{\Phi,\Psi}$.

□

5 Discussion

5-1 The above results show that progress measures can be derived in a direct manner from rank predicates and vice versa. Thus, their existence is guaranteed for programs that are correct with respect to Σ_1^1 fairness conditions and Π_1^1 correctness conditions. As is shown in [Arn83, Kla90], the analog of Theorem 2.1 holds also in a topological (rather than a recursion-theoretical) setting[2] (i.e., a language is *analytic* iff it can be defined by means of a Wolper automaton and iff it can be defined by means of a Büchi automaton), which means that Theorem 3.1 and Corollary 3.2 can also be stated in a topological setting. Thus, these results are broad enough to cover many cases of interest, and in particular they cover the Rabin fairness conditions of [KK91], the safety correctness conditions of [KS93], and the strong fairness conditions of [Kla92].

5-2 Consequently, the goal of research in this area should not be merely to prove the existence of progress measures, but rather to prove the existence of progress measures with some *desirable* properties. It is the intended application of progress measures—*concurrent-program verification*—that should determine what properties ought to be desired. Indeed, while Theorem 4.1 just proves the existence of progress measures, the progress measures defined by Klarlund possess specific, interesting properties that are spelled out in the various papers [KK91, Kla90, Kla91, Kla92, KS93, Kla94].

Acknowledgement of support: Work done at the IBM Almaden Research Center.

References

[Arn83] A. Arnold. Toplogical characterizations of infinite behaviours of transition systems. In *Proceedings of the 10th International Colloquium on Automata, Languages and Programming*, volume 154 of *Lecture Notes in Computer Science*, pages 28–38, Barcelona, 1983. Springer-Verlag.

[GFMdR85] O. Grumberg, N. Francez, J. A. Makowsky, and W. P. de Roever. A proof rule for fair termination of guarded commands. *Information and Control*, 66:83–102, 1985.

[KK91] N. Klarlund and D. Kozen. Rabin measures and their applications to fairness and automata theory. In *Proceedings of the 6th IEEE Symposium on Logic in Computer Science*, pages 256–265, Amsterdam, The Netherlands, 1991.

[2] One can define a topology on ω^ω by taking the basic neighborhoods to be the balls $Ball(j_1,\ldots,j_k) = \{ v \in \omega^\omega \mid v(i) = j_i$ for $0 \le i \le k \}$ for $k \ge 0$ and $j_1,\ldots,j_k \in \omega$, yielding the *Baire space* [Mos80].

[Kla90] N. Klarlund. *Progress Measures and Finite Arguments for Infinite Computations.* PhD thesis, Cornell University, August 1990. Technical Report TR-1153.

[Kla91] N. Klarlund. Liminf progress measures. In S. Brookes, M. Main, A. A. Melton, M. Mislove, and D. Schmidt, editors, *Proceedings of the 7th International Conference on Mathematical Foundations of Programming Semantics*, volume 598 of *Lecture Notes in Computer Science*, pages 477–491. Springer-Verlag, 1991.

[Kla92] N. Klarlund. Progress measures and stack assertions for fair termination. In *Proceedings of the 11th ACM Symposium on Principles of Distributed Computing*, pages 229–240, Vancouver, Canada, 1992.

[Kla94] N. Klarlund. The limit view of infinite computations. In *Proceedings of the 5th International Conference on Concurrency Theory (CONCUR '94)*, volume 836 of *Lecture Notes in Computer Science*, pages 351–368. Springer-Verlag, 1994.

[KS93] N. Klarlund and F. B. Schneider. Proving nondeterministically specified safety properties using progress measures. *Information and Computation*, 107:151–170, 1993.

[LPS81] D. Lehmann, A. Pnueli, and J. Stavi. Impartiality, justice and fairness: the ethics of concurrent termination. In *Proceedings of the 8th International Colloquium on Automata, Language, and Programming*, volume 115 of *Lecture Notes in Computer Science*, pages 264–277, Acre, Israel, 1981. Springer-Verlag.

[Mos80] Y. N. Moschovakis. *Descriptive Set Theory.* North Holland, 1980.

[Rog67] H. Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw Hill, New York, 1967.

[Var87] M. Y. Vardi. Verification of concurrent programs: the automata-theoretic framework. In *Proceedings of the 2nd IEEE Symposium on Logic in Computer Science*, pages 167–176, Ithaca, NY, 1987. Preliminary version of [Var91].

[Var89] M. Y. Vardi. Unified verification theory. In B. Banieqbal, H. Barringer, and A. Pnueli, editors, *Temporal Logic in Specification*, volume 398 of *Lecture Notes in Computer Science*, pages 202–212, Altrincham, UK, 1989. Springer-Verlag. Proceedings of 1987 workshop.

[Var91] M. Y. Vardi. Verification of concurrent programs—the automata-theoretic framework. *Annals of Pure and Applied Logic*, 51:79–98, 1991. Extended version of [Var87].