Farn Wang (Ed.)

Formal Techniques for Networked and Distributed Systems – FORTE 2005

LNCS 3731

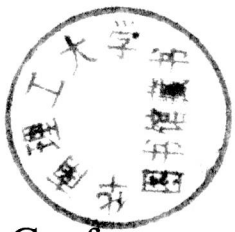
25th IFIP WG 6.1 International Conference
Taipei, Taiwan, October 2005
Proceedings

ifip

Springer

Farn Wang (Ed.)

Formal Techniques for Networked and Distributed Systems – FORTE 2005



25th IFIP WG 6.1 International Conference
Taipei, Taiwan, October 2-5, 2005
Proceedings

Springer

Volume Editor

Farn Wang
National Taiwan University
Department of Electrical Engineering
1, Sec. 4, Roosevelt Rd., Taipei, Taiwan 106, ROC
E-mail: farn@cc.ee.ntu.edu.tw

Library of Congress Control Number: 2005932939

CR Subject Classification (1998): C.2.4, D.2.2, C.2, D.2.4-5, D.2, F.3, D.4

ISSN 0302-9743
ISBN-10 3-540-29189-X Springer Berlin Heidelberg New York
ISBN-13 978-3-540-29189-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11562436 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3731

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Preface

FORTE (Formal Techniques for Networked and Distributed Systems) 2005 was sponsored by Working Group 6.1, Technical Committee 6 (TC6) of the International Federation for Information Processing (IFIP). The conference series started in 1981 under the name PSTV (Protocol Specification, Testing, and Verification). In 1988, a second series under the name FORTE was started. Both series merged to FORTE/PSTV in 1996. The conference name was changed to FORTE in 2001. During its 24-year history, many important contributions have been reported in the conference series. The last five meetings of FORTE were held in Pisa (Italy), Cheju Island (Korea), Houston (USA), Berlin (Germany), and Madrid (Spain).

The 25th FORTE was held from Sunday to Wednesday, October 2–5, 2005 on the beautiful campus of the National Taiwan University (NTU), Taipei, Taiwan, ROC. The scope covered formal description techniques (MSC, UML, Use cases, ...), semantic foundations, model-checking, SAT-based techniques, process algebrae, abstractions, protocol testing, protocol verification, network synthesis, security system analysis, network robustness, embedded systems, communication protocols, and several promising new techniques. In total, we received 88 submissions and accepted 33 regular papers and 6 short papers. All submissions received three reviews. The final program also included 3 keynote speeches, respectively by Prof. Amir Pnueli, Dr. Constance Heitmeyer, and Prof. Teruo Higashino, and 3 tutorials, respectively by Prof. Rance Cleaveland, Dr. Constance Heitmeyer, and Prof. Teruo Higashino. The proceedings also include the text of the keynote speeches. In addition, there were social events, including a reception, a banquet, and an excursion.

FORTE 2005 was co-located with ATVA (Automated Technology for Verification and Analysis) 2005 with a two-day overlap. Prof. Amir Pnueli was the joint keynote speaker. The participants of FORTE 2005 and ATVA 2005 also enjoyed a joint banquet/reception and a joint excursion.

FORTE 2005 was organized under the auspices of IFIP TC 6 by the Department of Electrical Engineering, NTU. It was generously supported by the National Science Council, Taiwan, ROC; the Ministry of Education, Taiwan, ROC; the Institute of Information Science, Academia Sinica, Taiwan, ROC; the Center for Information and Electronic Technologies, NTU; the Graduate Institute of Communication Engineering, NTU; and the Computer and Information Networking Center, NTU.

We would like to thank the Steering Committee members of FORTE for all their suggestions, guidance, and assistance. We also owe a lot to all the Program Committee members and reviewers for their effort in compiling rigorous reviews. Prof. Manuel Nunez deserves special thanks for unselfishly passing on his experience as organizer of FORTE 2004. We would also like to thank Prof. Jin-Fu Chang, Prof. Chuan Yi Tang, Prof. Wanjiun Liao, Prof. Sy-Yen Kuo, Prof. Ming-Syan Chen, Dr. Churn-Jung Liau, and Dr. Ching-Tarng Hsieh for their assistance. Finally special thanks go to Ms. Lih-Chung Lin, Ms. Tz-Hua Chen, Mr. Rong-Shiung Wu, and Mr. Zawa Chu for their energetic and careful planning in the local arrangements and webpage management.

October 2005 Farn Wang

Organization

Steering Committee

Gregor v. Bochmann (Canada)
John Derrick (UK)
Ken Turner (UK)
Tommaso Bolognesi (Italy)
Guy Leduc (Belgium)

General Chair

Jin-Fu Chang (Taiwan)

Program Chair

Farn Wang (Taiwan)

Program Committee

Gregor v. Bochmann (Canada)
Mario Bravetti (Italy)
Jin Song Dong (Singapore)
Colin Fidge (Australia)
Reinhard Gotzhein (Germany)
Holger Hermanns (Germany)
Dieter Hogrefe (Germany)
Ching-Tarng Hsieh (Taiwan)
Myungchul Kim (Korea)
David Lee (USA)
Wanjiun Liao (Taiwan)
Huimin Lin (China)
Elie Najm (France)
Kenji Suzuki (Japan)
Ken Turner (UK)
Farn Wang (Taiwan)
Tomohiro Yoneda (Japan)
Tommaso Bolognesi (Italy)
Ana Cavalli (France)
Khaled El-Fakih (UAE)
David de Frutos-Escrig (Spain)
Constance Heitmeyer (USA)
Teruo Higashino (Japan)
Gerald J. Holzmann (USA)
Claude Jard (France)
Hartmut Koenig (Germany)
Chin-Laung Lei (Taiwan)
Churn-Jung Liau (Taiwan)
Nancy Lynch (USA)
Manuel Nunez (Spain)
Alex Petrenko (Canada)
Chuan Yi Tang (Taiwan)
Hasan Ural (Canada)
Hsu-Chun Yen (Taiwan)

Additional Reviewers

Akira Idoue
Alessandro Fantechi
Alex Groce
Arnaud Dury
Baptiste Alcalde
Bassel Daou and Daniel Amyot

Carlos Gregorio -Rodríguez
Carron Shankland
Clara Segura
Claudio Guidi
Claudio Sacerdoti Coen
Constantin Werner
Dario VIEIRA
Dong Wang
Dongluo Chen
Elisangela Rodrigues-Vieira
Fabio Martinelli
Fida Dankar
Guoqiang Shu
Hesham Hallal
Hidetoshi Yokota
Hong PAN
Ingmar Fliege
Ismael Rodriguez
Jalal Kawash
Jiale Huo
Jian Liu
Jinzhi Xia
Joachim Thees
Jun Lei
Jun Sun
Keqin Li
Li Jiao
Lijun Zhang
Luis Llana
Manuel Mazzara
Marco Zibull
Maurice ter Beek
Mercedes G. Merayo
Michael Ebner
Miguel Palomino
Natalia Lopez
Nicola Tonellotto
O. Marroquin-Alonso
Ping Hao
Rene Soltwisch
Robert Clark
Roberto Gorrieri
Roberto Lucchi
Rüdiger Grammes
Rui Xue
Salvatore Rinzivillo
Sam Owre
Savi Maharaj
Sebastian Schmerl
Serge Boroday
Soonuk Seol
Stefania Gnesi
Stephan Schröder
Stephane Maag
Sungwon Kang
Takashi Kitamura
Thomas Kuhn
Tomohiko Ogishi
Toru Hasegawa
Wenhui Zhang
Xinxin Liu
Yolanda Ortega-Mallén
Yuan Fang Li
Yutaka Miyake

Sponsoring Institutions

National Science Council, Taiwan, ROC
Ministry of Education, Taiwan, ROC
Institute of Information Science, Academia Sinica, Taiwan, ROC
Center for Information and Electronic Technologies, NTU
Graduate Institute of Communication Engineering, NTU
Computer and Information Networking Center, NTU

Lecture Notes in Computer Science

For information about Vols. 1–3636

please contact your bookseller or Springer

Vol. 3739: W. Fan, Z. Wu, J. Yang (Eds.), Advances in Web-Age Information Management. XXII, 930 pages. 2005.

Vol. 3738: V.R. Syrotiuk, E. Chávez (Eds.), Ad-Hoc, Mobile, and Wireless Networks. XI, 360 pages. 2005.

Vol. 3731: F. Wang (Ed.), Formal Techniques for Networked and Distributed Systems - FORTE 2005. XII, 558 pages. 2005.

Vol. 3728: V. Paliouras, J. Vounckx, D. Verkest (Eds.), Integrated Circuit and System Design. XV, 753 pages. 2005.

Vol. 3726: L.T. Yang, O.F. Rana, B. Di Martino, J. Dongarra (Eds.), High Performance Computing and Communcations. XXVI, 1116 pages. 2005.

Vol. 3725: D. Borrione, W. Paul (Eds.), Correct Hardware Design and Verification Methods. XII, 412 pages. 2005.

Vol. 3724: P. Fraigniaud (Ed.), Distributed Computing. XIV, 520 pages. 2005.

Vol. 3718: V.G. Ganzha, E.W. Mayr, E.V. Vorozhtsov (Eds.), Computer Algebra in Scientific Computing. XII, 502 pages. 2005.

Vol. 3717: B. Gramlich (Ed.), Frontiers of Combining Systems. X, 321 pages. 2005. (Subseries LNAI).

Vol. 3715: E. Dawson, S. Vaudenay (Eds.), Progress in Cryptology – Mycrypt 2005. XI, 329 pages. 2005.

Vol. 3714: H. Obbink, K. Pohl (Eds.), Software Product Lines. XIII, 235 pages. 2005.

Vol. 3713: L. Briand, C. Williams (Eds.), Model Driven Engineering Languages and Systems. XV, 722 pages. 2005.

Vol. 3712: R. Reussner, J. Mayer, J.A. Stafford, S. Overhage, S. Becker, P.J. Schroeder (Eds.), Quality of Software Architectures and Software Quality. XIII, 289 pages. 2005.

Vol. 3711: F. Kishino, Y. Kitamura, H. Kato, N. Nagata (Eds.), Entertainment Computing - ICEC 2005. XXIV, 540 pages. 2005.

Vol. 3710: M. Barni, I. Cox, T. Kalker, H.J. Kim (Eds.), Digital Watermarking. XII, 485 pages. 2005.

Vol. 3708: J. Blanc-Talon, W. Philips, D. Popescu, P. Scheunders (Eds.), Advanced Concepts for Intelligent Vision Systems. XXII, 725 pages. 2005.

Vol. 3707: D.A. Peled, Y.-K. Tsay (Eds.), Automated Technology for Verification and Analysis. XII, 506 pages. 2005.

Vol. 3706: H. Fuks, S. Lukosch, A.C. Salgado (Eds.), Groupware: Design, Implementation, and Use. XII, 378 pages. 2005.

Vol. 3703: F. Fages, S. Soliman (Eds.), Principles and Practice of Semantic Web Reasoning. VIII, 163 pages. 2005.

Vol. 3702: B. Beckert (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. XIII, 343 pages. 2005. (Subseries LNAI).

Vol. 3701: M. Coppo, E. Lodi, G. M. Pinna (Eds.), Theoretical Computer Science. XI, 411 pages. 2005.

Vol. 3699: C.S. Calude, M.J. Dinneen, G. Păun, M. J. Pérez-Jiménez, G. Rozenberg (Eds.), Unconventional Computation. XI, 267 pages. 2005.

Vol. 3698: U. Furbach (Ed.), KI 2005: Advances in Artificial Intelligence. XIII, 409 pages. 2005. (Subseries LNAI).

Vol. 3697: W. Duch, J. Kacprzyk, E. Oja, S. Zadrożny (Eds.), Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005, Part II. XXXII, 1045 pages. 2005.

Vol. 3696: W. Duch, J. Kacprzyk, E. Oja, S. Zadrożny (Eds.), Artificial Neural Networks: Biological Inspirations – ICANN 2005, Part I. XXXI, 703 pages. 2005.

Vol. 3695: M.R. Berthold, R. Glen, K. Diederichs, O. Kohlbacher, I. Fischer (Eds.), Computational Life Sciences. XI, 277 pages. 2005. (Subseries LNBI).

Vol. 3694: M. Malek, E. Nett, N. Suri (Eds.), Service Availability. VIII, 213 pages. 2005.

Vol. 3693: A.G. Cohn, D.M. Mark (Eds.), Spatial Information Theory. XII, 493 pages. 2005.

Vol. 3692: R. Casadio, G. Myers (Eds.), Algorithms in Bioinformatics. X, 436 pages. 2005. (Subseries LNBI).

Vol. 3691: A. Gagalowicz, W. Philips (Eds.), Computer Analysis of Images and Patterns. XIX, 865 pages. 2005.

Vol. 3690: M. Pěchouček, P. Petta, L.Z. Varga (Eds.), Multi-Agent Systems and Applications IV. XVII, 667 pages. 2005. (Subseries LNAI).

Vol. 3688: R. Winther, B.A. Gan, G. Dahll (Eds.), Computer Safety, Reliability, and Security. XI, 405 pages. 2005.

Vol. 3687: S. Singh, M. Singh, C. Apte, P. Perner (Eds.), Pattern Recognition and Image Analysis, Part II. XXV, 809 pages. 2005.

Vol. 3686: S. Singh, M. Singh, C. Apte, P. Perner (Eds.), Pattern Recognition and Data Mining, Part I. XXVI, 689 pages. 2005.

Vol. 3685: V. Gorodetsky, I. Kotenko, V. Skormin (Eds.), Computer Network Security. XIV, 480 pages. 2005.

Vol. 3684: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part IV. LXXIX, 933 pages. 2005. (Subseries LNAI).

Vol. 3683: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part III. LXXX, 1397 pages. 2005. (Subseries LNAI).

Vol. 3682: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part II. LXXIX, 1371 pages. 2005. (Subseries LNAI).

Vol. 3681: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part I. LXXX, 1319 pages. 2005. (Subseries LNAI).

Vol. 3679: S.d.C. di Vimercati, P. Syverson, D. Gollmann (Eds.), Computer Security – ESORICS 2005. XI, 509 pages. 2005.

Vol. 3678: A. McLysaght, D.H. Huson (Eds.), Comparative Genomics. VIII, 167 pages. 2005. (Subseries LNBI).

Vol. 3677: J. Dittmann, S. Katzenbeisser, A. Uhl (Eds.), Communications and Multimedia Security. XIII, 360 pages. 2005.

Vol. 3676: R. Glück, M. Lowry (Eds.), Generative Programming and Component Engineering. XI, 448 pages. 2005.

Vol. 3675: Y. Luo (Ed.), Cooperative Design, Visualization, and Engineering. XI, 264 pages. 2005.

Vol. 3674: W. Jonker, M. Petković (Eds.), Secure Data Management. X, 241 pages. 2005.

Vol. 3673: S. Bandini, S. Manzoni (Eds.), AI*IA 2005: Advances in Artificial Intelligence. XIV, 614 pages. 2005. (Subseries LNAI).

Vol. 3672: C. Hankin, I. Siveroni (Eds.), Static Analysis. X, 369 pages. 2005.

Vol. 3671: S. Bressan, S. Ceri, E. Hunt, Z.G. Ives, Z. Bellahsène, M. Rys, R. Unland (Eds.), Database and XML Technologies. X, 239 pages. 2005.

Vol. 3670: M. Bravetti, L. Kloul, G. Zavattaro (Eds.), Formal Techniques for Computer Systems and Business Processes. XIII, 349 pages. 2005.

Vol. 3669: G.S. Brodal, S. Leonardi (Eds.), Algorithms – ESA 2005. XVIII, 901 pages. 2005.

Vol. 3668: M. Gabbrielli, G. Gupta (Eds.), Logic Programming. XIV, 454 pages. 2005.

Vol. 3666: B.D. Martino, D. Kranzlmüller, J. Dongarra (Eds.), Recent Advances in Parallel Virtual Machine and Message Passing Interface. XVII, 546 pages. 2005.

Vol. 3665: K. S. Candan, A. Celentano (Eds.), Advances in Multimedia Information Systems. X, 221 pages. 2005.

Vol. 3664: C. Türker, M. Agosti, H.-J. Schek (Eds.), Peer-to-Peer, Grid, and Service-Orientation in Digital Library Architectures. X, 261 pages. 2005.

Vol. 3663: W.G. Kropatsch, R. Sablatnig, A. Hanbury (Eds.), Pattern Recognition. XIV, 512 pages. 2005.

Vol. 3662: C. Baral, G. Greco, N. Leone, G. Terracina (Eds.), Logic Programming and Nonmonotonic Reasoning. XIII, 454 pages. 2005. (Subseries LNAI).

Vol. 3661: T. Panayiotopoulos, J. Gratch, R. Aylett, D. Ballin, P. Olivier, T. Rist (Eds.), Intelligent Virtual Agents. XIII, 506 pages. 2005. (Subseries LNAI).

Vol. 3660: M. Beigl, S. Intille, J. Rekimoto, H. Tokuda (Eds.), UbiComp 2005: Ubiquitous Computing. XVII, 394 pages. 2005.

Vol. 3659: J.R. Rao, B. Sunar (Eds.), Cryptographic Hardware and Embedded Systems – CHES 2005. XIV, 458 pages. 2005.

Vol. 3658: V. Matoušek, P. Mautner, T. Pavelka (Eds.), Text, Speech and Dialogue. XV, 460 pages. 2005. (Subseries LNAI).

Vol. 3657: F.S. de Boer, M.M. Bonsangue, S. Graf, W.-P. de Roever (Eds.), Formal Methods for Components and Objects. VIII, 325 pages. 2005.

Vol. 3656: M. Kamel, A. Campilho (Eds.), Image Analysis and Recognition. XXIV, 1279 pages. 2005.

Vol. 3655: A. Aldini, R. Gorrieri, F. Martinelli (Eds.), Foundations of Security Analysis and Design III. VII, 273 pages. 2005.

Vol. 3654: S. Jajodia, D. Wijesekera (Eds.), Data and Applications Security XIX. X, 353 pages. 2005.

Vol. 3653: M. Abadi, L. de Alfaro (Eds.), CONCUR 2005 – Concurrency Theory. XIV, 578 pages. 2005.

Vol. 3652: A. Rauber, S. Christodoulakis, A M. Tjoa (Eds.), Research and Advanced Technology for Digital Libraries. XVIII, 545 pages. 2005.

Vol. 3651: R. Dale, K.-F. Wong, J. Su, O.Y. Kwong (Eds.), Natural Language Processing – IJCNLP 2005. XXI, 1031 pages. 2005. (Subseries LNAI).

Vol. 3650: J. Zhou, J. Lopez, R.H. Deng, F. Bao (Eds.), Information Security. XII, 516 pages. 2005.

Vol. 3649: W.M. P. van der Aalst, B. Benatallah, F. Casati, F. Curbera (Eds.), Business Process Management. XII, 472 pages. 2005.

Vol. 3648: J.C. Cunha, P.D. Medeiros (Eds.), Euro-Par 2005 Parallel Processing. XXXVI, 1299 pages. 2005.

Vol. 3646: A. F. Famili, J.N. Kok, J.M. Peña, A. Siebes, A. Feelders (Eds.), Advances in Intelligent Data Analysis VI. XIV, 522 pages. 2005.

Vol. 3645: D.-S. Huang, X.-P. Zhang, G.-B. Huang (Eds.), Advances in Intelligent Computing, Part II. XIII, 1010 pages. 2005.

Vol. 3644: D.-S. Huang, X.-P. Zhang, G.-B. Huang (Eds.), Advances in Intelligent Computing, Part I. XXVII, 1101 pages. 2005.

Vol. 3643: R. Moreno Díaz, F. Pichler, A. Quesada Arencibia (Eds.), Computer Aided Systems Theory – EUROCAST 2005. XIV, 629 pages. 2005.

Vol. 3642: D. Ślezak, J. Yao, J.F. Peters, W. Ziarko, X. Hu (Eds.), Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Part II. XXIII, 738 pages. 2005. (Subseries LNAI).

Vol. 3641: D. Ślezak, G. Wang, M. Szczuka, I. Düntsch, Y. Yao (Eds.), Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Part I. XXIV, 742 pages. 2005. (Subseries LNAI).

Vol. 3639: P. Godefroid (Ed.), Model Checking Software. XI, 289 pages. 2005.

Vol. 3638: A. Butz, B. Fisher, A. Krüger, P. Olivier (Eds.), Smart Graphics. XI, 269 pages. 2005.

Vol. 3637: J. M. Moreno, J. Madrenas, J. Cosp (Eds.), Evolvable Systems: From Biology to Hardware. XI, 227 pages. 2005.

¥868.48元

Table of Contents

KEYNOTE SPEECHES

REGULAR PAPERS

SHORT PAPERS

Ranking Abstraction as Companion to Predicate Abstraction*

Ittai Balaban[1], Amir Pnueli[1,2], and Lenore D. Zuck[3]

[1] New York University, New York
{balaban, amir}@cs.nyu.edu
[2] Weizmann Institute of Science
[3] University of Illinois at Chicago
lenore@cs.uic.edu

Abstract. Predicate abstraction has become one of the most successful methodologies for proving safety properties of programs. Recently, several abstraction methodologies have been proposed for proving liveness properties. This paper studies "ranking abstraction" where a program is augmented by a non-constraining progress monitor, and further abstracted by predicate-abstraction, to allow for automatic verification of progress properties. Unlike most liveness methodologies, the augmentation does not require a complete ranking function that is expected to decrease with each step. Rather, the inputs are component rankings from which a complete ranking function may be formed.

The premise of the paper is an analogy between the methods of ranking abstraction and predicate abstraction, one ingredient of which is refinement: When predicate abstraction fails, one can refine it. When ranking abstraction fails, one must determine whether the predicate abstraction, or the ranking abstraction, need be refined. The paper presents strategies for determining which case is at hand.

The other part of the analogy is that of automatically deriving deductive proof constructs: Predicate abstraction is often used to derive program invariants for proving safety properties as a boolean combination of the given predicates. Deductive proof of progress properties requires well-founded ranking functions instead of invariants. We show how to obtain concrete global ranking functions from abstract programs.

We demonstrate the various methods on examples with nested loops, including a bubble sort algorithm on linked lists.

1 Introduction

Predicate abstraction has become one of the most successful methodologies for proving safety properties of programs. However, with no extension it cannot be used to verify general liveness properties. In this paper, we present a framework, based on predicate abstraction and *ranking abstraction*, for verification of both safety and progress properties. Ranking abstraction, introduced in [7], is based on an augmentation of the concrete program. The augmentation is parameterized by a set of well founded ranking

* This research was supported in part by NSF grant CCR-0205571, ONR grant N00014-99-1-0131, and Israel Science Foundation grant 106/02-1.

F. Wang (Ed.): FORTE 2005, LNCS 3731, pp. 1–12, 2005.

functions. Based on these, new *compassion* (strong fairness) requirements as well as transitions are generated, all of which are synchronously composed with the program in a non-constraining manner. Unlike most methodologies, the ranking functions are not expected to decrease with each transition of the program. Rather, a further step of state abstraction is performed such that, coupled with the new compassion, it serves to construct a complete ranking function.

The basic premise presented in this paper is that there is a duality between the activities that lead to verification of safety properties via predicate abstraction, and those that lead to verification of progress properties via ranking abstraction. This duality is expressed through the following components:

- *The initial abstraction.* Heuristics are applied to choose either an initial set of predicates, or a set of core well founded ranking functions.
- *Refinement.* A too-coarse initial abstraction leads to spurious abstract counterexamples. Depending on the character of the counterexample, either a predicate, or a ranking, refinement is performed.
- *Generation of deductive proof constructs.* Predicate abstraction is often used as an automatic method to generate an inductive invariant as a boolean combination of the given predicates. Dually, ranking abstraction can be used to generate a global concrete ranking function that decreases with every step of the program, as a lexicographical combination of the core ranking functions.

We demonstrate the use of ranking refinement in order to prove termination of a canonical program with nested loops and unbounded random assignments, as well as a bubble sort algorithm on unbounded linked lists. Both examples entail the use of additional heuristics in order to synthesize core ranking functions.

The framework, as well as all experiments, have been implemented using the TLV interactive model-checker [1]. The contribution of the paper is as follows: At the informal, philosophical level, it strives to convince the reader that the duality between invariance and progress, present in deductive frameworks, extends to how one approaches automatic verification of each kind of property. More concretely, it suggests a formal framework, based on two specific abstraction methods for proving both safety and progress properties. This includes heuristics for choosing separate refinement methodologies based on the form of counterexamples, and a method for automatically deriving a global well founded program ranking function.

The paper is organized as follows: Section 2 describes the computational model of *fair discrete systems* as well as predicate and ranking abstractions. Furthermore, it motivates the use of ranking abstraction by demonstrating its value, compared to a typical deductive method. Section 3 formalizes the different notions of abstraction refinement. Section 4 presents a method for extracting a global ranking function from an abstract program. Finally, Section 5 summarizes and concludes.

Related Work

Dams, Gerth, and Grumberg [6] point out the duality between verification of safety and progress of programs. Like us, they aim to lift this duality to provide tools for proving progress properties, whose functionality is analogous to similar tools used for

safety. Specifically, they propose a heuristic for discovering ranking functions from a program's text. In contrast, we concentrate on an analogy with predicate abstraction, a particular method for safety. Our approach is broader, however, in that we suggest a general framework for safety and progress properties where each of the activities in a verification process has an instantiation with respect to each of the dualities.

In [10] Podelski and Rybalchenko present a method for synthesis of linear ranking functions. The method is complete for unnested loops, and is embedded successfully in a broader framework for proving liveness properties [9].

The topic of refinement of state abstraction, specifically predicate abstraction, has been widely studied. A number of existing works in this area are [5,3], and [4].

2 The Formal Framework

In this section we present our computational model, as well as the methods of predicate abstraction and ranking abstraction.

2.1 Fair Discrete Systems

As our computational model, we take a *fair discrete system* (FDS) $S = \langle V, \Theta, \rho, \mathcal{J}, \mathcal{C} \rangle$, where

- V — A set of *system variables*. A *state* of S provides a type-consistent interpretation of the variables V. For a state s and a system variable $v \in V$, we denote by $s[v]$ the value assigned to v by the state s. Let Σ denote the set of all states over V.
- Θ — The *initial condition*: An assertion (state formula) characterizing the initial states.
- $\rho(V, V')$ — The *transition relation*: An assertion, relating the values V of the variables in state $s \in \Sigma$ to the values V' in an S-successor state $s' \in \Sigma$.
- $\mathcal{J}$ — A set of *justice* (*weak fairness*) requirements (assertions); A computation must include infinitely many states satisfying each of the justice requirements.
- $\mathcal{C}$ — A set of *compassion (strong fairness)* requirements: Each compassion requirement is a pair $\langle p, q \rangle$ of state assertions; A computation should include either only finitely many p-states, or infinitely many q-states.

For an assertion ψ, we say that $s \in \Sigma$ is a ψ-state if $s \models \psi$.

A *computation* of an FDS S is an infinite sequence of states $\sigma : s_0, s_1, s_2, ...$, satisfying the requirements:

- *Initiality* — s_0 is initial, i.e., $s_0 \models \Theta$.
- *Consecution* — For each $\ell = 0, 1, ...$, the state $s_{\ell+1}$ is an S-successor of s_ℓ. That is, $\langle s_\ell, s_{\ell+1} \rangle \models \rho(V, V')$ where, for each $v \in V$, we interpret v as $s_\ell[v]$ and v' as $s_{\ell+1}[v]$.
- *Justice* — for every $J \in \mathcal{J}$, σ contains infinitely many occurrences of J-states.
- *Compassion* – for every $\langle p, q \rangle \in \mathcal{C}$, either σ contains only finitely many occurrences of p-states, or σ contains infinitely many occurrences of q-states.

2.2 Predicate Abstraction

The material here is a summary of [7] and [2]. We fix an FDS $S = \langle V, \Theta, \rho, \mathcal{J}, \mathcal{C} \rangle$ whose set of states is Σ. A *predicate abstraction* is a mapping $\alpha\colon \Sigma \to \{0,1\}^n$ for some positive n. The set of tuples $\{0,1\}^n$ is referred to as the set of *abstract states*. We focus on abstractions that can be represented by a set of equations of the form $\{u_i = P_i(V) \mid i = 1, \ldots, n\}$, where the P_i's are assertions over the concrete variables V, to which we refer as *predicates*, and $U = \{u_1, \ldots, u_n\}$ is the set of boolean *abstract variables*. The mapping α can also be expressed more succinctly by:

$$U = \mathcal{P}(V)$$

For an assertion $p(V)$, we define its abstraction by:

$$\alpha(p)\colon \ \exists V.(U = \mathcal{P}(V) \ \wedge \ p(V))$$

The semantics of $\alpha(p)$ is $\|\alpha(p)\| = \{\alpha(s) \mid s \in \|p\|\}$. Note that $\|\alpha(p)\|$ is, in general, an over-approximation – an abstract state is in $\|\alpha(p)\|$ iff *there exists* some concrete p-state that is abstracted into it. An assertion $p(V, V')$ over both primed and unprimed variables is abstracted by:

$$\alpha(p)\colon \ \exists V, V'.(U = \mathcal{P}(V) \ \wedge \ U' = \mathcal{P}(V') \ \wedge \ p(V, V'))$$

The assertion p is said to be *precise with respect to the abstraction* α if $\|p\| = \alpha^{-1}(\|\alpha(p)\|)$, i.e., if two concrete states are abstracted into the same abstract state, they are either both p-states, or they are both $\neg p$-states. For a temporal formula ψ in positive normal form (where negation is applied only to state assertions), ψ^α is the formula obtained by replacing every maximal state sub-formula p in ψ by $\alpha(p)$. The formula ψ is said to be *precise with respect to* α if each of its maximal state sub-formulas are precise with respect to α.

In all cases discussed in this paper, the formulae are precise with respect to the relevant abstractions. Hence, we can restrict to the over-approximation semantics.

The abstraction of S by α is the system

$$S^\alpha \ = \ \langle U, \alpha(\Theta), \alpha(\rho), \bigcup_{J \in \mathcal{J}} \alpha(J), \ \bigcup_{(p,q) \in \mathcal{C}} (\alpha(p), \alpha(q)) \rangle$$

The soundness of predicate abstraction is derived from [7]:

Theorem 1. *For a system S, abstraction α, and a positive normal form temporal formula ψ:*

$$S^\alpha \models \psi^\alpha \qquad \Longrightarrow \qquad S \models \psi$$

Thus, if an abstract system satisfies an abstract property, then the concrete system satisfies the concrete property.

2.3 Ranking Abstraction

State abstraction often does not suffice to verify progress properties. We consider *ranking abstraction*, a method of augmenting the concrete program in a non-constraining