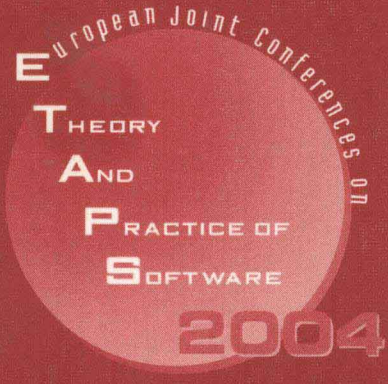
Evelyn Duesterwald (Ed.)

LNCS 2985

Compiler Construction

13th International Conference, CC 2004
Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2004
Barcelona, Spain, March/April 2004, Proceedings



Springer

Evelyn Duesterwald (Ed.)

Compiler Construction

13th International Conference, CC 2004
Held as Part of the Joint European Conferences
on Theory and Practice of Software, ETAPS 2004
Barcelona, Spain, March 29 – April 2, 2004
Proceedings

Springer

Volume Editor

Evelyn Duesterwald
IBM T.J. Watson Research Center
1101 Kitchawan Rd./Rte 134
Yorktown Heights, NY 10598, USA

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.

CR Subject Classification (1998): D.3.4, D.3.1, F.4.2, D.2.6, F.3, I.2.2

ISSN 0302-9743
ISBN 3-540-21297-3 Springer-Verlag Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by PTP-Berlin, Protago-TeX-Production GmbH
Printed on acid-free paper SPIN: 10990834 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 2985

Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Foreword

ETAPS 2004 was the seventh instance of the European Joint Conferences on Theory and Practice of Software. ETAPS is an annual federated conference that was established in 1998 by combining a number of existing and new conferences. This year it comprised five conferences (FOSSACS, FASE, ESOP, CC, TACAS), 23 satellite workshops, 1 tutorial, and 7 invited lectures (not including those that are specific to the satellite events).

The events that comprise ETAPS address various aspects of the system development process, including specification, design, implementation, analysis and improvement. The languages, methodologies and tools that support these activities are all well within its scope. Different blends of theory and practice are represented, with an inclination towards theory with a practical motivation on the one hand and soundly based practice on the other. Many of the issues involved in software design apply to systems in general, including hardware systems, and the emphasis on software is not intended to be exclusive.

ETAPS is a loose confederation in which each event retains its own identity, with a separate program committee and independent proceedings. Its format is open-ended, allowing it to grow and evolve as time goes by. Contributed talks and system demonstrations are in synchronized parallel sessions, with invited lectures in plenary sessions. Two of the invited lectures are reserved for "unifying" talks on topics of interest to the whole range of ETAPS attendees. The aim of cramming all this activity into a single one-week meeting is to create a strong magnet for academic and industrial researchers working on topics within its scope, giving them the opportunity to learn about research in related areas, and thereby to foster new and existing links between work in areas that were formerly addressed in separate meetings.

ETAPS 2004 was organized by the LSI Department of the Catalonia Technical University (UPC), in cooperation with:

European Association for Theoretical Computer Science (EATCS)
European Association for Programming Languages and Systems (EAPLS)
European Association of Software Science and Technology (EASST)
ACM SIGACT, SIGSOFT and SIGPLAN

The organizing team comprised

Jordi Cortadella (Satellite Events), Nikos Mylonakis, Robert Nieuwenhuis, Fernando Orejas (Chair), Edelmira Pasarella, Sonia Perez, Elvira Pino, Albert Rubio

and had the assistance of TILESA OPC.
ETAPS 2004 received generous sponsorship from:

UPC, Spanish Ministry of Science and Technology (MCYT), Catalan Department for Universities, Research and Information Society (DURSI), IBM, Intel.

Overall planning for ETAPS conferences is the responsibility of its Steering Committee, whose current membership is:

Ratislav Bodik (Berkeley), Maura Cerioli (Genoa), Evelyn Duesterwald (IBM, Yorktown Heights), Hartmut Ehrig (Berlin), José Fiadeiro (Leicester), Marie-Claude Gaudel (Paris), Andy Gordon (Microsoft Research, Cambridge), Roberto Gorrieri (Bologna), Nicolas Halbwachs (Grenoble), Görel Hedin (Lund), Kurt Jensen (Aarhus), Paul Klint (Amsterdam), Tiziana Margaria (Dortmund), Ugo Montanari (Pisa), Hanne Riis Nielson (Copenhagen), Fernando Orejas (Barcelona), Mauro Pezzè (Milan), Andreas Podelski (Saarbrücken), Mooly Sagiv (Tel Aviv), Don Sannella (Edinburgh), Vladimiro Sassone (Sussex), David Schmidt (Kansas), Bernhard Steffen (Dortmund), Perdita Stevens (Edinburgh), Andrzej Tarlecki (Warsaw), Igor Walukiewicz (Bordeaux), Michel Wermelinger (Lisbon)

I would like to express my sincere gratitude to all of these people and organizations, the program committee chairs and PC members of the ETAPS conferences, the organizers of the satellite events, the speakers themselves, and finally Springer-Verlag for agreeing to publish the ETAPS proceedings. This year, the number of submissions approached 600, making acceptance rates fall to 25%. I congratulate the authors who made it into the final program! I hope that all the other authors still found a way of participating in this exciting event and I hope you will continue submitting.

In 2005, ETAPS will be organized by Don Sannella in Edinburgh. You will be welcomed by another "local": my successor as ETAPS Steering Committee Chair – Perdita Stevens. My wish is that she will enjoy coordinating the next three editions of ETAPS as much as I have. It is not an easy job, in spite of what Don assured me when I succeeded him! But it is definitely a very rewarding one. One cannot help but feel proud of seeing submission and participation records being broken one year after the other, and that the technical program reached the levels of quality that we have been witnessing. At the same time, interacting with the organizers has been a particularly rich experience. Having organized the very first edition of ETAPS in Lisbon in 1998, I knew what they were going through, and I can tell you that each of them put his/her heart, soul, and an incredible amount of effort into the organization. The result, as we all know, was brilliant on all counts! Therefore, my last words are to thank Susanne Graf (2002), Andrzej Tarlecki and Paweł Urzyczyn (2003), and Fernando Orejas (2004) for the privilege of having worked with them.

Leicester, January 2004

José Luiz Fiadeiro
ETAPS Steering Committee Chair

Preface

The CC program committee is pleased to present this volume with the proceedings of the 13th International Conference on Compiler Construction (CC 2004). CC continues to provide an exciting forum for researchers, educators, and practitioners to exchange ideas on the latest developments in compiler technology, programming language implementation, and language design. The conference emphasizes practical and experimental work and invites contributions on methods and tools for all aspects of compiler technology and all language paradigms.

This volume serves as the permanent record of the 19 papers accepted for presentation at CC 2004 held in Barcelona, Spain, during April 1–2, 2004. The 19 papers in this volume were selected from 58 submissions. Each paper was assigned to three committee members for review. The program committee met for one day in December 2003 to discuss the papers and the reviews. By the end of the meeting, a consensus emerged to accept the 19 papers presented in this volume. However, there were many other quality submissions that could not be accommodated in the program; hopefully they will be published elsewhere.

The continued success of the CC conference series would not be possible without the help of the CC community. I would like to gratefully acknowledge and thank all of the authors who submitted papers and the many external reviewers who wrote reviews.

I especially thank the program committee members for their efforts and commitment in providing thoughtful and knowledgeable reviews. Special thanks go to our invited keynote speaker, Prof. Mary Lou Soffa from the University of Pittsburgh. Finally, many thanks to the entire ETAPS committee for making CC 2004 possible, especially to Fernando Orejas, the ETAPS 2004 organizing committee chair and to José Luiz Fiadeiro, the current chair of the ETAPS steering committee.

New York, January 2004 — Evelyn Duesterwald

Conference Organization

Program Chair

Evelyn Duesterwald, *IBM T.J. Watson Research Center*

Program Committee

Rastislav Bodik, *UC Berkeley, USA*
Evelyn Duesterwald, *IBM T.J. Watson Research Center, USA*
Christine Eisenbeis, *INRIA, France*
Paul Feautrier, *École Normale Supérieure de Lyon, France*
Jeanne Ferrante, *UC San Diego, USA*
Thomas Gross, *ETH Zürich, Switzerland*
Görel Hedin, *Lund University, Sweden*
Michael Hind, *IBM T.J.Watson Research Center, USA*
Nigel Horspool, *University of Victoria, B.C., Canada*
Susan Horwitz, *University of Wisconsin, USA*
Ulrich Kremer, *Rutgers University, USA*
Rainer Leupers, *Technical University of Aachen, Germany*
Josep Llosa, *UPC Barcelona, Spain*
Eduard Mehofer, *University of Vienna, Austria*
Samuel Midkiff, *Purdue University, USA*
Reinhard Wilhelm, *University of Saarbrücken, Germany*
Ben Zorn, *Microsoft, USA*

Referees

G. Almasi
J. Bauer
R. Brown
L. Carter
P. Clauss
A. Cohen
M. Daumas
B. Decker
A. Diwan
A. Ertl
L. Fei
S. Fink
R. Govindarajan
D. Grove
T. Harris
R. Henriksson
M. Jimenez
T. Johnson
M. O'Boyle
R. O'Callahan
J. Knoop
C. Krintz
J. Krinke
K. Lee
D. Liang
A. Nilsson
I. Pechtchanski
T. Proebsting
B. Rajkishore
F. Reig
T. Risset
S. Scoller
H. Seidl
V. Soni
S. Thesing
X. Vera
C. von Praun
S. Winkel

Lecture Notes in Computer Science

For information about Vols. 1–2864

please contact your bookseller or Springer-Verlag

Vol. 2996: V. Diekert, M. Habib (Eds.), STACS 2004. XVI, 658 pages. 2004.

Vol. 2993: R. Alur, G.J. Pappas (Eds.), Hybrid Systems: Computation and Control. XII, 674 pages. 2004.

Vol. 2992: E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, E. Ferrari (Eds.), Advances in Database Technology - EDBT 2004. XVIII, 877 pages. 2004.

Vol. 2991: R. Alt, A. Frommer, R.B. Kearfott, W. Luther (Eds.), Numerical Software with Result Verification. X, 315 pages. 2004.

Vol. 2985: E. Duesterwald (Ed.), Compiler Construction. X, 313 pages. 2004.

Vol. 2983: S. Istrail, M. Waterman, A. Clark (Eds.), Computational Methods for SNPs and Haplotype Inference. IX, 153 pages. 2004. (Subseries LNBI).

Vol. 2982: N. Wakamiya, M. Solarski, J. Sterbenz (Eds.), Active Networks. XI, 308 pages. 2004.

Vol. 2981: C. Müller-Schloer, T. Ungerer, B. Bauer (Eds.), Organic and Pervasive Computing – ARCS 2004. XI, 339 pages. 2004.

Vol. 2980: A. Blackwell, K. Marriott, A. Shimojima (Eds.), Diagrammatic Representation and Inference. XV, 448 pages. 2004. (Subseries LNAI).

Vol. 2978: R. Groz, R.M. Hierons (Eds.), Testing of Communicating Systems. XII, 225 pages. 2004.

Vol. 2976: M. Farach-Colton (Ed.), LATIN 2004: Theoretical Informatics. XV, 626 pages. 2004.

Vol. 2973: Y. Lee, J. Li, K.-Y. Whang, D. Lee (Eds.), Database Systems for Advanced Applications. XXIV, 925 pages. 2004.

Vol. 2970: F. Fernández Rivera, M. Bubak, A. Gómez Tato, R. Doallo (Eds.), Grid Computing. XI, 328 pages. 2004.

Vol. 2964: T. Okamoto (Ed.), Topics in Cryptology – CT-RSA 2004. XI, 387 pages. 2004.

Vol. 2962: S. Bistarelli, Semirings for Soft Constraint Solving and Programming. XII, 279 pages. 2004.

Vol. 2961: P. Eklund (Ed.), Concept Lattices. IX, 411 pages. 2004. (Subseries LNAI).

Vol. 2958: L. Rauchwerger (Ed.), Languages and Compilers for Parallel Computing. XI, 556 pages. 2004.

Vol. 2957: P. Langendoerfer, M. Liu, I. Matta, V. Tsaoussidis (Eds.), Wired/Wireless Internet Communications. XI, 307 pages. 2004.

Vol. 2954: F. Crestani, M. Dunlop, S. Mizzaro (Eds.), Mobile and Ubiquitous Information Access. X, 299 pages. 2004.

Vol. 2953: K. Konrad, Model Generation for Natural Language Interpretation and Analysis. XIII, 166 pages. 2004. (Subseries LNAI).

Vol. 2952: N. Guelfi, E. Astesiano, G. Reggio (Eds.), Scientific Engineering of Distributed Java Applications. X, 157 pages. 2004.

Vol. 2951: M. Naor (Ed.), Theory of Cryptography. XI, 523 pages. 2004.

Vol. 2949: R. De Nicola, G. Ferrari, G. Meredith (Eds.), Coordination Models and Languages. X, 323 pages. 2004.

Vol. 2947: F. Bao, R. Deng, J. Zhou (Eds.), Public Key Cryptography – PKC 2004. XI, 455 pages. 2004.

Vol. 2946: R. Focardi, R. Gorrieri (Eds.), Foundations of Security Analysis and Design II. VII, 267 pages. 2004.

Vol. 2943: J. Chen, J. Reif (Eds.), DNA Computing. X, 225 pages. 2004.

Vol. 2941: M. Wirsing, A. Knapp, S. Balsamo (Eds.), Radical Innovations of Software and Systems Engineering in the Future. X, 359 pages. 2004.

Vol. 2940: C. Lucena, A. Garcia, A. Romanovsky, J. Castro, P.S. Alencar (Eds.), Software Engineering for Multi-Agent Systems II. XII, 279 pages. 2004.

Vol. 2939: T. Kalker, I.J. Cox, Y.M. Ro (Eds.), Digital Watermarking. XII, 602 pages. 2004.

Vol. 2937: B. Steffen, G. Levi (Eds.), Verification, Model Checking, and Abstract Interpretation. XI, 325 pages. 2004.

Vol. 2934: G. Lindemann, D. Moldt, M. Paolucci (Eds.), Regulated Agent-Based Social Systems. X, 301 pages. 2004. (Subseries LNAI).

Vol. 2930: F. Winkler (Ed.), Automated Deduction in Geometry. VII, 231 pages. 2004. (Subseries LNAI).

Vol. 2926: L. van Elst, V. Dignum, A. Abecker, Agent-Mediated Knowledge Management. XI, 428 pages. 2004. (Subseries LNAI).

Vol. 2923: V. Lifschitz, I. Niemelä (Eds.), Logic Programming and Nonmonotonic Reasoning. IX, 365 pages. 2004. (Subseries LNAI).

Vol. 2919: E. Giunchiglia, A. Tacchella (Eds.), Theory and Applications of Satisfiability Testing. XI, 530 pages. 2004.

Vol. 2917: E. Quintarelli, Model-Checking Based Data Retrieval. XVI, 134 pages. 2004.

Vol. 2916: C. Palamidessi (Ed.), Logic Programming. XII, 520 pages. 2003.

Vol. 2915: A. Camurri, G. Volpe (Eds.), Gesture-Based Communication in Human-Computer Interaction. XIII, 558 pages. 2004. (Subseries LNAI).

Vol. 2914: P.K. Pandya, J. Radhakrishnan (Eds.), FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science. XIII, 446 pages. 2003.

Vol. 2913: T.M. Pinkston, V.K. Prasanna (Eds.), High Performance Computing - HiPC 2003. XX, 512 pages. 2003. (Subseries LNAI).

Vol. 2911: T.M.T. Sembok, H.B. Zaman, H. Chen, S.R. Urs, S.H. Myaeng (Eds.), Digital Libraries: Technology and Management of Indigenous Knowledge for Global Access. XX, 703 pages. 2003.

Vol. 2910: M.E. Orlowska, S. Weerawarana, M.M.P. Papazoglou, J. Yang (Eds.), Service-Oriented Computing - ICSOC 2003. XIV, 576 pages. 2003.

Vol. 2909: R. Solis-Oba, K. Jansen (Eds.), Approximation and Online Algorithms. VIII, 269 pages. 2004.

Vol. 2909: K. Jansen, R. Solis-Oba (Eds.), Approximation and Online Algorithms. VIII, 269 pages. 2004.

Vol. 2908: K. Chae, M. Yung (Eds.), Information Security Applications. XII, 506 pages. 2004.

Vol. 2907: I. Lirkov, S. Margenov, J. Wasniewski, P. Yalamov (Eds.), Large-Scale Scientific Computing. XI, 490 pages. 2004.

Vol. 2906: T. Ibaraki, N. Katoh, H. Ono (Eds.), Algorithms and Computation. XVII, 748 pages. 2003.

Vol. 2905: A. Sanfeliu, J. Ruiz-Shulcloper (Eds.), Progress in Pattern Recognition, Speech and Image Analysis. XVII, 693 pages. 2003.

Vol. 2904: T. Johansson, S. Maitra (Eds.), Progress in Cryptology - INDOCRYPT 2003. XI, 431 pages. 2003.

Vol. 2903: T.D. Gedeon, L.C.C. Fung (Eds.), AI 2003: Advances in Artificial Intelligence. XVI, 1075 pages. 2003. (Subseries LNAI).

Vol. 2902: F.M. Pires, S.P. Abreu (Eds.), Progress in Artificial Intelligence. XV, 504 pages. 2003. (Subseries LNAI).

Vol. 2901: F. Bry, N. Henze, J. Ma luszyński (Eds.), Principles and Practice of Semantic Web Reasoning. X, 209 pages. 2003.

Vol. 2900: M. Bidoit, P.D. Mosses (Eds.), Casl User Manual. XIII, 240 pages. 2004.

Vol. 2899: G. Ventre, R. Canonico (Eds.), Interactive Multimedia on Next Generation Networks. XIV, 420 pages. 2003.

Vol. 2898: K.G. Paterson (Ed.), Cryptography and Coding. IX, 385 pages. 2003.

Vol. 2897: O. Balet, G. Subsol, P. Torguet (Eds.), Virtual Storytelling. XI, 240 pages. 2003.

Vol. 2896: V.A. Saraswat (Ed.), Advances in Computing Science – ASIAN 2003. VIII, 305 pages. 2003.

Vol. 2895: A. Ohori (Ed.), Programming Languages and Systems. XIII, 427 pages. 2003.

Vol. 2894: C.S. Laih (Ed.), Advances in Cryptology - ASIACRYPT 2003. XIII, 543 pages. 2003.

Vol. 2893: J.-B. Stefani, I. Demeure, D. Hagimont (Eds.), Distributed Applications and Interoperable Systems. XIII, 311 pages. 2003.

Vol. 2892: F. Dau, The Logic System of Concept Graphs with Negation. XI, 213 pages. 2003. (Subseries LNAI).

Vol. 2891: J. Lee, M. Barley (Eds.), Intelligent Agents and Multi-Agent Systems. X, 215 pages. 2003. (Subseries LNAI).

Vol. 2890: M. Broy, A.V. Zamulin (Eds.), Perspectives of System Informatics. XV, 572 pages. 2003.

Vol. 2889: R. Meersman, Z. Tari (Eds.), On The Move to Meaningful Internet Systems 2003: OTM 2003 Workshops. XIX, 1071 pages. 2003.

Vol. 2888: R. Meersman, Z. Tari, D.C. Schmidt (Eds.), On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE. XXI, 1546 pages. 2003.

Vol. 2887: T. Johansson (Ed.), Fast Software Encryption. IX, 397 pages. 2003.

Vol. 2886: I. Nyström, G. Sanniti di Baja, S. Svensson (Eds.), Discrete Geometry for Computer Imagery. XII, 556 pages. 2003.

Vol. 2885: J.S. Dong, J. Woodcock (Eds.), Formal Methods and Software Engineering. XI, 683 pages. 2003.

Vol. 2884: E. Najm, U. Nestmann, P. Stevens (Eds.), Formal Methods for Open Object-Based Distributed Systems. X, 293 pages. 2003.

Vol. 2883: J. Schaeffer, M. Müller, Y. Björnsson (Eds.), Computers and Games. XI, 431 pages. 2003.

Vol. 2882: D. Veit, Matchmaking in Electronic Markets. XV, 180 pages. 2003. (Subseries LNAI).

Vol. 2881: E. Horlait, T. Magedanz, R.H. Glitho (Eds.), Mobile Agents for Telecommunication Applications. IX, 297 pages. 2003.

Vol. 2880: H.L. Bodlaender (Ed.), Graph-Theoretic Concepts in Computer Science. XI, 386 pages. 2003.

Vol. 2879: R.E. Ellis, T.M. Peters (Eds.), Medical Image Computing and Computer-Assisted Intervention - MICCAI 2003. XXXIV, 1003 pages. 2003.

Vol. 2878: R.E. Ellis, T.M. Peters (Eds.), Medical Image Computing and Computer-Assisted Intervention - MICCAI 2003. XXXIII, 819 pages. 2003.

Vol. 2877: T. Böhme, G. Heyer, H. Unger (Eds.), Innovative Internet Community Systems. VIII, 263 pages. 2003.

Vol. 2876: M. Schroeder, G. Wagner (Eds.), Rules and Rule Markup Languages for the Semantic Web. VII, 173 pages. 2003.

Vol. 2875: E. Aarts, R. Collier, E.v. Loenen, B.d. Ruyter (Eds.), Ambient Intelligence. XI, 432 pages. 2003.

Vol. 2874: C. Priami (Ed.), Global Computing. XIX, 255 pages. 2003.

Vol. 2871: N. Zhong, Z.W. Raś, S. Tsumoto, E. Suzuki (Eds.), Foundations of Intelligent Systems. XV, 697 pages. 2003. (Subseries LNAI).

Vol. 2870: D. Fensel, K.P. Sycara, J. Mylopoulos (Eds.), The Semantic Web - ISWC 2003. XV, 931 pages. 2003.

Vol. 2869: A. Yazici, C. Şener (Eds.), Computer and Information Sciences - ISCIS 2003. XIX, 1110 pages. 2003.

Vol. 2868: P. Perner, R. Brause, H.-G. Holzhütter (Eds.), Medical Data Analysis. VIII, 127 pages. 2003.

Vol. 2866: J. Akiyama, M. Kano (Eds.), Discrete and Computational Geometry. VIII, 285 pages. 2003.

Vol. 2865: S. Pierre, M. Barbeau, E. Kranakis (Eds.), Ad-Hoc, Mobile, and Wireless Networks. X, 293 pages. 2003.

the algorithm will update the sampled set with this new subpath id. Finally, the algorithm will decide how many blocks are to be skipped before sampling begins again, and will switch into skipping mode.

Every time subpath [3456] is sampled, its count in the sample is incremented. Note that it will be sampled at a rate about 3 times the rate of subpath $[4567_b]$, about 6 times the rate of subpath $[4567_a]$, and over 20 times the rate of subpaths $[12_b34]$ and $[2_b345]$. Also note that even for a sampling probability of about $\frac{1}{40}$, it is expected to be sampled approximately 150 times, enabling a very accurate estimate of its count.

2.2 Complexity Analysis

The skipping overhead, in the `enterBlock` method, is $O(1)$ operations per block, with a constant depending on the exact implementation of the skipping process. The sampling overhead, in the `sampleBlock` method, is $O(1)$ operations per sampled block. The cost of table resampling is controlled by setting the new sampling probability, and can be made to be amortized $O(1)$ per sampled block [8]. Since the number of sampled blocks is a small fraction of the total number of executed blocks, the total sampling overhead is $o(n)$, where n is the number of executed blocks, and is $o(1)$ amortized per executed block. A more detailed analysis is given in [12,13].

2.3 Special Considerations

Sampling and Skipping The sampling and counting are performed using a hot-list algorithm [8]. The hot-list algorithm is given an estimate of the input size, and a permissible memory footprint. From these values an initial sampling frequency f is computed, and each subpath is sampled with probability $\frac{1}{f}$.

Let m be the permissible memory footprint, G the expected gain and n the expected input size, then

$$f = \frac{n}{m \times G} \tag{1}$$

Instead of deciding for each subpath whether it should be sampled or not, a *skip value* is computed [18]. This value represents how many subpaths must be skipped before one should be sampled. The skip values are chosen so that their expected value is f, and for large values of f the performance gain can be important.

Subpaths For performance reasons, we observe that it is advantageous to only consider subpaths whose length is a power of two. Since the number of subpaths increases (quadratically) with the number of basic blocks, and the number of subpaths in the input affects accuracy for a given sample size, we improve performance by limiting the input set. Our choice provides significant reduction in the noise that exists in the sample set. Moreover, for any hot subpath of length k, we can find a subpath of length at least $\frac{k}{2}$ which is part of the sample space.

Table of Contents

Optimization

Code Generation and Backend Optimizations

Compiler Construction

Developing a Foundation for Code Optimization

Mary Lou Soffa

Department of Computer Science
University of Pittsburgh,
Pittsburgh, PA
soffa@cs.pitt.edu
www.cs.pitt.edu/~soffa

Abstract. Although optimization technology has been successful over the past 40 years, recent trends are emerging that demand we reconsider the paradigm that we are using for code optimization. In particular, the trends toward dynamic optimization, writing embedded system software in high level languages and the lack of significant performance improvement from current optimization research are forcing us to rethink what we know and do not know about optimization. A number of problems dealing with both semantic and application properties of optimizations have always existed but have been mostly ignored. The challenge in optimization today is to explore properties of optimizations and develop a framework for better understanding and use of optimizations. Fortunately, research is starting to explore properties, including proving the soundness of optimizations, proving the correctness of optimizations, specifications of optimizations, and the predictability of profits of optimizations. Not only must we understand the properties, but we also need to integrate the properties into a framework. Only then can we address the problems of the developing trends.

1 Introduction

The field of optimization has been extremely successful over the past 40+ years. As new languages and new architectures have been introduced, new and effective optimizations have been developed to target and exploit both the software and hardware innovations. Various reports from both research and commercial projects have indicated that the performance of software can be improved by 20% to 40% by applying levels of aggressive optimizations.

Most of the success in the field has come from the development of particular optimizations, such as partial redundancy elimination, speculation, and path sensitive optimization. Although we knew that there were various problems with optimization that were not well understood, they were mostly ignored. Thus instead of trying to understand and solve the problems, we avoided them for the most part because we were getting performance improvements. These problems included knowing when, where and what optimizations to apply for the best improvement. Other problems

E. Duesterwald (Ed.): CC 2004, LNCS 2985, pp. 1-4, 2004.

include showing the soundness of optimizations (an optimization does not change the semantics of a program) and the correctness of the optimizer that implements the optimizations. When optimizations are introduced, seldom is their soundness proved, and likewise optimizers are notorious for being buggy.

2 Technical Challenges

A number of recent events are forcing us to finally take up the challenge of the optimization problems. Because of the continued growth of embedded systems and the competitive market, where time-to-market is critical, there is a movement to write software for embedded system in high level languages. This movement requires an optimizing compiler to generate code that is near the quality of the manually produced code. Today's optimization technology is not able to adequately handle some of the challenges offered by embedded systems. For example, the resource constraints are more severe than in desktop computers and thus optimizations must be able to satisfy the constraints. Furthermore, embedded systems have multiple constraints, including execution time, memory and energy. Most of the prior work in optimization has really focused on a single constraint, namely time. The optimization technology has not been developed to handle the multiple constraints and the trade-offs. Another activity that has brought optimization problems to the forefront is the trend toward dynamic optimization. Dynamic optimization requires that we understand certain properties of optimizations in order in order for them to be effective. Currently, it is unclear when and where to apply optimizations dynamically and how aggressive the optimizations can be and still be profitable after factoring in the cost of applying the optimization. The third challenge is the lack of performance improvement that we are currently experiencing with optimization research. Although new optimizations continue to be developed, the performance improvement is shrinking. The question then is whether the field has reached its limit or is the problems that we have ignored simply limiting our progress. Lastly, the robustness of software has become a major concern, mandating that we ensure that the optimizing compilers are correct and optimizations sound.

To tackle these problems, we need to better understand the properties of optimization. We categorize optimization properties as either (1) semantic or (2) operational. Semantic properties deal with the semantics of the optimization and include

- correctness – the correctness of the implementation of optimizations,
- soundness – the semantics of a program do not change after applying an optimization, and
- specification – being able to specify conditions both the conditions needed and the semantics of applying the optimization.

Operational properties target the application of optimizations and their performance and include

- interaction – the conditions under which optimizations enable and/or disable other optimizations,
- profitability – the profit of applying an optimization at a particular point in the code given the resources of the targeted platform,
- order – the order that optimizations should be applied, based on the interaction property,
- configuration - the best optimization configuration, including tile size of loop tiling and the unroll factor, considering the resources.
- automatic generation – the conditions under which we can specify optimizations and have a tool that automatically implements the optimizations, and
- portability – the conditions to develop plug-in optimization components.

Research on these properties has been limited. However, more recently there has been a flurry of research activity focusing on optimization properties. There are two approaches to exploring the properties. One approach is through formal techniques. These include developing formal specifications of optimizations, analytical models, and proofs through model checking and theorem provers. Another approach is experimental. That is, the properties are explored by actually implementing optimizations and executing the optimized code, using the information gained to determine properties. The experimental approach can be performed prior to or in conjunction with actual program executions.

3 Related Research

Semantic properties have been formally tackled by proving the soundness of the optimizations [1] [2], and the correctness of the optimizers [3]. Formal specifications of optimizations were introduced a number of years ago by Whitfield and Soffa [4] for automatic implementation of optimizations. Recently, specification techniques have been developed to prove the soundness of optimizations [1] [2]. An experimental approach to the correctness of an optimizer involved checking both unoptimized and optimized code [5].

For operational properties, determining the order or the configuration to apply optimizations has recently been experimentally explored by Triantafyllis et al. [6], Cooper et al. [7], and Kisuki et al. [8]. The interaction and ordering property was formally explored by Whitfield and Soffa [9]. The profitability and predictability of optimizations is also being explored [10] [11].

4 Summary

The important challenge of optimization research today is to better understand properties of optimizations and use these properties when decisions about what optimization to apply, when to apply it and under what conditions to apply it. Also, it

tion to apply, when to apply it and under what conditions to apply it. Also, it is imperative that we know optimizations are sound and that the implementation of optimizations is robust. We need to develop a foundation that can be used to help us determine the properties both for existing optimizations and for future ones, and to integrate the properties. Such a foundation would enable us to better understand optimizations and help us meet the challenges of emerging technologies.

References

[1] Lacey, E., Jones, N.D., Eric Van Wyk, E., Christian Frederiksen, C., Proving Correctness of Compiler Optimizations by Temporal Logic, Proceedings of the ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Portland, Oregon, (2002) 283-294.

[2] Lerner, S., Millstein, T., and Chambers, C., Automatically Proving the Correctness of Compiler Optimizations, Proceedings of ACM SIGPLAN Conference on Programming Language Design and Implementation, (2003) 1-19.

[3] Necula, G.C., Translation Validation for an Optimizing Compiler, Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation, Vancouver, British Columbia, Canada (2000) 83-94.

[4] Whitfield D., Soffa, M.L., An Approach for Exploring Code Improving Transformations. ACM Transactions on Programming Languages, 19(6) (1997) 1053-1084.

[5] Jaramillo, C., Gupta, R., Soffa, M.L., Comparison Checking: An approach to avoid debugging of optimized code, ACM SIGSOFT Proceedings of Foundation of Software Engineering, (1999) 268-284.

[6] Triantafyllis, S., Vachharajani, M., Vachharajani, N., August, D., Compiler Optimization-space Exploration. 1st International Symposium on Code Generation and Optimization (2003) 204-215.

[7] K. Cooper, K., D. Subramanian, D., Torczon, L., Adaptive Optimizing Compilers for the 21st Century, Proceedings of the 2001 LACSI Symposium, Santa Fe, NM, USA, October (2001).

[8] Kisuki, T., Knijnenburg, P.M.W., O'Boyle, M.F.P., Combined Selection of Tile Size and Unroll Factors Using Iterative Compilation, International Conference on Parallel Architectures and Compilation Techniques (2000) 237-246.

[9] Whitfield, D., Soffa, M.L. An Approach to Ordering Optimizing Transformations, Proceedings ACM SIGPLAN Symposium on Principles & Practice of Parallel Programming, (1990) 137-146.

[10] Zhao, M., Childers, B., Soffa, M.L., Predicting the Impact of Optimizations for Embedded Systems, 2003 ACM SIGPLAN Conference on Languages, Compilers, and Tools for Embedded Systems, San Diego, CA. (2003) 1-11.

[11] Zhao, M., Childers, B., Soffa, M.L., Profit Driven Optimizations, Technical Report, University of Pittsburgh, Jan. (2004).