SpringerBriefs in Applied Sciences and Technology

PoliMI SpringerBriefs

Amir H. Ashouri · Gianluca Palermo
John Cavazos · Cristina Silvano

Automatic Tuning of Compilers Using Machine Learning

POLITECNICO
MILANO 1863

Springer

Amir H. Ashouri
Edward S. Rogers Sr. Department of Electrical and Computer Engineering (ECE)
University of Toronto
Toronto, ON
Canada

Gianluca Palermo
Department of Electronics, Information and Bioengineering (DEIB)
Politecnico di Milano
Milan
Italy

John Cavazos
Computer and Information Sciences (CIS)
University of Delaware
Newark, DE
USA

Cristina Silvano
Department of Electronics, Information and Bioengineering (DEIB)
Politecnico di Milano
Milan
Italy

ISSN 2191-530X ISSN 2191-5318 (electronic)
SpringerBriefs in Applied Sciences and Technology
ISSN 2282-2577 ISSN 2282-2585 (electronic)
PoliMI SpringerBriefs
ISBN 978-3-319-71488-2 ISBN 978-3-319-71489-9 (eBook)
https://doi.org/10.1007/978-3-319-71489-9

Library of Congress Control Number: 2017959897

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Learning never exhausts the mind
Imparare non stanca mai la mente
—Leonardo da Vinci

Foreword

Compilers have two jobs: translating programs into a form understandable by machines and making the translated code run efficiently. This second role, compiler optimization is a long-standing research problem. It has led to a large number of compiler heuristics or optimizations, each of which is designed to improve system performance. While each of these optimizations may deliver good performance individually, when combined they may degrade performance. Determining what optimizations to use and in what order depends on the program and the target platform. The different combinations and orderings quickly create a massive optimization space greater than the number of atoms in the known universe. The complexity of this problem prevents innovation in compiler research and leads to a loss of performance. In recent years, researchers have looked to search and machine learning-based approaches to navigate this complex space and select the best combination and sequence of optimizations.

This book tackles the difficult problem of determining the best set of compiler optimizations for a range of platforms. It addresses this problem using innovative machine learning-based solutions that exploit prior knowledge. This knowledge is used to build models that predict the right optimizations for unseen programs. It succinctly describes the fundamental research problem and extensively surveys the large body of prior work. This survey provides an excellent background to the topic.

This book makes four specific technical contributions. The first considers how to co-design VLIW micro-architecture and compiler optimizations using a performance/area Pareto curve. The second contribution is the use of a novel Bayesian network to predict the best optimizations using a method that explains how program features correlate with output. The third contribution is the use of a performance predictor to guide and select compiler optimizations without running the code. The final contribution is the most ambitious chapter, tackling phase order based on a unique optimization clustering approach.

This book provides an excellent state-of-the-art survey of compiler optimization, develops innovative solutions to long-standing problems, and most importantly of all opens up new lines of research in compiler optimization.

October 2017

Michael O'Boyle
University of Edinburgh,
Edinburgh UK

Preface

The diversity of today's architectures has forced programmers to spend additional efforts to port and tune their application source code across different platforms. In this process, compilers need additional tuning to generate better code. Recent compilers offer a vast number of multilayered optimizations, capable of targeting different code segments of an application. Choosing the right set of optimizations can significantly impact the performance of the code. This is a challenge made more complicated by the need to find the best order in which they should be applied given an application. Finding the best ordering is a long-standing problem in compilation research called the phase-ordering problem. The traditional approach for constructing compiler heuristics to solve it simply cannot cope with the enormous complexity of choosing the right ordering of optimizations for each code segment in an application. The current research focuses on exploring, studying, and developing an innovative approach to the problem of identifying the compiler optimizations that maximize the performance of a target application.

Overview of this Book

This book addresses two fundamental problems involved in compilation research: the problem of *selecting* the best compiler optimizations and the *phase-ordering* problem. Statistical analyses were extensively used to relate the performance of an application to the applied optimizations. More precisely, machine learning models were adapted to predict an outcome. Here, an outcome is described either in terms of performance metrics, or the use of a certain compiler optimization. Similar to other machine learning approaches, we use a set of training applications to learn the statistical relationships between application features and compiler optimizations. For instance, Bayesian networks are used to learn the statistical model to which an application can be represented with. We call these representations an application feature. Thus, given a new application not included in the training set, its features are fed to the Bayesian network as evidence. This evidence generates a bias on the distribution, as compiler optimizations are correlated with software features. The obtained probability distribution is application-specific and effectively focuses on the prediction of the most promising compiler optimizations. This will be discussed in detail in Chap. 3.

Who is this Book for?

This is a textbook that aims to showcase the very recent developments of research approaches in the compilation research, specifically autotuning. Therefore, all researchers in the compiler community, computer architecture, parallel computing, and machine learning can benefit from reading it. Additionally, given the potential industrial impact of the provided approaches, it is recommended to read to other technical professionals as well.

Summary and Organization

This book tackles the major problems of compiler autotuning. We use machine learning, DSE, and meta-heuristic techniques to construct efficient and accurate models to induce prediction models.

It is organized as follows: First, we provide an extensive review of the state of the art in Chap. 1. We survey more than hundred recent papers of the past twenty-five years since when the applications of machine learning have been introduced to compiler optimization field. Following the literature review, in Chap. 2, we provide a co-exploration approach using design space exploration technique for an embedded domain, namely VLIW. We show that this technique can speed up the performance of an application by using certain optimizations pass over our proposed VLIW micro-architecture. In Chap. 3, we present a novel machine learning approach to selecting the most promising compiler optimizations using Bayesian networks. This technique significantly improves application's performance against using fixed optimization available at GCC where our Bayesian network selects the most promising compiler passes. Chapters 4 and 5 are presenting our novel machine learning predictive models on how to tackle the phase-ordering problem. The former presents an intermediate approach, and the latter showcases a complete sequence speedup predictor on the very problem.

Finally, we present some concluding remarks and future works. Note that in this book, the bibliography is chapter-wise.

We hope this book brings the latest research done to a wide range of readers and promotes the use of machine learning on the field of compilation.

Toronto, ON, Canada	Amir H. Ashouri
Milan, Italy	Gianluca Palermo
Newark, DE, USA	John Cavazos
Milan, Italy	Cristina Silvano

Acknowledgements

The majority of the research related to this book has been carried out in the Department of Electronics, Information and Bioengineering (DEIB) at Politecnico di Milano. Additionally, I had the chance to be a Visiting Scholar in the Department of Electrical Engineering and Computer Science at the University of Delaware, USA. The collaboration allowed me to carry out further elaborations and analyses. The work described in this book was partially supported by the European Commission Call H2020-FET-HPC program under the grant ANTAREX-671623.

I would like to thank all my colleagues and advisers including postdoctoral and Ph.D. fellows with whom I had the opportunity to collaborate, specially Cristina Silvano, Gianluca Palermo, John Cavazos, Giovanni Mariani, Sotiris Xydis, Marco Alvarez, Eunjung Park, Sameer Kulkarni, William Kilian, Andrea Bignoli, and Robert Searles. The teamwork was truly fun and challenging at the same time, and I was grateful to participate in numerous constructive discussions. Many thanks to Michael O'Boyle and Erven Rohou for their valuable comments and provided reviews. Their insight on the compiler optimization field is truly inspiring.

Last but not least, I would like to appreciate the lifetime support of my lovely family: mother, father, and the younger brother who always have been my backbone during the hard times and the good times. Thank you for giving me the positive energy to carry on and for urging me to choose this path for my life.

Amir H. Ashouri
University of Toronto
Toronto, Canada

October 2017

Contents