

Parallel and Distributed Autotuning for High-Performance Computing

Pedro Bruel

TEXT SUBMITTED
TO THE
INSTITUTE DE MATHEMATICS AND STATISTICS
OF THE
UNIVERSITY OF SÃO PAULO
FOR THE
QUALIFYING EXAMINATION
FOR THE
DOCTORAL DEGREE IN
COMPUTER SCIENCE

Advisor: Professor Alfredo Goldman Vel Lejbman

During the development of this work, the author was supported by CNPq №XXXX,
Hewlett-Packard Enterprise and CAPES

São Paulo, March 7, 2017

Parallel and Distributed Autotuning for High-Performance Computing

This is the original version
submitted for evaluation
by Pedro Bruel

Abstract

BRUEL, P. **Parallel and Distributed Autotuning for High-Performance Computing**. 2017. xxx f. Qualifying Examination (PhD) - Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2017.

Elemento obrigatório, elaborado com as mesmas características do resumo em língua portuguesa. De acordo com o Regimento da Pós- Graduação da USP (Artigo 99), deve ser redigido em inglês para fins de divulgação.

Keywords: keyword1, keyword2, keyword3.

Contents

List of Symbols	iv
List of Abbreviations	v
List of Figures	vi
List of Tables	vi
1 Introduction	1
1.1 Objectives and Expected Contributions	1
1.2 Text Structure	1
2 Autonomous Search	3
2.1 Algorithm Selection and Autonomous Solvers	3
2.2 Off-line Configuration	3
2.2.1 Evolutionary Computing	3
2.2.2 Stochastic Local Search	3
2.2.3 Machine Learning	3
2.3 On-line Control	3
2.3.1 Adaptive Parameter Configuration	3
2.3.2 Credit Assignment	3
2.3.3 Reinforcement Learning	3
3 Autotuning	5
3.1 OpenTuner	5
3.1.1 Context	5
3.1.2 Software Architecture	5
3.2 Search Techniques	5
3.2.1 Numerical Methods	5
3.2.2 Evolutionary Computation	5
3.2.3 Stochastic Local Search	5
3.2.4 Machine Learning	5
3.3 Benchmarks	5
3.3.1 Solvers of NP-Complete Problems	5
3.3.2 Algorithm Selection and Configuration	5

3.3.3 Compiler Configuration	5
3.3.4 Measurement Time	5
4 Case Studies	7
4.1 Selecting Compiler Parameters for GPUs	7
4.1.1 Introduction	7
4.1.2 Results	7
4.1.3 Small Measurement Time	7
4.1.4 Summary and Future Work	7
4.2 Autotuning and Distributed Computing	7
4.2.1 Introduction	7
4.2.2 Results	7
4.2.3 Parallel and Distributed Programming in OpenTuner	7
4.2.4 Summary and Future Work	7
4.3 Selecting Parameters for High-Level Synthesis for FPGAs	7
4.3.1 Introduction	7
4.3.2 Results	7
4.3.3 Large Measurement Time	7
4.3.4 Summary and Future Work	7
5 Parallel and Distributed Autotuning in the Julia Language	9
5.1 The Julia Language	9
5.2 StochasticSearch.jl	9
5.2.1 Objective	9
5.2.2 Software Architecture	9
5.2.3 Results	9
5.2.4 Summary and Future Work	9
6 Summary and Discussion	11
6.1 Objective and Future Work	11
6.2 Schedule	11
Bibliography	13
Index	14

List of Symbols

ω	Frequência angular
ψ	Função de análise <i>wavelet</i>
Ψ	Transformada de Fourier de ψ

List of Abbreviations

CFT	Transformada contínua de Fourier (<i>Continuous Fourier Transform</i>)
DFT	Transformada discreta de Fourier (<i>Discrete Fourier Transform</i>)
EIP	Potencial de interação elétron-íon (<i>Electron-Ion Interaction Potentials</i>)
STFT	Transformada de Fourier de tempo reduzido (<i>Short-Time Fourier Transform</i>)

List of Figures

List of Tables

Introduction

A: Bilmes *et al.* (1997)

1.1 Objectives and Expected Contributions

1.2 Text Structure

Autonomous Search

2.1 Algorithm Selection and Autonomous Solvers

2.2 Off-line Configuration

2.2.1 Evolutionary Computing

2.2.2 Stochastic Local Search

2.2.3 Machine Learning

2.3 On-line Control

2.3.1 Adaptive Parameter Configuration

2.3.2 Credit Assignment

2.3.3 Reinforcement Learning

Autotuning

3.1 OpenTuner

3.1.1 Context

3.1.2 Software Architecture

3.2 Search Techniques

3.2.1 Numerical Methods

3.2.2 Evolutionary Computation

3.2.3 Stochastic Local Search

3.2.4 Machine Learning

3.3 Benchmarks

3.3.1 Solvers of NP-Complete Problems

3.3.2 Algorithm Selection and Configuration

3.3.3 Compiler Configuration

3.3.4 Measurement Time

Case Studies

4.1 Selecting Compiler Parameters for GPUs

4.1.1 Introduction

4.1.2 Results

4.1.3 Small Measurement Time

4.1.4 Summary and Future Work

4.2 Autotuning and Distributed Computing

4.2.1 Introduction

4.2.2 Results

4.2.3 Parallel and Distributed Programming in OpenTuner

4.2.4 Summary and Future Work

4.3 Selecting Parameters for High-Level Synthesis for FPGAs

4.3.1 Introduction

4.3.2 Results

4.3.3 Large Measurement Time

4.3.4 Summary and Future Work

Parallel and Distributed Autotuning in the Julia Language

5.1 The Julia Language

5.2 StochasticSearch.jl

5.2.1 Objective

5.2.2 Software Architecture

5.2.3 Results

5.2.4 Summary and Future Work

Summary and Discussion

6.1 Objective and Future Work

6.2 Schedule

Bibliography

Bilmes et al. (1997) Jeff Bilmes, Krste Asanovic, Chee-Whye Chin e Jim Demmel. Optimizing matrix multiply using phipac: a portable, high-performance, ansi c coding methodology. Em *Proceedings of International Conference on Supercomputing, Vienna, Austria*. Citado na pág. [1](#)

Index

DFT, *see* transformada discreta de Fourier

DSP, *see* processamento digital de sinais

Fourier

transformada, *see* transformada de Fourier

STFT, *see* transformada de Fourier de tempo
reduzido

TBP, *see* periodicidade região codificante