Autotuning under Budget Constraints: a Design of Experiments Approach

Pedro Bruel[*][†], Arnaud Legrand[*], Brice Videau[*] and Alfredo Goldman[†]
[*]Univ. Grenoble Alpes, CNRS, INRIA, LIG - Grenoble, France
Email: {arnaud.legrand, brice.videau}@imag.fr
[†]Univ. of São Paulo - São Paulo, Brazil
Email: {phrb,gold}@ime.usp.br

Abstract—Abstract

I. Introduction

Optimizing code for objectives such as performance and power consumption is fundamental to the success and cost effectiveness of industrial and scientific endeavours related to High Performance Computing. A considerable amount of highly specialized time and effort is spent in porting code to GPUs, FPGAs and other hardware accelerators. Experts are also needed in leveraging bleeding edge software improvements in compilers, languages, libraries and frameworks.

The automatic configuration and optimization of High Performance Computing applications, or autotuning, decreases the cost and time needed to adopt efficient hardware and sofware. Typical targets for autotuning include algorithm selection, source-to-source transformations and compiler configuration.

The autotuning of Hgh Performance Computing applications can be studied as a search problem, where the objective is to minimize single or multiple sofware of hardware metrics. The exploration of the search spaces defined by configurations and optimizations present interesting challenges to search algorithms. These search spaces grow exponentially with the number of considered configuration parameters and their possible values, and it is also difficult to explore these search spaces extensively, due to the often prohibitive costs of hardware utilization and program execution times. Developing efficient autotuning strategies is therefore essential for decreasing optimization cost and time.

II. Related Work

A. Source-to-source Transformation

B. Autotuning

C. Search Space Exploration Strategies

III. Design of Experiments

A. D-Optimal Designs

IV. Applying Design of Experiments to Autotuning

A. Experimental Methodology

B. Performance on a GPU Laplacian Kernel

V. Results on the SPAPT Benchmark

VI. Conclusion