A Performance Evaluation of Programming Languages Operating in Single Core Instructions

Parallel and Distributed Computing
Bachelors in Informatics and Computer Engineering

3L.EIC01_G5
Joel Fernandes up201904977@up.pt
Mário Travassos up201905871@up.pt
Tiago Rodrigues up201907021@up.pt

March 22, 2022

Contents

Chapter 1

Introduction

This project intends to show and evaluate the effect of processor performance when accessing large amounts of data, performing the same instructions multiple times. In this study, the product of two matrices was used as the base calculation.

Also, a comparison of how different programming languages interact with memory and impact the processor speed is shown. It is important to highlight that these tests were performed on a single core, so no parallelism optimizations are made.

Finally, performance measures were made using the Performance API (PAPI), which will be analyzed and discussed in further detail.

1.1 Problem Description

The problem used to evaluate the performance was the matrix multiplication. It was chosen because the amount of instructions does not impact performance tremendously, with the greatest bottleneck being memory access.

That way, we can measure more truthfully how much time does the processor spend accessing memory, and the impact that cache hits and misses have on a program.

Even though the main intention is to measure memory access performance, we also could see how some improvements in the algorithms used could make the processing time differ.

The first improvement was to multiply by line instead of the usual matrix multiplication. Then, a further improvement made was multiplying by block, which is shown to reduce running times.

Chapter 2

Algorithm Analysis

2.1 Normal Multiplication

2.2 Line Multiplication

2.3 Block Multiplication

Chapter 3

Performance Evaluation

3.1 Metrics Used

Cringe

3.2 Results Analysis

Cringe

Chapter 4

Conclusion

The way and order in which we perform computations can have a great impact on performance. The use of the proper algorithms, even in single core programs, can have a significant impact on processing time as well. Even more, the programming language used and the way it handles memory can also affect the time, as we have shown in this report.

The result turns what to be what was expected, with a language designed for speed like C++ beating out Java in every metric. Mention cache hits and misses whenever they are available

With this in mind, the report is satisfactory, as it concludes what was thought to be true beforehand, and it contributed to our knowledge of how machine instructions, order of operations and memory access can have a significant impact on the speed of programs.