POLITECNICO DI TORINO

Bachelor of Science in Mathematical Engineering

Bachelor degree thesis

Analysis of matrix vector product

Supervisor:
Prof. Stefano Berrone
Dr. Federico Tesser

Candidate:
Ludovico Bessi

July 2019

Ai miei genitori

Abstract

We made a comparison between a naive implementation of the matrix vector product and the Eigen's library implementation. We found out that on equal hardware power our code runs FASTEERR

Contents

Chapter 1

Introduction

Let A be a $n \times n$ dense matrix and v be a vector of n entries. We define the kernel as the following operation:

$$result[i] = result[i] + A[i, j] * vector[i]$$

. We used LIKIWID to profile the code and gather insights as well as stream. We went through different implementation of our kernel to get to the theoretical limit of our implementation using one core.

- Naive implementation
- Using a temporary variable to speed up cache lookup
- Loop unrolling to exploit sum and multiplication pipelines

With the last implementation, we managed to get to the theoretical limit. After that, we adressed scalability. That is, we looked at how our kernel worked in parallel on the supermegacomputer of politenico (inserire specifiche)

Chapter 2

Naive implementation

At first we started with the simplest thing one could think of:

```
for(int i = 0; i < n; i++){
  result[i] = 0;
  for(int j = 0; j < n; j++){
    result[i] += matrix[i][j]*vector[j];
  }
}
```

Chapter 3

Using a temporary variable to speed up cache lookup

Chapter 4

Loop unrolling to exploit sum and multiplication pipelines

Chapter 5

Parallel computations

Bibliography

[1] Jan Treibig, Georg Hager, Gerhard Wellein, *LIKWID: A lightweight performance-oriented tool suite for x86 multicore environments.*

[2] Mark bull, *A Short Introduction to OpenMP*

[3] Gaël Guennebaud, Benoît Jacob, *Eigen v3*

[4] Samuel Williams, Andrew Waterman, David Patterson, *An insightful visual performance model for floating point programs and multicore architectures*