Profiling Report

Group 1

May 2, 2022

Setup

To profile the code we utilised the programs `vtune`[1] and `perf`[2]. We ran the profiling on an Intel Whiskey Lake Processor. The input matrix was of dimension 1000×1000 with the expected results of dimension 1000×50 and 50×1000. We profiled both baselines to get a good feeling of where the bottlenecks in the straightforward and slightly optimized implementations are.

We focused on three measures to quantify the profile of the tested implementations. These were:

- Hotspot Analysis: We wanted to find out what parts of the code utilize the most CPU time to identify potentially expensive functions and calculations.
- Memory Bound: An approximation of the last level cache misses as well as the influence of cache misses on the computation related to the different functions.
- Micro-architecture Exploration: An estimate of how well the functions utilize the microarchitecture as well as a comparison on compute and memory boung parts of the functions.

Following we will discuss both baselines as well as their numbers with regards to these measures.

Baseline 1

Hotspot Analysis

Analysing the CPU time used by the different functions of baseline 1 has shown that especially our straightforward implementation of left transposed matrix multiplication has significant performance costs. This function alone accounted for 50.7% of total CPU time. Following this comes the right transposed matrix multiplication with also around 25% of CPU time. The straightforward matrix multiplication accounted for around 20% of CPU time. This information shows that those three functions have the highest influence on total CPU time and if we could at the same time improve their runtimes as well as the number of utilizations of these functions by reuse or similar initiatives, we could vastly save CPU time and thus most probably improve performance drastically.

[1] `https://www.intel.com/content/www/us/en/developer/tools/oneapi/vtune-profiler.html`
[2] `https://man7.org/linux/man-pages/man1/perf.1.html`

Memory Bound

This analysis has resulted in a very similar picture to the previously discussed analysis. The approximate number of last level cache misses of baseline 1 was two billions. The fraction of computation slots where the pipeline was stalled due to memory loads was around 28%. Also, once again left transposed matrix multiplication resulted in the most cache misses. Interestingly, also error and norm had significant memory traffic.

Micro-architecture Exploration

According to `vtune` micro-architecture performance analysis, which gives an approximate estimate on how well a given code is adapted to the micro-architecture of the profiling machine, left transposed matrix multiplication is 25% efficient, right transposed matrix multiplication 44% of efficiency and normal matrix multiplication 64% efficient.

The estimates also show that the micro-architecture is running around 25% of the time in a memory bound computation with less than optimal CPU usage and around 31% with a core bound that limits the CPU usage. Core bound means for example inefficient usage of pipelines due to dependencies. This means that around 44% of the time the CPU is running at full capacity of its pipelines.

Baseline 2

Hotspot Analysis

The profiling of baseline 2 showed, as expected, that the `dgemm` calls were the most CPU time consuming. Around 74% of CPU time was consumed by these calls. This shows once again the importance of reducing the number of matrix multiplications.

Memory Bound

The number of last level cache misses was around 300 millions and thus significantly better than baseline 1. Interestingly, once again also error and norm had significant memory traffic.

Micro-architecture Exploration

The efficiency indicators for baseline 2 showed that `dgemm` was adapted to utilize around 82% of the of the micro-architectures pipelines.

Additionally, the pipeline usage showed that baseline 2 was around 11% memory bound and 17% core bound. This displays a significantly more efficient CPU pipeline usage compared to baseline 1.