**TDT 4260: Computer Architecture**

**Climbing Mount Blanc – Optimization Log**

Submitted By: Amna Waris

Zahra Jenab Mahabadi

1. **Handed out code:**

* Virtual Machine

1. Execution time/CPU usage:

User time 53.36 s

System time 0.35 s

Total time elapsed: 54.08 s

CPU: 99%

1. Top 5 code portions with longest execution time:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Children** | **Self** | **Command** | **Shared Object** | **Symbol** |
| 99.91% | 0.00% | image\_processin | libc-2.31.so | [.]\_\_libc\_start\_main |
| 99.91% | 0.00% | image\_processin | image\_processing\_c | [.] main |
| 99.07% | 98.95% | image\_processin | image\_processing\_c | [.]blurIteration |
| 0.61% | 0.19% | image\_processin | image\_processing\_c | [.]convertToAccurate |
| 0.45% | 0.00% | image\_processin | [kernel.kallsyms] | [k]page\_fault |

* Climbing Mount Blanc Website

*Program was timed out after 90 seconds!*

1. **Cache and access patterns:**

As can be seen above the blurIteration function takes the most time. As the first step to optimize it, the ‘for’ loops in the function were interchanged to computer values in the horizontal direction first and then the vertical direction. This would be beneficial as the cache usually fetches data that is placed horizontally together first.

* Virtual Machine

1. Execution time/CPU usage:

User time 44.51 s

System time 0.30 s

Total time elapsed: 45.17 s

CPU: 99%

1. Top 5 code portions with longest execution time:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Children** | **Self** | **Command** | **Shared Object** | **Symbol** |
| 99.90% | 0.00% | image\_processin | libc-2.31.so | [.]\_\_libc\_start\_main |
| 99.90% | 0.00% | image\_processin | image\_processing\_c | [.] main |
| 98.81% | 98.67% | image\_processin | image\_processing\_c | [.]blurIteration |
| 0.80% | 0.22% | image\_processin | image\_processing\_c | [.]convertToAccurate |
| 0.60% | 0.00% | image\_processin | [kernel.kallsyms] | [k]page\_fault |

* Climbing Mount Blanc Website

*Program was timed out after 90 seconds!*

1. **Useless code and branches:**

It was noticed that blurIteration (function that takes the longest time to run), was being called several times in main to run for each color channel separately. This is redundant and all the calculations can just be done in one go. This will also remove the for loop in the main function and if/else statements (branches) in the blurIteration function.

* Virtual Machine

1. Execution time/CPU usage:

User time 11.95 s

System time 0.32 s

Total time elapsed: 12.55 s

CPU: 97%

1. Top 5 code portions with longest execution time:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Children** | **Self** | **Command** | **Shared Object** | **Symbol** |
| 99.68% | 0.00% | image\_processin | libc-2.31.so | [.]\_\_libc\_start\_main |
| 99.68% | 0.00% | image\_processin | image\_processing\_c | [.] main |
| 96.44% | 98.17% | image\_processin | image\_processing\_c | [.]blurIteration |
| 2.21% | 0.82% | image\_processin | image\_processing\_c | [.]convertToAccurate |
| 1.47% | 0.00% | image\_processin | [kernel.kallsyms] | [k]page\_fault |

* Climbing Mount Blanc Website

Time: 32.60 s

Energy: 105.00 J

EDP:3423.07 Js

A picture containing background pattern

Description automatically generated

1. **Changing algorithm:**

It was noticed that blurIteration (function that takes the longest time to run), was being called several times in main to run for each color channel separately. This is redundant and all the calculations can just be done in one go. This will also remove the for loop in the main function and if/else statements (branches) in the blurIteration function.

* Virtual Machine

1. Execution time/CPU usage:

User time 2.15 s

System time 0.39 s

Total time elapsed: 2.74 s

CPU: 92%

1. Top 5 code portions with longest execution time:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Children** | **Self** | **Command** | **Shared Object** | **Symbol** |
| 98.21% | 0.00% | image\_processin | libc-2.31.so | [.]\_\_libc\_start\_main |
| 98.21% | 0.00% | image\_processin | image\_processing\_c | [.] main |
| 79.03% | 78.09% | image\_processin | image\_processing\_c | [.]blurIteration |
| 14.27% | 3.63% | image\_processin | image\_processing\_c | [.]convertToAccurate |
| 11.44% | 0.04% | image\_processin | [kernel.kallsyms] | [k]page\_fault |

* Climbing Mount Blanc Website

Time: 10.88 s

Energy: 32.28 J

EDP:351.21 Js

A picture containing graphical user interface

Description automatically generated

1. **Parallelization:**

Parallelization can be used to run the code on different cores to make the execution time faster. The parallelization was done using OpenMP. Running the code on VM gave almost the same speed up because it has only one core, but a significant difference can be noted when it ran on CMB website.

* Virtual Machine

User time 2.23 s

System time 0.34 s

Total time elapsed: 2.77 s

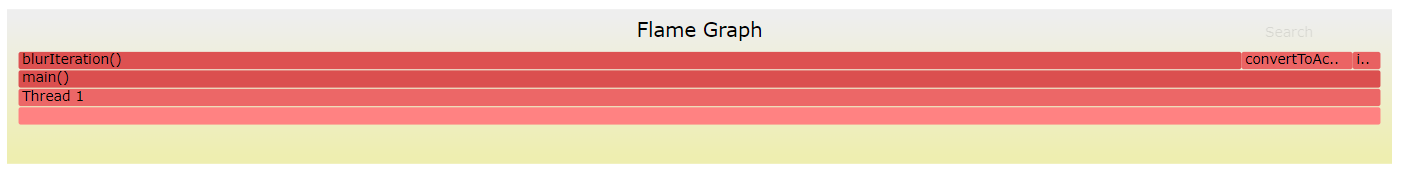
CPU: 92%

* Climbing Mount Blanc Website

Time: 6.56 s

Energy: 26.17 J

EDP: 171.71 Js



**SUMMARY**

The graphs below do not consider version 1 and version 2 for CMB because it was not optimized enough to run on CMB.