King Saud University
Collage of Computer and Information Sciences
Computer Science Department

INDIVIDUAL REPORT

Faisal Aldhuwayhi
438102142
CSC453 – Parallel Processing

Team Members Names

Faisal Aldhuwayhi – 438102142
Hassan Jaafar – 438105959
Fahad Aldeham – 438100439
Ahmad Almosallam – 438103307
Majed Alkhraiji - 438104708

Concluding the Findings:

From the experience that has been done, parallel processing has proven its power in providing excellent performance for solving such problems, especially with big size problems.

During the application of the experiment, one important thing that has been noticed is that the efficiency keeps decreasing as the number of processing units increase. Particularly, in a specific number of processing units, the efficiency would drop down noticeably. That would lead us to an induction, that there is a threshold of number processing units in each size of the problem in which having more processing units would affect the efficiency significantly in a negative way. In other words, the processing units would not work efficiently as desired to solve the problem.

The Limitations:

As seen from the outputs of the experiment, parallel processing metrics depend largely on the hardware resources/configuration, as well as the software implementation side.

So, hardware configuration (like memory limitations, number of processors) and software implementation (like the algorithm, the language) play a great role in restricting the results.

Trying a new hardware configuration, also a new method of software implementation would be a good attempt to the way of getting better results.

How to Enhance the Code in the Future:

Besides the methods that have been discussed in the team report about enhancing the code, the *false sharing* problem is one of the problems that need to be addressed when trying to enhance the experiment.

False sharing problem occurs when two or more cores hold a copy of the same memory cache line, and since one core updates, a variable would force other cores to update cache either. If this occurs frequently, the performance and scalability of the application will suffer significantly, especially for the execution time.

One way to solve that is cache padding, which means, padding some meaningless variables between variables. That would force one variable to occupy a core's cache line alone, so when other cores update other variables would not make that core reload the variable from memory. That would affect the performance in a good way and would lead us to better run-time.

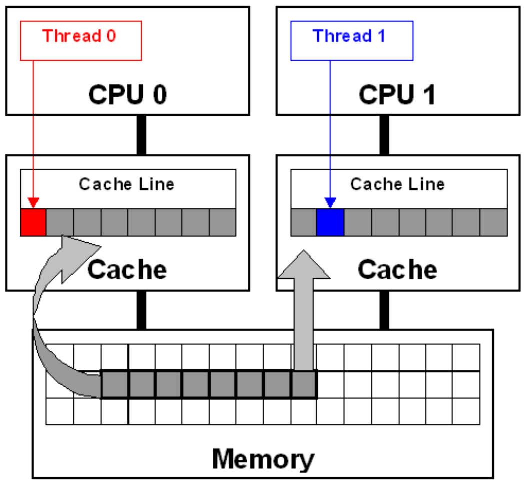

Figure 1: False Sharing

Finally, I believe that there is room for development and optimization in this experiment. And would be much beneficial to try other solutions and to get more knowledge about that.