Swetha Jayapathy
Student ID : 934041047
Instructor : Mike Bailey
CS575: Introduction to Parallel Programming
May 12, 2020

**Project #04**

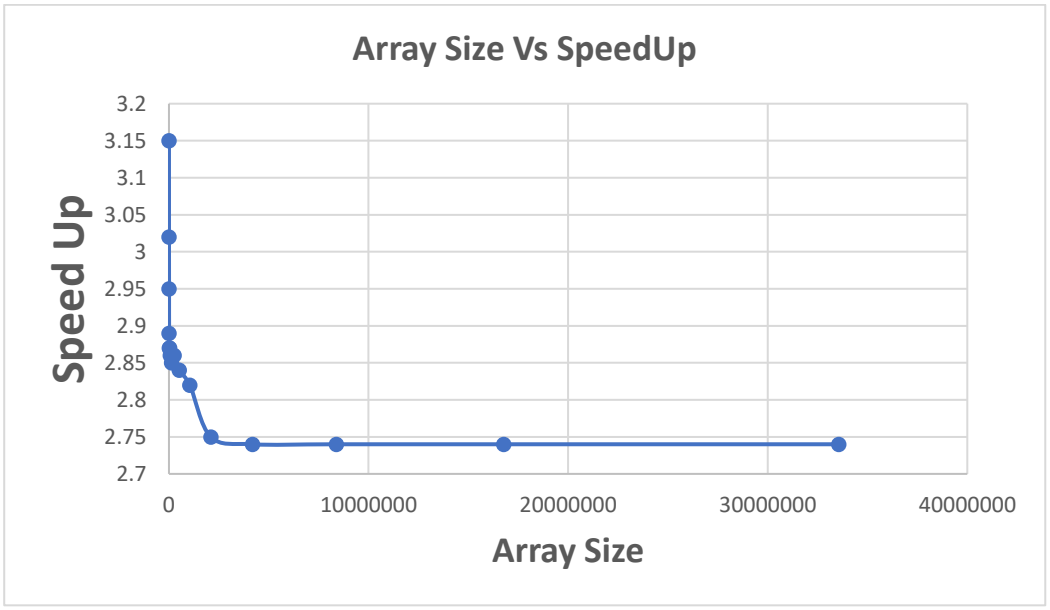
1. What machine you ran this on

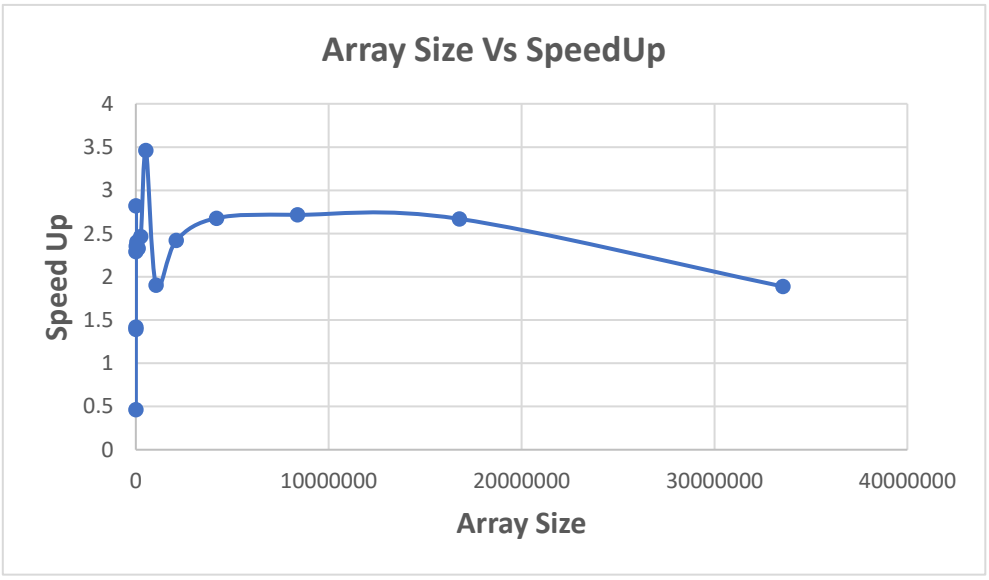
I ran my code on OSU Flip Server

2. Show the table of performances for each array size and the corresponding speedups

| SSE(MegaMults/Sec) | Non SSE(MegaMults/Sec) | ArraySize | SpeedUp |
|---|---|---|---|
| 628.65 | 199.61 | 1024 | 3.15 |
| 637.81 | 211.27 | 2048 | 3.02 |
| 641.42 | 217.5 | 4096 | 2.95 |
| 642.02 | 221.87 | 8192 | 2.89 |
| 643.07 | 223.94 | 16384 | 2.87 |
| 643.63 | 224.46 | 32768 | 2.87 |
| 642.84 | 225.03 | 65536 | 2.86 |
| 642.15 | 225.06 | 131072 | 2.85 |
| 642.12 | 224.46 | 262144 | 2.86 |
| 637.35 | 224.15 | 524288 | 2.84 |
| 626.13 | 222.29 | 1048576 | 2.82 |
| 605.56 | 220.3 | 2097152 | 2.75 |
| 602.51 | 219.99 | 4194304 | 2.74 |
| 602.49 | 220.02 | 8388608 | 2.74 |
| 601.6 | 219.66 | 16777216 | 2.74 |
| 601.63 | 219.49 | 33554432 | 2.74 |

3. Show the graph of SIMD/non-SIMD speedup versus array size (one curve only)



4. What patterns are you seeing in the speedups?

Initially, I could see that for lesser array size the SpeedUp is pretty high and then it goes low and up for moderate values of the data size. I could also see that the Speedup drops as the data size gets bigger. Overall I could see a 3x in performance.We could also see the vast difference between the performance of SIMD and Non SIMD, it can be seen that SSE is much better than Non SSE.

In the SIMD plus Multi threading graph, we could see how effective it is. There is a 16x performance with SIMD plus 4 cores. It can also be seen that it drops at an array size of 500000.

5. Are they consistent across a variety of array sizes?

We could see clearly from the first graph that the SpeedUp is not consistent across the data sizes. Initially the value of the speed up fluctuates from high to low and then drops suddenly and maintains over a period and again drops as the data size gets bigger. This can be seen from both the graphs. Especially in the SIMD plus Muticores graph, we could see the consistency for 1,2,3 threads and the $4^{th}$ thread drops at 500000.

6. Why or why not, do you think?

The Speed up drops as the data size increases, this happens due to Cache issues. Here we are using the data only once and therefore it causes an issue with temporal coherence. Temporal coherence is that a data when fetched by the processor may be required again by the processor. Temporal coherence is good for high speed performance when the same data is accessed again and again over a time span. This is because the data is present in cache and hence the processor can access it soon. But here for large data size, the processor may need to fetch data from Main memory due to Cache Miss and hence this lag in SpeedUp.

Hence as data size gets bigger, the time spent waiting on data grows proportionally. Over a large number, it flattens as the memory access dominates the processing speed.

**Extra Credit :**

**Table for SSE with Multithreading Performance and SpeedUp**

| #Thread/ArraySize | 1 | 2 | 3 | 4 | 6 |
|---|---|---|---|---|---|
| 1024 | 2.350436 | 2.183253 | 2.938984 | 2.84326 | 2.946518 |
| 2048 | 2.560313 | 3.538458 | 3.892766 | 4.288167 | 4.16607 |
| 4096 | 2.66702 | 4.448318 | 5.166481 | 6.472458 | 8.549745 |
| 8192 | 2.748085 | 4.913022 | 6.569338 | 8.25185 | 8.846775 |
| 16384 | 2.807198 | 5.301022 | 7.304785 | 9.269988 | 11.52869 |
| 32768 | 2.814059 | 5.539505 | 7.772919 | 10.28072 | 13.73405 |
| 65536 | 2.821802 | 5.593314 | 7.951272 | 10.82173 | 15.11261 |
| 131072 | 2.840977 | 5.666622 | 7.988968 | 11.03695 | 13.96233 |
| 262144 | 2.847937 | 5.767027 | 8.166005 | 11.26433 | 16.13618 |
| 524288 | 2.841534 | 5.711869 | 8.375783 | 11.61722 | 12.03912 |
| 1048576 | 2.789685 | 5.727153 | 8.314834 | 11.6415 | 16.54361 |
| 2097152 | 2.743618 | 4.855607 | 6.943337 | 10.23474 | 15.91965 |

**Graph for SIMD plus Multithreading :**

**Array Size Vs SpeedUp**