# Project #4 – Vectorized Array Multiplication and Multiplication/Reduction using SSE

Young-Joon Park

parky8@oregonstate.edu

1. What machine you ran this on

The program was run on the OSU flip servers.

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

```
[parky8@flip1 simd_test]$ bash loop.bash
        1024    N     164.55    S    1090.51    (  6.63)       N     166.32    S    1267.08    (  7.62)
        4096    N     221.39    S    1757.11    (  7.94)       N     242.86    S    1885.75    (  7.76)
       16384    N     221.52    S    1760.58    (  7.95)       N     241.72    S    1855.28    (  7.68)
       65536    N     221.18    S    1369.38    (  6.19)       N     241.15    S    1812.89    (  7.52)
      262144    N     219.55    S    1362.06    (  6.20)       N     240.20    S    1812.83    (  7.55)
     1048576    N     214.84    S     729.98    (  3.40)       N     238.70    S    1584.37    (  6.64)
     4194304    N     215.11    S     678.62    (  3.15)       N     234.93    S     902.53    (  3.84)
     8388608    N     214.45    S     708.27    (  3.30)       N     237.31    S    1069.59    (  4.51)
```

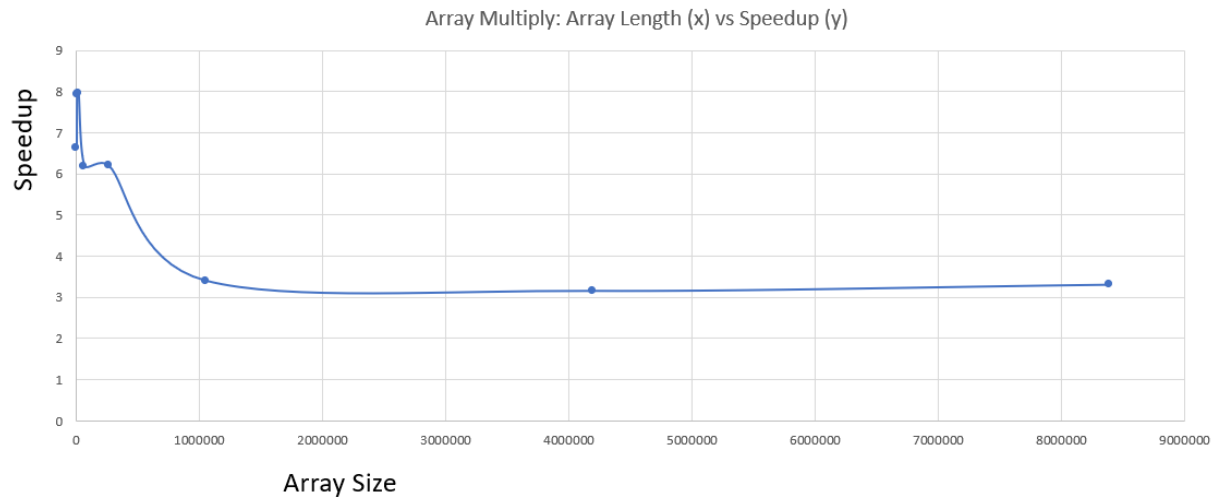
| Array Size | Speedup |
|---|---|
| 1024 | 6.63 |
| 4096 | 7.94 |
| 16384 | 7.95 |
| 65536 | 6.19 |
| 262144 | 6.2 |
| 1048576 | 3.4 |
| 4194304 | 3.15 |
| 8388608 | 3.3 |

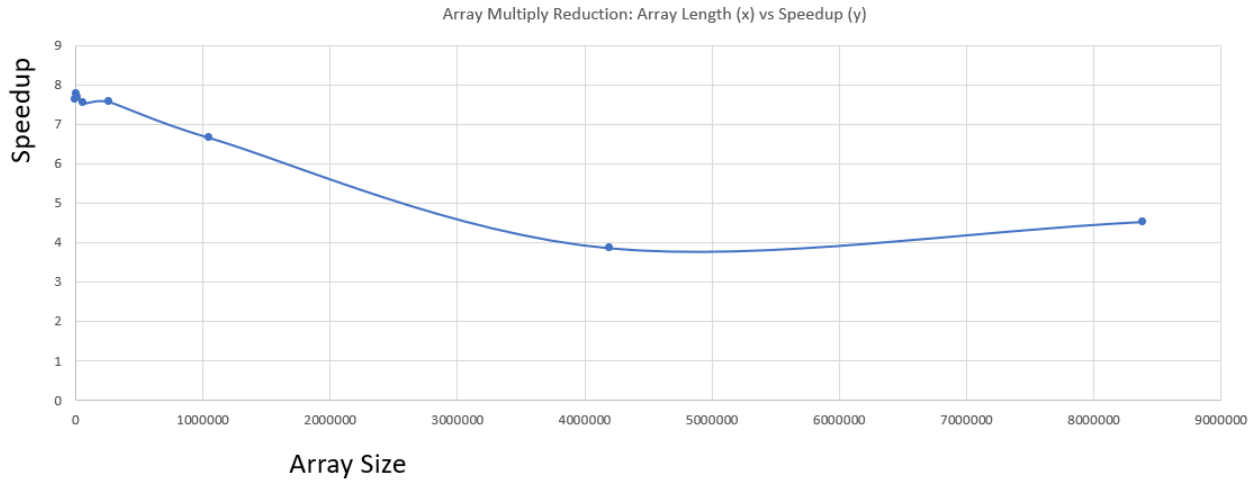
Array Mul

| Array Size | Speedup |
|---|---|
| 1024 | 7.62 |
| 4096 | 7.76 |
| 16384 | 7.68 |
| 65536 | 7.52 |
| 262144 | 7.55 |
| 1048576 | 6.64 |
| 4194304 | 3.84 |
| 8388608 | 4.51 |

Array Mul Reduction

3. Show the graphs (or graph) of SIMD/non-SIMD speedup versus array size (either one graph with two curves, or two graphs each with one curve)




4. What patterns are you seeing in the speedups?

This part is not on the graph, but if you use simd on very small arrays (1, 16, 32), the speedup is very low. Once the array size is relatively large, speedups stay at around 7 to 8, until it starts to decrease after the array gets very long.

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

Speedup is not consistent across a variety of array sizes. When array sizes are very small, speedup is very low. When array sizes are relatively long, simd performs optimally. When array sizes are very large, speedup decreases compared to “relatively large” arrays.

6. Why or why not, do you think?

When the array size is very small, the overhead required for simd instructions outweighs the benefits of simd speedups. When there are relatively larger arrays, the speed and efficiency of simd can be fully harnessed. When the arrays are very long, the speedup decrease can happen due to a few factors. One possible factor could be that the multiplies are happening so quick that cache reload needs to happen often. Under this scenario, memory bandwidth may become an issue as the speed of memory read and writes could become a bottleneck if simd is so quick that it is waiting in idle for caches to fill up after it finishes one set of cache. This does not take temporal coherence into consideration working with caches, and prefetching may help in this situation.