Amy Stockinger (stockina@oregonstate.edu)
CS475 Project 4 – Vectorized Array Multiplication and Reduction using SSE
Due: May 13, 2019

For this experiment, I used the given code and wrote a separate program04.cpp with my own non-SIMD C++ code. I ran my program on flip3 and compiled as instructed. Uptimes were around 5.5-6.0, but I have not found much lower times than that on flip recently. Below are my results:

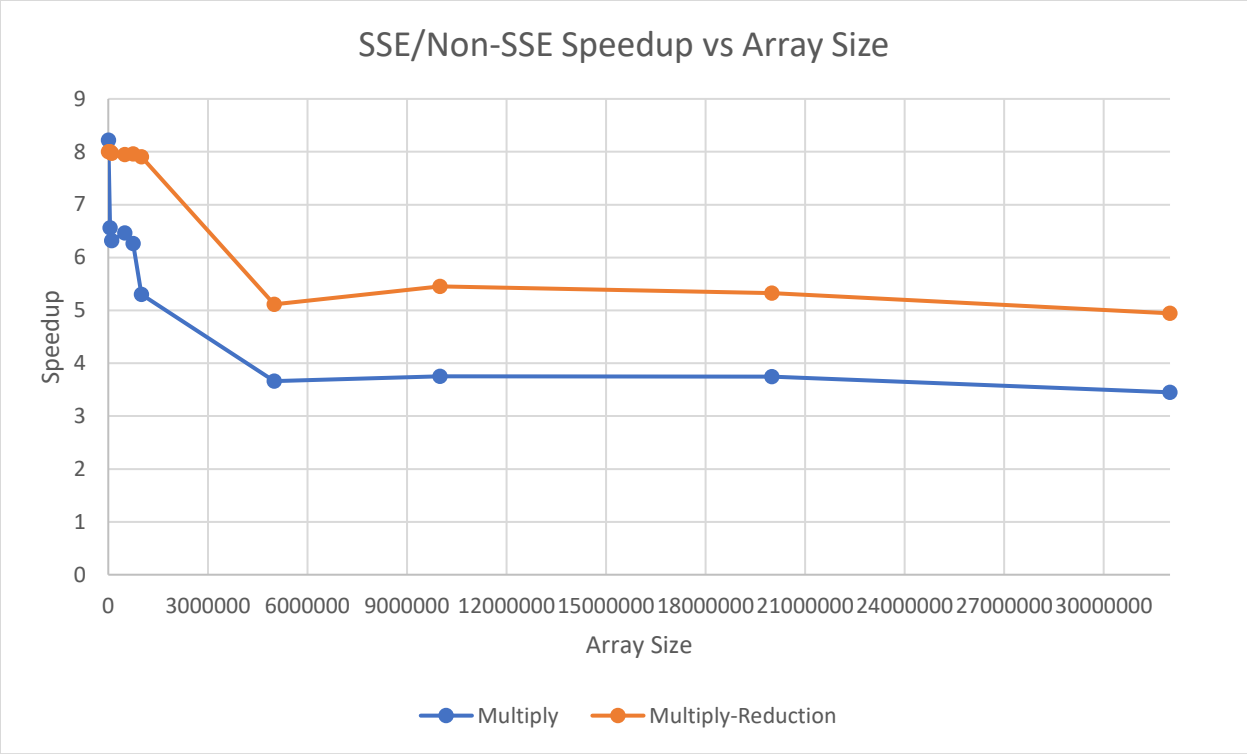
SIMD vs non-SIMD Multiply (peak and avg times in ms)

	non-SIMD Peak	non-SIMD Avg	SIMD Peak	SIMD Avg	Speedup
1000	0.004513	0.004626	0.000549	0.000555	8.220401
50000	0.226371	0.228823	0.034507	0.034892	6.560147
100000	0.453296	0.457548	0.071766	0.073036	6.316306
500000	2.273235	2.289696	0.351693	0.355177	6.463691
750000	3.423112	3.469626	0.546565	0.561021	6.262955
1000000	4.608041	4.650001	0.869416	0.917668	5.300157
5000000	23.452386	23.716925	6.403273	6.581669	3.662562
10000000	47.153426	47.986794	12.571637	12.879907	3.750779
20000000	94.023015	96.351308	25.104051	25.422555	3.745332
32000000	149.343719	153.877907	43.283978	44.314894	3.450323

SIMD vs non-SIMD Multiply & Reduction (peak and avg times in ms)

	non-SIMD Peak	non-SIMD Avg	SIMD Peak	SIMD Avg	Speedup
1000	0.004474	0.004477	0.000559	0.000561	8.003578
50000	0.221577	0.222369	0.027699	0.027882	7.999458
100000	0.443374	0.445021	0.055611	0.05596	7.972775
500000	2.225623	2.233614	0.280058	0.281622	7.947007
750000	3.338849	3.35196	0.419477	0.423589	7.959552
1000000	4.458493	4.471238	0.563903	0.570777	7.906489
5000000	22.928257	23.057736	4.484056	4.560233	5.113285
10000000	45.865931	46.203754	8.408261	8.512267	5.454865
20000000	91.873927	92.300722	17.248321	17.675587	5.326543
32000000	145.159114	146.294766	29.360961	29.898595	4.94395

The above graphs show array sizes (on the left) and program output times in milliseconds for each run (SSE or non-SSE). The speedup (on the right) is calculated using the peak times, $T_{non\text{-}sse}/T_{sse}$. The tables clearly show that larger arrays take more time (as expected), but speedup decreases with array size. The graph of the speedups vs array size is below:



We can see from the graph that speedup starts out fairly high around 8.0 and then each curve starts to level out around the 5,000,000 array size. The array-multiply-reduction is consistently faster than the array-multiply except for the very first run with the array size of 1000. However, the runs with array size 1000 are extremely close that it does not seem significant. I would expect the array-multiply-reduction to be faster given that it does not need to access 3 arrays, but only two.

Knowing that SSE SIMD is 4-floats-at-a-time, a speedup of 4.0 is significant. Both curves are leveling off yet still trending down, with the SIMD-mul hanging out just below the 4.0 speedup line. That is because the cache is unable to keep up with speed of the CPU and we are not prefetching. The multiplication-reduction again only accesses two arrays, so the speedup will be greater than 4.0 for a longer time.