Bryce Egley
CS475

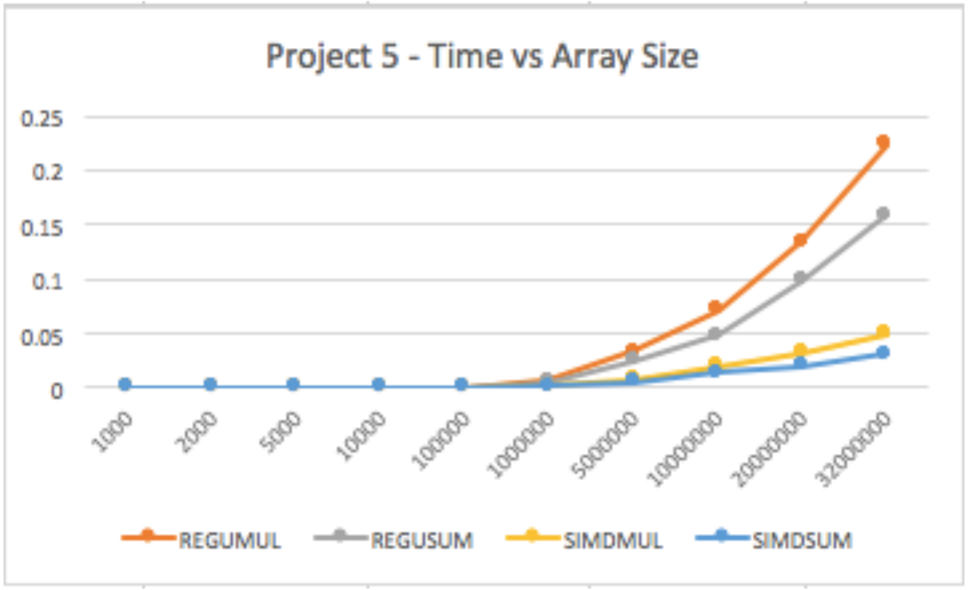
Project 5 Write Up

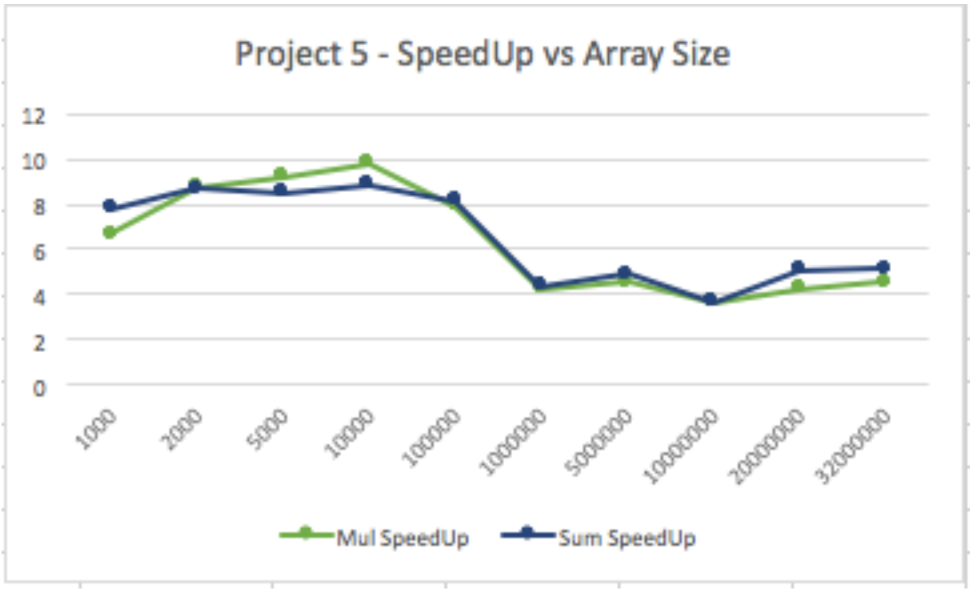
1. What machine you ran this on

   I ran this on Flip(Linux) which has 24 processors.

2. Show the table and graph

Size	REGUMUL Time	REGUSUM Time	SIMDMUL Time	SIMDSUM Time	Mul Speedup	Sum SpeedUp
1000	5.06E-06	4.81E-06	7.60E-07	6.17E-07	6.66E+00	7.80E+0
2000	1.14E-05	9.56E-06	1.30E-06	1.10E-06	8.73E+00	8.68E+0
5000	3.26E-05	2.40E-05	3.55E-06	2.81E-06	9.19E+00	8.52E+0
10000	6.16E-05	4.79E-05	6.29E-06	5.42E-06	9.79E+00	8.84E+0
100000	0.000628723	0.000481037	7.93E-05	5.89E-05	7.93E+00	8.16E+0
1000000	0.00666576	0.0049264	0.00157232	0.00112023	4.24E+00	4.40E+0
5000000	0.0330092	0.0243909	0.00726098	0.00494685	4.55E+00	4.93E+0
10000000	0.0710737	0.048542	0.0192851	0.0131609	3.69E+00	3.69E+0
20000000	0.134348	0.0987941	0.0316629	0.0194907	4.24E+00	5.07E+0
32000000	0.223578	0.158251	0.0489047	0.0307664	4.57E+00	5.14E+0





3. What patterns are you seeing in the speedups?

Both of the speed ups for multiply and sum appear to stay at about 9 times speed up and then drop down to 5 time speed up after 1 million array size.

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

   Yes, in the graph both speedups follow the same pattern.

5. Why or why not, do you think?

   I think they follow the same patter because the same concepts is being applied to both of them. The C++ code I wrote is pretty straight forward and is just one extra instruction for += to get the sum. This is why both speed up curves are the same.

6. Knowing that SSE SIMD is 4-floats-at-a-time, why could you get a speed-up of < 4.0 or > 4.0 in the array mutiplication?

Because it uses assembly language which is more efficient than writing code in C++ and having the translator to convert it to machine language. As you mentioned in the video most of the benefits of SIMD go away if you don't use assembly.

7. Knowing that SSE SIMD is 4-floats-at-a-time, why could you get a speed-up of < 4.0 or > 4.0 in the array mutiplication-reduction?

Because it is written in assembly language which is more efficient than writing code in C++ and having the translator to convert it to machine language. As you mentioned in the video most of the benefits of SIMD go away if you don’t use assembly.