CS 575 – Spring 2022

Project 4

Vectorized Array Multiplication/Reduction using SSE

Vishwas Somashekara Reddy

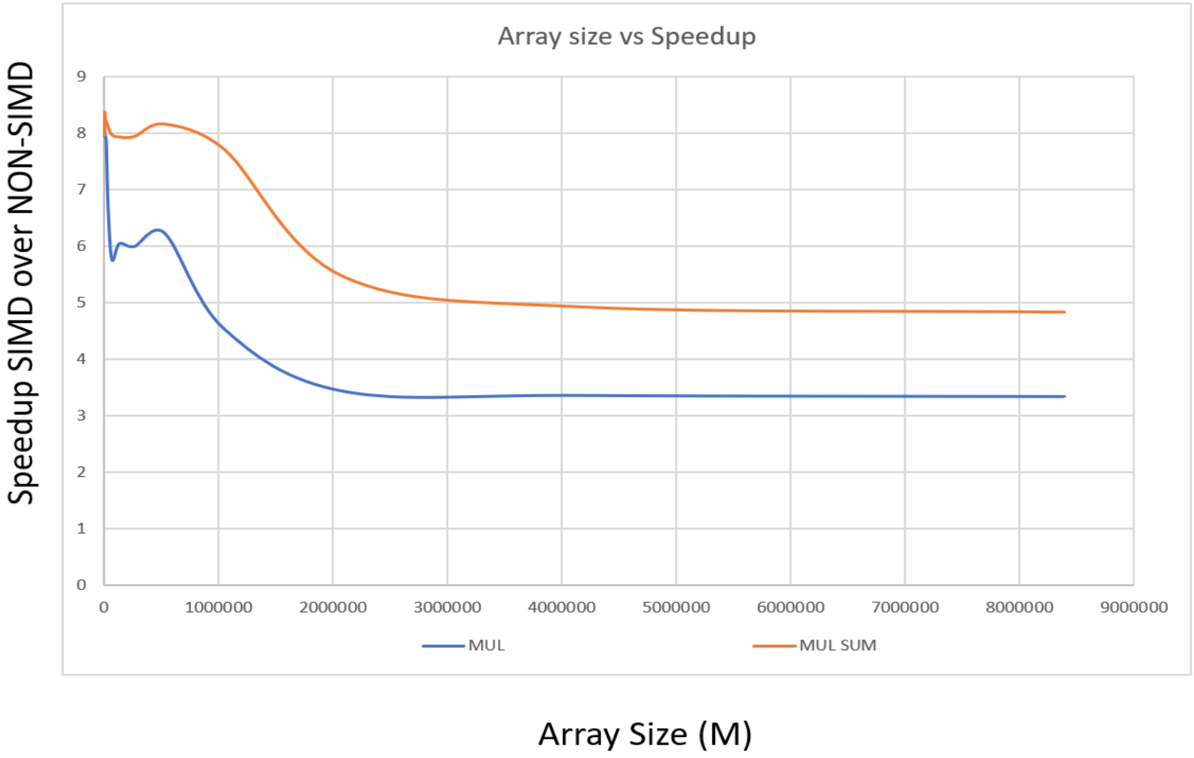
ONID - 934402783

1. What machine you ran this on?
   I ran the code on flip-1 server.

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

SIZE	NON SIMD MUL	SIMD MUL	MUL	NON SIMD MUL SUM	SIMD MUL SUM	MUL SUM
1000	221.6	1618.3	7.3	223.53	1791.06	8.01
2048	221.66	1877.1	8.47	224.28	1865.16	8.32
4096	221.78	1759.57	7.93	225.32	1885.15	8.37
8192	160.92	1134.47	7.05	167.3	1240.02	7.41
16384	221.81	1765.66	7.96	225.94	1849.67	8.19
32768	221.38	1397.34	6.31	225.79	1811.76	8.02
65536	231.71	1456.19	6.28	236.05	1888.43	8
131072	221.25	1362.98	6.16	225.8	1806.6	8
262144	225.68	1409.63	6.25	231.16	1890.88	8.18
524288	219.6	1315.57	5.99	234.54	1871.07	7.98
1048576	224.32	1143.95	5.1	233.68	1805.6	7.73
2097152	214.9	756.64	3.52	225.42	1219.56	5.41
4194304	218.31	781.54	3.58	224.67	1317.95	5.87
8388608	219.83	827.21	3.76	227.67	1334.98	5.86

3. Show the graph of SIMD/non-SIMD speedup versus array size



4. What patterns are you seeing in the speedups?
   Answer- According to the chart, array multiplication sum stays unchanged across array sizes, but array multiplication drops slightly at first but then remains constant.

5. Are they consistent across a variety of array sizes?
   Answer- Matrix multiplication was constant initially and started decreasing gradually but became constant after 4000000. Matrix Multiplication sum was constant across.

6. Why or why not, do you think?
   Answer- The Multiplication sum is showing a constant performance as the value are summed and stored cache which is being reused. Thereby producing the constant performance. However, in the multiplication, there is a dip in the performance because for every array size there is new multiplication value.