CS 475

5/12/2021

Kyle Huang

huangky@oregonstate.edu

Project 4

1. What machine you ran this on

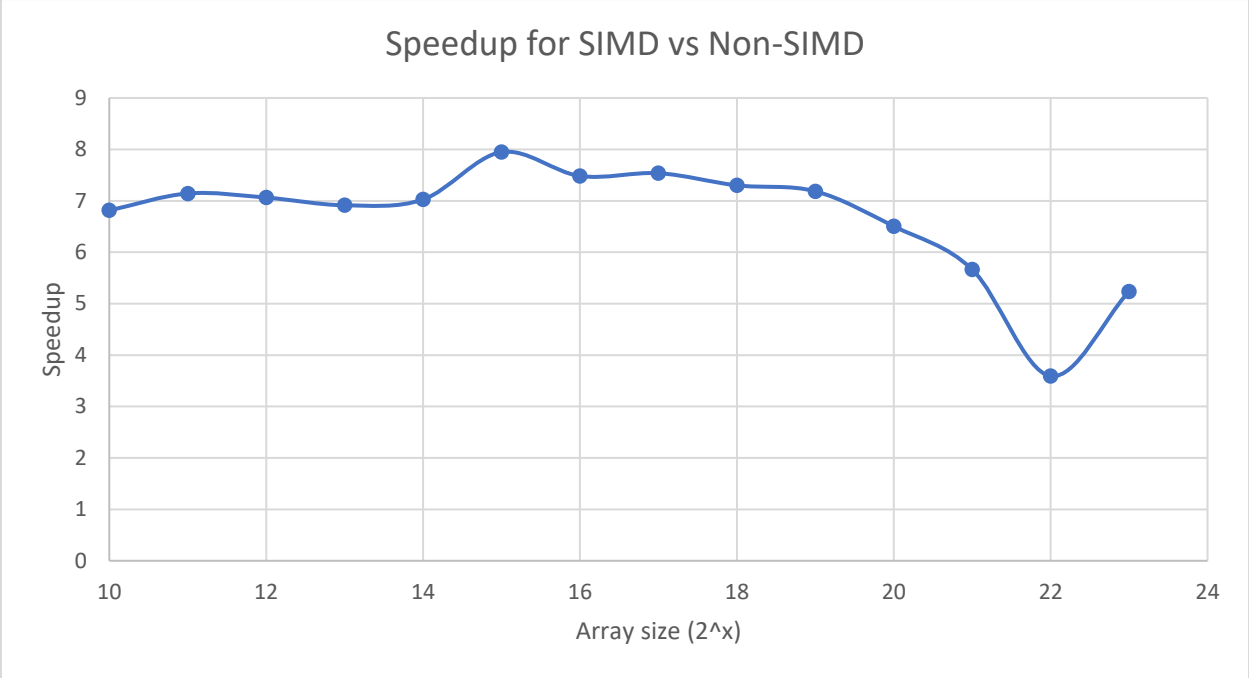
   On Flip Server.

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

Array Size (2^x)	Non-SIMD	SIMD	Speedup
10	115.592055	787.615779	6.813754
11	121.305343	866.439423	7.142632
12	121.291961	856.651054	7.062719
13	120.921792	835.971585	6.913324
14	63.252121	444.404235	7.025918
15	121.098257	962.426065	7.947481
16	120.943822	904.901293	7.481997
17	118.767817	895.151368	7.536986
18	119.92376	875.527157	7.300698
19	118.313675	850.138712	7.185465
20	117.118641	761.844767	6.504898
21	115.317373	653.231437	5.66464
22	110.833109	398.141901	3.592265
23	114.496856	599.690039	5.237611

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

Not using OMP Parallel



4. What patterns are you seeing in the speedups?

   SIMD adds a significant performance boost across all observed array sizes. There is a noticeable speedup decrease when the array size is 1MB or greater. The rebound in speedup at an array size of 4MB is interesting, I'm not sure why 3MB would run slower.

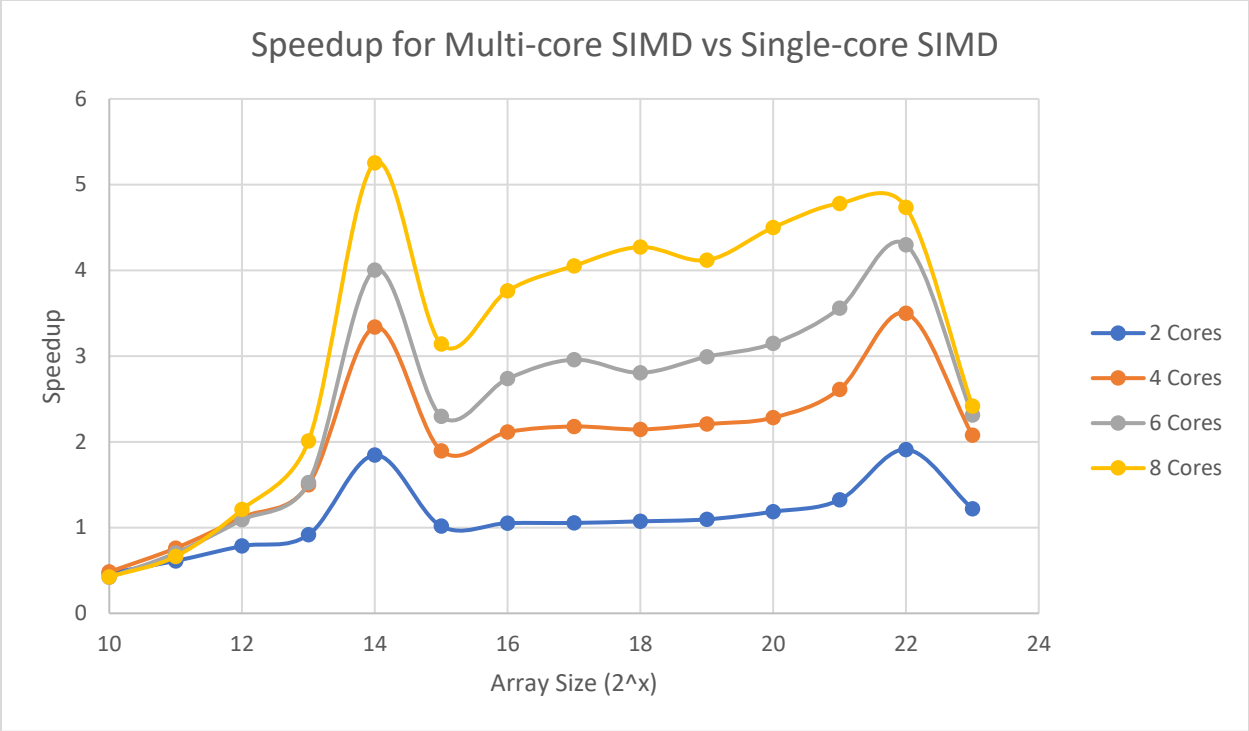
5. Are they consistent across a variety of array sizes?

   The speedups are within 1x speedup of each other up until the array size gets to 1MB. Then it seems to lose consistency.

6. Why or why not, do you think?

   I would think this is because the array is too large to cache and the program will have to go out and fetch normally out in the memory. A fix could be extending the SSE_WIDTH so that there are more vectors so that each one is doing calculations on less elements.

Extra-Credit:



This graph is showing the speedups of multi-core SIMD vs single-core SIMD. The single-core SIMD is not using omp parallel for loop; I did this because if you were to implement this in a real application you wouldn't use omp parallel and only use one core. These curves are showing speedup for different core counts as array size increases. Adding more cores increases the speedup as it would be expected. The spike at 2^{14} array size I would assume it is something to do with setup time. The large drop in speedup at array size 2^{23} would probably be due to cache size since all the speedups are converging, it would appear to be some hardware limitations.

Array Size (2^x)	Core Count	Speedup	Single-core SIMD	Multi-core SIMD
10	2	0.468770987	787.6158	369.2114
11	2	0.611271677	866.4394	529.6299
12	2	0.784775298	856.6511	672.2786
13	2	0.916231278	835.9716	765.9433
14	2	1.84584538	444.4042	820.3015
15	2	1.016120963	962.4261	977.9413
16	2	1.051916782	904.9013	951.8809
17	2	1.05392216	895.1514	943.4199
18	2	1.072898867	875.5272	939.3521
19	2	1.096007812	850.1387	931.7587
20	2	1.185358552	761.8448	903.0592

21	2	1.322984094	653.2314	864.2148
22	2	1.910524012	398.1419	760.6597
23	2	1.220125175	599.69	731.6969
10	4	0.483379502	787.6158	380.7173
11	4	0.76033553	866.4394	658.7847
12	4	1.118518519	856.6511	958.1801
13	4	1.4988604	835.9716	1253.005
14	4	3.338336987	444.4042	1483.571
15	4	1.896555303	962.4261	1825.294
16	4	2.114301251	904.9013	1913.234
17	4	2.179038695	895.1514	1950.569
18	4	2.144681191	875.5272	1877.727
19	4	2.206948269	850.1387	1876.212
20	4	2.282148448	761.8448	1738.643
21	4	2.612731494	653.2314	1706.718
22	4	3.500647419	398.1419	1393.754
23	4	2.07727677	599.69	1245.722
10	6	0.419471154	787.6158	330.3821
11	6	0.702657807	866.4394	608.8104
12	6	1.091875798	856.6511	935.3566
13	6	1.523602665	835.9716	1273.689
14	6	4.003438511	444.4042	1779.145
15	6	2.297511313	962.4261	2211.185
16	6	2.736627253	904.9013	2476.378
17	6	2.958637561	895.1514	2648.428
18	6	2.805634098	875.5272	2456.409
19	6	2.992471215	850.1387	2544.016
20	6	3.147584138	761.8448	2397.971
21	6	3.558618825	653.2314	2324.602
22	6	4.299371789	398.1419	1711.76
23	6	2.313688537	599.69	1387.496
10	8	0.424832624	787.6158	334.6049
11	8	0.663355985	866.4394	574.7578
12	8	1.211420481	856.6511	1037.765
13	8	2.008781979	835.9716	1679.285
14	8	5.254313776	444.4042	2335.039
15	8	3.139642735	962.4261	3021.674
16	8	3.762167394	904.9013	3404.39
17	8	4.05336702	895.1514	3628.377
18	8	4.272886762	875.5272	3741.028
19	8	4.119199285	850.1387	3501.891
20	8	4.499689435	761.8448	3428.065
21	8	4.78130791	653.2314	3123.301

22	8	4.735954845	398.1419	1885.582
23	8	2.416501916	599.69	1449.152