IN4330 Oblig 1 repport

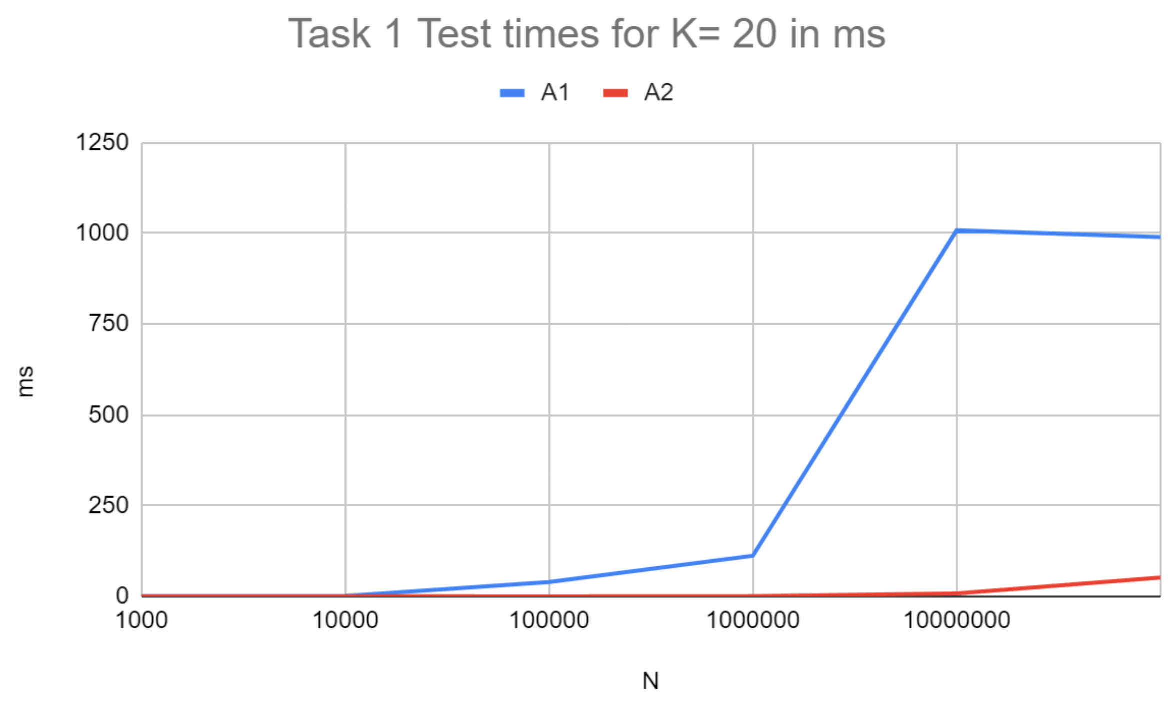
Computing Specification used for testing

Processor is an AMD Ryzen 5 5625U, 2,30 GHZ with 6 cores and 12 logical processors

Task 1

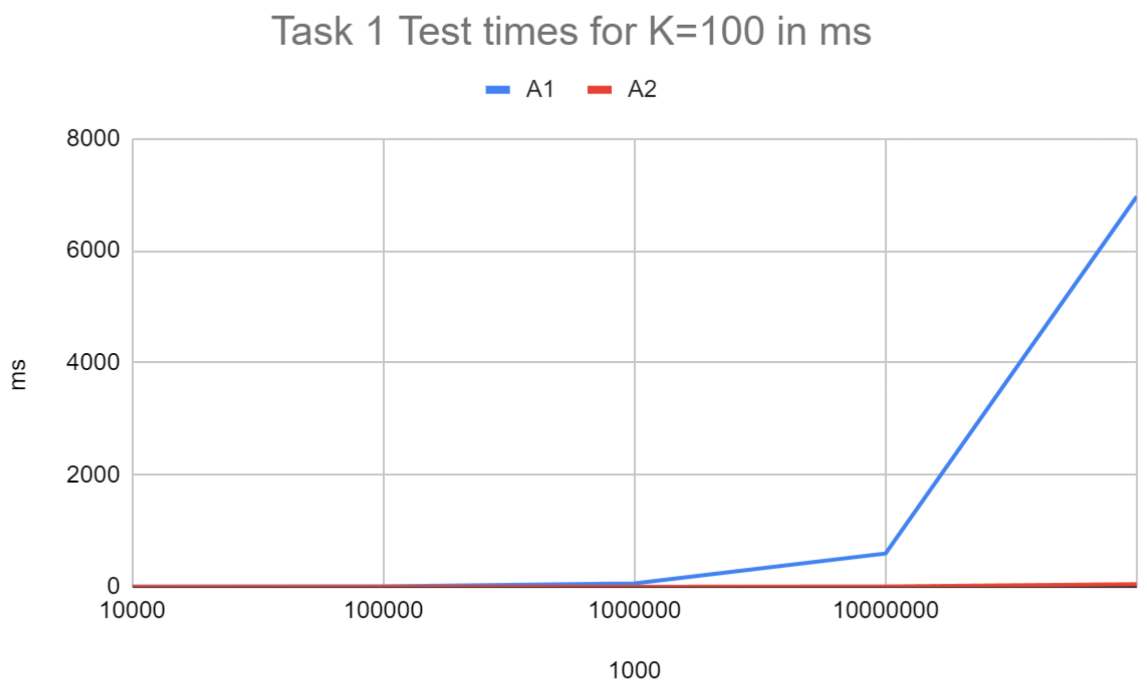
The test result is from a single run off the program that runs both algorithms 7 times and captures the median of all 7 runs. As we can see from the test result of K, the A2 algorithm is better for every value of N. This is due to the complexity of each sorting method.

Task 1 Test times for K= 20 in ms			
N	A1	A2	Speedup
1000	0,5483	0,1817	6.835
10000	1,2838	0,3977	3.99873
100000	39,7402	0,1627	122.76861
1000000	111,79	0,6049	190.0727
1000 0000	1008,2106	8,1389	120.945
100000000	989,9728	51,9753	168.68402



For K= 100 the same pattern of A2 being significantly better than A1 follows. Furthermore the increase in K from 20 to 1000 also significantly decreased A1 performance, meanwhile A2 saw mostly the same result across multiple runs for both K 20 and 100.

Task 1 Test times for K=100 in ms			
N	A1	A2	Speed up
1000	0.4518	0.178	2.115
10000	1.0268	0.5675	1.726
100000	8.7599	0.1705	37.316
1000000	60.3874	0.4884	122.947
1000 0000	595.6875	5.4662	106.7994
100000000	6966.7103	47.2557	149.551

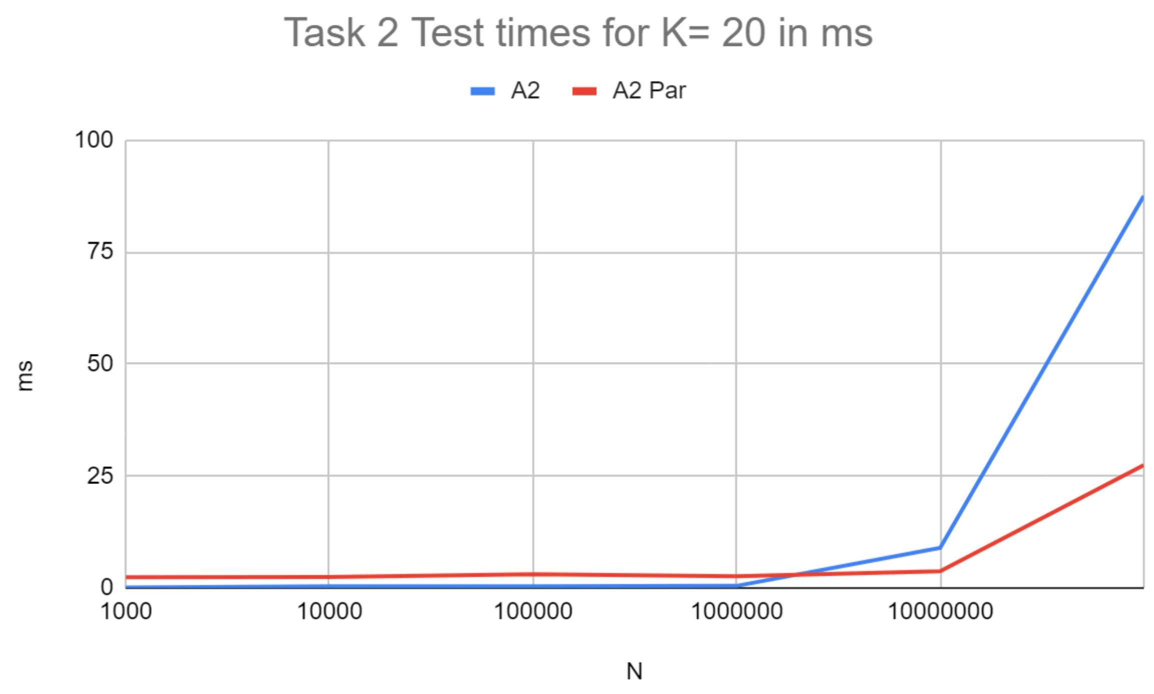


Task2

The sequential A2 has better results in most of the tests in the lower end of N large arrays for K 20. By that a significant amount too, with the parallel solution only reaching speeds between 10%-30% of the sequential one. When we approach larger arrays above a million we immediately see speedup, with arrays even larger gaining even more speed up. The low speed up in lower numbers of N is probably due to the way threads are divided. During testing there was a pattern of lower numbers of N preferring less threads, probably due to the overhead of using as many cores as possible. But for larger arrays it is necessary to divide the work into that many threads in order to gain better performance.

Task 2 Test times for K= 20 in ms			
N	A2	A2 Paralelirized	Speedup
1000	0.0907	2.3798	0.0345
10000	0.3648	2.4642	0.1453
100000	0.391	3.039	0.1736
1000000	0.4593	2.6027	0.2023
1000 0000	8.9507	3.7341	2.2918

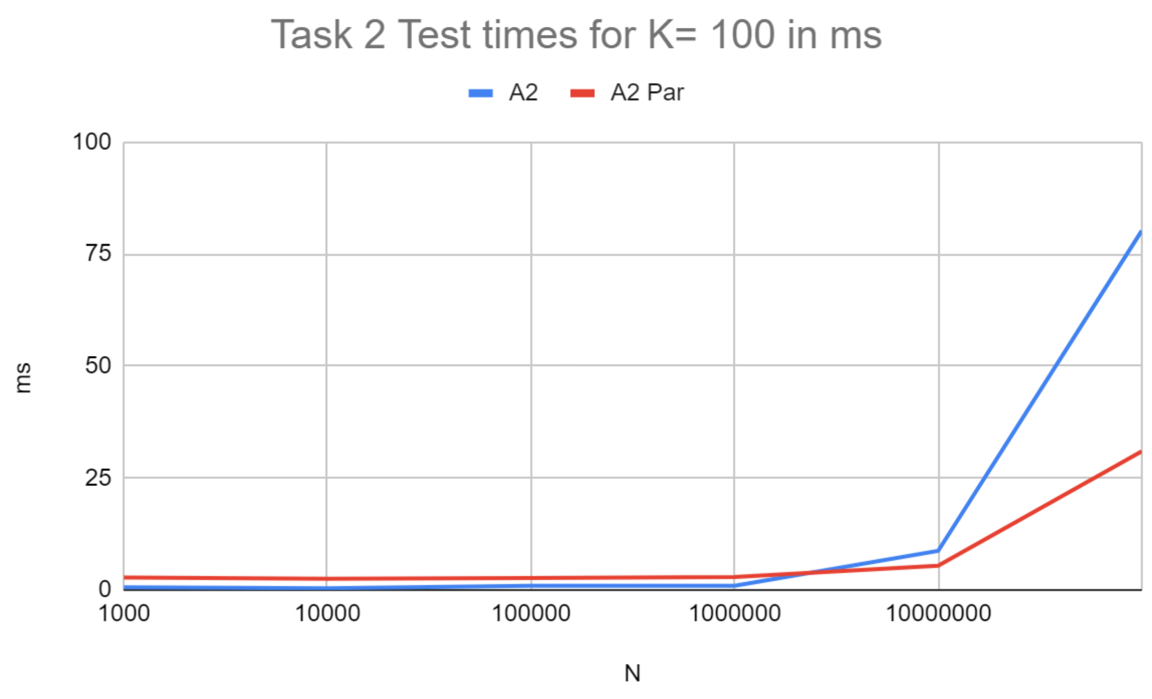
100000000	77.5322	27.4285	3.1792

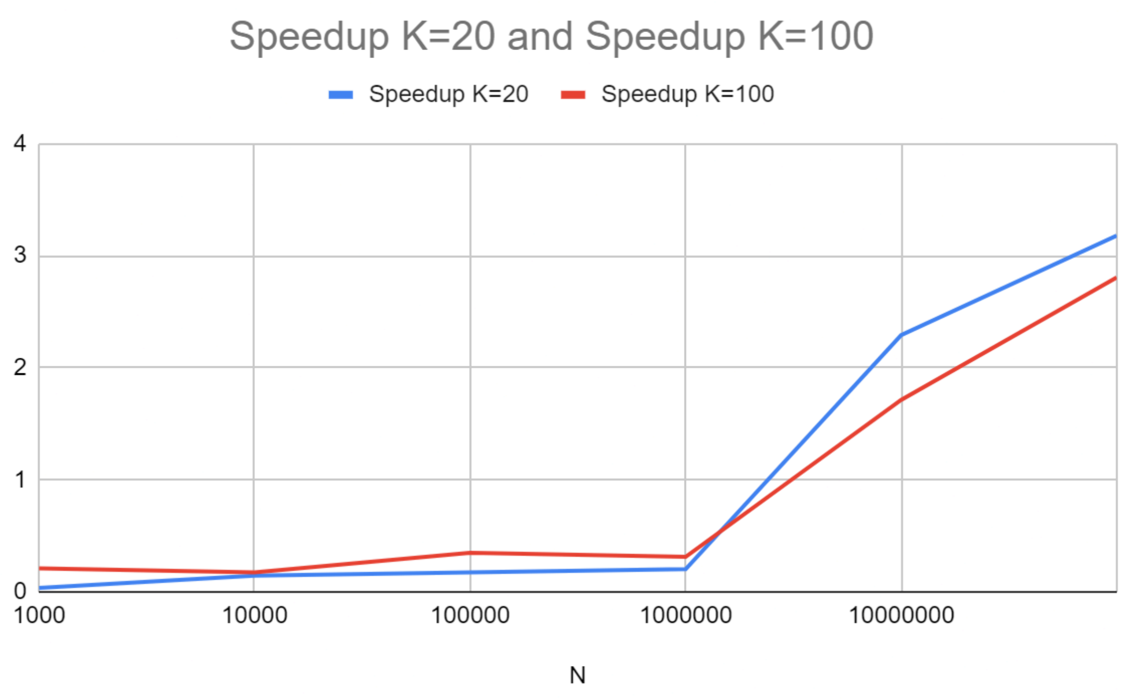


K 100 gave very similar results to K 20, although with slightly worse performance probably due to higher workload of sorting 100 elements instead of just 20.

Task 2 Test times for K= 100 in ms			
N	A2	A2 Paralelirized	Speedup
1000	0.5664	2.7746	0.20983
10000	0.3739	2.4858	0.1747
100000	0.9225	2.6761	0.3487

1000000	0.9084	2.9073	0.3124
10000000	8.7182	5.3976	1.7133
100000000	80.2027	30.9741	2.8064





As we can see both K 20 and 1000 see speed up with arrays larger than 10 million.
Furthermore K 20 consistently sees better speed up after 10 million. I
Added two more tables showing results for all algorithms just for easy comparison.

Task 2 Test times All algs for K= 20 in ms			
N	A1	A2	A2 Paralelirized
1000	0,5483	0.0907	2.3798
10000	1,2838	0.3648	2.4642
100000	39,7402	0.391	3.039
1000000	111,79	0.4593	2.6027
1000 0000	1008,2106	8.9507	3.7341
100000000	989,9728	87.5322	27.4285

Task 1 and 2 Test times All algorithms for K= 100 in ms			
N	A1	A2	A2 Paralelirized
1000	0.4518	0.5664	2.7746
10000	1.0268	0.3739	2.4858
100000	8.7599	0.9225	2.6761
1000000	60.3874	0.9084	2.9073
1000 0000	595.6875	8.7182	5.3976
100000000	6966.7103	80.2027	30.9741