Parallel Search and Sort

Philip Nelson

2017 February 11

Hypothesis

Running quick sort and linear search in parallel should be faster than the serial versions because the work is executed simultaneously therefore terminating faster.

Process

I wrote a parallel version of quick sort and linear search and tested them using a thread pool with 2-8 threads on data sets of 100-1E6. I then tested std::sort and std::find from the standard template library on the same data sets and compared them to the parallel versions. The resulting data can be found in figure 1

Findings

The results were surprising, the parallel versions performed much worse in comparison to their serial competitors. However, parallel sort was better compared to itself with larger data sets. figure **??**.

Conclusion

I believe that the slower times were a result of large amounts of overheads required to spin up threads, lock mutexes and make extra function calls. std::sort and std::find require little to no overhead before executing therefore they can begin working on the given task long before the parallel versions being.

Data

Parallel Search					Parallel Quick Sort				
Threads	Average	Standard Dev	Speed Up	Efficency	Threads	Average	Standard Dev	Speed Up	Efficency
size: 10					size: 10				
1	0.00109883	9.75E-05	1	1	1	0.00115783	0.00036966	1	1
2	0.0838852	0.048311	0.013099212	0.006549606	2	0.108691	0.0334647	0.010652492	0.005326246
3	0.0948037	0.0119409	0.011590581	0.003863527	3	0.485558	2.00208	0.002384535	0.000794845
4	0.116456	0.0303609	0.009435581	0.002358895	4	0.131474	0.0184649	0.008806532	0.002201633
5	0.134663	0.0404128	0.008159851	0.00163197	5	0.160914	0.0659779	0.007195334	0.001439067
6	0.228933	0.032371	0.004799789	0.000799965	6	0.208442	0.0304715	0.005554687	0.000925781
7	0.338482	0.0934163	0.003246347	0.000463764	7	0.295069	0.0555615	0.00392393	0.000560561
8	0.390265	0.0369591	0.0028156	0.00035195	8	0.439591	0.258808	0.00263388	0.000329235
size: 100					size: 100				
1	0.00113543	5.92E-05	1	1	1	0.00666657	0.000464939	1	1
2	0.0884459	0.067039	0.012837565	0.006418783	2	0.490875	0.0918289	0.013580993	0.006790497
3	0.0976943	0.026482	0.011622275	0.003874092	3	1.4829	4.11087	0.00449563	0.001498543
4	0.113652	0.0251875	0.009990409	0.002497602	4	0.528837	0.0640797	0.012606096	0.003151524
5	0.142026	0.0631873	0.007994522	0.001598904	5	0.62413	0.144586	0.01068138	0.002136276
6	0.214697	0.0325827	0.005288523	0.00088142	6	1.41122	3.23537	0.004723976	0.000787329
7	0.316053	0.119454	0.00359253	0.000513219	7	2.81782	5.57412	0.002365861	0.00033798
8	0.387871	0.146035	0.002927339	0.000365917	8	2.6501	4.05272	0.002515592	0.000314449
size: 1000					size: 1000				
1	0.0011511	4.72E-05	1	1	1	0.0831023	0.00562988	1	1
2	0.0794755	0.0233377	0.014483709	0.007241854	2	4.00187	0.0599992	0.020765867	0.010382933
3	0.10662	0.052978	0.010796286	0.003598762	3	3.99073	0.409395	0.020823834	0.006941278
4	0.122146	0.0507727	0.009423968	0.002355992	4	4.12348	0.457998	0.020153438	0.00503836
5	0.130677	0.0320689	0.008808742	0.001761748	5	4.42435	0.397332	0.01878294	0.003756588
6	0.225852	0.0792761	0.0050967	0.00084945	6	5.67854	2.17408	0.014634448	0.002439075
7	0.293173	0.035291	0.003926351	0.000560907	7	7.07625	5.29651	0.011743833	0.00167769
8	0.361657	0.038773	0.00318285	0.000397856	8	8.26828	5.23118	0.010050736	0.001256342
size: 10000					size: 10000				
1	0.000909733	0.000363517	1	1	1	1.06443	0.0108955	1	1
2	0.137196	0.0382624	0.0066309	0.00331545	2	40.0676	0.615596	0.026565854	0.013282927
3	0.146305	0.0452038	0.006218058	0.002072686	3	38.5397	0.178644	0.027619053	0.009206351
4	0.168274	0.0223914	0.00540626	0.001351565	4	40.5261	1.06493	0.026265296	0.006566324
5	0.179777	0.0361006	0.005060341	0.001012068	5	41.4663	0.885705	0.025669761	0.005133952
6	0.277753	0.0370767	0.003275331	0.000545888	6	42.9187	2.08228	0.024801077	0.004133513
7	0.35356	0.0282543	0.002573065	0.000367581	7	43.0287	2.11797	0.024737675	0.003533954
8	0.4572	0.0469362	0.001989792	0.000248724	8	44.9917	6.04658	0.023658364	0.002957295
size: 100000					size: 100000				
1	0.00181603	0.000179806	1	1	1	13.2489	0.0699823	1	1
2	0.861477	0.752543	0.002108042	0.001054021	2	439.235	2.2118	0.03016358	0.01508179
3	0.674031	0.409022	0.002694283	0.000898094	3	425.102	2.50317	0.031166402	0.010388801
4	0.701187	0.270446	0.002589937	0.000647484	4	425.335	6.34818	0.031149329	0.007787332
5	0.653432	0.223635	0.002779218	0.000555844	5	455.39	18.0211	0.029093524	0.005818705
6	0.644309	0.142543	0.00281857	0.000469762	6	473.004	8.70292	0.028010123	0.004668354
7	0.824715	0.472154	0.002202009	0.000314573	7	476.345	19.6875	0.027813664	0.003973381
8	0.778974	0.149995	0.00233131	0.000291414	8	491.035	23.6542	0.02698158	0.003372697
size: 1000000					size: 1000000				
1	0.00218417	0.00028675	1	1	1	160.293	1.07272	1	1
2	8.00296	8.7056	0.00027292	0.00013646	2	4702.48	44.0468	0.034086907	0.017043454
3	11.7047	12.1585	0.000186606	6.22021E-05	3	4888.36	189.288	0.032790752	0.010930251
4	7.56533	6.91935	0.000288708	7.2177E-05	4	6034.33	134.34	0.026563512	0.006640878
5	9.1333	7.84554	0.000239144	4.78287E-05	5	6499.34	71.2609	0.024662966	0.004932593
6	10.5738	8.02952	0.000206564	3.44274E-05	6	6816.53	65.5735	0.023515337	0.003919223
7	12.807	11.2234	0.000170545	2.43636E-05	7	6647.19	181.389	0.0241144	0.003444914
8	11.6175	10.6694	0.000188007	2.35009E-05	8	6599.45	200.242	0.024288842	0.003036105

Figure 1: Data

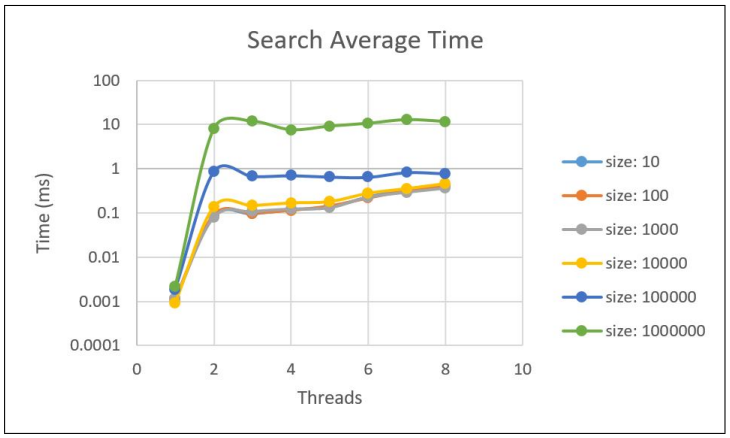

Figure 2: Average Search Time

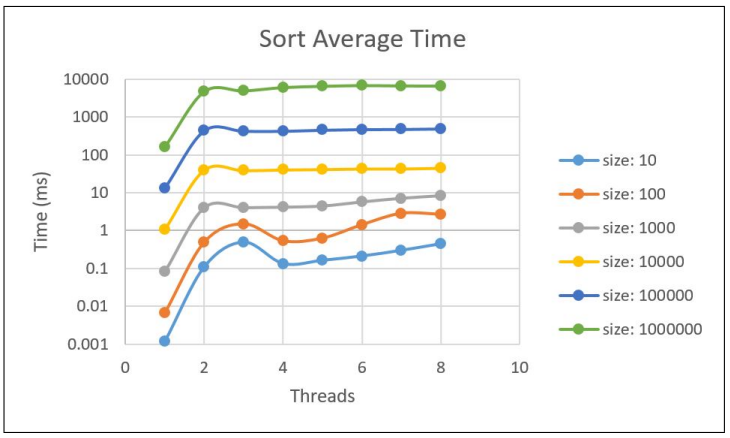

Figure 3: Average Sort Time

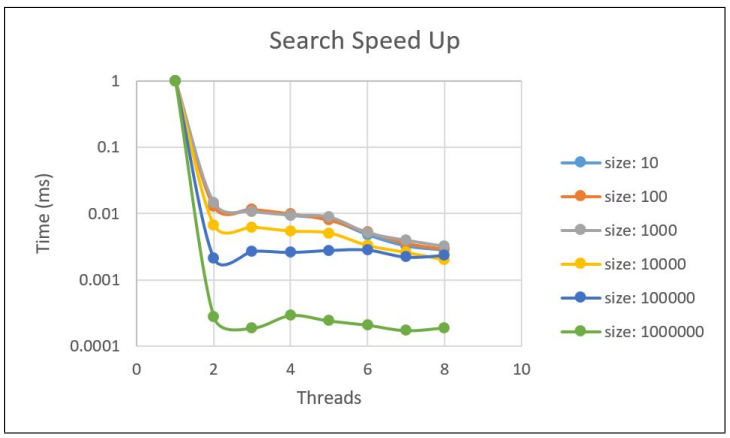

Figure 4: Search Speed Up

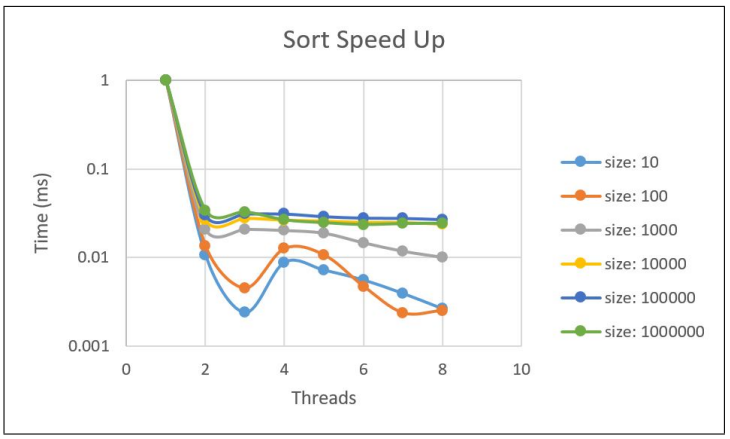

Figure 5: Sort Speedup

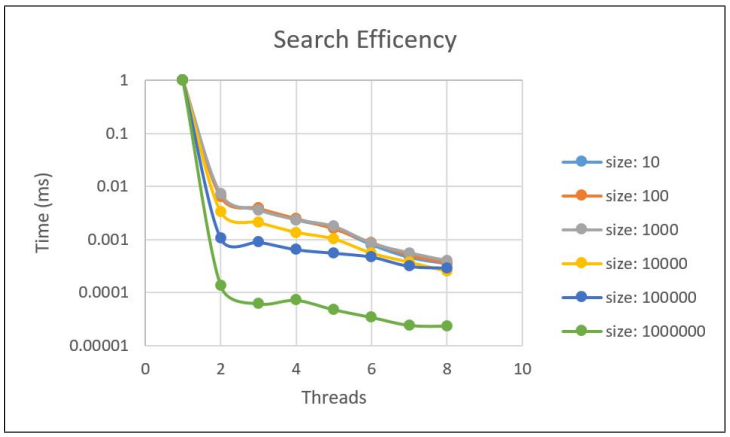

Figure 6: Search Efficiency

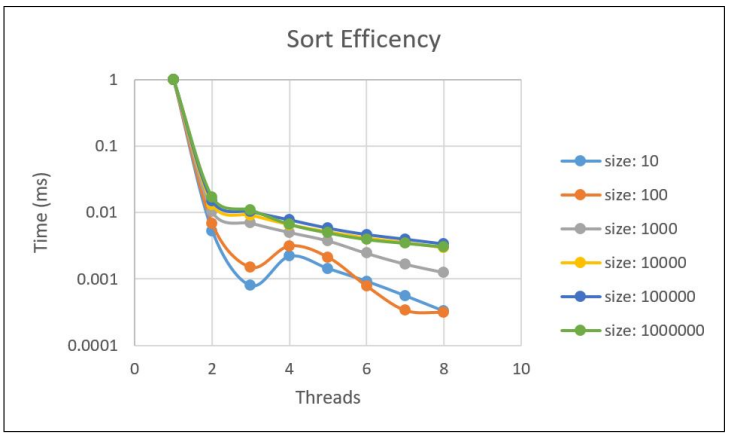

Figure 7: Sort Efficiency