Assignment 5 Parallel Sorting

By:
Amretasre Rengarajan Thiruvengadam,
NUID: 002762670.

Observations:

Firstly, when the cutoff is at 1000 or 10000 for an array length of 1000000 it sorts faster compared to the normal sorting technique. This is evident from the below table. To sort an array with cutoff of 1000 takes only 51.4 whereas when we use normal sorting technique it takes 89.4 ms.

Parallel Sorting	Time (ms)	Normal sorting	Time(ms)
0.001	**51.4**	**0.02**	**89.4**
0.002	31.5	**0.04**	73.6
0.003	30.3	**0.06**	74.8
0.004	30.2	**0.08**	73.2
0.005	29.8	**0.1**	72.9
0.006	30.3	**0.12**	72.9
0.007	30.4	**0.14**	74.1
0.008	30.5	**0.16**	75.1
0.009	30.7	**0.18**	73.8
0.01	30.8	**0.2**	74.2
0.011	31	**0.22**	74.9
0.012	30.4	**0.24**	74.8
0.013	31.6	**0.26**	74.9
0.014	31.3	**0.28**	75.1

When we try sorting an array of length 2000 with parallel sorting technique it takes 2.6 ms whereas normal sorting technique takes only 0.4 ms. The table below is generated for array length of 2000.

Parallel Sort/Array Length	Time (ms)	**Normal Sort**	Time (ms)
0.01	1.5	**0.01**	0.1
0.015	1	**0.015**	0.1
0.02	0.8	**0.02**	0.2
0.025	0.7	**0.025**	0.1
0.03	0.8	**0.03**	0.1
0.035	0.8	**0.035**	0.2
0.04	0.8	**0.04**	0.1
0.045	0.8	**0.045**	0.1
0.05	0.6	**0.05**	0.1
0.055	0.8	**0.055**	0.1
0.06	0.7	**0.06**	0.1
0.065	0.4	**0.065**	0.2

0.07	0.4	**0.07**	0.1

Hence, when the array length is less it is better to use normal sorting algorithm compared to parallel sorting.

Secondly, when considering the thread count, threads with power of 2 implement sorting faster compared to other thread counts. When running the Parallel sort for an array length of 1000000 with cutoff of 1000, the sorting works faster when the thread is 4 compared to 3. So in multi thread algorithms it is better to use thread counts in the power of 2.

1000 with 3 threads		1000 with 4	
0.001	**50.6**	**0.001**	**50.7**
0.002	31.1	**0.002**	29.5
0.003	31.3	**0.003**	28.7
0.004	31.5	**0.004**	29.3
0.005	30.6	**0.005**	28.9
0.006	32	**0.006**	31.2
0.007	31.2	**0.007**	29.5
0.008	30.6	**0.008**	29
0.009	31.8	**0.009**	29.4
0.01	31.9	**0.01**	29.9
0.011	32	**0.011**	29.1
0.012	31.5	**0.012**	28.7
0.013	31.6	**0.013**	29.1
0.014	31.4	**0.014**	29.1
0.015	31.4	**0.015**	29.2
0.016	31.4	**0.016**	29.7
0.017	31.3	**0.017**	30.2

Now when comparing different threads, the higher the power of 2 the lower the array takes to sort.

1000 with 2 thread		1000 with 4 thread		1000 with 8 thread		1000 with 16 thread		1000 with 32 thread	
0.001	**51.6**	**0.001**	**50.7**	**0.001**	**50**	**0.001**	**48.9**	**0.001**	**55.8**
0.002	34.2	**0.002**	29.5	**0.002**	33.1	**0.002**	30.2	**0.002**	26.6
0.003	35	**0.003**	28.7	**0.003**	29.8	**0.003**	29.1	**0.003**	27.7
0.004	34.6	**0.004**	29.3	**0.004**	27.4	**0.004**	28.7	**0.004**	26.3
0.005	35.1	**0.005**	28.9	**0.005**	27.3	**0.005**	28.7	**0.005**	26.4
0.006	34.5	**0.006**	31.2	**0.006**	27.3	**0.006**	28.3	**0.006**	26.4
0.007	34.2	**0.007**	29.5	**0.007**	27.3	**0.007**	28.5	**0.007**	26.4
0.008	34.5	**0.008**	29	**0.008**	27.9	**0.008**	27.8	**0.008**	25.9
0.009	36.5	**0.009**	29.4	**0.009**	30.3	**0.009**	29.2	**0.009**	27
0.01	35	**0.01**	29.9	**0.01**	31.4	**0.01**	28.4	**0.01**	25.9
0.011	35.2	**0.011**	29.1	**0.011**	29.7	**0.011**	28.2	**0.011**	25.9
0.012	35.3	**0.012**	28.7	**0.012**	27.2	**0.012**	28.4	**0.012**	26.1
0.013	36.6	**0.013**	29.1	**0.013**	26.3	**0.013**	26.9	**0.013**	25.8
0.014	36.4	**0.014**	29.1	**0.014**	25.9	**0.014**	31.5	**0.014**	26.1

The sorting is faster when the thread count is 32 compared to 16 or 8.

For considering the cutoff, the ratio of cutoff/array length should be less than or equal to 0.01. Generally, for an array length of 1000000, 2000000 and 10000000 for a cutoff of 1000 to 10000 sorted the array in less time compared to that of a cutoff more than 10000. For an array length of 2000000, the below table shows the values generated for cutoffs 10000, 100000, 200000. For a cutoff of 10000 and 20000 it sorts using normal sorting algorithm as the loop includes the array length 2000000 while incrementing, when the cutoff Is equal to the array length it should sort using normal algorithm.

10000 with 64 thread		100000		200000	
0.005	**80.6**	**0.05**	**177.8**	**0.1**	**176.9**
0.01	58.8	**0.1**	161.5	**0.2**	168.5
0.015	58.2	**0.15**	167.6	**0.3**	166.1
0.02	59.6	**0.2**	179.6	**0.4**	158.6
0.025	59.3	**0.25**	171.8	**0.5**	152.7
0.03	58.9	**0.3**	164.2	**0.6**	155.1
0.035	58.8	**0.35**	163	**0.7**	153.6
0.04	58.9	**0.4**	163.8	**0.8**	160.8
0.045	59.5	**0.45**	174.8	**0.9**	168.3
0.05	58.7	**0.5**	180	**1**	172.1
0.055	59	**0.55**	156.6	**1.1**	169.4
0.06	59.1	**0.6**	161.8	**1.2**	167.2
0.065	59.8	**0.65**	169.1	**1.3**	175.3
0.07	58.3	**0.7**	160.7	**1.4**	171.4
0.075	58.6	**0.75**	160.7	**1.5**	169.4
0.08	58.1	**0.8**	165.9	**1.6**	174.3
0.085	59.2	**0.85**	176.3	**1.7**	162.8
0.09	58.8	**0.9**	165.8	**1.8**	158.7
0.095	58.8	**0.95**	168.7	**1.9**	162.1
0.1	58.3	**1**	164.6	**2**	158
0.105	59	**1.05**	164.2	**2.1**	159.7
0.11	60	**1.1**	168.9	**2.2**	155.8
0.115	60.1	**1.15**	169	**2.3**	154.1
0.12	58.9	**1.2**	165.8	**2.4**	157.6
0.125	58.7	**1.25**	160.1	**2.5**	159.4
0.13	58.4	**1.3**	163.8	**2.6**	158.9
0.135	59.9	**1.35**	166.2	**2.7**	156.9
0.14	59.7	**1.4**	160.5	**2.8**	161.8

Conclusion:
The best cutoff can be obtained at Array size/ Number of threads. Also, multi thread sorting works better than single thread sorting when the array size is bigger.