

Program Structures and Algorithms
Spring 2024

NAME: Ting Guo

NUID: 002834835

GITHUB LINK:<https://github.com/Nangongnuanshan/INFO6205>

Task: Assignment 6

Conclusion:

I think the best predictor of total execution time is swaps.

Evidence to support that conclusion:

MergeSort							
	Memory(bytes)	Compares	Fixes	Swaps	Copies	Per Time(mSec)	Normalized Time
10000	24710136	121504	24992889	9755	110000	3.5	4.92
20000	12467240	263008	100017231	19518	240000	8.58	5.57
40000	35335432	565993	399901957	39020	520000	15.67	4.72
QuickSortDualPivot							
	Memory	Compares	Fixes	Swaps	Copies	Per Time(mSec)	Normalized Time
10000	104931016	156005	28337977	66443	0	644.91	907.37
20000	87958648	340795	112903507	141614	0	2541.37	1648.69
40000	overflow	737540	453938379	298108	0	10248.51	3084.24
HeapSort							
	Memory	Compares	Fixes	Swaps	Copies	Per Time(mSec)	Normalized Time
10000	overflow	234372	75567121	124201	0	1235.33	1738.07
20000	overflow	510745	302561332	268405	0	4876.33	3163.46
40000	overflow	1101465	1210614525	576786	0	19259.46	5796.05

Obviously, the running time of Merge Sort is the shortest. Therefore, we can rule out copies. Merge Sort use the most copies but using the lest time.

Then, memory used is not the best predictor, because the use of 10000, 20000, 40000 of Merge Sort is similar, but the per time of these is different.

When dealing with the same number of elements, the use of compares and fixes of three sort are similar, especially the fixes of Merge Sort and Quick Sort Dual Pivot, but time is obviously different. So, compares and fixes are not the best.

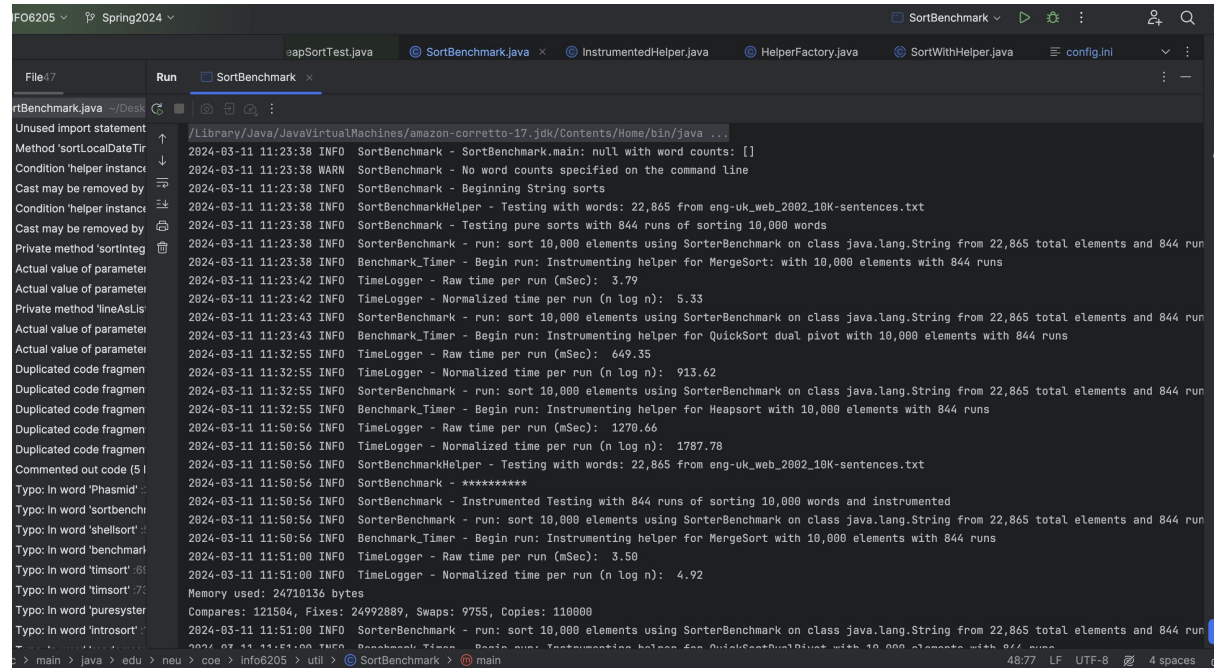
When it comes swaps, through vertical comparison and horizontal comparison, we can find that time and swaps have a linear relationship. The time of 10000 of Merge Sort is 3.5 with 9755 swaps, and the time of 20000 of Merge Sort is 8.58 with 19518 swaps, we can find that when swaps are doubled(about), the running time is almost doubled. When we compare the same elements(like 10000), we can find, the more swaps, the more time.

Therefore, I think the best predictor of total execution time is swaps.

(ps. My computer/icloud met some problem, I cannot download eng-uk_web_2002_100K-sentences, so I try some smaller number of elements, I get same conclusion)

Unit Test Screenshots:

Oncd for timing, and once for the instrument(Instrument Test)



```
F06205 Spring2024 SortBenchmark
File47 Run SortBenchmark
rtBenchmark.java ~/Desk
Unused import statement
Method 'sortLocalDateTir
Condition 'helper Instance
Cast may be removed by
Condition 'helper Instance
Cast may be removed by
Private method 'sortInteg
Actual value of parameter
Actual value of parameter
Private method 'lineAsLis
Actual value of parameter
Actual value of parameter
Duplicated code fragmen
Duplicated code fragmen
Duplicated code fragmen
Duplicated code fragmen
Duplicated code fragmen
Commented out code (5)
Type: In word 'Phasmid'
Type: In word 'sortbench
Type: In word 'shellsort'
Type: In word 'benchmark
Type: In word 'timsort'
Type: In word 'timsort'
Type: In word 'puresyster
Type: In word 'introsort'
2024-03-11 11:23:38 INFO SortBenchmark - SortBenchmark.main: null with word counts: []
2024-03-11 11:23:38 WARN SortBenchmark - No word counts specified on the command line
2024-03-11 11:23:38 INFO SortBenchmark - Beginning String sorts
2024-03-11 11:23:38 INFO SortBenchmarkHelper - Testing with words: 22,865 from eng-uk_web_2002_10K-sentences.txt
2024-03-11 11:23:38 INFO SortBenchmark - Testing pure sorts with 844 runs of sorting 10,000 words
2024-03-11 11:23:38 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 run
2024-03-11 11:23:38 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort: with 10,000 elements with 844 runs
2024-03-11 11:23:42 INFO TimeLogger - Raw time per run (mSec): 3.79
2024-03-11 11:23:42 INFO TimeLogger - Normalized time per run (n log n): 5.33
2024-03-11 11:23:43 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 run
2024-03-11 11:23:43 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 10,000 elements with 844 runs
2024-03-11 11:32:55 INFO TimeLogger - Raw time per run (mSec): 649.35
2024-03-11 11:32:55 INFO TimeLogger - Normalized time per run (n log n): 913.62
2024-03-11 11:32:55 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 run
2024-03-11 11:32:55 INFO Benchmark_Timer - Begin run: Instrumenting helper for Heapsort with 10,000 elements with 844 runs
2024-03-11 11:50:56 INFO TimeLogger - Raw time per run (mSec): 1270.66
2024-03-11 11:50:56 INFO TimeLogger - Normalized time per run (n log n): 1787.78
2024-03-11 11:50:56 INFO SortBenchmarkHelper - Testing with words: 22,865 from eng-uk_web_2002_10K-sentences.txt
2024-03-11 11:50:56 INFO SortBenchmark - *****
2024-03-11 11:50:56 INFO SortBenchmark - Instrumented Testing with 844 runs of sorting 10,000 words and instrumented
2024-03-11 11:50:56 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 run
2024-03-11 11:50:56 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with 10,000 elements with 844 runs
2024-03-11 11:51:00 INFO TimeLogger - Raw time per run (mSec): 3.50
2024-03-11 11:51:00 INFO TimeLogger - Normalized time per run (n log n): 4.92
Memory used: 24710136 bytes
Compares: 121504, Fixes: 24992889, Swaps: 9755, Copies: 110000
2024-03-11 11:51:00 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 run
2024-03-11 11:51:00 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 10,000 elements with 844 runs
48/77 LF UTF-8 4 spaces
```