CSS 342 - Lab 4 Performance Report

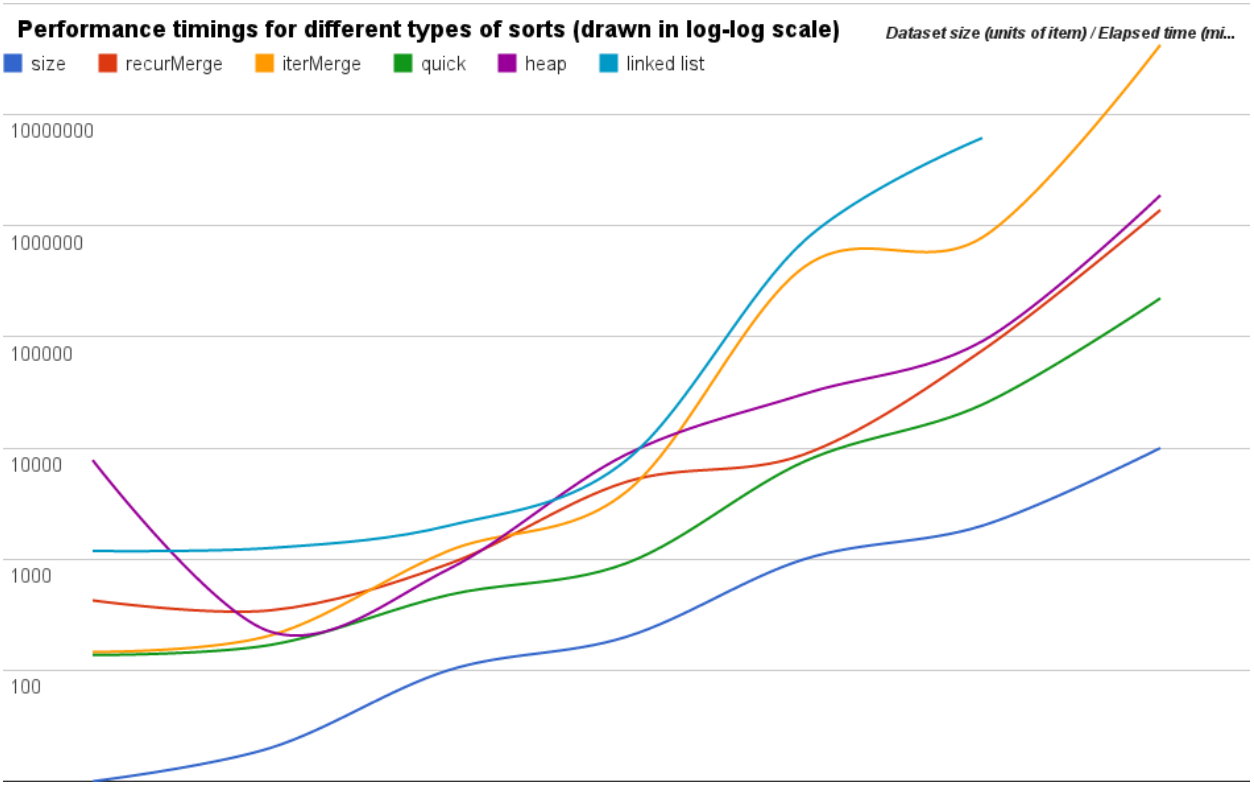
Comparing running times of different sorting algorithms

James Murphree
Vu Dinh

Feb 23 2014

Analysis and Rationale

Figure 1: Performance timings for different types of sorting algorithms.



Performance data sheet for sizes 10, 20, 100, 200, 1000, 2000, and 10000, 100000, and 1000000; 34 iterations each size.

Results: All sorts (except for linked list merge) perform roughly similarly when optimized, i.e. use bubble with size = 10 for merges. Without bubble-sort optimization, both merge sorts fall far behind in-place quick sort.

Iterative Merge Sort performs the poorest in all testing cases, surprisingly.

Analysis: The log-log scale representation of the data shows a few things worthy of notice:

- The slopes of the lines indicate the exponents of the polynomial growth.
- The vertical shifts indicate a multiplicative constant

As expected, all sort functions have larger constants than the problem size itself (all lines are above the blue line). Also expectedly, they all have roughly the same average slope as the problem size, indicating a growth of nearly O(n), i.e. O(nlogn).

Interestingly, iterative merge sort performs more poorly than its recursive counterpart, the **opposite** of which is what this assignment is supposed to **prove**.

Additionally, Linked List Merge Sort does not perform as well as expected. In fact, it could not handle 34 repetitions of problem size 10,000 in reasonable amounts of time.

Other than that, the results are as expected:

- **In-place Quick Sort with Random pivot choice** performs the best
- **Heap Sort** is among the poorest performers due to its overhead in maintaining the heap structure
- **Recursive Merge Sort** performs worse than Quick Sort does

We suspect that Iterative Merge Sort's poor performance, despite what we believe to be a **proper implementation** of it, is due to the fact that the 34 iterations don't all get new, unsorted data.

Raw Data

size	recurMerge	iterMerge	quick	heap	linked list
10	425	146	137	7790	1184
20	346	206	169	222	1260
100	905	1188	476	812	1999
200	4965	4048	919	8769	7791
1000	8784	427890	7572	30701	735867
2000	75729	786920	24627	91591	6182878
10000	1389611	42598345	222713	1883541	N/A

Another test:

Size	100	1514	2928	4342	5756	7170	8584
Recursive MergeSort	780	94641	91613	286016	1492838	286493	1414562
Optimized Recursive Merge Sort	359	9460	102446	102378	1177978	211510	179775
Iterative Merge Sort	1083	93575	4482206	9895196	14914909	23184934	41410870
Optimized Iterative Merge Sort	300	115131	137079 3	182861 2	1505713	2208176	4712425
Quick Sort	365	10566	42120	105339	1183983	285893	1298280
Heap Sort	692	72837	121841	1309901	1447654	1502061	1723275
LinkedList Merge Sort	1176	123822	207129	2226831	2461011	2553504	2929567

Figure 2. Performance timings for different types of sorts in Linear-Log scale.