Ellie Fitzpatrick

Michael Rabinovich

Data Structures

21 April, 2017

Statistical Analysis of Sorting Algorithms

1 Chart depicting the log of the times for heap quick, and merge sort for a sorted, reversed, and random array. Data collected from one run of the program.

In observing the data, one can immediately see that quicksort is the quickest of all the algorithms for sorted, reversed, and random arrays with the length of one million. This is due to its divide and conquer nature. It is also clear that for every list sorted except for the random quicksort, the order of elements does not matter all that much. Although, it does matter for the random test. This is because quicksort works by choosing a number as a partition and then sorting in regards to that partition. If the list is sorted or reverse, it either goes through and verifies that all the elements are in the correct place or swaps each value once. For random values, it has to loop through the array more often because the numbers are not in any order. Another trend that sticks out when looking through the data is how both heap and quicksort have a large variance, but merge sort does not. This is because merge sort algorithmically does the same thing no matter how the data is oriented. It always partitions to size two, then works its way back to the size of the array by combining the partitions. This analysis shows that the best method to use is quicksort in all situations.

Raw Data

\*\*\*\* Heap Sort \*\*\*\*

Time in nanoseconds

[1000, 10000, 100000, 1000000]

Sorted: Median Time, Median Time, Variance Time

[236986, 871420, 13141798, 87758841], [463150, 871265, 40190918, 112643865], [256681440688, 6525542, 2344771213089790, 2002452071457313]

Reverse: Median Time, Median Time, Variance Time

[72812, 925217, 9075942, 90774261], [83478, 979014, 8808348, 89695227], [1279065050, 9611574926, 4003773712111, 13975536263967]

Random: Mean Times, Variance Time

[77032, 1049507, 12234944, 122036637], [151342825, 26024127240, 2004051176586, 13331305537364]

\*\*\*\* Quick Sort \*\*\*\*

Time in nanoseconds

[1000, 10000, 100000, 1000000]

Sorted: Median Time, Median Time, Variance Time

[9275, 67710, 50550, 562550], [10048, 78376, 54879, 570280], [12545017, 356396630, 62448073, 898447764]

Reverse: Median Time, Median Time, Variance Time

[928, 5565, 51943, 532406], [773, 6183, 59362, 601507], [72075, 2221943, 165161780, 15509742559]

Random: Mean Times, Variance Time

[2921, 37008, 332475, 2906295], [1340651, 32252545, 15619173547, 2928573845641]

\*\*\*\* Merge Sort \*\*\*\*

Time in milliseconds

[1000, 10000, 100000, 1000000]

Sorted: Median Time, Mean Time, Variance Time

[2532, 987, 5702, 362610], [6409, 973, 6029, 366496], [65465029, 62890, 364819, 6075298202]

Reverse: Median Time, Mean Time, Variance Time

[1655, 1001, 4247, 302814], [8857, 1047, 4617, 300468], [173322439, 15090, 1275564, 24019387]

Random: Mean Times, Variance Time

[3122, 527, 4663, 316603], [42061200, 79454, 976113, 2026484585]

Data for 1,000,000 Items

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | Array | Median Time | Mean Time | Variance |
| Heap Sort (ns) | Sorted | 87758841 | 87758841 | 2002452071457313 |
| Heap Sort (ns) | Reversed | 90774261 | 89695227 | 13975536263967 |
| Heap Sort (ns) | Random |  | 122036637 | 13331305537364 |
| Quick Sort (ns) | Sorted | 595942 | 684676 | 24877412923 |
| Quick Sort (ns) | Reverse | 603362 | 665816 | 12410512086 |
| Quick Sort (ns) | Random |  | 2873182 | 28731822873182 |
| Merge Sort (ns) | Sorted | 362610 \* 106 | 366496\* 106 | 366496\* 106 |
| Merge Sort (ns) | Reverse | 302814\* 106 | 300468\* 106 | 24019387\* 106 |
| Merge Sort (ns) | Random |  | 316603\* 106 | 316603\* 106 |