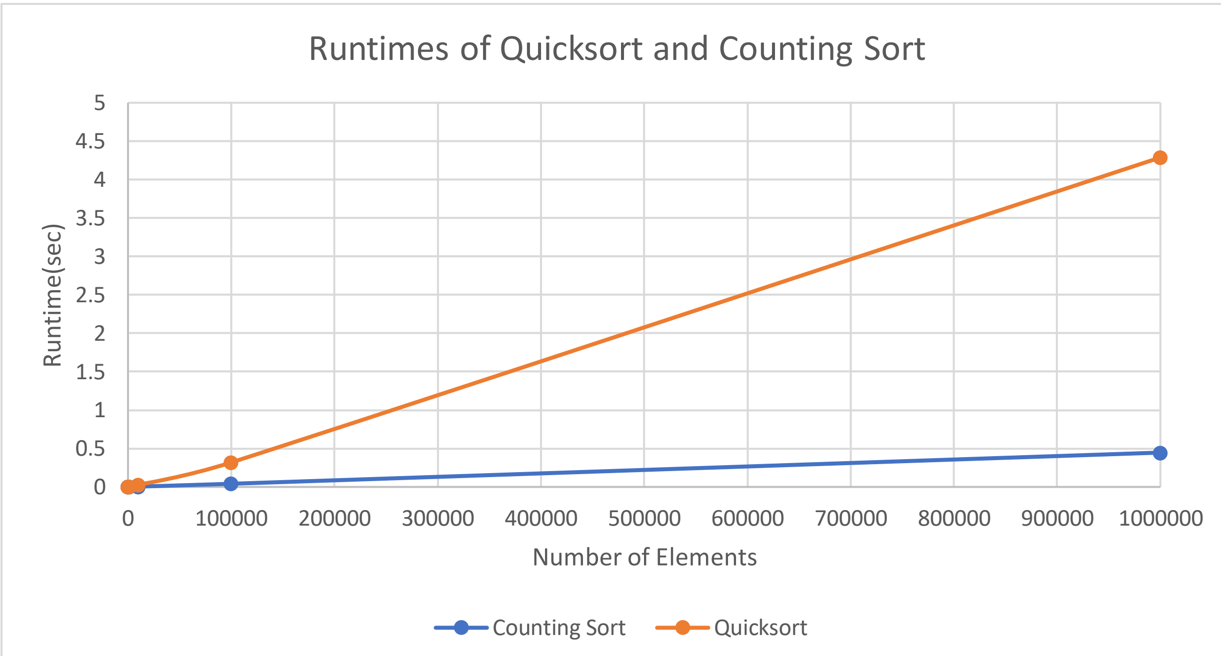
By: Nicholas Soucy

For: Dr Roy Turner

Quicksort versus Counting Sort comparison

For the runtime comparisons, each sort was running with a randomly generated array of sizes 10 – 1,000,000 in powers of 10. Each n ran 50 times to get a good average time for each data point. The random array was populated with positive integers in the range of 0 to 2n for more variability. The graph below shows the differences in quicksort versus counting sort.



Quicksort and Counting sort were basiclly identical at small n as expected. Once we hit an n = 100,000 we started seeing a difference. As expected, counting sort is quicker than quicksort.

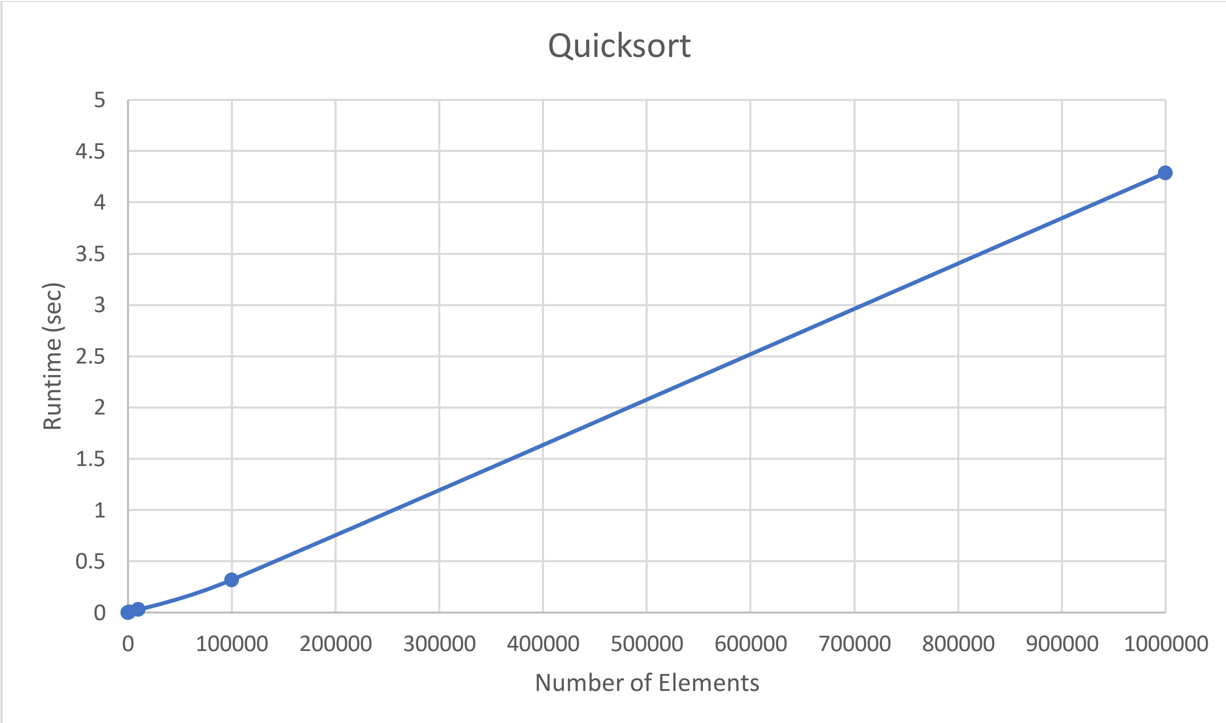
Lets now determine via the ratio test how quicksort and counting sort grow.

Quicksort:

Ratio Test

n	runtimes	n^2	n log(n)	n	log(n)
10	2.19E-05	2E-07	7E-07	2.2E-06	6.59E-06
100	0.000124	1E-08	2E-07	1.2E-06	1.87E-05
1000	0.002055	2E-09	2E-07	2.1E-06	0.000206
10000	0.02803	3E-10	2E-07	2.8E-06	0.002109
1E+05	0.31901	3E-11	2E-07	3.2E-06	0.019206
1E+06	4.2877	4E-12	2E-07	4.3E-06	0.215121

Here is the graph of just Quicksort:

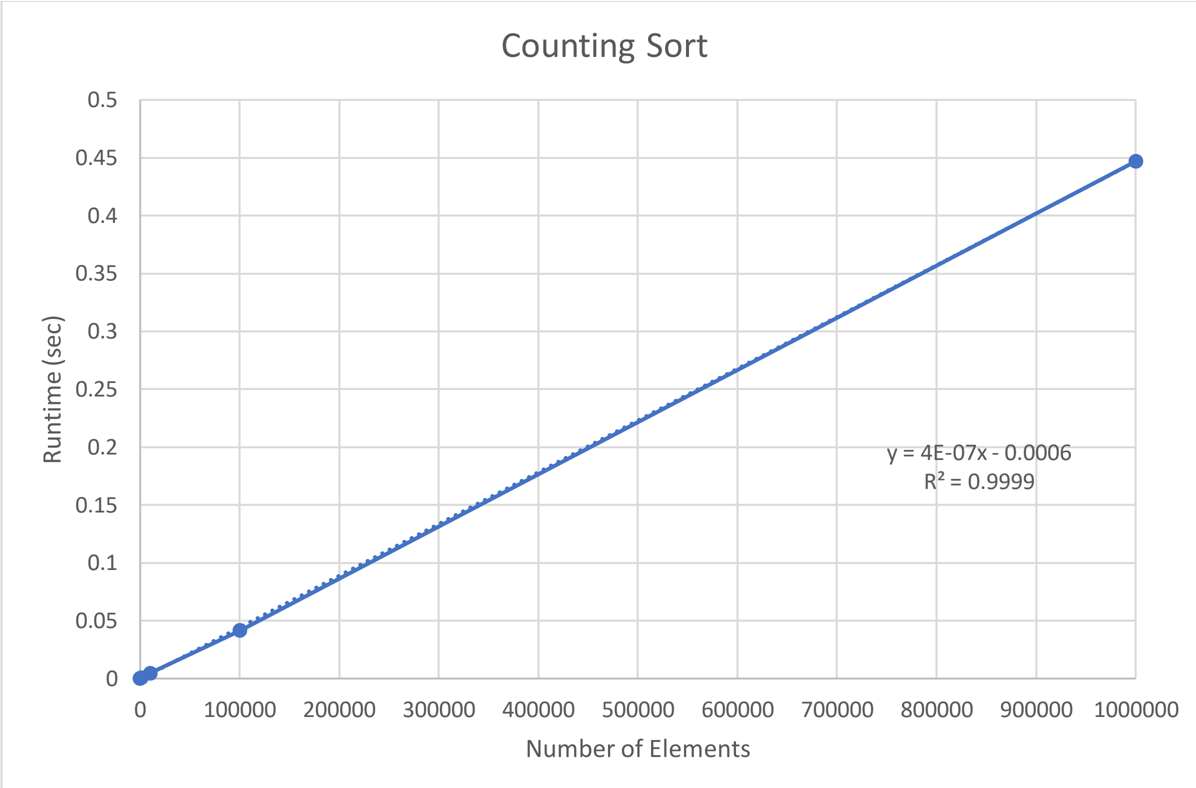


Counting Sort:

Ratio Test

n	runtimes	n^2	n log(n)	n	log(n)
10	5.54E-06	5.5E-08	1.67E-07	6E-07	2E-06
100	7.2E-05	7.2E-09	1.08E-07	7E-07	1E-05
1000	0.000483	4.8E-10	4.84E-08	5E-07	5E-05
10000	0.004545	4.5E-11	3.42E-08	5E-07	3E-04
1E+05	0.04126	4.1E-12	2.48E-08	4E-07	0.002
1E+06	0.44685	4.5E-13	2.24E-08	4E-07	0.022

Here is the graph of just Counting Sort:



We can see clearly from the ratio test for quicksort that n^2 is going to zero so it is an overestimate and n & log(n) are going to infinity so they are underestimates. <u>We can see clearly that nlog(n) is the correct growth rate for quicksort based on the ratio test above because it is going towards the constant 2E-07.</u>

For counting sort, we can see that n^2 and n log(n) are overestimates while log(n) is an underestimate. <u>We can see clearly that n is the correct growth rate for counting sort based on the ratio test above because it is going towards the constant 4E-07.</u>

From the Counting Sort graph, we can also see how clearly linear that is, with a best-fit linear fit with an R^2 value of 0.9999 is too strong to ignore. Excel does not have a best-fit for n log(n), so the ratio test needed to be used to determine the runtime pattern for Quicksort.

Conclusion:

In conclusion, we emperically detemined that Counting sort is indeed faster than Quicksort for large enough n. By the ratio test, we detemined that Quicksort grows as n log(n) and Counting Sort grows linerarly (n). This shows why Counting Sort out performed Quicksort for large n. These growth functions also verify the theoretical growth functions for Quicksort and Counting sort.