Eliezer Amponsah

02/16/2024

CSE 431 – Algorithm Engineering

Dr. Charles Ofria

Asymptotic Analysis for Insertion Sort and Merge Sort Algorithm

Hypothesis:

It is well known that asymptotically speaking, the sorting algorithm, merge sort will outperform insertion sort. For small values of N however, where N represents the amount of data to be sorted, insertion sort may outperform merge sort. Asymptotically, insertion sort is characterised as $O(n^2)$ while Merge sort is characterised as an $O\left(nlog_2(n)\right)$. Mathematically, there is never a case where $0 = nlog_2(n) - n\text{^}2$ is true. This led me to believe that it will not be the case that insertion sort will out perform Merge sort even for small values of N (counter to what I do know by observation).

Method:

To test this hypothesis, I wrote a program in C++ that implemented the two sorting algorithms. Credits to Stack Overflow and Bing AI for the merge sort algorithm. Checks were put in place to make sure the data is indeed sorted by comparing it with the C++ STL Vector::Sort algorithm. With the two algorithms in place, the program simply asks the user for how many trial runs, how many N values, the minimum possible value and the maximum possible value. The GenerateVector() function is then called to populate a vector with N values in the range of minimum value and maximum value. Once the vector is generated, a clock timer starts right before we start the insertion sort algorithm and stops right afterwards. The stop and start time are subtracted and outputted to the console to communicate the length of time it took to return from the function. Similarly, the same procedure is done right before and right after merge sort algorithm is executed. When the values are done, the program notifies the user of the faster algorithm for the specified amount of N values and prompts for a re-run. This is repeated for x amount of trial runs and recorded. I pick 20 trials: the first 5 trials are for values of 10, 100, 500, 1000 repeated twice. The last trial tests an N value of values of 3,000

Results:

The results showed that I was indeed wrong and that merge sort was slower than insertion sort for small values of N. After about an N value of 500, Merge sort started winning the race.

Conclusion:

The results of the data proved my initial hypothesis wrong in that Merge sort is not faster than Insertion sort all the time. I had the initial hypothesis that insertion sort could never beat Merge sort due to the fact that there is no instrrance in which n^2 <= nlog(n). However, if merge sort had a higher constant than insertion sort did, then it would stand to prove that there are points where insertion sort would perform better than merge sort for smaller values of N.