Stone Gulliksen

CSE 310

6/1/2019

Written Response for Recitation Assignment 2-part d)

d) Compare the graphs in (c) and the asymptotic analysis based running time of above sorting algorithms. Can you conclude that asymptotic running time is a good indicator of actual performance of the algorithm after comparing the graphs in (c) and the asymptotic analysis based running time? Explain your answer.

Selection sort runs in Θ(n2) time. The best- and worst-case scenario don’t have that much of an effect on the running time, the index will always perform the same function of having to search the whole array for a minimum value in each loop cycle. Therefore, the time of execution will always be Θ(n2). This can be seen in the graph as the size of the array increases in the same fashion as the notation. When comparing the run time of selection sort to that of insertion sort (Which both have an average runtime of Θ(n2)) It is important to notice the difference in how the algorithms run. The best case and worst-case scenario for insertion sort differ from the fact that the algorithm accounts for the integers that are already sorted, this helps because it won’t have to search all indexes. This can lead to cases where no shifting is required for the key which allows the algorithm to sort at a more efficient pace. In the best-case scenario where the elements are already sorted the run time will be Θ(n). In contrast, selection sort has to search the whole array to find the next minimum value. When comparing the graphs, you will notice the insertion sort doesn’t increase at the same rate but at a slightly rate better due to the difference in variation of the best- and worst-case situations that insertion sort has. Both algorithms sort in place.

Both Merge sort and Quick sort run at Θ (nlog2n) time. They both apply the divide and conquer strategy of breaking up a big problem into smaller sub problems using recursion. This leads to major improvements in running time when comparing these algorithms to insertion and selection sort. The only major difference between these tow algorithms is the fact merge sort needs to allocate more memory for every sub array created when the recursion takes place. This differs from Quick sort where the algorithm doesn’t need to create any more space in order to perform its function. Quick sort for the most part performs slightly better then quick sort. However, there can be rare occurrences where merge sort can perform better then quick sort. Over all the graphs indicate and illiterate a great visual representation of the performance of the algorithms discussed here.