CS 301 Assignment 2

Beyzagul Demir, 28313

November 8, 2022

1 Problem 1 (Order Statistics)

1.1 (a)

First of all, we need to sort elements. We can use different algorithms but I choose Merge Sort since it is asymptotically optimal. Sorting them using merge sort will take θ(n log n) time complexity for the best asymptotic worst-case running time. After that, finding and printing k smallest numbers will have $\theta(k)$ time complexity since our elements are sorted. As a result, we have $\theta(k + n \log n)$. However, we know that k < n for all cases so we can ignore k, and finally we have $\theta(n \log n)$ time complexity to sort the numbers using a comparison-based algorithm and return the k smallest numbers.

1.2 (b)

First of all, using an order statistics algorithm, we can choose the randomized selection. In that way, we can find the k^{th} smallest number in $\theta(n)$ time complexity. After that, partitioning around that number to get the k smallest numbers, again take θ(n) time complexity. As the last step, we will sort these k smallest numbers using a comparison-based sorting algorithm. I choose to merge sort for this and we have θ(k log k) time complexity for it. Finally, we have 2n + klog k as an outcome function and it's time complexity is $\theta(n + k \log k)$.

1.3 Compare Method (a) and (b)

When we compare (n log n) from part a and (n + k log k) from part b, we know that k is always smaller than n. That means (n + k log k) will always be a better choice since (n log n) dominates (n + k log k) for all cases. Therefore, I would prefer to use method b.

2 Problem 2 (Order Statistics)

2.1 (a)

The Radix sort algorithm takes integers like a string of digits. Before starting to sort, we need to look at the length of the strings. Since every string can have a different length we should equalize their length to the longest one's length by adding dummy

characters. We can choose '*' as a dummy variable. We will add '*' at the end of the string to equalize their length to the longest one. After that, we will identify our base. In this case, we are sorting strings so we have 26 alphabetical characters from 'a' to 'z' and 1 dummy character which is '*'. As a result, we have 27 as a size to save the occurrences of the characters. Now we are ready to use the radix sort algorithm. The algorithm will start from the least significant digit (LSD) and continue to compare and sort until the most significant character(MSD)(MSD included).

2.2 (b)

As we mentioned in part a, the radix sort algorithm will start from the least significant digit (LSD) and continue to compare and sort until the most significant character(MSD)(MSD included).

Finally, our array is compared and sorted. It is: ["BARIS**" , "BATURAY" , "GIRAY**", "GORKEM*" , "TAHIR**"]

2.3 (c)

We can see that number of steps gives us the length of the longest element. Let's say k for the length of the longest element and then we have $\theta(k)$. After that our base size is 27 which consists of alphabetical characters from 'a' to 'z' and '*'. Now our time complexity is $\theta(27k)$. As a next step, let's assume n is the number of elements. Then we have $\theta((27 + n)k$ as time complexity. It is equal to $\theta(27k + nk)$ but since 27 is a constant value, we can say the time complexity is $\theta(nk)$ where n is the number of elements and k is the length of the longest string.