Jose Marquez Jaramillo
EN.605.621 Foundations of Algorithms - Spring 2023
March 6, 2023

Homework 3

Question 1.

Assume that the algorithm has n in total. Starting from the root node, let k be the number of nodes on one side of the root and the remaining $n-k-1$ be the number of nodes on the other side. The complexity to traverse this tree can then be defined by

$$T(n) = T(k) + T(n-k-1) + c \tag{1}$$

Let's proceed with the two extreme cases, when one of the sub-trees is empty and when there is an equal number of nodes in both trees:

- When one of the sub-trees is empty then $k = 0$:

$$\begin{aligned} T(n) &= T(0) + T(n-1) + c \\ T(n-1) &= T(0) + T(n-1) + c \\ T(n-2) &= T(0) + T(n-3) + c \\ &\dots \\ T(n(n-1)) &= T(0) + T(0) + c \end{aligned} \tag{2}$$

  This indicates that when the tree is empty then the time required to process the algorithm would be a constant as the root is of NoneType. Adding all the equations above we get:

$$\begin{aligned} T(n) &= n \times T(0) + n \times c \\ &= nk + nc \\ &= n(k+c) \\ &= O(n) \end{aligned} \tag{3}$$

- When both the sub-trees are of equal size, that is, $k = (n/2)$:

$$\begin{aligned} T(n) &= T(n/2) + T(n/2 - 1) + c \\ &= 2T(n/2) \end{aligned} \tag{4}$$

  By the master theorem, this results in $T(n) = O(n)$

Because all other cases will be bounded by these two edge cases, we can indicate that this algorithm is $O(n)$

Question 2.

We can denote $Optimal(i)$ as the cost for the optimal schedule over the weeks 1-i. We can then consider how to find the optimal schedule at the ith week:

If we were to hire company A:

- If we were to hire company A, we would need to pay rs_i for week i,
- We also need to pay for the previous weeks of work. We need to decide the best schedule for week i, knowing that in previous weeks we have already made decisions from week 1 to week $i-1$. The total cost on the previous weeks is therefore given by $Optimal(i-1)$.
- The total cost of hiring company A is therefore

$$rs_i + Optimal(i-1) \tag{5}$$

If we were to hire company B:

- We can hire company B for week i, but it needs to happen in 4 week blocks. This means we would need to hire Company B for blocks i,$i-1$,$i-2$, and $i-3$.

- We would therefore pay Company B a weekly rate of c for a total of 4 weeks, or $4c$.
- Note that the difference with company A is that in company B we don't make weekly decisions but we make decisions every 4 weeks. Therefore we make decisions taking into account that 4 weeks ago we made the decisions corresponding to the following 4 weeks. The total cost on the previous weeks is therefore given by $Optimal(i-4)$.
- The total cost of hiring company A is therefore
- The total cost of hiring company A is therefore

$$4c + Optimal(i-4) \tag{6}$$

We seek to minimize the overall cost of hiring either company, the Bellman equation therefore becomes:

$$Optimal(i) = \min\{rs_i + Optimal(i-1), 4c + Optimal(i-4)\} \tag{7}$$

Question 3.

We know that we will be given an input sequence $y_1y_2...y_n$ of length n, we also know that there is a $Quality()$ function that given any sequence of letters would return the quality of the sequence. The larger the value of quality returned, the better the quality.

We need to compute the optimal segmentation of the input string into words such that the sum of these word's qualities is the highest possible. We can therefore follow a dynamic program with the following recursive function:

$$Opt(j) = \max_{j \leq i}\{Opt(j-1) + Quality(y_j...y_n)\} \tag{8}$$

Where:

- $Quality(y_j...y_n)$ gives us the quality of the word that is formed by the characters starting from and ending in position n.
- $Opt(i)$ is the score of the best segmentation of the prefix consisting of the first characters of y.

This means that we would like to maximize the value of $Opt(n)$. That is, find the value that maximizes the quality considering all the previous maximizing values. If you think about it, we are going through the string and maximizing as a character is evaluated.

Now, we can look at the induction provided by the i value:

1) In the base case we have a one-character word,
2) In the inductive step since we know the past score from the first iteration, we know that $Opt()$ finds the optimal solution for the indices less than the current iteration. Because of this, we know that $Opt(j-1)$ provides the optimal solution up to the jth character. Therefore the previous $j-1$ characters must be optimal. Now, the optimal cost $Opt(i)$ will yield the value of the segmentation we mentioned. The cost $Opt(i)$ is the cost of $Opt(j)$ and the cost of adding the last character.

Because we are finding the value that maximizes the value of $Opt()$ at each step, the recurrence will provide a result that maximizes its value at each step.

Question 4.

(a) Let $W = \{w_1, w_2, w_3, w_4\} = \{2, 3, 2, 3\}$ and $K = 2$. In this case, the greedy algorithm would load 2 onto the first truck. The second weight is too heavy for the first truck and the algorithm would send it away and load 2 onto the next truck. This truck is not fully loaded and must be sent off. This continues with the third and fourth trucks as well. The optimal number of trucks is three. By loading the two weights of 2 into a single truck and loading the two weights of 3 onto two other trucks we achieve the optimal number of trucks. This is even though a more optimal solution could be to use 2 trucks instead of three.

(b) W is the total number of trucks used by the algorithm. This is calculated by $W = \sum_i w_i$. Each truck can hold at most K units of weight. The minimum number of trucks needed by the algorithm can be approximated by W/K. Because of how the algorithm behaves, the number of trucks used

by the greedy algorithm would be an odd number which can be denoted by $t = 2v + 1$ where t is even. We can divide the number of trucks into 2 consecutive groups giving a total of $v + 1$ groups of trucks. Each group but the last would have a total weight of containers which must be strictly greater than K. Otherwise, the second truck in the group would not have been started then. It is then the case that $W > vK$ and $W/K > v$. From this, it follows that the optimum solution uses at least $v + 1$ trucks, which is within a factor of 2 of $t = 2v + 1$.