Jose Marquez Jaramillo
EN.605.621 Foundations of Algorithms - Spring 2023
March 20, 2023

Homework 4

Question 1.

A simple algorithm can be inplemented where we traverse the adjacent list for every vertex. Let's imagine that we are at vertex i. We traverse the list of vertex i, if the size of the adjancent list of vertex i is x then the out degree for $i = x$ and we increment the in degree of every vertex that has an incoming edge from i. We then repeat the steps for every vertex and print the in and out degree for all the vertices at the end. Below is a pseudo-code implementation:

Algorithm 1. PrintInOut(AdjList,n)

Inputs: List of m adjacent edges AdjList and number of vertices n

```
1: In $\leftarrow$ Initiate list of $n$ zeroes
2: Out $\leftarrow$ Initiate list of $n$ zeroes
3: for $i \in$ AjdList do
4:     List $\leftarrow$ AdjList[$i$]
5:     Out[$i$] $\leftarrow$ $sizeof$(List)
6:     for $j \in$ List do
7:         In[List[$j$]] $\leftarrow$ In[List[$j$]] $+ 1$
8: for $k \in range(n)$ do
9:     $print(k, \text{In}[k], \text{Out}[k])$
```

Now, for the complexity, we have that the loop from lines $3 - 7$ will run the number of vertices in the graph m while the loop in lines $8 - 9$ will run a total of the number of edges n in the graph respectively.

Question 2.

Upon inspecting Figure 9.2 in the textbook, we can visualize an $x - y$ plane. In this $x - y$ plane the main pipeline would be parallel to the x axis. In this way, the y coordinate of ith point would be y_i. Therefore, if the main pipeline is separated from the x-axis by y point, then the distance of all pipelines from that line can be denoted as $y_i - y$. The total (absolute) length of the pipe would therefore be $\sum_{i=1}^{n} \|y_i - y\|$. The objective of the problem is then to find a value for y such that it is minimized.

Next, let's consider the implications of these operations. Consider that if i points lie above the y line and j points lie below y. Whenever $i > j$ then if we increase y to $y + \delta_y$. The total difference will then increase by $(i - j)\delta_y$. On the other hand, if $i < j$, then if we were to decrease y to $y - \delta_y$, the sum will decrease by $(j - i)\delta_y$.

Under this reasoning, we find that it is optimal to have the same number of points above and below y. This is equivalent to using the median value for all the pipelines.

Question 3.

Denote the following random variables:

- For $j = 1, 2, ..., n$, denote X_j as the increase in value given to the counter by the jth INCREMENT function.
- Let V_n represent the value of the counter after n INCREMENT operations

(a) We would then like to compute $\mathbb{E}[V_n]$ which by linearity of expectations is given by:

$$\begin{aligned}\mathbb{E}[V_n] &= \mathbb{E}[X_1 + X_2 + ... + X_n]\\ &= \mathbb{E}[X_1] + \mathbb{E}[X_2] + ... + \mathbb{E}[X_n]\\ &= \sum_{i=1}^{n} \mathbb{E}[X_i]\end{aligned} \tag{1}$$

For $\mathbb{E}[V_n] = n$, we would need that $\mathbb{E}[X_1] = 1, \mathbb{E}[X_2] = 1, ..., \mathbb{E}[X_n] = 1$. Imagine that at the beginning of the jth INCREMENT operation, the counter holds the value i representing n_i. If this counter increase because of the INCREMENT operation, the value represented by such an increase can be represented by $n_{i+1} - n_i$. By the problem statement, the counter increases with probability $\frac{1}{(n_{i+1}-n_i)}$. We can therefore generalize that:

$$\begin{aligned}\mathbb{E}[X_j] &= (0 \times \mathbb{P}\{\text{counter increases}\}) + ((n_{i+1} - n_i) \times 1 - \mathbb{P}\{\text{counter increases}\})\\ \mathbb{E}[X_j] &= \left(\left(0 \times \left(1 - \frac{1}{(n_{i+1} - n_i)}\right)\right) + \left((n_{i+1} - n_i) \times \frac{1}{(n_{i+1} - n_i)}\right)\right)\\ &= 1\end{aligned} \tag{2}$$

Because $\mathbb{E}[X_j] = 1$, then $\mathbb{E}[X_1] = 1, \mathbb{E}[X_2] = 1, ..., \mathbb{E}[X_n] = 1$ and $\mathbb{E}[V_n] = n$.

(b) Since $n_i = 100i$, at the increment we can see that $n_{i+1} - n_i = 100(i + 1) - 100i = 100$. It is then the case that with probability 0.99, the increase in the value given by the counter due to the jth INCREMENT operation is 0. By total probability, we get that with probability 0.01, the value represented increases by 100:

$$\begin{aligned}\mathbb{V}[X_j] &= \mathbb{E}[X_j^2] - \mathbb{E}^2[X_j]\\ &= \left(\left(0^2 \times 0.99\right) + \left(100^2 \times 0.01\right)\right) - 1^2\\ &= 100 - 1 = 99.\end{aligned} \tag{3}$$

(c) One possible reason for using different bases such as the Fibonacci sequence could be the rate of growth on those particular sequences. In the case of Fibonacci, it has been shown that it grows at a rate that approximates 1.61803. This rate is less than $2^b - 1$. In the case of $n_i = 2^{i-1}$, the reason could be similar as the rate of growth is lower. These lesser rates of growth could provide efficiencies in computational and storage costs.

Question 4.

(a) In the case that both operations have to pop the entire stack, the running time of each operation would be the size of the entire stack. Therefore the worst-case MultiPop(A) runs in $O(n)$ and MultiPop(B) runs in $O(m)$. Transfer pops elements off of A and pushes them onto B. The worst case happens when all of the elements of A are popped and transferred onto B. Therefore n pops happen from A while n pushes happen onto B, this gives a worst-case running time of $O(n)$.

(b) Consider the function $\Phi(n, m) = 3n + m$. At the beginning the potential is 0, also, for non-empty stacks, the potential should always be positive. This provides that the amortized costs can be computed in the following way:

- Push(A,x):

$$\begin{aligned}\hat{c}_i &= c_i + \Phi(D_{i+1}) - \Phi(D_i) \\ &= 1 + 3(n+1) + m - (3n + m) \\ &= 4\end{aligned} \tag{4}$$

- Push(B,x):

$$\begin{aligned}\hat{c}_i &= c_i + \Phi(D_{i+1}) - \Phi(D_i) \\ &= 1 + 3n + (m+1) - (3n + m) \\ &= 2\end{aligned} \tag{5}$$

- MultiPop(A,k):

$$\begin{aligned}\hat{c}_i &= k + 3(n-k) + m - (3n + m) \\ &= -2k\end{aligned} \tag{6}$$

- MultiPop(B,k):

$$\begin{aligned}\hat{c}_i &= k + 3n + (m-k) - (3n + m) \\ &= 0\end{aligned} \tag{7}$$

- Transfer(k):

$$\begin{aligned}\hat{c}_i &= 2k + 3(n-k) + (m+k) - (3n + m) \\ &= 0\end{aligned} \tag{8}$$

Therefore, the 5 operations have constant worst-time complexity or are $O(n1)$.