Worksheet 15 Review

April 1, 2020

Question 1

a. First, we will evaluate the cost of he inner most loop.

Because the loop runs from $j = i = 1$ to $j = n - 1$, with each iteration costing 1 step, we can conclude that the inner most loop has cost of at most

$$\lceil (n-1) - (i+1) + 1 \rceil = n - i - 1 \tag{1}$$

steps.

Next, we will evaluate the cost of the outer most loop.

Because the loop runs from $i = 0$ to $i = n - 1$ with each iteration costing $(n - i - 1)$ steps, we can conclude the outer most loop has cost of at most

$$\begin{aligned}
\sum_{i=0}^{n-1}(n-i-1) &= \left[\sum_{i=0}^{n-1}(n-1) - \sum_{i=0}^{n-1} i\right] && (2)\\
&= \left[\frac{2n(n-1)}{2} - \frac{(}{n}(n-1))2\right] && (3)\\
&= \frac{n(n-1)}{2} && (4)
\end{aligned}$$

steps.

Next, we will bring everything together.

Since the lines **n = len(lst)** and **return False** have cost of 1 step each, the total cost of the algorithm is

$$\frac{n(n-1)}{2} + 2 \tag{5}$$

steps.

Then, it follows from above that the algorithm has runtime of $\Theta(n^2)$.

Correct Solution:

First, we will evaluate the cost of he inner most loop.

Because the loop runs from $j = i = 1$ to $j = n - 1$, with each iteration costing 1 step, we can conclude that the inner most loop has cost of at most

$$\lceil (n-1) - (i+1) + 1 \rceil = n - i - 1 \tag{6}$$

steps.

Next, we will evaluate the cost of the outer most loop.

Because the loop runs from $i = 0$ to $i = n - 1$ with each iteration costing $(n - i - 1)$ steps, we can conclude the outer most loop has cost of at most

$$\begin{align} \sum_{i=0}^{n-1} (n-i-1) &= \left[\sum_{i=0}^{n-1} (n-1) - \sum_{i=0}^{n-1} i \right] \tag{7} \\ &= \left[\frac{2n(n-1)}{2} - \frac{(}{n}(n-1))2 \right] \tag{8} \\ &= \frac{n(n-1)}{2} \tag{9} \end{align}$$

steps.

Next, we will bring everything together.

Since the lines **n = len(lst)** and **return False** have cost of 1 step each, the total cost of the algorithm is at most

$$\frac{n(n-1)}{2} + 2 \tag{10}$$

steps.

Then, it follows from above that the algorithm has runtime of $\mathcal{O}(n^2)$.

Notes:

- Noticed that in here, professor considers the cost of loop variables and other lines with constant time.
- $\mathcal{O}$ used since we are determining the upper bound.
- In worksheet 14, the cost of loop variables is not required.

b. Let $n \in \mathbb{N}$, and $lst = [0, 1, 2, 3, \ldots, n-3, n-1, n-1]$.

First, we will calculate the cost of the inner most loop.

Because we know the inner most loop will terminate when **if lst[i] == lst[j]** and because we know this condition occurs when $i = n-2$, we can conclude the loop will start at $i = 0$ and run until $i = n-2$.

Because we know the condition of the inner most loop **for j in range(i+1,n)** stays true until $i = n-2$ even at the worst case, we can conclude that the cost of inner loop is the same as the cost of the inner loop at worst case, that is

$$n - i - 1 \tag{1}$$

Next, we will evaluate the cost of the outer most loop.

Since the outer most loop starts at $i = 0$ and ends at $i = n-1$ with each iteration costing $(n-i-1)$ steps, the outer most loop has cost of

$$\sum_{i=0}^{n-1}(n-i-1) = \frac{n(n-1)}{2} \tag{2}$$

steps.

Next, we will combine everything together.

Since each of the lines **n = len(lst)** and **return True** have cost of 1 step, we can conclude that the algorithm has total cost of

$$\frac{n(n-1)}{2} + 2 \tag{3}$$

steps.

Then, we can conclude the algorithm has runtime of $\Omega(n^2)$.

Because we know both $\mathcal{O}(n^2)$ and $\Omega(n^2)$ are true, we can also conclude the algorithm has runtime of $\Theta(n^2)$.

Notes:

- Does if condition counted towards the total cost?

Question 2