Worksheet 14 Review

April 1, 2020

Question 1

a. Since the inner loop starts at $j = 0$ and finishes at $j = n - 1$ with j increasing by 1 per iteration, we can conclude that the inner loop has

$$\lceil n - 1 - 0 + 1 \rceil = n \tag{1}$$

iterations.

Since the inner loop takes 1 step per iteration, we can conclude that the inner loop has the total cost of

$$n \cdot 1 = n \tag{2}$$

steps.

For the outer loop, because it starts at $i = 0$ and ends at $i = n - 1$ with i increasing by 5 per iteration, we can conclude that the outer loop has

$$\left\lceil \frac{n - 1 - 0 + 1}{5} \right\rceil = \left\lceil \frac{n}{5} \right\rceil \tag{3}$$

iterations.

Since each iteration in the outer loop takes n steps, we can conclude the outer loop has the total cost of

$$n \cdot n = n^2 \tag{4}$$

steps.

Since we are ignoring the cost of the loop variables, the total cost of the algorithm is $n^2 + n$ steps.

Then, because we know the algorithm takes total of $n^2 + n$ steps, we can conclude the algorithm has the runtime of $\Theta(n^2)$.

b. We will determine the exact cost and theta of this algorithm by first calculating the exact cost of inner loop 1

```
    j = 1
    while j < n:
        j = j * 3

```

and then, calculating the exact cost of inner loop 2

```
    k = 0
    while k < n:
        k = k + 2

```

and then, calculating the exact cost of the outer loop using the information from the exact cost of inner loop 1 and inner loop 2

```
    i = 4
    while i < n:
        j = 1
        while j < n:
            j = j * 3
        k = 0
        while k < n:
            k = k + 2
        i = i + 1

```

and then, we will finish off by calculating the theta of the outer loop.

Part 1 (Calculating the exact cost of loop 1):

Because we kow $j = j \cdot 3$, we can calculate

$$\begin{aligned} i_1 &= 3 \\ i_2 &= 9 \\ i_3 &= 27 \\ &\vdots \\ i_j &= 3^j \end{aligned}$$

Then, using the fact that loop termination occurs when $i_j \geq n$, we can conclude

$$3^j \geq n \tag{1}$$
$$j \geq \log_3 n \tag{2}$$

Since we are looking for the smallest value of j resulting in loop termination, we can conclude the value of j is $\lceil \log_3 n \rceil$.

Since the inner loop 1 takes constant step per iteration, we can conclude that the loop has exact cost of

$$\lceil \log_3 n \rceil \cdot 1 = \lceil \log_3 n \rceil \tag{3}$$

steps.

Part 2 (Calculating the exact cost of loop 2):

Since the loop starts from $k = 0$ and ends at $k = n - 1$, with k increasing by 2 per iteration, we can conclude that the loop has

$$\left\lceil \frac{n-1-0+1}{2} \right\rceil = \left\lceil \frac{n}{2} \right\rceil \tag{4}$$

iterations.

Since the loop takes 1 step per iteration, the loop has total cost of

$$\left\lceil \frac{n}{2} \right\rceil \cdot 1 = \left\lceil \frac{n}{2} \right\rceil \tag{5}$$

steps.

Part 3 (Calculating the exact cost of outer loop):

Since the loop runs from $i = 4$ to $i = n - 1$ with i increasing by 1 per iteration, we can conclude the loop has

$$\left\lceil \frac{n-1-4+1}{1} \right\rceil = n - 4 \tag{6}$$

iterations.

Since each iteration takes $\lceil \log_3 n \rceil + \left\lceil \frac{n}{2} \right\rceil$ steps, we can conclude the outer loop has total of

$$(n-4) \cdot \left(\lceil \log_3 n \rceil + \left\lceil \frac{n}{2} \right\rceil \right) \tag{7}$$

steps.

Part 4 (Calculating Theta):

Because we know the loop in total has exact cost of $(n-4) \cdot \left(\lceil \log_3 n \rceil + \left\lceil \frac{n}{2} \right\rceil \right)$, we can conclude that the algorithm has total runtime of $\Theta(n^2)$.

<u>Correct Solution:</u>

We will determine the exact cost and theta of this algorithm by first calculating the exact cost of inner loop 1

```
        j = 1
        while j < n:
            j = j * 3

```

and then, calculating the exact cost of inner loop 2

```
        k = 0
        while k < n:
            k = k + 2

```

and then, calculating the exact cost of the outer loop using the information from the exact cost of inner loop 1 and inner loop 2

```
        i = 4
        while i < n:
            j = 1
            while j < n:
                j = j * 3
            k = 0
            while k < n:
                k = k + 2
            i = i + 1

```

and then, we will finish off by calculating the theta of the outer loop.

Part 1 (Calculating the exact cost of loop 1):

Because we kow $j = j \cdot 3$, we can calculate

$$
\begin{aligned}
i_1 &= 3 \\
i_2 &= 9 \\
i_3 &= 27 \\
&\vdots \\
i_j &= 3^j
\end{aligned}
$$

Then, using the fact that loop termination occurs when $i_j \geq n$, we can conclude

$$3^j \geq n \tag{1}$$
$$j \geq \log_3 n \tag{2}$$

Since we are looking for the smallest value of j resulting in loop termination, we can conclude the value of j is $\lceil \log_3 n \rceil$.

Since the inner loop 1 takes constant step per iteration, we can conclude that the loop has exact cost of

$$\lceil \log_3 n \rceil \cdot 1 = \lceil \log_3 n \rceil \tag{3}$$

steps.

Part 2 (Calculating the exact cost of loop 2):

Since the loop starts from $k = 0$ and ends at $k = n - 1$, with k increasing by 2 per iteration, we can conclude that the loop has

$$\left\lceil \frac{n-1-0+1}{2} \right\rceil = \left\lceil \frac{n}{2} \right\rceil \tag{4}$$

iterations.

Since the loop takes 1 step per iteration, the loop has total cost of

$$\left\lceil \frac{n}{2} \right\rceil \cdot 1 = \left\lceil \frac{n}{2} \right\rceil \tag{5}$$

steps.

Part 3 (Calculating the exact cost of outer loop):

Since the loop runs from $i = 4$ to $i = n - 1$ with i increasing by 1 per iteration, we can conclude the loop has

$$max(\left\lceil \frac{n-1-4+1}{1} \right\rceil, 0) = max(n-4, 0) \tag{6}$$

iterations.

Since each iteration takes $\lceil \log_3 n \rceil + \lceil \frac{n}{2} \rceil$ steps, we can conclude the outer loop has total of

$$max(n-4, 0) \cdot \left(\lceil \log_3 n \rceil + \left\lceil \frac{n}{2} \right\rceil \right) \tag{7}$$

steps.

Part 4 (Calculating Theta):

Because we know the loop in total has exact cost of $max(n-4,0) \cdot \left(\lceil \log_3 n \rceil + \lceil \frac{n}{2} \rceil\right)$, we can conclude that the algorithm has total runtime of $\Theta(n^2)$.

Notes:

- Noticed professor uses $max(f(n),0)$ when a loop variable doesn't start at $i=0$.
- Noticed professor skipped the detailed explanation on the evaluation of the number of iterations.

c. Since the inner most loop has j iterations, and since it has cost of 1 step per iteration, we can conclude the inner most loop has cost of

$$j \cdot 1 = j \tag{1}$$

steps.

For the intermediate loop, because we know it runs n iterations with the cost of j steps per iteration, we can conclude the intermediate loop has cost of

$$\left[\sum_{j=0}^{n-1} j\right] \cdot 1 = \frac{n(n-1)}{2} \cdot 1 \tag{2}$$

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

steps.

For the outer loop, because we know it has $\lceil \frac{n}{4} \rceil$ iterations with each iteration taking $\frac{n(n-1)}{2}$ steps, we can conclude the the outer loop has cost of

$$\left\lceil \frac{n}{4} \right\rceil \cdot \frac{n(n-1)}{2} \tag{4}$$

steps.

Because we know the loop has exact cost of $\lceil \frac{n}{4} \rceil \cdot \frac{n(n-1)}{2}$ steps, we can conclude that the algorithm has runtime of $\Theta(n^3)$.

Correct Solution:

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

Since the inner most loop has j iterations, and since it has cost of 1 step per iteration, we can conclude the inner most loop has cost of

$$j \cdot 1 = j \tag{1}$$

steps.

Next, we calculate the cost of the intermediate loop.

Because we know the loop is in reverse from $j = n$ to $j = 1$ with j decreasing by 1 per iterations, we can conclude this is the same as going from $j = 1$ to $j = n$ with j increasing by 1.

Because we know the loop has the cost of j steps per iteration, we can conclude the intermediate loop has cost of

$$\left[\sum_{j=1}^{n} j \right] \cdot 1 = \frac{n(n+1)}{2} \cdot 1 \tag{2}$$

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

steps.

Finally, we calculate the cost of the outer loop.

Because we know it has $\lceil \frac{n}{4} \rceil$ iterations with each iteration taking $\frac{n(n+1)}{2}$steps, we can conclude the the outer loop has cost of

$$\left\lceil \frac{n}{4} \right\rceil \cdot \frac{n(n+1)}{2} \tag{4}$$

steps.

Because we know the loop has exact cost of $\lceil \frac{n}{4} \rceil \cdot \frac{n(n+1)}{2}$ steps, we can conclude that the algorithm has runtime of $\Theta(n^3)$.

Notes:

- Noticed professor is being very specific about parts of proof he is working on.
- Would it be a good idea if I sketch on paper a skeleton of proof (what needs to be worked on, what we know, and what is missing) before writing a full proof?
- How does professor create a sketch to a proof, and what strategies does he employ that a proof is neither incomplete at the end or gets stuck half way?

d. First, we need to determine the cost of inner loop.

Since the inner loop starts from $j = 0$ until $j = i - 1$, we can conclude that the inner loop has

$$i - 1 - 0 + 1 = i \tag{1}$$

iterations.

Because we know each iteration takes 1 step, we can conclude the inner loop has cost of

$$i \cdot 1 = i \tag{2}$$

steps.

Now, we need to calculate the cost of outer loop.

Because we know the outer loop runs from $i = 1$ until $i = n - 1$ with i increasing by 2^i per iteration, and because we know each iteration takes i steps, we can conclude that the loop has cost of

$$\sum_{i=\{1,2,4,8,\dots\}}^{n-1} i = \sum_{i'=0}^{\lceil \log(n-1) \rceil} 2^{i'} \tag{3}$$

steps.

Then, using geometric series $\sum_{k=0}^{n-1} r^k = \frac{1-r^n}{1-r}$, where $r \neq 1$, we can calculate

$$\begin{aligned}
\sum_{i'=0}^{\lceil \log(n-1) \rceil} 2^{i'} &= \sum_{i'=0}^{\lceil \log(n-1) \rceil - 1} 2^{i'} + 2^{\lceil \log(n-1) \rceil} && (4)\\
&= (2^{\lceil \log(n-1) \rceil} - 1) + 2^{\lceil \log(n-1) \rceil} && (5)\\
&= (2 \cdot 2^{\cdot \lceil \log(n-1) \rceil} - 1) && (6)
\end{aligned}$$

Then, because we know $2^{\cdot \lceil \log(n-1) \rceil}$ is roughly $n - 1$, we can conclude the runtime of the algorithm is $\Theta(n)$

Correct Solution:

First, we need to determine the cost of inner loop.

Since the inner loop starts from $j = 0$ until $j = i - 1$, we can conclude that the inner loop has

$$i - 1 - 0 + 1 = i \tag{1}$$

iterations.

Because we know each iteration takes 1 step, we can conclude the inner loop has cost of

$$i \cdot 1 = i \tag{2}$$

steps.

Now, we need to calculate the cost of outer loop.

Because we know the outer loop runs from $i = 1$ until $i = n - 1$ with i increasing by 2^i per iteration, and because we know each iteration takes i steps, we can conclude that the loop has cost of

$$\sum_{i=\{1,2,4,8,\dots\}}^{n-1} i = \sum_{i'=0}^{\lceil \log(n) \rceil - 1} 2^{i'} \tag{3}$$

steps.

Then, using geometric series $\sum_{k=0}^{n-1} r^k = \frac{1-r^n}{1-r}$, where $r \neq 1$, we can calculate

$$\sum_{i'=0}^{\lceil \log(n) \rceil - 1} 2^{i'} = (2^{\lceil \log(n) \rceil} - 1) \tag{4}$$

Then, because we know $2^{\cdot \lceil \log(n) \rceil}$is roughly n, we can conclude the runtime of the algorithm is $\Theta(n)$

Question 2