

Worksheet 15 Solution

March 26, 2020

Question 1

a. **Inner Loop Iterations (upper bound):** n

Inner Loop Step Size: 1

Inner Loop Steps Total: n

Outer Loop Iterations (upper bound): n

Outer Loop Step Size: 1

Outer Loop Steps Total: n

Steps Total: $n \cdot n = n^2$

Correct Solution:

Since the inner loop starts at $i+1$ and ends at $n-1$, where i represents the variable in outer loop, the inner loop has $(n-1-(i+1)+1) = n-i-1$ iterations.

Since each iteration takes 1 step, the total steps taken by inner loop is:

$$(n-i-1) \cdot 1 = (n-i-1) \tag{1}$$

Now, we will evaluate total steps taken by outer loop.

Since the outer loop starts at $i = 0$, and ends at $n - 1$, the loop runs at most n iterations.

Since each iteration takes $(n - i - 1)$ steps, the total steps of outer loop is:

$$\sum_{i=0}^{n-1} (n - i + 1) = \sum_{i=0}^{n-1} [(n - 1) - i] \quad (2)$$

$$= \sum_{i=0}^{n-1} (n - 1) - \sum_{i=0}^{n-1} i \quad (3)$$

$$= n(n - 1) - \frac{n(n - 1)}{2} \quad (4)$$

$$= \frac{n^2 - n}{2} \quad (5)$$

Then, since the last **return** statement takes 1 step, it follows that the total number of steps of this algorithm is at most $\frac{n^2 - n}{2} + 2$, or $\mathcal{O}(n^2)$.

- b. Consider the input family where none of the values in a list are the same (i.e. $[1, 2, 3, 4, 5, 6, 7, 8, 9]$).

Since all values in the input list are not matching, both the inner and the outer loop will run, giving the loops the total number of steps of $\frac{n^2 - n}{2}$.

Since the last **return** statement takes 1 step, the total number of steps of this algorithm is $\frac{n^2 - n}{2} + 1$, or $\Omega(n^2)$.

Correct Solution:

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

Since the inner loop will run without interruptions until the end, the inner loop has

$$n - 1 - (i + 1) + 1 = n - i - 1 \quad (1)$$

iterations.

Then, since the inner loop takes 1 step per iteration, the total steps taken by the inner loop is

$$(n - i - 1) \cdot 1 = (n - i - 1) \quad (2)$$

Since the **if condition** $lst[i] == lst[j]$ and the **return** statement are activated when $i = n - 2$, the outer loop will run until $i = n - 2$, where j is the variable of the inner loop and i is the variable of the outer loop.

Since the outer loop starts at 0 and ends at $n - 2$, it has

$$n - 2 + 1 = n - 1 \quad (3)$$

iterations.

Since each iteration in the outer loop takes $(n - i - 1)$ steps, the outer loop has total cost of

$$\sum_{i=0}^{n-2} (n - i - 1) = \sum_{i=0}^{n-2} (n - 1) + \sum_{i=0}^{n-2} i \quad (4)$$

$$= (n - 1)(n - 1) - \frac{(n - 2)(n - 1)}{2} \quad (5)$$

$$= \frac{(n - 1)n}{2} \quad (6)$$

Since each of the **if condition** and **return** statement has cost of 1, the total cost of algorithm is $\frac{n(n-1)}{2} + 2$, or $\Omega(n^2)$

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

Since the inner loop will run from $j = i + 1$ until the end without interruptions, the loop has

$$(n - 1) - (i + 1) + 1 = n - i - 1 \quad (1)$$

iterations.

Since the inner loop takes 1 step per iteration, the loop takes total of

$$(n - i - 1) \cdot 1 = (n - i - 1) \quad (2)$$

steps.

Now, because we know that the **if condition** and **return** statement will occur at $i = 0$, the outer loop has at most 1 iteration.

Because we know that the outer loop terminates at $i = 0$, the total cost of inner loop can be simplified to

$$(n - i - 1) = n - 1 \quad (3)$$

Since the outer loop has 1 iteration and takes $n - 1$ steps, the loop has total cost of $n - 1$.

Lastly, since each of the **if condition** and **return** statement has cost of 1, the total cost of the algorithm is

$$n - 1 + 2 = n + 1 \quad (4)$$

steps, or $\Theta(n)$.

Note

- What's the lower/upper bound of this input family? How can I find them?
- $[1, 2, 3, \dots, 1, n - 1]$ returns total cost of algorithm of n . Does it imply $[1, 2, 3, \dots, 1, n - 1]$ is in different input family than $[1, 2, 3, \dots, n - 1, 1]$?
- $g \in \mathcal{O}(f) : \exists c, n_o \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_o \Rightarrow g(n) \leq cf(n)$, where $f, g : \mathbb{N} \rightarrow \mathbb{R}^{\geq 0}$
- $g \in \Omega(f) : \exists c, n_o \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_o \Rightarrow g(n) \geq cf(n)$, where $f, g : \mathbb{N} \rightarrow \mathbb{R}^{\geq 0}$
- $g \in \Theta(f) : g \in \mathcal{O}(f) \wedge g \in \Omega(f)$

Question 2

a. Since $j = \text{len}(lst) = n$ and $i = 0$ initially, the initial value of r is

$$r = j - i \quad (1)$$

$$= n - 0 \quad (2)$$

$$= n \quad (3)$$

b. The loop terminates when $r \leq 0$.

c. Let $k \in \mathbb{N}$, and $j, i \in \mathbb{Z}$. Assume $j > i$.

We will prove the statement by separating into two cases, and combining them at the end.

Case1 ($lst[mid] < x$):

Assume $lst[mid] < x$.

Then, it follows from the fact $i = \lfloor \frac{i+j}{2} \rfloor + 1$ and $j = j$ that the value of r at $k + 1^{th}$ step is

$$r_{k+1} = j - \left(\left\lfloor \frac{i+j}{2} \right\rfloor + 1 \right) \quad (1)$$

Then,

$$r_{k+1} \leq j - \left(\left\lfloor \frac{i+j}{2} \right\rfloor \right) \quad (2)$$

$$\leq \frac{2j}{2} - \left(\left\lfloor \frac{i+j}{2} \right\rfloor \right) \quad (3)$$

$$\leq - \left(\left\lfloor \frac{i+j}{2} \right\rfloor + \frac{(-2j)}{2} \right) \quad (4)$$

$$\leq - \left(\left\lfloor \frac{i+j}{2} \right\rfloor + \frac{(-2j)}{2} \right) \quad (5)$$

$$\leq - \left(\left\lfloor \frac{i+j}{2} + \frac{(-2j)}{2} \right\rfloor \right) \quad (6)$$

by using the fact $\forall x \in \mathbb{Z}, \forall y \in \mathbb{R}, \lfloor x + y \rfloor = x + \lfloor y \rfloor$.

Then,

$$-\left(\left\lfloor \frac{i+j}{2} + \frac{(-2j)}{2} \right\rfloor\right) \leq -\left(\left\lfloor \frac{i-j}{2} \right\rfloor\right) \quad (7)$$

$$\leq -\left(\frac{i-j}{2}\right) \quad (8)$$

$$\leq \left(\frac{j-i}{2}\right) \quad (9)$$

$$\leq \frac{1}{2}r_k \quad (10)$$

Case2 ($lst[mid] > x$):

Assume $lst[mid] \geq x$.

Then, it follows from the fact $i = 1$ and $j = \lfloor \frac{i+j}{2} \rfloor$ that the value of r at $k + 1^{th}$ step is

$$r_{k+1} = \left\lfloor \frac{i+j}{2} \right\rfloor - i \quad (11)$$

$$(12)$$

Then,

$$\left\lfloor \frac{i+j}{2} \right\rfloor - i \leq \left(\frac{i+j}{2}\right) - i \quad (13)$$

by the fact $\forall x \in \mathbb{R}, \lfloor x \rfloor \leq x < 1 + \lfloor x \rfloor$.

Then,

$$\left(\frac{i+j}{2}\right) \leq \frac{i+j}{2} - \frac{2i}{2} \quad (14)$$

$$\leq \frac{j-i}{2} \quad (15)$$

$$\leq \frac{1}{2}r_k \quad (16)$$

Then, it follows from proof by cases that the statement $r_{k+1} \leq \frac{1}{2}r_k$ is true.

Notes:

- External properties of ceiling and floor

1. $\forall x \in \mathbb{R}, 0 \leq x - \lfloor x \rfloor < 1$
2. $\forall x \in \mathbb{R}^{\geq 0}, x \geq 4 \Rightarrow (\lfloor x \rfloor)^2 \geq \frac{1}{2}x^2$
3. $\forall x \in \mathbb{R}^{\geq 0}, x \geq 4 \Rightarrow \frac{1}{2}x^2 \geq 2x$
4. $\forall x \in \mathbb{Z}, \forall y \in \mathbb{R}, \lfloor x + y \rfloor = x + \lfloor y \rfloor$

d. Let $n \in \mathbb{N}, k \in \mathbb{Z}^{\geq 0}$ and $r = j - i$. Assume $j > i$.

Because we know $r_{k+1} \geq \frac{1}{2}r_k$, we can conclude that the size of r at $k+1^{th}$ step is

$$\left\lfloor \frac{n}{2^{k+1}} \right\rfloor \quad (1)$$

Then, because we know the loop terminates when $0 < \frac{n}{2^k} < 1$, we can also conclude that loop terminates when

$$2^{k+1} > n \quad (2)$$

$$k+1 > \log n \quad (3)$$

$$k+1 > \lfloor \log n \rfloor \quad (4)$$

Since $\lfloor \log n \rfloor$ is the lower bound of loop termination, $\lfloor \log n \rfloor + 1$ is the value of iteration that results in termination.

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

$$(\lfloor \log n \rfloor + 1) \cdot 1 = \lfloor \log n \rfloor + 1 \quad (5)$$

steps.

Since the **return** statement occurs at the end and since it has cost of 1, the total cost of algorithm is at most $\lfloor \log n \rfloor + 2$, or $\mathcal{O}(\log n)$.

Note

- A through understanding of algorithm is required. Maybe try few examples before proof?
- iteration is in terms of k, but we want in terms of n.

e. Let $n \in \mathbb{N}, k \in \mathbb{Z}^{\geq 0}$, $lst = [1, 2, 3, 4, 5, \dots, n - 1]$, and $x = 0$.

Because we know $r_{k+1} \geq \frac{1}{2}r_k$, we can conclude that the value of r at $k+1^{th}$ step is

$$\left\lfloor \frac{n}{2^k} \right\rfloor \quad (1)$$

Because we know the loop terminates when $j \leq i$, or when $r \leq 0$, we can also conclude that, in terms of equation 1, the termination occurs when

$$2^k > n \quad (2)$$

$$k > \log n \quad (3)$$

Since $\log n$ represents the lower bound of loop termination, the loop terminates when $k = \log n + 1$.

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

$$(\log n + 1) \cdot 1 = \log n + 1 \tag{4}$$

steps.

Since **return** statement occurs at the end and since it has cost of 1, the total cost of algorithm is $\log n + 2$, or $\Omega(\log n)$.

Since $\log n + 2$ is the same for both the upper bound and the lower bound, $\Theta(\log n)$ is true.