CS-202

ASSIGNMENT-01

S. SAMARTH REDDY
B19109
GROUP - 25

---

① (1) Yes, it is possible that an algorithm takes $O(n^2)$ worst-case time, and $O(n)$ on some inputs, because there is no requirement for the function in big-oh to be tight. Also the given big-oh bound refers to worst-case input, and some inputs may not take worst-case time.

(2) Yes, a said in first part, the function in the big-oh does not have to be tight. A function may have tight bound of $O(n)$ but, it is not wrong to say that it has $O(n^2)$ complexity for all inputs.

(3) Yes, it is possible an algorithm takes $\Theta(n^2)$ worst-case time, but $O(n)$ on some inputs. The $\Theta(n^2)$ worst-case time algorithm may take for some inputs, but there may also be inputs such that it takes lesser time like $O(n)$ or even lesser.

(2) I have used the following identities for solving the question. It is similar to the binary GCD algorithm.

(1) $gcd(0, v) = v$

This is true because every number divides 0 and v is the largest divisor of v. Similarly, $gcd(u, 0) = u$

(2) $gcd(2u, 2v) = 2 \times gcd(u, v)$

This is because if 2 is a factor of both numbers, then it will also be a factor of GCD.

(3) $gcd(2u, v) = gcd(u, v)$, if v is odd

and

$gcd(u, 2v) = gcd(u, v)$, if u is odd

If 2 is a factor of only one number, then it won't be present in gcd so can be safely dropped.

(4) $gcd(u, v) = gcd(|u-v|, \min(u, v))$

This is similar to the Euclidean algorithm of gcd.

Proof: Let $b = aq + r$.

To Prove: $gcd(b, a) = gcd(a, r)$

Let $m = gcd(b, a)$ and $n = gcd(a, r)$

- m divides both b and a, so it must also divide $r = b - aq$. $\therefore$ m is common divisor of a and r. and $m \leq n$ ($\because n = gcd(a, r)$)
- Likewise, n divides both a and r, so it must divide $b = aq + r$. $\therefore n \leq m$

Since, $m \leq n$ and $n \leq m \Rightarrow n = m$.

Using these identities repititively, we keep decreasing either u, or v till one of them becomes zero, and then we use the first identity.

Hence, the algo always finds the right answer.

(3)

(1) $T(n^2) = 7T(n^2/4) + cn^2$

Substitute $n^2 = k$

$T(k) = 7T(k/4) + ck$

We can use Master Method to solve this recurrence as it is of the form

$T(k) = aT(k/b) + f(k)$, where $a = 7, b = 4$

$f(k) = ck$

Clearly, $f(k) = O(k)$

$= O(k^{\log_4 7 - \epsilon})$, where $\epsilon > 0$

Therefore, $T(k) = \Theta(k^{\log_4 7})$ or $\boxed{T(n) = \Theta(n^{\log_4 7})}$

---

(2) $T(n) = nT(\sqrt{n})$ and $T(2) = 4$ [Given]

Let $n = 2^{2^i}$

$T(n) = nT(n^{1/2})$

$= n \cdot n^{1/2} T(n^{1/4}) = n \cdot n^{1/2} T(n^{1/2^2})$

$= n^{1 + 1/2 + 1/2^2} \cdot T(n^{1/2^3})$

For a general integer j,

$T(n) = n^{1 + 1/2 + \dots + 1/2^{j-1}} T(n^{1/2^j})$

$= n^{\left(\frac{1 - (1/2)^j}{1/2}\right)} T(n^{1/2^j})$

$$= n^{2(1-1/2^j)} \cdot T(n^{1/2^j})$$

let $j = i$ and Substitute $n = 2^{2^i}$

$$T(n) = 2^{2^i \times 2 - 2^i \times 2 \times \frac{1}{2^i}} \cdot T\left(2^{2^i \times \frac{1}{2^i}}\right)$$

$$= 2^{2^i \times 2 - 2} \cdot T(2)$$

$$= 4 \times \frac{n^2}{4}$$ $\left[T(2) = 4 \text{ and } n = 2^{2^i}\right]$

$$T(n) = n^2 = \Theta(n^2)$$

$$\therefore \boxed{\Theta(n^2) = n^2 = T(n)}$$

---

(3) $T(n) = T(n/2) + 2T(n/4) + 3n/2 \quad \forall n > 3$,

$T(1) = 0, \ T(2) = 2$

we use the <u>Akra Bazzi Method</u>

The recursion is of the form

$$T(n) = \sum_{i=1}^{k} a_i T(n/b_i) + g(n) \qquad \forall\ n > n_0$$

$a_i > 0, \ b_i > 1$

$$g(n) = 3n/2$$

$\Rightarrow$ find p Such that $\sum_{i=1}^{k} a_i/(b_i)^p = 1$

$$1 \cdot \left(\frac{1}{2}\right)^p + 2 \cdot \left(\frac{1}{4}\right)^p = 1$$

$p = 1$ Satisfies this equation

$$\therefore T(n) = \Theta\left(n^p \left(1 + \int_1^n \frac{g(u)}{u^{p+1}} \cdot du\right)\right)$$

$$= \Theta\left(n\left(1 + \int_1^n \frac{3}{2} \frac{u}{u^{1+1}} \cdot du\right)\right)$$

$$= \Theta\left(n\left(1 + \int_1^n \frac{3}{2} \frac{du}{u}\right)\right)$$

$$= \Theta\left(n\left(1 + \frac{3}{2}\left[\log n\right]_1^n\right)\right)$$

$$= \Theta\left(n + \frac{3}{2} n \log n\right)$$

$$\therefore \quad \boxed{T(n) = \Theta(n \log n)}$$

---

(4) $T(n) = 4T(n/2) + n^3$ and $T(1) = 1$

Using master method as the recursion is of the form $T(n) = aT(n/b) + f(n)$

where $a = 4$, $b = 2$, $f(n) = n^3$

$$f(n) = \cancel{\Theta}\, n^3$$

$$= \Omega(n^3) = \Omega(n^{\log_2 4 + \epsilon}), \quad \epsilon = 1 > 0$$

Now, checking Condition:

$$a f(n/b) \le c f(n) \quad \text{for Some } c < 1$$

$$\frac{4n^3}{8} \le cn^3$$

$$c \ge 1/2$$

Condition is satisfied for $c = 1/2 < 1$

By master method

$$T(n) = \Theta(n^3)$$

(5) $T(n) = T(n/2) + c\log n$

$= T(n/2^2) + c\log(n/2) + c\log(n)$

$= T(n/2^3) + c\log(n \cdot n/2 \cdot n/2^2)$

for a general integer i,

$T(n) = T\left(\frac{n}{2^i}\right) + c\log\frac{n^i}{2^0 \cdot 2^1 \cdot 2^2 \cdots 2^{i-1}}$

$= T\left(\frac{n}{2^i}\right) + c\log\frac{n^i}{2^{(i-1)i/2}}$

$= T(n/2^i) + c\,i\log n - c\frac{i(i-1)}{2}\log 2$

we know, $2^{\log n} = n$ $[\log n = \log_2 n]$

Put $i = \log n$

$T(n) = T(1) + c(\log n)^2 - c\frac{(\log n)(1+\log n)}{2}$

$= T(1) + c\frac{(\log n)^2}{2} - c\frac{\log n}{2}$

$\Rightarrow \boxed{T(n) = O(\log^2 n)}$

④ Given functions:

$$\frac{n^{1.2}}{\log n}, \quad n^2, \quad n\log n, \quad 1.1^n, \quad 0.9^n, \quad \log^3 n$$

We pair-wise compare these functions using limits and $n \to \infty$.

- $\lim_{n\to\infty} 0.9^n = 0$ $\quad \therefore$ 0.9^n is the Smallest

Now, let's compare $\frac{n^{1.2}}{\log n}$ and n^2

- $$\lim_{n\to\infty} \frac{n^{1.2}}{\log n} \times \frac{1}{n^2} = \lim_{n\to\infty} \frac{1}{n^{0.8}\log n} = 0$$

$$\therefore \quad 0.9^n < \frac{n^{1.2}}{\log n} < n^2 \longrightarrow ①$$

- Now, checking n^2 and $n\log n$

$$\lim_{n\to\infty} \frac{n^2}{n\log n} = \lim_{n\to\infty} \frac{n}{\log n} = \lim_{n\to\infty} \frac{1}{1/n} \quad [\text{L Hospital}]$$

$$= \infty$$

$$\therefore \quad n\log n < n^2 \longrightarrow ②$$

- Comparing $\frac{n^{1.2}}{\log n}$ and $n\log n$

$$\lim_{n\to\infty} \frac{n^{1.2}}{\log n} \times \frac{1}{n\log n} = \lim_{n\to\infty} \frac{n^{0.2}}{(\log n)^2} = \lim_{n\to\infty} \frac{0.2\times n}{n^{0.8}\, 2\log n}$$

$$= \lim_{n\to\infty} \frac{0.1\times n^{0.2}}{\log n}$$

$$= \lim_{n\to\infty} \frac{0.02\times n}{n^{0.8}} = \infty$$

③

From ①, ② and ③;

$$\therefore \quad 0.9^n < n\log n < \frac{n^{1.2}}{\log n} < n^2$$

- Now Comparing $n \log n$ and $\log^3 n$

$$\lim_{n\to\infty} \frac{n \log n}{\log^3 n} = \lim_{n\to\infty} \frac{n}{\log^2 n} = \lim_{n\to\infty} \frac{1 \times n}{2 \times \log n}$$

$$= \lim_{n\to\infty} \frac{n}{2} = \infty$$

$$\therefore\ 0.9^n < \log^3 n < n \log n < \frac{n^{1.2}}{\log n} < n^2$$

- Finally comparing n^2 and 1.1^n

$$\lim_{n\to\infty} \frac{(1.1)^n}{n^2} = \lim_{n\to\infty} \frac{(1.1)^n \log 1.1}{2n} = \lim_{n\to\infty} \frac{(1.1)^n (\log 1.1)^2}{2}$$

$$= \infty$$

$\therefore$ we get the ascending order of these funct^n:

$$\Rightarrow \boxed{0.9^n < \log^3 n < n \log n < \frac{n^{1.2}}{\log n} < n^2 < (1.1)^n}$$

---

(5) If each operation takes $O(1)$ time, then the run time of given function can be estimated by counting the total number of operations, which is equal to the returned value of r.

```
function XYZ(n):
r = 0;

    for i = 1 to n :
        for j = 1 to i :
            for k = j to (i+j) :
                for l = 1 to (i+j-k):
                    r = r+1;

return (r)
```

$$r = \sum_{i=1}^{n} \sum_{j=1}^{i} \sum_{k=j}^{(i+j)} \sum_{r=0}^{(i+j+k)} (1)$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{i} \sum_{k=j}^{(i+j)} (i+j-k)$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{i} \left[(i+j)[i+j-j] - \left(\sum_{k=1}^{i+j} k - \sum_{k=1}^{j} k \right) \right]$$

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

$$= \sum_{i=1}^{n} \sum_{j=1}^{i} \left[\frac{i^2}{2} - \frac{i}{2} \right]$$

$$= \sum_{i=1}^{n} i \left(\frac{i^2}{2} - \frac{i}{2} \right) = \sum_{i=1}^{n} \frac{i^3}{2} - \frac{i^2}{2}$$

$$= \frac{1}{2} \left[\left[\frac{n(n+1)}{2} \right]^2 - \frac{n(n+1)(2n+1)}{6} \right]$$

$$= \frac{3n^4 + 2n^3 - 3n^2 - 2n}{24} = \frac{n^4}{8} + \frac{n^3}{12} - \frac{n^2}{8} - \frac{n}{16}$$

$$f(n) = \frac{n^4}{8} + \frac{n^3}{12} - \frac{n^2}{8} - \frac{n}{16} \leq \frac{n^4}{8} + \frac{n^4}{12} + \frac{n^4}{8} + \frac{n^4}{16} = \frac{5}{12} n^4$$

Thus $f(n) = O(n^4)$. because $\frac{f(n)}{(n^4)} \leq M = 5/12$