FARHAN AKBOR KHAN

ID: 20301230

Section: 08

Lab: 01

CSE'221

2.

Implentation : 1

$T(n)$ be the function which refers to fibonacci - 1.

Here, $T(0) = 1$; $T(1) = 1$; $T(2) = 1$

$\therefore T(n) = T(n-1) + T(n-2) + c$

we can assume,

$\therefore T(n-1) \approx T(n-2)$

$\therefore T(n) = 2\,T(n-1) + c$

$= 2\left(2\,T(n-2)\right) + 2c + c$

$= 4\,T(n-2) + 3c$

$= 2^2\,T(n-2) + 3c$

$$= 2^3 T(n-3) + 7C$$

So, $n - k = 0$

$$k = n$$

$$\therefore T(0) = 1$$

$$\Rightarrow 2^k T(n-k) + (2^k - 1)C$$

$$\Rightarrow 2^n \cdot 1 + (2^n - 1)C$$

$$\Rightarrow 2^n + 2^n C - C$$

$$\therefore O(2^n) \quad \text{(Ans)}$$

Implementation : 2

In this case, we used memoization to avoid landy calculation. It means for any problems, we will not consider any subproblems without one.

The tree = $n \rightarrow n-1 \rightarrow n-2 \rightarrow n-3$
$\hookleftarrow n-4 \ldots\ldots$

Which gives us a linear runtime which is $O(n)$.

(Ans).

Problem: 4

```
for i = 0 to n-1                          → O(n)
    for j = 0 to n-1                      → O(n)
        for k = 0 to n-1                  → O(n)
            C[i,j] += A[i,k] * B[k,j]
        end for
    end for
end for
```

$\therefore T(n) = O(n * n * n)$

$= O(n^3)$

(Ans)

Problem : 5

(i) $T(n) = T(n/2) + n - 1$, $T(1) = 0$

Ans:

Imagine, $n = 2^m$

$\therefore T(2^m) = T(2^{m-1}) + 2^m - 1$

$= T(2^{m-2}) + 2^{m-1} - 1 + 2^m - 1$

$= T(2^{m-3}) + 2^{m-2} - 1 + 2^{m-1} - 1 + 2^m - 1$

$= T(2^{m-3}) + 2^{m-2} + 2^{m-1} + 2^m - 1 - 1 - 1$

$= T(2^{m-m}) + 2^{m-3} + 2^{m-2} + 2^{m-1} + 2^m - 1 - 1 - 1$

$= 2^0 + \ldots + 2^{m-2} + 2^{m-1} + 2^m - m$

$= 2^{m-1} - 1 - m$

$= 2^n \cdot 2^{-1} - 1 - n$

$= 2^n - n$

$= n - n \quad [n = 2^n]$

$= n - \log_2 n$

$= O(n) \quad [\text{Ignore } \log_2 n]$

OR using master-theorem;

$T(n) = T(n/2) + n - 1, \quad T(1) = 0$

$a = 1, \quad b = 2, \quad c = 1, \quad k = 1$

$\therefore b^k = 2^1$

$\therefore 2 > a$

$\therefore O(n^1) = O(n)$

(Ans)

(ii)

$T(n) = T(n-1) + n - 1$, $T(1) = 0$

Ans:

$T(n) = T(n-1) + n - 1$

$= T(n-2) + (n-1) + n - 1 - 1$

$= T(n-3) + (n-2) - 1 + (n-1) + n - 1 - 1$

$= T(n-n+1) + (n-n+2) + \ldots\ldots$

$\ldots\ldots + (n-3) + (n-2) + (n-1) + n$

$= 0 + 1 + 2 + 3 + 4 \ldots + n$

Arithmatic Series : $\frac{n(n+1)}{2}$

$= \frac{n^2+n}{2}$ $\therefore O(n^2)$

(Ans)

(iii)

$$T(n) = T(n/3) + 2T(n/3) + n$$

$$\Rightarrow T(n) = 3T(n/3) + n$$

Here, $a = 3, \; b = 3, \; c = 1, \; k = 1$

$$\therefore b^k = 3^1 = 3$$

$$\therefore 3 = a = 3 = b^k$$

$$\therefore O(n^k \log n)$$

$$\Rightarrow O(n^1 \log(n))$$

$$= O(n \log n)$$

(Ans)

(iv)

$T(n) = 2T(n/2) + n$

Ans: Using Master-theorem →

$a = 2, \quad b = 2, \quad c = 1, \quad k = 2$

$\therefore b^k = 2^2 = 4$

$\therefore 4 > a$

$\therefore O(n^k) = O(n^2)$

∴ For the worst case complexity will be n^2.

(Proved)