CSA0666 - Design and Analysis of Algorithm

— Assignment - 01 —

By:
E. Sachin
192324045
06/06/2024

To Solve the recurrence relation $x(t)$

a) $x(n) = x(n-1) + 5$ for $n > 1$ $x(1) = 0$

Soln

$n = 2$

$x(2) = x(2-1) + 5$

$= x(1) + 5 = 0 + 5 = 5$

$n = 3$

$x(3) = x(3-1) + 5$

$= x(2) + 5 = 5 + 5 = 10$

$n = 4$

$x(4) = x(4-1) + 5$

$= x(3) + 5 = 10 + 5 = 15$

$n = 5$

$x(5) = x(5-1) + 5$

$= x(4) + 5 = 15 + 5 = 20$

The $x(n)$ increases by 5 for each increment of n

$$x(n) = 5(n-1)$$

$x(k) = 5(k-1)$ is true for some $k \geq 1$ we need to show that

$$x(k+1) = x(k) + 5$$

$$x(k) = 5(k-1)$$

$$x(k+1) = 5(k-1) + 5 = 5k - 5 + 5 = 5k$$

$x(n) = 5(n-1)$ for all $n \geq 1$

(b) $x(n) = 3x(n-1)$ for $n > 1$ $x(1) = 4$

Let $n = 2$

$$x(2) = 3x(1) = 3(4) = 12$$

$n = 3$

$$x(3) = 3x(3-1)$$
$$= 3x(2) = 3(12) = 36$$

$n = 4$

$$x(4) = 3x(4-1)$$
$$= 3x(3) = 3(36) = 108$$

$n = 5$

$$x(5) = 3x(5-1)$$
$$= 3x(4) = 3(108) = 324$$

$x(n)$ is multiplied by 3 for each increment

n

$$x(n) = 3^k x(n-k)$$

$n - k = 1$

$k = n - 1$

$$= 3^{n-1} x(\cancel{n} - \cancel{n} - 1)$$
$$= 3^{n-1} \cdot 4$$

$$O(3^n)$$

$x(n) = x(n/2) + n$ for $n > 1$ $x(1) = 1$ (Solve for $n = 2^k$)

$x(n) = 1 \times (n/2) + n$

$T(n) = a\,T(n/b) + f(n) \rightarrow$ master theorm

so $a = 1$
$b = 2$
$f(n) = n$

step 1 :- $\log_2 1 \Rightarrow \log_a b$

$f(n) = n^k + \log_n p$

where in question no log value

$p = 1$ $k = 1$ as $n^1 \rightarrow n$

case 3 $\log_2 1 < k$

$0 < 1$

check $p \geq 0$ $n^k \log_n p$

$p \leq 0$ n^k

$1 \geq 0$ $n \log n^1 \Rightarrow O(n)$

d) $x(n) = x(n/3) + 1$ for $n > 1$ $\quad x(1) = 1$

$$x(n) = x(n/3) + 1 \quad \text{—①}$$

$$x(n/3) = x\left(\frac{n/3}{3}\right) + 1$$

$$= x(n/3^2) + 1 \quad \text{—②}$$

① $\quad x(n) = (x(n/3^3) + 1) + \cancel{\ }$

$$= x(n/a) + 1$$

② $\quad x(n) = x(n/a) + 1 + 2$

$$= x(n/9) + 3$$

Observed Series

$$x(n) = x\left(\frac{n}{3^i}\right) + i \qquad x(1) = 1$$

$$\frac{n}{3^i} = 1$$

apply log $\longrightarrow$ $\log_2 n = \log_3 i$

$\log_2 n = i \log_2 3$

$$x(n) = x\left(\frac{n}{3\log_3 n}\right) + \log_3 n$$

$n = 3k$

$$3(3k) = x\left(\frac{3k}{3\log_3 k}\right) + \log_3 k$$

$$= x\left(\frac{3}{n}\right) + \log_3 n$$

$$= x(1) + \log_3 n$$

$x(3k) \Rightarrow \log_3 k + 1$

$x(n) \Rightarrow \log_3 n + 1$ $\Big\}$ $O(\log n)$

2) Evaluate the following recurrence completely

i) $T(n) = T(n/2) + 1$ when $n = 2k$ for all $k \geq 0$

Sol

$$T(n) = T\left(\frac{n}{2}\right) + 1 \quad — ①$$

$n = n/2$

$$T(n/2) = T(n/4) + 1 \quad — ②$$

Sub ② in ①

$$T(n) = T(n/4) + 1 + 1$$

$$T(n) = T(n/4) + 2 \quad — ③$$

$n = n/4$

$$T(n/4) = T(n/8) + 1 \quad — ④$$

Sub eq ④ in ③

$$T(n) = T(n/8) + 2 + 1$$

$$= T(n/8) + 3 \quad — ⑤$$

$$T(n) = T\left(\frac{n}{2^k}\right) + k$$

$n = 2^k$ then $\frac{n}{2^k} = 1$

$$T(n) = T(1) + k \qquad T(1) = 1$$

$$T(n) = 1 + k \quad — ⑥$$

Taking logn of both side

$$k = \log_2 n$$

sub k in eq (6)

$$T(n) = 1 + \log_2 n$$

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

$$T(n/3) = T(n/3) + T(2n/3^2) + C\left(\frac{2n}{3^2}\right)$$

$$T(2n/3) = T(2n/3^2) + T\left(\frac{4n}{3^2}\right) + C.\frac{2n}{3^2}$$

$$T(n) = T(n/3^2) + T\left(\frac{2n}{3^2}\right) + T\left(\frac{2n}{3^2}\right) + \left(\frac{4n}{3^2}\right) + \frac{C.n}{3^2} + \frac{C.2n}{3^2}$$

$$T(n/3^2) = T(n/3^3) + T\left(\frac{2n}{3^3}\right)$$

$$T(2n/3^2) = T\left(\frac{2n}{3^3}\right) + T\left(\frac{4n}{3^2}\right)$$

$$T\left(\frac{4n}{3^2}\right) = T\left(\frac{4n}{3^3}\right) + T\left(\frac{8n}{3^3}\right)$$

$$T(n) = T(n/3^3) + T\left(\frac{2n}{3^3}\right) + T\left(\frac{2n}{3^3}\right) + T\left(\frac{2n}{3^3}\right)$$

$$+ T\left(\frac{4n}{3^3}\right) + T\left(\frac{4n}{3^3}\right) + T\left(\frac{8n}{3^3}\right) \ldots$$

Consider the following recursion algorithm

min1(A[0.....n-1])

if n=1 return A[0]

Else temp= Min1(A[0...n-2])

if temp<= A[n-1] return temp

Else
return A[n-1]

a) what does this algorithm compute?

Sol In this algorith computes the minimum value of the Array

- If the size of array n is 1, it return the only element A[0]
- Otherwise, it recursively calls itself to find the minimum element in subarray A[0...n-2].
- It then compares the result of this recursive call with the last element of the array A[n
- Finally, it returns the smaller of the two values

b) setup a recurrence relation for the algorithm basic operation count and solve it.

Sol

Sol

* The comparison temp <= A[n-1]
* The recursive call itself

Let's denote $T(n)$

if $n=1$, the algorithm return $A[0]$ with no comparison

$$T(n) = T(n-1) + c \quad \text{—} ①$$

$n = n-1$

$$T(n-1) = T(n-1-1) + 1$$
$$= T(n-2) + 2C \quad \text{—} ②$$

~~n=n-2~~

$$\cancel{T(n-2) = T(n-3) + 1}$$

Sub ② in ①

$$T(n) = T(n-2) \longrightarrow ③$$

Find $T(n-2) = T(n-2-1) \Rightarrow T(n-3)$ — ④

Sub ④ in ③

$$T(n) = T(n-3) + 3C$$

by the seri

$$T(n-2) \cdot T(n-3) \qquad T(1) = 1$$

$$T(n) = T(n-k) + kC$$

$$\left.\begin{matrix} n-k=0 \\ n=k \end{matrix}\right\} \quad \begin{matrix} n-k=1 \\ n-1=k \end{matrix}$$

$$T(n) = T(1) + (n-1)C \cdot 1 + (n-1)C \rightarrow O(n)$$

nalyze the order of growth

$F(n) = 2n^2 + 5$ and $g(n) = 7n$ use the $\Omega(g(n))$ notat.

$F(n) \geq C \cdot g(n)$ for all $n \geq n_0$

$F(n) = 2n^2 + 5$

$g(n) = 7n$

$2n^2 + 5 \geq C \cdot 7n$ 5 is Constant so negligible

$2n^2 \geq 7Cn$

÷ by n both side

$2n \geq 7C$ Divid by 2 on both side

$n \geq \frac{7C}{2}$

$F(n) = 2n^2 + 5 \in \Omega(7n)$