7/08/24

Design Analysis and Algorithm

S.R.Thirupugazh

(1)

1) Solve the following recurrence relations:

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

$x(n) = x(n-1)+5 \rightarrow (1)$

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

$= x(n-2)+5 \rightarrow (2)$

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

$= x(n-3)+5 \rightarrow (3)$

Sub eqn (3) in (2)

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

$= x(n-3)+10 \rightarrow (4)$

Sub eqn (4) in (1)

$x(n) = x(n-3)+10+5$

$= x(n-3)+15$

for some k,

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

$n-k=1$

$n-1=k$

Eqn (5)

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

$x(n) = 0+5n-5$

$O(n)$

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

$x(n) = 3x(n-1) \rightarrow (1)$

$x(n-1) = 3x(n-1-1) = 3x(n-2)$

$x(n-2) = 3x(n-3)$

③

Sub eq ③ in ②,

$x(n-1) = 3[3x(n-3)]$

$x(n-1) = 9x(n-3) \to ④$

Sub eq ④ in ①,

$x(n) = 3[9x(n-3)]$

$x(n) = 27x(n-3)$

At some k

$x(n) = 3^k x(n-k) \to ⑤$

$n-k=1$

$k=n-1$

For ⑤ => $x(n) = 3^{n-1} x(1)$

$= 3^{n-1} \cdot 4$

$= 3^n \cdot 3^{-1} \cdot 4$

$= 3^n$

time complexity $= O(3^n)$

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

$x(n) = x(n/2) + c \to ①$

$x(n/2) = x(n/4) + c \to ②$

$x(n/4) = x(n/8) + c \to ③$

sub ② in ①

$x(n) = x(n/4) + c + c$

$x(n) = x(n/4) + 2c \to ④$

$= x(n/2^2) + 2c$

sub ③ in ④

$x(n) = x(n/8) + c + 2c$

2)

$x(n) = x(n/2^3) + 3c$

$x(n) = x(n/2^k) + kc$

$n = 2^k, \ x(1) = 1$

$x(n) = x(n/n) + kc$

$x(n) = 1 + kc$

$x(n) = 1 + \log n \cdot c$

Time complexity $= O(\log n)$

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

(solve for $n = 3^k$)

$x(n) = x(n/3) + 1 \rightarrow (1)$

$x(n/3) = x(n/4) + 1 \rightarrow (2)$

$x(n/4) = x(n/27) + 1 \rightarrow (3)$

sub (2) in (1)

$x(n) = x(n/a) + 2 \rightarrow (4)$

sub (3) in (4).

$x(n) = x(n/27) + 3 \rightarrow (5)$

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

$x(n) = x(n/3^k) + k$

$x(n) = x(n/3^k) + k$

$= x(n/n) + k$

$= x(1) + k$

$= 1 + k$

$x(n) = \log n$

Time complexity $= O(\log n)$

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

where $n = 2^k$ for all $k \geq 0$

$n = 2^k$

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

$n = k-1$

$T(2^{k-1}) = T\left(\frac{2^{k-1}}{2}\right) + 1 = T(2^{k-2}) + 1$

$n = k-2$

$T(2^{k-2}) = T\left(\frac{2^{k-2}}{2}\right) + 1 = T(2^{k-3}) + 1$

$T(2^1) + T(2^0) + 1$

$n = 2^k \Rightarrow k = \log_2 n$

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

Since,

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

$T(2^k) = 1 + k$

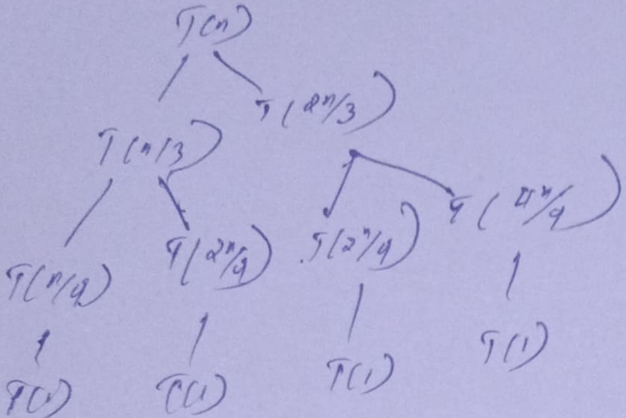
$T(n) = 1 + \log_2 n$

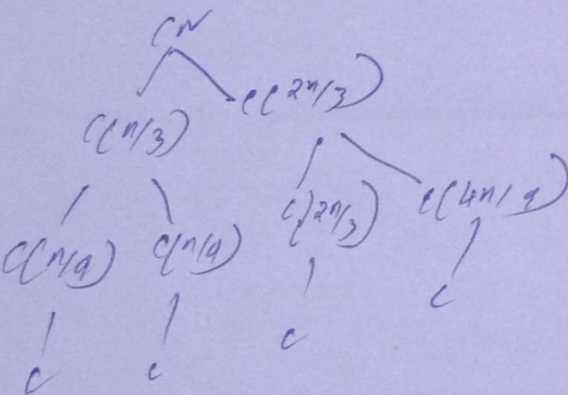
Time complexity $= O(\log n)$

(3)

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

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




Consider following algorithm

Min1 (A [0 .. n-1])

If n=1 return A[0]-1

Else

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

If temp ≤ A[n-1] return temp

else

return A[n-1]-n-1

(4)

a) This algorithm computes minimum element in an array A of size n.

b) So, $T(n) = T(n-1) + 1$ when $n > 1$ (one comparison of every step except $n = 1$)

$T(1) = 0$ (no compare when $n=1$)

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

$= 0 + (n-1)$

$= n-1$

$= O(n)$

4. $F(n) = 2n^2+5$ and $g(n) = 7n$ use $(a+b)^2 = a^2 + 2ab + b^2$

$F(n) = 2n^2+5$ $\qquad$ $F(n) \geq c \cdot g(n)$

$c \cdot g(n) = 7n$

$n=1$

$F(1) = 2(1)^2+5 = 7$

$g(1) = 7$

$n=2$

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

$= 8+5 = 13$

$g(2) = 7 \times 2 = 14$

$n=3$

$F(3) = 2(3)^2$

$= 18+5$

$= 23$

$g(3) = 21$

F(n) is always greater than or equal to g(x) when n value is greater or equal to 3.

$$F(n) = \Omega(g(n))$$

F(n) is grows more than g(n) from below asymptotically