DESIGN AND ANALYSIS OF ALGORITHM

ANALYTICAL QUESTION - ASSIGNMENT

NAME: VIJAYALAKSHMI BAI.K

REGNO: 192321031

COURSE
COD: CSA0669

) Solve the following recurrence relation

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

at $n = 1$; $x(1) = 0$ (Given)

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

$= x(1) + 5$

$= 0 + 5 = 5$

$x(2) = 5$

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

$= x(2) + 5$

$= 5 + 5 = 10$

$x(3) = 10$

at $n = 4$; $x(4) = x(4-1) + 5$

$= x(3) + 5$

$= 10 + 5$

$x(4) = 15$

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

difference $(d) = 5$

$x(n) = x(1) + (n-1) \cdot d$ { formula for n-th term to find general form of $x(n)$

Here, $x(1) = 0, d = 5$

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

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

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

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

$n = 1$: $x(1) = 4$ (Given)

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

$= 3x(1)$

$= 3 \times 4 = 12$

$x(2) = 12$

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

$= 3x(2)$

$= 3 \times 12$

$x(3) = 36$

$n = 4$; $x(4) = 3x(4-1)$

$= 3x(3)$

$= 3 \times 36$

$x(4) = 108$.

∴ $x(n)$ Obtained by multiplying the previous term by 3

Ratio : 3

$$x(n) = x(1) \cdot r^{n-1}$$

Here, $x(1) = 4$, $r = 3$

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

Ans: $x(n) = 4 \times 3^{n-1}$

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

$n = 2^k$

$n = 1$; $x(1) = 1$

$n = 2$: $x(2) = x(2/2) + 2 = x(1) + 2$

$= 1 + 2 = 3$

$x(2) = 3$

$n = 4$: $x(4) = x(4/2) + 4 = x(2) + 4$

$= 3 + 4 = 7$

$x(4) = 7$

$n = 8$; $x(8) = x(8/2) + 8 = x(4) + 8$

$x(8) = 7 + 8 = 15$

$n = 16$: $x(16) = x(16/2) + 16 = x(8) + 16$

$= 15 + 16 = 31$

$x(16) = 31$

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

$x(2^k) = 2^{k+1} + 1$

$\because 2^k = n.$

$x(n) = x(2^k) = 2^{(\log_2 n)+1} - 1$

$= 2 \cdot 2^{\log_2 n} - 1$

$= 2n - 1$

Ans: $x(n) = 2n - 1$

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

$x(1) = 1$ (Given)

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

$= 1 + 1 = 2$

$x(3) = 2$

$n = 9$ $x(9) = x(9/3) + 1$

$= x(3) + 1$

$x(9) = 2 + 1 = 3$

$n = 27$ $x(27) = x(27/3) + 1$

$= x(9) + 1$

$= 3 + 1$

$= 4$

$x(n) = 1 + \log_3 n$ hold true for $n = 3^k$

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

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

2) Evaluate the following recurrences completely

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

Assume $n = 2^k$ i.e. $k = \log n$

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

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

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

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

$$= [T(2^{k-3}) + 1] + 2$$

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

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

$$= T(2^0) + k$$

$$= T(1) + k$$

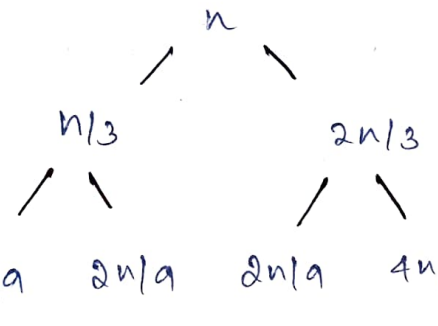
If $T(1) = 1$, we get

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

i.e. $T(n) = \log n + 1$

Thus, we get $T(n) = \Theta(\log n)$

ii) $T(n) = T(n/3) + T(2n/3) + cn$, where 'c' is a constant and 'n' is the input size



$T(n) =$ "+n" = sum of the all numbers in this tree

length $= \log_3 n$

$T(n) \geq n \log_3 n$

[$\because T$ is $\Omega(n \log n)$

depth $= \log_{3/2} n$

$T(n) \leq n \log_{3/2} n$

T is $\Theta(n \log n)$

3) Consider the following recursion algorithm

```
Mini[A[0 .... n-1])
if n-1 return A[0]
Else temp = Mini(A[0 ... n-2])
     if temp <= A[n-1] return temp
Else
     Return A[n-1]
```

a) What does this algorithm Compute?

This algorithm computes the minimum value in an Array A.

1. Best case (n=1):

if n=1, only one element. It returns the A[0] as it's the minimum value in a single-element array.

2 Recursive case (n>1):

=> If n>1, creates the temporary variable (temp)

=> Call recursively (A[0 to n-2) : first n-1 elements

=> Comparing temp with last element (A[n-1]).

```
if temp ≤ A[n-1]
     return temp
else
     return A[n-1].
```

b) setup a recurrance relation for the algorithm basic Operation count and solve it

Base case : $T(1) = C_1$ [C_1 is constant -> return single element

Recursive case : $T(n) = T(n-1) + C_2$ [C_2 -> constant representing the basic operations for comparison and assignment

final solution :

$$T(n) = C_2 * n^2 + (C_1 - C_2)$$

$$T(n) = O(n^2)$$

A) Analyze the order of growth.

P) $F(n) = 2n^2 + 5$ and $g(n) = 7n$. Use the $\Omega(g(n))$ notation.

As n grows, $2n^2$ grows much faster than $7n$.

$$F(n) = 2n^2 + 5 >= c * 7n$$

if $n = 1$, $7 = 7$

$n = 2$ $13 = 14$

$n = 3$ $23 = 14$

$n = 4$ $37 = 28$

$n = 5$ $55 = 35$

$n \geq 4$, $F(n) = 2n^2 \geq 7n$

F(n) is always greather than or equal to c * g(n)

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

∴ F(n) is at least as fast as the order of growth of g(n). F(n) grows at least as fast as 7n as n approaches positive infinity