Assignment -1

Design and Analysis of Algorithm

CSA0671

M. Raghavendra

192225022

AIML.

① Find the efficiency and order of notation for recursive algorithm factorial of a given number.

Sol:- General plan:-

1) Integer n.

2) Multiplication.

3) n times

4) $f(n) = f(n-1) + n$.

$m(n) = m(n-1) + 1.$

↓ To compute $f(n-1)$ ⟶ Constant k.

$n = 0$

$0! = 0$

$m(0) = 0.$

c) solving.

pseudo code:-

Algorithm fact (n)

problem description: computers fact of n.

Input : Any integer n.

output : factorial of n.

If (n==0)

return 1;

else

return fact (n-1) x f(n).

substitution Method :-

1) Forward substitution 2) Backward substitution.

Forward substitution :

$m(n) = m(n-1)+1 \rightarrow 0$.

$m(0) = 0$

$n = 1$.

$m(1) = m(1-1)+1$.

$m(1) = 1$.

$n = 2$.

$m(2) = m(2-1)+1$.

$= 1+1$.

$m(2) = 2$.

$\vdots \quad \vdots$

$n = i$

$m(i) = m(n-i)+1$.

Backward substitution :-

$m(n) = m(n-1)+1 \rightarrow (1)$

$m(0) = 0$.

$m = n-1$.

$m(n-1) = m(n-2)+1 \rightarrow (2)$

sub (2) in (1)

$$m'(n) = m(n-2)+2 \rightarrow ③$$

$$m(n-2) = m(n-3)+1 \rightarrow ④$$

sub ④ in ③

$$m(n) = m(n-3)+3 \rightarrow ⑤$$

⋮

$$n=(n-i) = m(n-i-1)+1.$$

$$T(n) \leq O(n) \rightarrow \text{Time complexity}.$$

② Find the efficiency and order of notation for the non recursion algorithm. Find the maximum value in a list.

Ⓐ General plan:

1) Input
2) Basic operation
3) No. of times.
4) summation (Σ)
5) solving summation.

pseudo code :-

Algorithm Max-element [A (0, 1, 2, . . . n-1)]
// problem description
// Input : Given Array
// output : Maximum element in the array

Max-value ← A[0]

for [A(i > Max-value).

Max-value ← A[i]

return Max_Value

Iteration :-

[5, 8, 4, 7, 9]

Max_value = 5

i = 1

If [A[1]] > 5

if 8>5 satisfies

Iteration 2 :-

Max_value = 8

i = 2

if A[2] > 8

if 4>8 not satisfied

return 8.

similarly it compares by iteration 3,4 and it find max_value is 9.

Time complexity :-

$$c(n) = \sum_{i=1}^{n-1} 1$$

Formula :- $\sum_{i=k}^{n} 1 = n-k+1$

$c(n) = (n-1) - \cancel{1} + \cancel{1}$

$c(n) = n-1.$

$C_n \in \Theta(n).$

③ Explain the steps to solve the Towers of Honai problem and also estimate the order of notation for n.

dist using the substitution method for to predict the order of growth.

Sol: Tower of Honai:- We have to more the dist from one pole to other by supportive.

General plan:-

1. n dist   2. Move   3. n time   4. Recure sence relation.

pseudo code:-

Algorithm TOH.

// problem Description

// Input : Any Intern.

// output : Tower of Honai n.

If (n==1)

{ write ("Disk mode from A to B").

return

} //Move to n-1 dist from A to B using C TOH.

// Move remaining size. disk

TOH.

}.

Recurrence relation:-.

If $n>1$

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

Initial condition

$n=1$.

$m(1)=1 \rightarrow$ only one disk contains.

solving:- Forward substitution:-

$m(n) = 2m(n-1)+1 \rightarrow ①$.

$m(1) = 1$.

$n=2 \rightarrow$ sub in eq ①.

$m(2) = 2m(1)+1$

$m(2) = 3$

$n = 3$

$m(3) = 4$

$\vdots \qquad \vdots$

$n=i \quad m(i) = 2m(n-i)+i$

Backward substitution :-

$m(n) = 2m(n-1)+1 \rightarrow$ ①

$m(1) = 1$

$n = n-1$.

$m(n-1) = 2m(n-2)+1 \rightarrow$ ②

sub ② in eq ①

$m(n) = 4m(n-2)+2+1 \rightarrow$ ③

$\vdots$

$m(n) = 2m(n-1)+2^{i-1} \ldots +2+1$

$$\boxed{x^{i-1}+x^{i-2}+\ldots\ldots+2+1 = \frac{1-x^{i}}{1-x}.}$$

$m(n) = 2^{i}m(n-i) + \dfrac{1-2^{i}}{1-2} = 2^{i}-1.$

$m(n) = 2^{i}\, m(n-i) + 2^{i}-1.$

sub $i = n-1$.

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

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

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

$= 2^{n}-1.$

$\boxed{T(n) \in O(2^{n})} \rightarrow$ Time Complexity.