Problem 2.3.2 Complexity analysis of the Tower of Hanoi algorithm

Answer:

Time Complexity Analysis:

Let,

the time required for N disks is $f(N)$
moving a disk from source to destination is a constant time operation, let it be 1

Now according to the recursive solution of Tower of Hanoi,

$$f(N) = f(N-1) + 1 + f(N-1)$$
$$f(N) = 2f(N-1) + 1 \quad (1)$$

Then,

$$f(N-1) = 2f(N-2) + 1 \quad (2)$$
$$f(N-2) = 2f(N-3) + 1 \quad (3)$$

Using the substitution law between equation (2) and (3),

$$f(N-1) = 2(2f(N-3) + 1) + 1$$
$$f(N-1) = 2^2 f(N-3) + 2 + 1 \quad (4)$$

Again, using the substitution law between equation (1) and (4),

$$f(N) = 2(2^2 f(N-3) + 2 + 1) + 1$$
$$f(N) = 2^3 f(N-3) + 2^2 + 2 + 1$$
$$f(N) = 2^3 f(N-3) + 2^2 + 2^1 + 2^0 \quad (5)$$

After the generalization of equation (5),

$$f(N) = 2^k f(N-k) + 2^{k-1} + 2^{k-2} + \dots\dots\dots + 2^2 + 2^1 + 2^0$$

For the base case, $f(1) = 1$, then $N - k = 1$, so $k = N - 1$

Therefore,

$$f(N) = 2^{N-1} f(1) + 2^{N-2} + 2^{N-3} + \dots\dots\dots + 2^2 + 2^1 + 2^0$$
$$f(N) = 2^0 + 2^1 + 2^2 + \dots\dots\dots + 2^{N-3} + 2^{N-2} + 2^{N-1} \quad (6)$$

Equation (6) is a geometric series, where

First term, $a = 2^0 = 1$

Common ratio, $r = \frac{2^1}{2^0} = \frac{2^2}{2^1} = 2\ (r > 1)$

Number of terms, $n = N$

So equation (6),

$$f(N) = \sum_{n=1}^{N} ar^{n-1} = \frac{a(1-r^n)}{1-r} = \frac{1(1-2^N)}{1-2} = \frac{1-2^N}{-1} = -(1-2^N) = 2^N - 1$$

Therefore, the time complexity of the Tower of Hanoi algorithm, $f(N) = O(2^N - 1) \approx O(2^N)$, which is exponential.

Space Complexity Analysis:

Let,

the space required for N disk is $f(N)$
space needed for each call is independent of N which is constant, let it be 1

Since, there are two recursive call and among them 2nd recursive call is invoked after 1st recursive call is over, we can reuse the space of 1st recursive call for 2nd recursive call. Therefore,

$f(N) = f(N-1) + 1$

$f(1) = 1$ (For base case, constant space is needed)

Then,

$f(2) = f(2-1) + 1 = f(1) + 1 = 1 + 1 = 2$

$f(3) = f(3-1) + 1 = f(2) + 1 = 2 + 1 = 3$

$f(4) = f(4-1) + 1 = f(3) + 1 = 3 + 1 = 4$

$f(5) = f(5-1) + 1 = f(4) + 1 = 4 + 1 = 5$ and so on

So, by looking into the $f(1), f(2), f(3), f(4)$, and so on, the required space is linearly increasing depending on the disk size. Therefore, the space complexity of the Tower of Hanoi algorithm, $f(N) = O(N)$.

Summary: Complexity of the Tower of Hanoi algorithm for N disk is as follows,

Complexity Category	**Complexity Function (for N Disk)**	**Complexity Class**
Time	$f(N) = O(2^N - 1) \approx O(2^N)$	exponential
Space	$f(N) = O(N)$	linear