S.KIMMEL

Recurrence Relations

Algorithm with loops → use summation/graph analysis to analyze runtime

Recursive algorithm → use recurrence relations

ex:

```
Factorial(n)
1. if(n==1): return 1
2. else: return n·Factorial(n-1)
```

Recurrence relation for time complexity T(n)

- $T(n) = C$ if $n = 1$ ← Base case(s)
- $T(n) = D + T(n-1)$ ← Recurrence

} Recurrence Relation

C, D are constants

all operations done in recursive call

To solve, want to not have T on right hand side:

e.g. $T(3) = D + T(2)$

plug in $T(2) = D + T(1)$

$T(3) = D + D + T(1) = 2D + T(1)$

plug in $T(1) = C$

$T(3) = 2D + C$

S. KIMMEL

Solving a Recurrence Relation (iterative method)

1. Plug in several times:

ex:

$$T(n) = D + T(n-1) \qquad \left(T(n-1) = D + T(n-2) \text{ plug in}\right)$$
$$= D + D + T(n-2) \qquad \left(T(n-2) = D + T(n-3) \text{ plug in}\right)$$
$$= D + D + D + T(n-3)$$
$$\vdots$$

2. Guess the pattern for each stage using variable k:

ex: Looks like pattern is

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

3. Find value of k such that $T(\cdot)$ on right hand side is base case, plug in to solve

ex: Base case is $T(1)$. Want $n-k=1$

$\Downarrow$ solve for k

$k = n-1$

Plug back in:

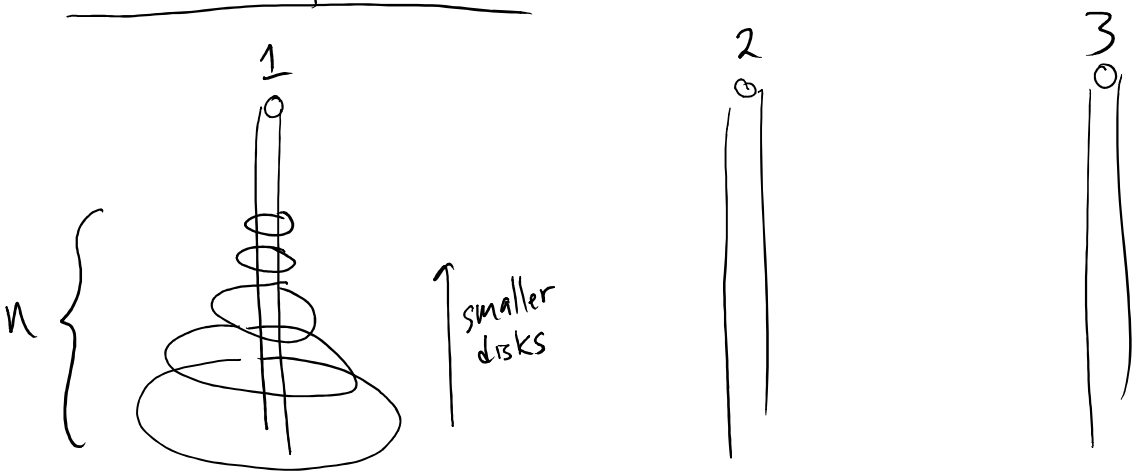
$$T(n) = (n-1)D + T(n-(n-1)) = (n-1)D + T(1)$$
$$= (n-1)D + (\ ) = O(n)$$

$$T(n) = O(n)$$

S. KIMMEL

Another example...

Towers of Hanoi



Task: Move Tower from 1 to 3 without putting larger disk on top of smaller.

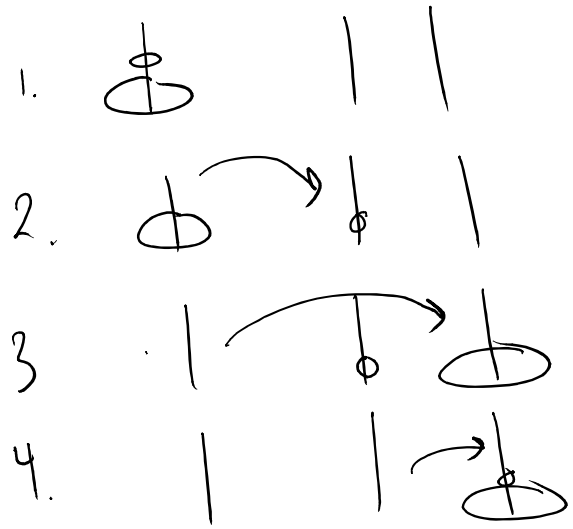
Q: How many moves do you need?

ex 1.

2.

3.

4.



$n=2 \rightarrow 3$ moves

S. KIMMEL

Q: Which of the following recurrence relations is correct if $T(n)$ = # of moves w/ n disks

A) $T(1) = 1$; $T(n) = 2 + T(n-1)$

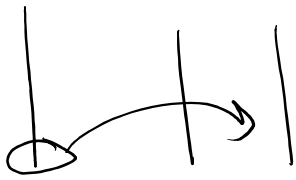
B) $T(2) = 1$; $T(n) = 2T(n-1)$

C) $T(1) = 1$; $T(n) = 2T(n-1)$

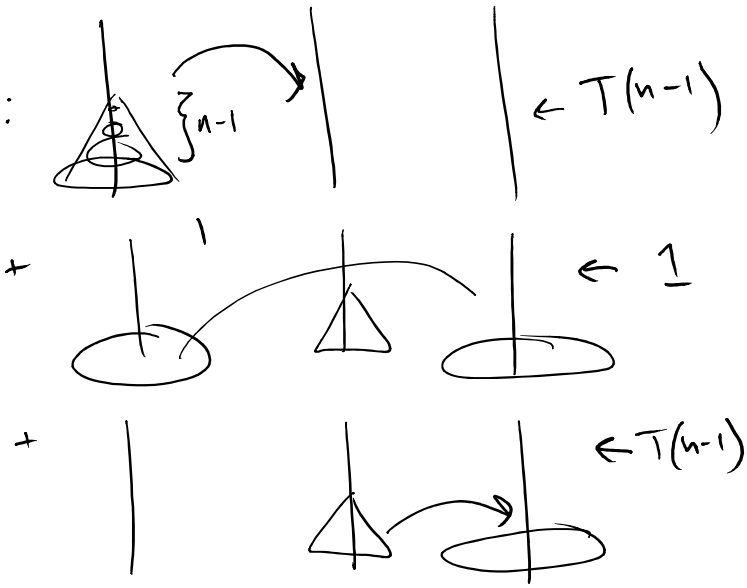
D) $T(1) = 1$; $T(n) = 2T(n-1) + 1$

Answer D:

$T(1) = 1$



$T(n)$:



Think about how you could solve using the fact that to move n-1 disks takes time $T(n-1)$

S. KIMMEL

Solve for $T(n)$ using iterative method

$$T(n) = 2T(n-1) + 1$$

$$= 2\left(2 \cdot T(n-2) + 1\right) + 1 = 4 \cdot T(n-2) + 2 + 1$$

$$= 4 \cdot \left(2 \cdot T(n-3) + 1\right) + 2 = 8 \cdot T(n-3) + 4 + 2 + 1$$

$$= 8\left(2 \cdot T(n-4) + 1\right) + 4 + 2 = 16 \cdot T(n-4) + 8 + 4 + 2 + 1$$

$$\vdots$$

$$= 2^k T(n-k) + \sum_{j=1}^{k-1} 2^j$$

Base case: $n - k = 1 \Rightarrow k = n-1$

$$= 2^{n-1} T(1) + \sum_{j=0}^{n-2} 2^j$$

$$= \sum_{j=0}^{n-1} 2^j = 1 + 2 + 4 + 8 + 16 + 32 + \cdots + 2^{n-2} = \frac{2^n - 1}{2 - 1}$$

Geometric series: Will prove in HW

$$= 2^n - 1 = O(2^n)$$

Worksheet!