Recurrences

Recurrence

Recall that we used induction to prove statements like

$$0+1+2+3+\ldots+n = \frac{n(n+1)}{2} \qquad \text{(Arithmetic series)}$$

$$1+a+a^2+a^3+\ldots+a^n = \frac{1-a^n}{1-a} \qquad \text{(Geometric series)}$$

Recurrence

Let's denote

$$Sum(n) = 1 + 2 + 3 + \ldots + n.$$

Question 1:
If we already know that $Sum(99) = 4950$, compute $Sum(100)$.
Question 2:
If we know $Sum(n-1)$, how to compute $Sum(n)$?

These observations suggest that this summation can be defined:

$$\begin{aligned} Sum(0) &= 0 \\ Sum(n) &= Sum(n-1) + n \qquad (\forall n > 0) \end{aligned}$$

This is a so called *recurrent* definition.

Recurrence

Exponentiation (to integer power n):

$$E(a,n) = a^n$$

Recurrently:

$$\begin{aligned} E(a,0) &= 1 \\ E(a,n) &= E(a,n-1)\cdot a \qquad (\forall n > 0) \end{aligned}$$

Recurrence

Factorial:

$$n! \ = 1 \cdot 2 \cdot \ldots \cdot n$$

Recurrently:

$$\begin{aligned} 0! &= 1 \\ n! &= (n-1)! \cdot n \qquad\qquad (\forall n > 0) \end{aligned}$$

Recurrence

Translating a word problem into a recurrence.

A farmer had 10 cows when he started in January, 2000. The farm is working great, and each year the number of cows doubles.

Unfortunately, on every December 31, one of the cows gets abducted by a UFO.

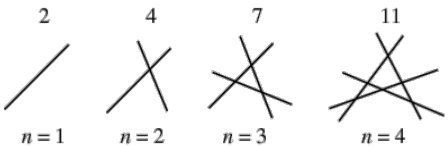
Question: Find a recurrent expression for the number of cows in the beginning of each year.

$(9 \times 2^{\Delta Y} + 1)$

Lines dividing the plane into regions

Let's consider lines dividing the plane into regions:

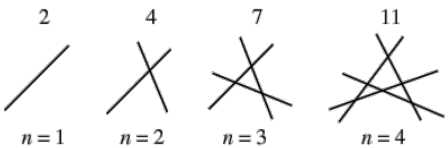
1 line divides it into 2 regions.
2 lines into 4 regions.
3 lines into 7 regions...



(It is assumed that each pair of lines intersects, and no three lines intersect in the same point.)

When there are $n-1$ lines and we add one more line. How does it change the number of regions?

Lines dividing the plane into regions



When there are $n-1$ lines and we add one more line. How does it change the number of regions?

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

Full recurrence:

$$\begin{aligned} R(0) &= 1 \\ R(n) &= R(n-1) + n \qquad \text{(when } n > 0) \end{aligned}$$

Okay, nice, but can we find a *closed-form* expression for it?

Lines dividing the plane into regions

Recurrence:

$$
\begin{aligned}
R(0) &= 1 \\
R(n) &= R(n-1) + n \qquad \text{(when } n > 0\text{)}
\end{aligned}
$$

Answer for the closed-form expression:

$$R(n) = \frac{n(n+1)}{2} + 1.$$

It is almost identical to the $Sum(n)$ recurrence we considered earlier, differing only in the way it is initialized.

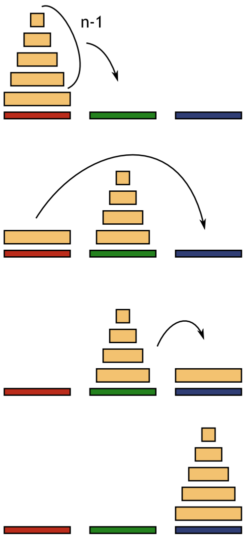
The towers of Hanoi

http://www.mathsisfun.com/games/towerofhanoi.html

The towers of Hanoi

Our recursive algorithm to move
a tower of height n from *#1* to *#3*:

1. Move an $(n-1)$-tower from *#1* to *#2*.
2. Move an 1-tower from *#1* to *#3*.
3. Move an $(n-1)$-tower from *#2* to *#3*.

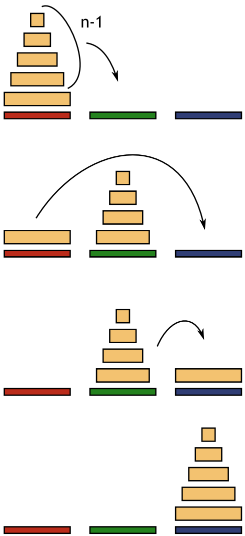


The towers of Hanoi

Our recursive algorithm to move
a tower of height n from *#1* to *#3*:

1. Move an $(n-1)$-tower from *#1* to *#2*.
2. Move an 1-tower from *#1* to *#3*.
3. Move an $(n-1)$-tower from *#2* to *#3*.

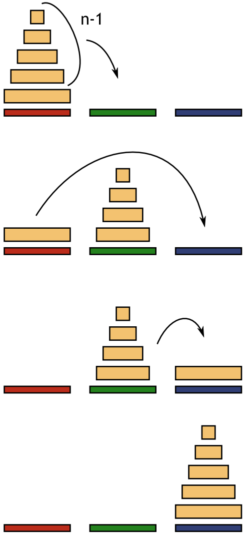
There is a way to find a recurrent formula for T_n, the total number of steps to move the tower from the peg 1 to the peg 3.



The towers of Hanoi

T_n, the time to move a tower of height n:

$$T_1 = 1$$
$$T_n = T_{n-1} + 1 + T_{n-1} \qquad (\forall n > 1)$$

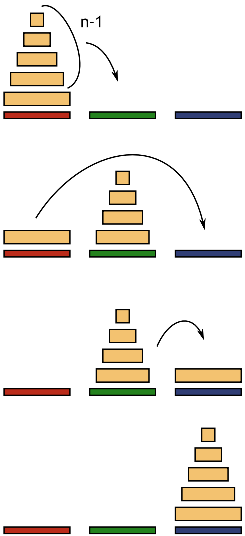


The towers of Hanoi

T_n, the time to move a tower of height n:

$$T_1 = 1$$
$$T_n = T_{n-1} + 1 + T_{n-1} \qquad (\forall n > 1)$$

There is a proof by induction that this time is optimal for any algorithm.



The towers of Hanoi

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \qquad (\forall n > 1)
\end{aligned}
$$

Our goal is to find a closed form expression for T_n as a function of n, without any recurrence.

The towers of Hanoi

Before we get a closed form formula for T_n, what are the numbers?

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \qquad (\forall n > 1)
\end{aligned}
$$

We can compute a list like this:

$$
\begin{aligned}
T_1 &= 1 \\
T_2 &= 3 \\
T_3 &= 7 \\
T_4 &= 15 \\
T_5 &= 31 \\
T_6 &= 63 \\
&\dots
\end{aligned}
$$

The towers of Hanoi

$$T_1 = 1$$
$$T_n = 2T_{n-1} + 1 \qquad (\forall n > 1)$$

$$T_1 = 1$$
$$T_2 = 3$$
$$T_3 = 7$$
$$T_4 = 15$$
$$T_5 = 31$$
$$T_6 = 63$$
$$\dots$$

Guess and verify method... Let's try $T_n = 2^n - 1$?

The towers of Hanoi

$$T_1 = 1$$
$$T_n = 2T_{n-1} + 1 \qquad (\forall n > 1)$$

Guess and verify method... Let's try $T_n = 2^n - 1$? We can show by induction that this formula is correct.

The base case, $n = 1$:

$$T_1 = 2^1 - 1 = 1.$$

Ok, the base case is true.

The towers of Hanoi

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \qquad (\forall n > 1)
\end{aligned}
$$

We want to prove the closed form formula $T_n = 2^n - 1$.

The inductive step, $n > 1$:
Assume that $T_n = 2^n - 1$, and show that then $T_{n+1} = 2^{n+1} - 1$.

Proof. From the recurrence:

$$T_{n+1} = 2T_n + 1$$

By the inductive hypothesis:

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

The towers of Hanoi

If it is difficult to guess the closed form expression for the recurrence, there is another technique:

$$T_1 = 1$$
$$T_n = 2T_{n-1} + 1$$

The towers of Hanoi

If it is difficult to guess the closed form expression for the recurrence, there is another technique:

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \\
&= 2(2T_{n-2} + 1) + 1
\end{aligned}
$$

The towers of Hanoi

If it is difficult to guess the closed form expression for the recurrence, there is another technique:

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \\
&= 2(2T_{n-2} + 1) + 1 \\
&= 2^2 T_{n-2} + 2 + 1
\end{aligned}
$$

The towers of Hanoi

If it is difficult to guess the closed form expression for the recurrence, there is another technique:

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \\
&= 2(2T_{n-2} + 1) + 1 \\
&= 2^2 T_{n-2} + 2 + 1 \\
&= 2^2(2T_{n-3} + 1) + 2 + 1 \\
&= 2^3 T_{n-3} + 2^2 + 2 + 1 \\
&\dots
\end{aligned}
$$

The towers of Hanoi

If it is difficult to guess the closed form expression for the recurrence, there is another technique:

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \\
&= 2(2T_{n-2} + 1) + 1 \\
&= 2^2 T_{n-2} + 2 + 1 \\
&= 2^2(2T_{n-3} + 1) + 2 + 1 \\
&= 2^3 T_{n-3} + 2^2 + 2 + 1 = \ldots = 2^k T(n-k) + 2^{k-1} + \ldots + 2 + 1 \\
&\ldots \quad \text{(can expand until } k = n-1 \text{ and } T(1) = 1\text{)}
\end{aligned}
$$

The towers of Hanoi

If it is difficult to guess the closed form expression for the recurrence, there is another technique:

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \\
&= 2(2T_{n-2} + 1) + 1 \\
&= 2^2 T_{n-2} + 2 + 1 \\
&= 2^2(2T_{n-3} + 1) + 2 + 1 \\
&= 2^3 T_{n-3} + 2^2 + 2 + 1 = \ldots = 2^k T(n-k) + 2^{k-1} + \ldots + 2 + 1 \\
&\ldots \quad \text{(can expand until } k = n-1 \text{ and } T(1) = 1) \\
&= 2^{n-1} \underbrace{T(n-(n-1))}_{=1} + \cdots + 4 + 2 + 1
\end{aligned}
$$

The towers of Hanoi

If it is difficult to guess the closed form expression for the recurrence, there is another technique:

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \\
&= 2(2T_{n-2} + 1) + 1 \\
&= 2^2 T_{n-2} + 2 + 1 \\
&= 2^2(2T_{n-3} + 1) + 2 + 1 \\
&= 2^3 T_{n-3} + 2^2 + 2 + 1 = \ldots = 2^k T(n-k) + 2^{k-1} + \ldots + 2 + 1 \\
&\ldots \quad \text{(can expand until } k = n-1 \text{ and } T(1) = 1) \\
&= 2^{n-1} \underbrace{T(n-(n-1))}_{=1} + \cdots + 4 + 2 + 1 \\
&= 2^{n-1} + \cdots + 4 + 2 + 1 = \sum_{k=0}^{n-1} 2^k =
\end{aligned}
$$

The towers of Hanoi

If it is difficult to guess the closed form expression for the recurrence, there is another technique:

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \\
&= 2(2T_{n-2} + 1) + 1 \\
&= 2^2 T_{n-2} + 2 + 1 \\
&= 2^2(2T_{n-3} + 1) + 2 + 1 \\
&= 2^3 T_{n-3} + 2^2 + 2 + 1 = \ldots = 2^k T(n-k) + 2^{k-1} + \ldots + 2 + 1 \\
&\ldots \quad \text{(can expand until } k = n-1 \text{ and } T(1) = 1\text{)} \\
&= 2^{n-1} \underbrace{T(n-(n-1))}_{=1} + \cdots + 4 + 2 + 1 \\
&= 2^{n-1} + \cdots + 4 + 2 + 1 = \sum_{k=0}^{n-1} 2^k = \frac{1-2^n}{1-2} = 2^n - 1.
\end{aligned}
$$

The towers of Hanoi

Why is it useful to know that the recurrence

$$
\begin{aligned}
T_1 &= 1 \\
T_n &= 2T_{n-1} + 1 \qquad (\forall n > 1)
\end{aligned}
$$

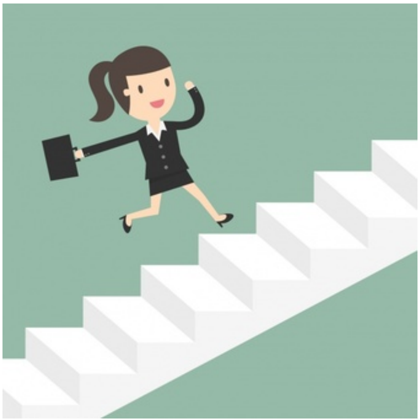
is equivalent to the closed form formula $T_n = 2^n - 1$?

The 7-disk puzzle will require $T_7 = 2^7 - 1 = 127$ moves to complete.

And the 100-disk puzzle will require

$T_{100} = 2^{100} - 1 = 1267650600228229401496703205375$ moves.

Fibonacci staircase



A staircase has n steps.

You walk up by taking 1 or 2 steps at a time.

Question: In how many different ways can you walk up the stairs?