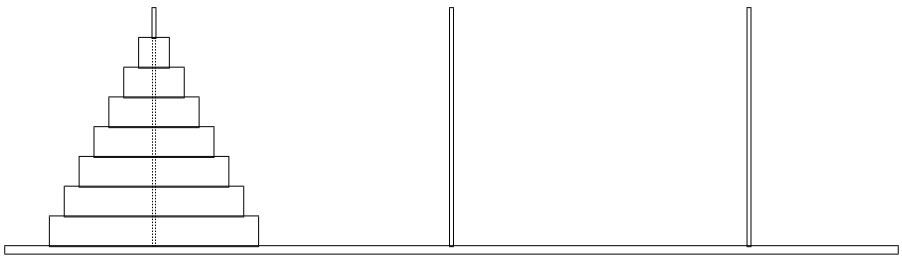
Recurrences I

This is the first of two lectures about solving recurrences and recurrent problems. Needless to say, recurrent problems come up again and again. In particular, recurrences often arise in the analysis of recursive algorithms.

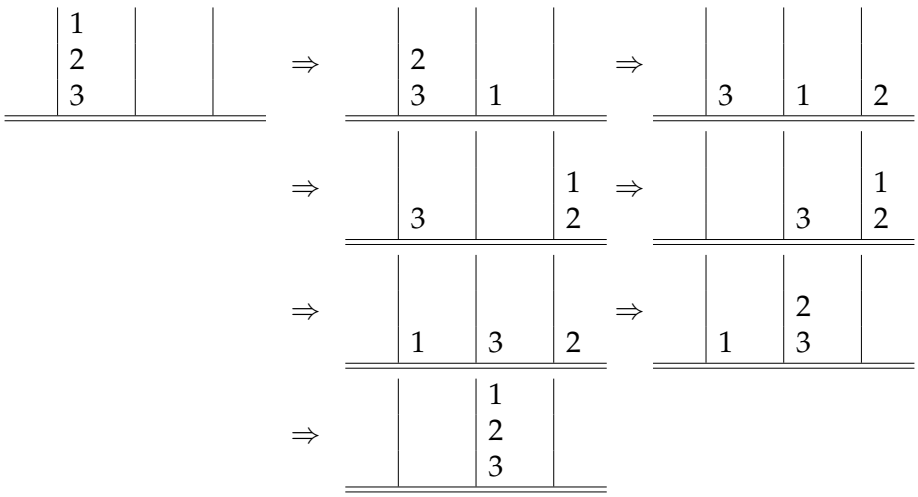
1 The Towers of Hanoi

In the Towers of Hanoi problem, there are three posts and seven disks of different sizes. Each disk has a hole through the center so that it fits on a post. At the start, all seven disks are on post #1 as shown below. The disks are arranged by size so that the smallest is on top and the largest is on the bottom. The goal is to end up with all seven disks in the same order, but on a different post. This is not trivial because of two restrictions. First, the only permitted action is removing the top disk from a post and dropping it onto another post. Second, a larger disk can never lie above a smaller disk on any post. (These rules imply, for example, that it is no fair to pick up the whole stack of disks at once and then to drop them all on another post!)



It is not immediately clear that a solution to this problem exists; maybe the rules are so stringent that the disks cannot all be moved to another post!

One approach to this problem is to consider a simpler variant with only three disks. We can quickly exhaust the possibilities of this simpler puzzle and find a 7-move solution such as the one shown below. (The disks on each post are indicated by the numbers immediately to the right. Larger numbers correspond to larger disks.)



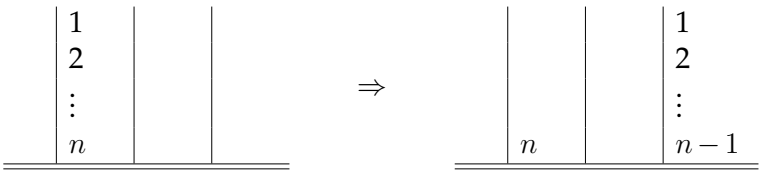
This problem was invented in 1883 by the French mathematician Edouard Lucas. In his original account, there were 64 disks of solid gold. At the beginning of time, all 64 were placed on a single post, and monks were assigned the task of moving them to another post according to the rules described above. According to legend, when the monks complete their task, the Tower will crumble and the world will end!

The questions we must answer are, "Given sufficient time, can the monks succeed?" and if so, "How long until the world ends?" and, most importantly, "Will this happen before the 6.042 final?"

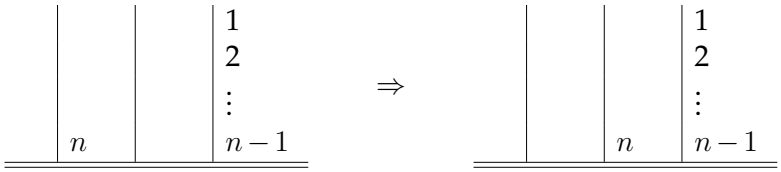
1.1 Finding a Recurrence

The Towers of Hanoi problem can be solved recursively as follows. Let T_n be the minimum number of steps needed to move an n-disk tower from one post to another. For example, a bit of experimentation shows that $T_1 = 1$ and $T_2 = 3$. For 3 disks, the solution given above proves that $T_3 \leq 7$. We can generalize the approach used for 3 disks to the following recursive algorithm for n disks.

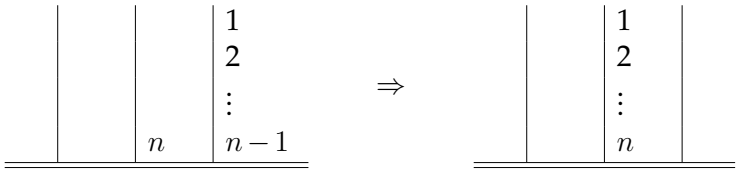
Step 1. Move the top $n-1$ disks from the first post to the third post. This can be done in T_{n-1} steps.



Step 2. Move the largest disk from the first post to the to the second post. This requires just 1 step.



Step 3. Move the $n-1$ disks from the third post onto the second post. Again, T_{n-1} steps are required.



This algorithm shows that T_n, the number of steps required to move n disks to a different post, is at most $2T_{n-1}+1$. We can use this fact to compute upper bounds on the number of steps required for various numbers of disks:

$$\begin{aligned} T_3 &\leq 2 \cdot T_2 + 1 \\ &= 7 \\ T_4 &\leq 2 \cdot T_3 + 1 \\ &\leq 15 \end{aligned}$$

The algorithm described above answers our first question: given sufficient time, the monks will finish their task and end the world. (Which is a shame. After all that effort they'd probably want to smack a few high-fives and go out for burgers and ice cream, but nope— world's over.)

1.2 A Lower Bound for Towers of Hanoi

We can not yet compute the exact number of steps that the monks need to move the 64 disks; we can only show an upper bound. Perhaps— having pondered the problem since the beginning of time— the monks have devised a better algorithm.

In fact, there is no better algorithm, and here is why. At some step, the monks must move the n-th disk from the first post to a different post. For this to happen, the $n-1$ smaller disks must all be stacked out of the way on the only remaining post. Arranging the $n-1$ smaller disks this way requires at least T_{n-1} moves. After the largest disk is moved, at least another T_{n-1} moves are required to pile the $n-1$ smaller disks on top.

This argument shows that the number of steps required is at least $2T_{n-1} + 1$. Since we gave an algorithm using exactly that number of steps, we now have a recurrence for T_n, the number of moves required to complete the Tower of Hanoi problem with n disks:

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

We can use this recurrence to conclude that $T_2 = 3, T_3 = 7, T_4 = 15, \ldots$.

1.3 Guess-and-Verify

Computing T_{64} from the recurrence would require a lot of work. It would be nice to have a closed form expression for T_n, so that we could quickly compute the number of steps required to solve the Towers of Hanoi problem for any given number of disks. (For example, we might want to know how much sooner the world would end if the monks melted down one disk to purchase burgers and ice cream *before* the end of the world.)

There are several different methods for solving recurrences. The simplest method is to *guess* the solution and then to *verify* that the guess is correct, usually with an induction proof. This method is called **guess-and-verify** or "substitution". As a basis for a good guess, let's tabulate T_n for small values of n:

n	T_n
1	1
2	3
3	7
4	15
5	31
6	63

Based on this table, a natural guess is that $T_n = 2^n - 1$.

Whenever you guess a solution to a recurrence, you should always verify it with a proof by induction or by some other technique; after all, your guess might be wrong. (But why bother to verify in this case? After all, if we're wrong, its not the end of the…no, let's check.)

Claim. *If:*

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

then:

$$T_n = 2^n - 1$$

Proof. The proof is by induction on n. Let $P(n)$ be the proposition that $T_n = 2^n - 1$.

Base case: $P(1)$ is true because $T_1 = 1 = 2^1 - 1$.

Inductive step: Now we assume $T_n = 2^n - 1$ to prove that $T_{n+1} = 2^{n+1} - 1$, where $n \geq 1$.

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

The first equality is the recurrence relation, and the second equation follows by the assumption $P(n)$. The last step is simplification. □

Our guess is now verified. This shows, for example, that the 7-disk puzzle will require $2^7 - 1 = 127$ moves to complete. We can also now resolve our remaining questions about the 64-disk puzzle. Since $T_{64} = 2^{64} - 1$, the monks must complete more than 18 billion billion steps before the world ends. Better study for the final.

1.4 The Plug-and-Chug Method

In general, guess-and-verify is a great way to solve recurrences. The only problem with the method is guessing the right solution. This was easy in the Towers of Hanoi example, but sometimes the solution has a strange form that is quite hard to guess. Practice helps, of course, but so can some other methods.

Plug-and-chug is one such alternative method for solving recurrences. Plug-and-chug is also sometimes called "expansion", "iteration", or "brute force". The method consists of four calculation-intensive steps. These are described below and illustrated with the Tower of Hanoi examle.

Step 1: Plug and Chug

Expand the recurrence equation by alternately "plugging" (applying the recurrence equation) and "chugging" (simplifying the resulting expression).

$$\begin{aligned} T_n &= 1 + 2T_{n-1} && \\ &= 1 + 2(1 + 2T_{n-2}) && \text{plug} \\ &= 1 + 2 + 4T_{n-2} && \text{chug} \\ &= 1 + 2 + 4(1 + 2T_{n-3}) && \text{plug} \\ &= 1 + 2 + 4 + 8T_{n-3} && \text{chug} \\ &= 1 + 2 + 4 + 8(1 + 2T_{n-4}) && \text{plug} \\ &= 1 + 2 + 4 + 8 + 16T_{n-4} && \text{chug} \end{aligned}$$