Workshop 2: Recurrence Relations
Where does it end?

- I have a pile of string
- I have one end of the string in my hand
- I need to find the other end
- How can I do this efficiently?
Rope End Finding

1. Set aside the already obtained end

2. Separate the pile of rope into 2 piles, note which connects to the known end (call it pile A, the other pile B)

3. Count the number of strands crossing the piles

4. If the count is even, pile A contains the end, else pile B does
How efficient is it?

- $T(n) = \text{count}(n) + T\left(\frac{n}{2}\right)$
- $T(n) = 5 + T\left(\frac{n}{2}\right)$
- Base case: $T(1) = 1$

Supposing the counting always requires looking at 5 inches of string
Let’s solve the recurrence!

\[ T(n) = 5 + T\left(\frac{n}{2}\right) \]

\[ T(1) = 1 \]

\[ T(n) = 5 + T\left(\frac{n}{4}\right) \]

\[ 5 + T\left(\frac{n}{8}\right) \]

\[ \cdots \]

Let’s talk about this

\[ \log_2 n \]

\[ T(n) = \sum_{i=0}^{\log_2 n} 5 + 1 = 5 \log_2 n + 1 \]
Structure of the Recursion

• Idea: cut the string in half until its length is no more than 1 inch long.

• What did we do between cuts?
  – Count the “crossing strands”
  – We estimate we need to look at 5 inches of string to do this

• Overall: we check 5 inches of string per cut.

• How many times must I cut it?
Binary Search

• How does it work?
  – Input
  – Output

• Implementation

• What is its “running time”?
  – Units?
Binary Search Running Time

• Counting Comparisons

• \( C(n) = x + C \left( \frac{n}{2} \right) \)

• \( x = ? \) (worst case number of comparisons)
What if the length is odd?

• What happens if the length of the list is odd?

• $C(13) = 4 + C\left(\frac{13}{2}\right) = 4 + C(6.5)$

• We can’t have a list of length 6.5, how do we count comparisons?
What if the length is odd?

• What happens if the length of the list is odd?
• \( C(17) = 4 + C\left(\frac{13}{2}\right) \)
• Certainly, \( C(13) \geq C(8) \)
• Also, \( C(13) \leq C(16) \)
• What is \( C(16) - C(8) \)?
What if the length is odd?

• Just round up to the next power of 2, we’re not going to be much worse than that (asymptotically)!
Towers of Hanoi

On the step of the altar in the temple of Benares, for many, many years Brahmins have been moving a tower of sixty-four golden disks from one pole to another, one by one, never placing a larger on top of a smaller. When all the disks have been transferred the Tower and the Brahmins will fall, and it will be the end of the world.
How many moves does it take?

• To solve for $n$ discs:
  – Move everything on top of the largest disc $(n - 1)$
  – Move the largest disc
  – Move everything back on top of the largest disc

• Define $h: \mathbb{N} \to \mathbb{N}$ s.t. $h(n)$ gives the number of moves required to move $n$ discs

• $h(n) = 2h(n - 1) + 1$

• $h(1) = 1$

• Recursive definition!
How many moves?

\[ h(n) = 2h(n - 1) + 1 \]

The diagram illustrates the recursive process of moving the disks in the Tower of Hanoi puzzle, where each step involves moving a stack of \( n \) disks, moving the largest disk, and recursively solving the puzzle for \( n - 1 \) disks.
How many moves?

\[ h(n) = 2h(n - 1) + 1 \]

Number of moves doubles per level
Should be \( \approx 2^n \)
How many exactly?

• If we have 1 disk:
  – $h(1) =$?

• If we have 2 disks:
  – $h(2) =$?

• If we have 3 disks:
  – $h(3) =$?

• Pattern?
Claim: Hanoi requires $2^n - 1$ moves

• To Show: $h(n) = 2h(n - 1) + 1 = 2^n - 1$

• Base Case: $h(1) = 1 = 2^1 - 1$

• Inductive Hypothesis: Assume $h(k) = 2^k - 1$ for arbitrary $k$

• Inductive Step: Show that $h(k + 1) = 2^{k+1} - 1$
  
  \[-h(k + 1) = 2h(k) + 1\]  
  \[-2h(k) + 1 = 2(2^k - 1) + 1\]  
  \[-2(2^k - 1) + 1 = 2^{k+1} - 1\]  
  Definition of $h$  
  Ind. Hyp.