# 2/28/2022 CS1

① Fri - no in person class
Video will be posted probably Tues Night.

## Mon + Wed

Recurrence Relations

Sums effective for analyzing code w/loops only.

Recursion - amt of time doesn't seem straightforward

towers(n, first, last)
1. towers(n-1, first, mid) - ? steps
2. move disk n from first to last - 1 step
3. towers(n-1, mid, last) - ? steps

Let $T(n)$ = # of steps it takes to move a tower of n disks

$T(0) = 0$

$T(n) = \underbrace{T(n-1)}_{\text{move } n-1 \text{ disks}} + \underbrace{1}_{\text{move bottom disk}} + \underline{T(n-1)}, \quad T(1) = 1$

$T(n) = 2T(n-1) + 1, \quad T(1) = 1$

↳ recurrence relation a function defined in terms of other values of the function.

Technique: Iteration technique to use a recurrence relation to calculate an equivalent closed-form function.

$$T(n) = 2T(n-1) + 1 \qquad T(n-1) = \boxed{2T(n-2)+1}$$

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

$$= 2\left[2T(n-2)+1\right] + 1 \qquad T(n-2) = \boxed{2T(n-3)+1}$$

$$= 4T(n-2) + (2+1)$$

$$= 4\left[2T(n-3)+1\right] + (2+1)$$

$$T(n) = 8T(n-3) + (4+2+1)$$

After $k$ iterations we have:

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

Since this it true for different values of $k$, and I know $T(1)$. I want to plug in the value of $k$ that makes $n-k=1$

$$n-1=k$$

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

$$= 2^{n-1} T(1) + \frac{2^{n-1}-1}{2-1}$$

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

$$= \boxed{2^n - 1} \qquad = 2^{n-1} \cdot 2^{1} - 1$$

$$= \boxed{2^n - 1}$$

# Binary Search analysis

1. Couple comparisons
2. Recursive call array size $\frac{n}{2}$.

Let $T(n)$ be runtime of binary search array size $n$.

$$T(n) = 1 + T\left(\frac{n}{2}\right) \quad , T(1) = 1$$

$$\begin{aligned} T(n) &= T\left(\frac{n}{2}\right) + 1 . \\ &= \left[T\left(\frac{n}{4}\right) + 1\right] + 1 \\ &= T\left(\frac{n}{4}\right) + 2 \\ &= \left[T\left(\frac{n}{8}\right) + 1\right] + 2 \\ &= T\left(\frac{n}{8}\right) + 3 \end{aligned}$$

After $k$ iterations, we get

$$T(n) = T\left(\frac{n}{2^k}\right) + k$$

Want value of $k$ such that $\frac{n}{2^k} = 1 \Rightarrow 2^k = n$
$\Rightarrow k = \log_2 n$

$$\begin{aligned} T(n) &= T(1) + \log_2 n \\ &= 1 + \log_2 n \\ &= O(\log n) \end{aligned}$$

~~Factorial~~ Run time
Permutation

Perm(size $n$) calls Perm(size $n-1$) $n$ times

Let $T(n)$ = run time of perm.

$$T(n) = n \cdot \underline{T(n-1)} + O(1)$$
$$= n\left[(n-1)T(n-2) + O(1)\right] + O(1)$$
$$= n(n-1)T(n-2) + \left[O(1) + O(n)\right]$$
$$= n(n-1)\left[(n-2)T(n-3) + O(1)\right] + \left[O(1) + O(n)\right]$$
$$= n(n-1)(n-2)T(n-3) + \left[O(1) + O(n) + O(n(n-1))\right]$$

After $k$ steps

$$T(n) = {}_nP_k\, T(n-k) + \sum_{i=0}^{k-1} {}_nP_i$$

Plug in $n-k=1$ so $k=n-1$.

$$T(n) = {}_nP_{n-1}\, T(1) + \sum_{i=0}^{n-2} {}_nP_i$$ Prove $\leq n!$

$$= n! + O(n!)$$
$$= O(n!)$$

## Next Rec Rel

MergeSort (n items)

1. MS(left half $\frac{n}{2}$)
2. MS(right half $\frac{n}{2}$)
3. Merge $\rightarrow O(n)$ time

Let $T(n)$ = Merge-Sort run time on $n$ elements

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + O(n)$$

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n)$$

$$T(n) = \boxed{2T\left(\frac{n}{2}\right) + \underline{n}},$$

$$T\left(\frac{n}{2}\right) = \boxed{2T\left(\frac{n}{4}\right) + \frac{n}{2}}$$

$$= 2\left[2T\left(\frac{n}{4}\right) + \frac{n}{2}\right] + n$$

$$T\left(\frac{n}{4}\right) = \boxed{2T\left(\frac{n}{8}\right) + \frac{n}{4}}$$

$$= 4T\left(\frac{n}{4}\right) + n + n$$

$$= \boxed{4T\left(\frac{n}{4}\right) + 2n}$$

$$= 4\left[2T\left(\frac{n}{8}\right) + \frac{n}{4}\right] + 2n$$

$$= 8T\left(\frac{n}{8}\right) + n + 2n$$

$$= \boxed{8T\left(\frac{n}{8}\right) + 3n}$$

$$T(1) = 1$$

After $k$ steps, we have

$$T(n) = 2^k T\left(\frac{n}{2^k}\right) + kn$$

Plug in $k$ such that $\frac{n}{2^k} = 1 \implies k = \log_2 n \quad n = 2^k$

$$T(n) = n\,T(1) + (\log_2 n) \cdot n$$

$$= \underline{n} + \underline{n \log_2 n} = \boxed{O(n \lg n)}$$

# Exact Rec Rel

$T(n) = 3T(n-1) + 1 \qquad T(1) = 1$

$= 3[3T(n-2)+1]+1 \qquad T(n-1) = 3T(n-2)+1$

$= 9T(n-2) + [3+1] \qquad T(n-2) = 3T(n-3)+1$

$= 9[3T(n-3)+1] + [3+1]$

$= 27T(n-3) + [9+3+1]$

After $k$ steps, we have:

$$T(n) = 3^k T(n-k) + \sum_{i=0}^{k-1} 3^i$$

We want $n-k=1 \Rightarrow k = n-1$. Plug in $k=n-1$:

$$T(n) = 3^{n-1} T(1) + \sum_{i=0}^{n-2} 3^i$$

$$= 3^{n-1} + \frac{3^{n-1}-1}{3-1}$$

$$= \frac{\boxed{2 \cdot 3^{n-1}} + \boxed{3^{n-1}} - 1}{2}$$

$$= \frac{\boxed{3 \cdot 3^{n-1}} - 1}{2} = \boxed{\frac{3^n - 1}{2}}$$

---

$T(n) = aT(n-1)+1 \Longrightarrow T(n) = \dfrac{a^n - 1}{a-1}$

for any pos int $a > 1$

Practice Prob