CSCI 3104: Algorithms
Spring 2022
Recitation #1 - Induction

Problem 1

Your friend excitedly shows you her proof that, for all $n \geq 1$, $\sum_{i=1}^{n}(4i-3) = n(2n-1)$. She tells you that she's pretty sure she's gotten it right. Find any errors in your friend's proof and explain why each is incorrect. You don't need to write a corrected version, only explain the errors.

Base Case. Let $k = 1$. Then, $\sum_{i=1}^{k}(4i-3) = 4 \cdot 1 - 3 = 1$, and $k(2k-1) = 1(2 \cdot 1 - 1) = 1$, and the statement holds.

for some $k \geq 1$

Inductive Hypothesis. Assume the statement holds for all $k \geq 1$.

Assume the claim is true and then use it to prove itself.

Inductive Step. We show it also holds for $k+1$. By the IH,

$$\sum_{i=1}^{k+1}(4i-3) = (k+1)(2(k+1)-1), \tag{1}$$

so we can conclude that

$$\sum_{i=1}^{k}(4i-3) = k(2k-1) - (4(k+1)-3) \tag{2}$$

$$\sum_{i=1}^{k}(4i-3) = k(2k-1) \tag{3}$$

as claimed.

It does not make sence because it works on both sides.

$$\sum_{i=1}^{k+1}(4i-3) = \left[\sum_{i=1}^{k}(4i-3)\right] + 4(k+1)-3$$

$$= k(2k-1) + 4k+1$$

$$= 2k^2+3k+1$$

$$= (k+1)(2k+1)$$

$$= (k+1)(2(k+1)-1)$$

The claim still holds.

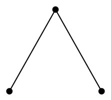
Problem 2

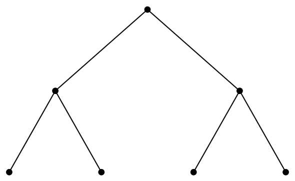
Definition 1. The *complete, balanced binary tree* of depth d for $d \in \mathbb{N}$, denoted $\mathcal{T}(d)$, is defined as follows:

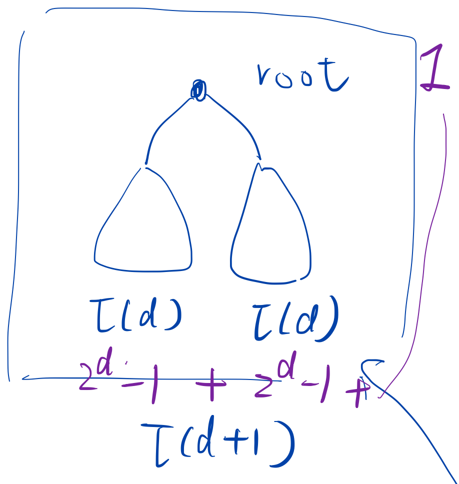
- $\mathcal{T}(0)$ is a single vertex
- for $d > 0$, $\mathcal{T}(d)$ is a root vertex with two children, each the root of $\mathcal{T}(d-1)$.

For example:

$\mathcal{T}(0)$.


$\mathcal{T}(1)$.


$\mathcal{T}(2)$.



Using induction, show that the tree $\mathcal{T}(d)$ has exactly $2^d - 1$ non-leaf nodes.

a. State your induction variable and base case. Show that your base case is correct.
the depth d. $d=0$

b. State your inductive hypothesis.

c. State your goal for the inductive step. Prove your inductive step. Where did you use your inductive hypothesis?

a. induction variable : d

base case: $d=0$

when $d=0$. there is no non-leaf nodes. and $2^0-1=0$. Hence, the claim holds.

b. for some fixed $d \geq 0$, the tree $T(d)$ has 2^d-1 non-leaf nodes.

c. The goal is to show

$T(d+1)$ has $2^{d+1}-1$ non-leaf nodes.

$$\underbrace{2^d-1}_{\text{left subtree}} + \underbrace{2^d-1}_{\text{right subtree}} + \underbrace{1}_{\text{new root.}}$$

$= 2^{d+1}-1$ #

Problem 3

Recall MERGESORT, which sorts an array $A[1,\ldots,n]$ (here, the notation $A[1,\ldots,n]$ indicates an array of length n with indices $1,\ldots,n$). MERGESORT recursively sorts $A[1,\ldots,n/2]$ and $A[n/2+1,\ldots,n]$, and then merges the two sorted sub-arrays into a single sorted array. Pseudocode for MERGESORT is below. You may assume the MERGE function correctly merges the two sorted sub-arrays into a single sorted array. The goal of this problem is to prove the correctness of MERGESORT by induction.

a. Clearly identify the base case(s). Why does MERGESORT correctly sort arrays in these cases?

b. Formulate an inductive hypothesis. [**Note:** Will you want to use a weak inductive hypothesis or strong inductive hypothesis? Try to answer this question and explain your reasoning. Don't hesitate to talk with your classmates or TA if you get stuck!]

c. We now consider the inductive step.

   i. What are the smaller cases that we need to consider? Why?

   ii. How do we know that MERGESORT correctly sorts these smaller cases?

   iii. Once MERGESORT sorts these smaller cases, how do we know that MERGESORT correctly sorts $A[1,\ldots,n]$, the original input array?

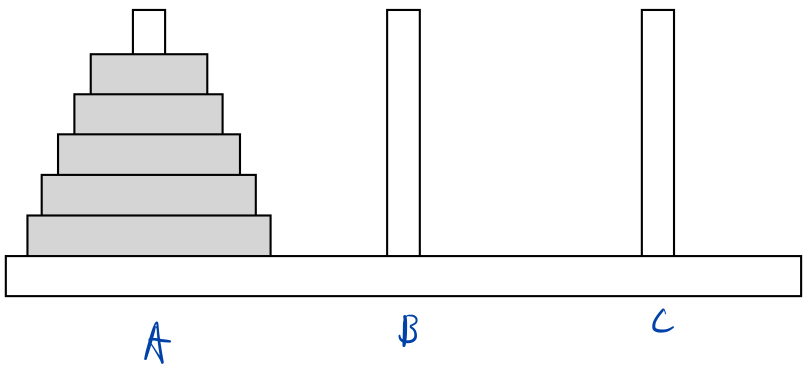
d. Using your work in parts (a)-(c), write a proof of correctness for MERGESORT.

Algorithm 1 Mergesort

```
procedure MERGESORT(Array$A[1,\ldots,n]$)
    if len(A) $\leq 1$ then return A
    Left $\leftarrow$ []
    Right $\leftarrow$ []

    for $i \leftarrow 1; i \leq \text{len}(A)/2; i \leftarrow i+1$ do
        Left[$i$] $\leftarrow A[i]$

    for $i \leftarrow \text{len}(A)/2+1, j \leftarrow 1; i \leq \text{len}(A); i \leftarrow i+1, j \leftarrow j+1$ do
        Right[$j$] $\leftarrow A[i]$

    Mergesort(Left)
    Mergesort(Right)
    Merge(Left, Right, A)
```

Problem 4 (Bonus)

The puzzle game the Tower of Hanoi consists of three upright pegs, and n disks of different diameters which fit over the pegs. The puzzle starts with the disks all stacked on the first peg, with the largest diameter on the bottom decreasing in size to the smallest disk on top, as follows:



The objective of the puzzle is to transfer this "pyramid" to the third peg in the same configuration: largest disk on the bottom up to the smallest on top. Each move consists of moving a single disk from its current peg to a different peg, and disks can only be placed onto either an empty peg or a disk of a larger diameter.

Use induction to show that a Tower of Hanoi with n disks can be solved in $2^n - 1$ moves.

(hint: without loss of generality, if you can move the tower to the third peg in k moves, you can also move the tower to the second peg in k moves)

$n = 0$, ...

IH : Assume $n = k$, the tower can be solved in $2^k - 1$ steps

IS : prove $n = k+1$, the tower can solved in $2^{k+1} - 1$ steps.

① move k disks from A to B and let them still in the correct order : $\boxed{2^k - 1}$

② move 1 remaining disk from A to C : $\boxed{1}$

③ move k disks from B to C and let them in the correct order : $\boxed{2^k - 1}$

$$\underline{2 \cdot (2^k - 1) + 1 = 2^{k+1} - 1}$$ #