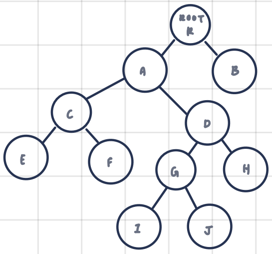
WK 8-1

HEY GUYS LISA HERE! I DO LOVE TAKING NOTES BUT WOW THEY DO TAKE 4EVER :')
I AM ONCE AGAIN ASKING 4 YOUR FINANCIAL SUPPORT
IF YOU WOULD LIKE, I WOULD MUCH APPRECIATE A DONATION @ LISAAWU ON VENMO ♥
$ WILL GO TOWARDS VAL SKINS 4 MY VALORANT ADDICTION

THANK U

Lisa Wu
@lisaawu
venmo

BINARY TREES CONTINUED



THE **HEIGHT** OF A ROOTED TREE : 4

THE LONGEST PATH (HIGHEST LEVEL)

THE **LEVEL** OF A VERTEX IN A ROOTED TREE :

OF EDGES IN THE DIRECTED PATH FROM THE ROOT

LEVEL OF G : 3

LEVEL OF C : 2

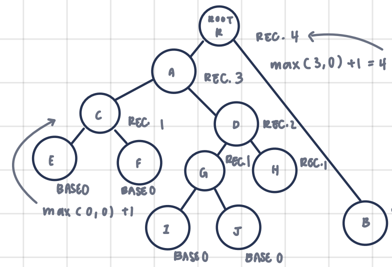
LEVEL 0 : ROOT

LEVEL 3 → E, F, G, H

LEVEL 2 : C, D

LEVEL 4 : J, K

LEVEL 1 : A, B



B IS STILL LEVEL 1

(DON'T GET TROLLED)

BINARY TREE ▪ RECURSIVE DEFINITIONS

- A single vertex is a full binary tree
- If T_1 + T_2 are full bin. trees, then $T_1 \cdot T_2$ is a full bin. tree

  FORMED BY A NEW VERTEX, THE ROOT OF THE NEW TREE, WHICH HAS EDGES TO THE ROOTS OF T_1 + T_2

WE WANT A RECURSIVE DEF. OF THE HEIGHT OF THE TREE :

BASE
IF T IS A SINGLE VERTEX, then $h(T) = 0$

RECURSIVE
IF $T = T_1 \cdot T_2$ for a f.b.t T_1 + T_2,

$h(T) = \max(h(T_1), h(T_2)) + 1$

BOUND THE # OF VERTICES IN THE TREE BASED ON ITS HEIGHT

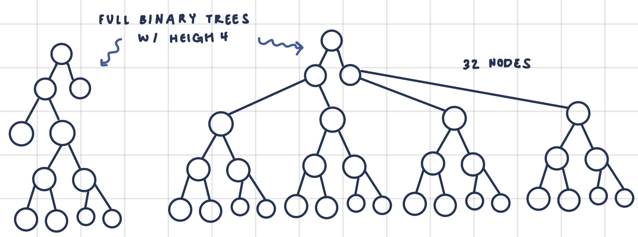
- IF THE HEIGHT OF A FULL BIN. TREE IS h,

  then there is at most $2^{h+1} + 1$ vertices and at least $h+1$ vertices

  ↑ NOT A TIGHT BOUND

max ≠ least because

FULL BINARY TREES W/ HEIGHT 4



HEIGHT WON'T TELL YOU # OF NODES

LET'S PROVE IT

EVERY f.b.t has at most $2^{h+1} - 1$ vertices where h is the height

BASIS :

IF T IS A SINGLE VERTEX, $v(T) = 1$ + $h(T) = 0$

PLUG : $2^{0+1} - 1 = 1$, $v(T) = 1$ ✓

INDUCTIVE STEP:

IF T IS $T_1 \cdot T_2$ for f.b.t T_1 + T_2,

assume $v(T) \leq 2^{h(T_1)+1} - 1$ + $v(T) \leq 2^{h(T_2)+1} - 1$ → assume true for all component trees, show this property holds for all binary trees

we need to prove: $v(T) \leq 2^{(h(T)+1)} - 1$

$v(T_1 \cdot T_2) = v(T_1) + v(T_2) + 1$

$\leq (2^{h(T_1)+1} - 1) + (2^{h(T_2)+1} - 1) + 1$ SUB IN THE HYPOTHESIS

$\leq (2^{h(T_1)+1} + 2^{h(T_2)+1}) - 1$ $-1 + -1 + 1 = -1$ SIMPLIFY

$\leq (2^{\max(h(T_1), h(T_2))+1}) + (2^{\max(h(T_1), h(T_2))+1}) - 1$ because each height $\leq$ max height

$\leq 2^{h(T_1 \cdot T_2)} + 2^{h(T_1 \cdot T_2)} - 1$ BY DEF OF HEIGHT

$\leq 2(2^{h(T_1 \cdot T_2)}) - 1$ DEF OF HEIGHT

$\leq 2^{(h(T_1 \cdot T_2)+1)} - 1$ LAW OF EXPONENTS

GOAL: $v(T) \leq 2^{(h(T)+1)} - 1$ ✓

STRINGS ~ RECURSIVE DEF.s

CH 5.3

GIVEN AN ALPHABET Σ,

WE DEFINE THE SET OF STRINGS OVER THE ALPHABET

- **BASIS:** λ is a string (this is the empty string)
- **RECURSIVE STEP:** if w is a string + $x \in \Sigma$, then wx is a string

  like putting string n another symbol together to make a new string

  EX: if $w = 0110$, $x = 1$, $wx = 01101$

NOW WE WANT A FUNCTION FOR THE **LENGTH OF THE STRING**

length $(\lambda) = 0$

length (wx), where w is a string n $x \in \Sigma$, = length$(w) + 1$

↳ all the symbols from w +1 from x

STRUCTURAL INDUCTION

CLAIM: FOR ALL STRINGS w, $\lambda w = w$

PROOF: BASIS - $\lambda \cdot \lambda = \lambda$, by def. of concatenation

INDUCTIVE - ASSUME $\lambda \cdot w = w$, show $\lambda \cdot wx = wx$ where $x \in \Sigma$

BASE ON STRINGS OF LENGTH K,

PROVE IT WORKS FOR THE NEXT LENGTH UP

we want to prove this claim for all strings

assume our claim is true for the component string

$[\underbrace{a\,b\,c\,d\,e\,d\,s}_{w}]\underbrace{a}_{x}$

$\lambda \cdot wx = (\lambda \cdot w)x$ def. of concatenation

$= wx$ by IH (ASSUME $\lambda \cdot w = w$)

so $\lambda \cdot wx = wx$

DEFINE CONCATENATION:

$w_1 \cdot w_2$ is the concatenation of w_1 + w_2

$w \cdot \lambda = w$

RECURSIVE: $w_1 \cdot w_2 x = (w_1 \cdot w_2)x$ → $1100 \cdot 1001$

$(1100 \cdot 100)1$

$((1100 \cdot 10)0)1$

$(((1100 \cdot 1)0)0)1$

$((((1100\ \lambda)1)0)0)1$

REDUCE TILL $w_2 = \lambda$

: RECONSTRUCT INTO 11001001

DEFINE REVERSAL:

REVERSE $(\lambda) = \lambda$

REVERSE (ABC) = CBA

RECURSIVE: REVERSE $(wx) = \lambda \cdot x \cdot \text{Rev}(w)$

Rev $(abc) = c \cdot \text{Rev}(ab)$

$c \cdot b \cdot \text{Rev}(a)$

$c \cdot b \cdot a \cdot \text{Rev}(\lambda)$

$c \cdot b \cdot a \cdot \lambda = c \cdot ((b \cdot \lambda)a)$ .. → cba

λ because x by itself is NOT a STRING, BY CONCATENATING λ w/ x IT MAKES IT A STRING!