Discrete Math

Structural Induction

Lect. 23

Basic blocks
(Base cases)

Rules for building
(Recursive Structure)

We use <u>Structural Induction</u> when there is no obvious or easy notion of "next" or "previous"

(we don't have a <u>total ordering</u> on our objects)

<u>note</u> "regular" induction inducts on $\mathbb{N}$

$\mathbb{N}$ is well-ordered $\qquad P(k) \Rightarrow P(k+1)$

when you're using a set that has recursive structure, you can instead induct <u>on</u> the recursive structure ex] $S \subseteq \mathbb{N} \times \mathbb{N}$

basis elements

- $(2,1) \in S$, $(1,3) \in S$

recursive structure
$$\left\{ \begin{array}{l} \bullet\ (x,y),\ (a,b) \in S \Rightarrow (xa,\ yb) \in S \\ \bullet\ (x,y) \in S \Rightarrow (x+2,\ y) \in S \end{array} \right.$$

is no obvious order on these elements

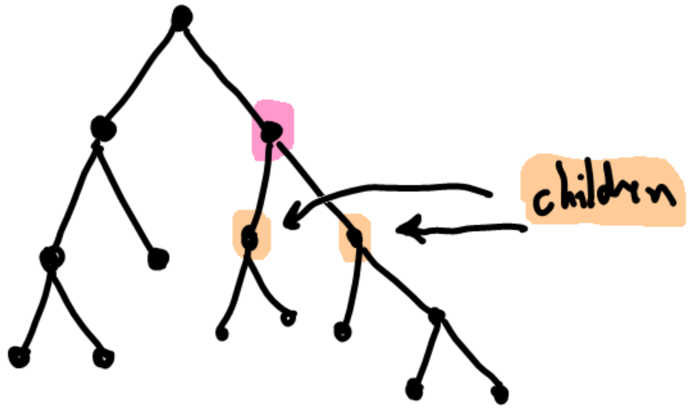
$(2,3) \in S$ $\qquad$ $(4,1)$, $(3,3)$

Full Binary Trees

Tree, rooted +

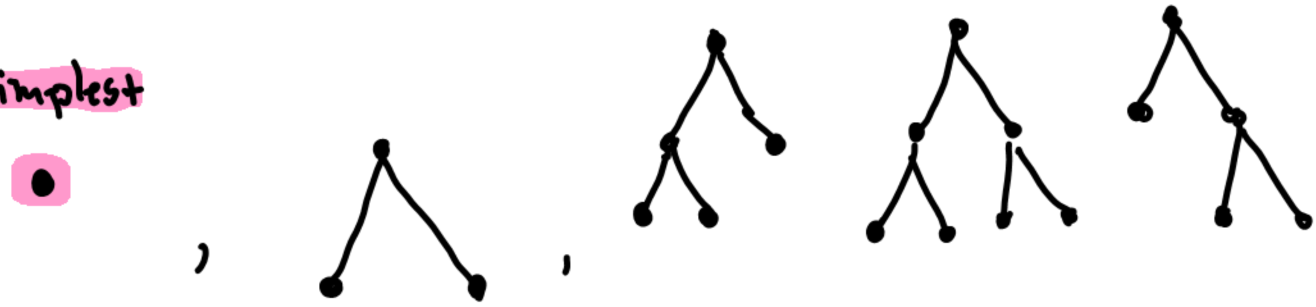
all nodes are joined to either one node or three nodes

all nodes have either 0 "children" or 2 "children"



the recursive structure of FBTs

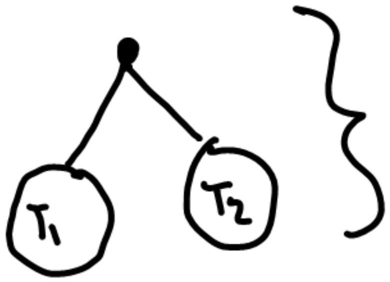
Simplest

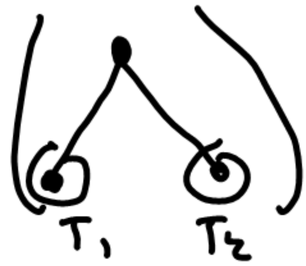


the set of FBTs has recursive structure but it lacks an obvious way to talk about "next" / "previous" $\longrightarrow$ it lacks a $\boxed{\text{total ordering}}$

$$h(T) = \begin{cases} 0 & \text{if } T \text{ consists of one node} \\ \max\{h(T_1), h(T_2)\} + 1 & \text{where } T \text{ is built from two} \\ & \text{full binary sub-trees } T_1 \text{ and } T_2 \end{cases}$$

$\bullet \longleftarrow h = 0$



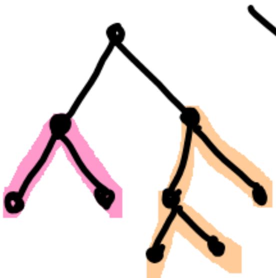
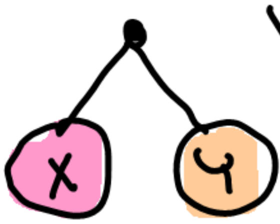
h 

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

$= \max\{h(\bullet), h(\bullet)\} + 1$

$= \max\{0, 0\} + 1$

$= 0 + 1 = 1$

$h\Big($  $\Big) = h\Big($  $\Big)$

$$= \max\{h(x), h(y)\} + 1$$

$= \max\Big\{1, h\Big($  $\Big)\Big\} + 1$

$h\Big($  $\Big) = \max\Big\{h\Big($  $\Big), h(\bullet)\Big\} + 1$

$$= \max\{1, 0\} + 1$$

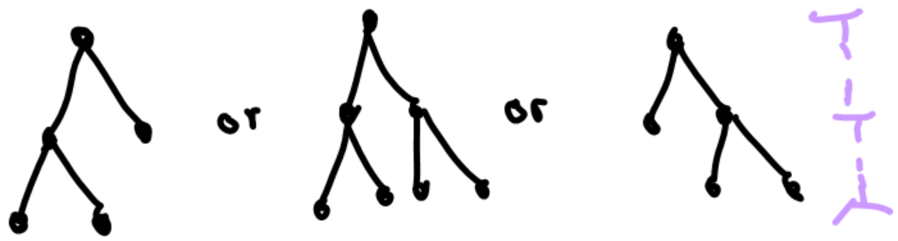
$$= 1 + 1 = \boxed{2}$$

$$= \max\{1, 2\} + 1 = 2 + 1 = \boxed{3}$$

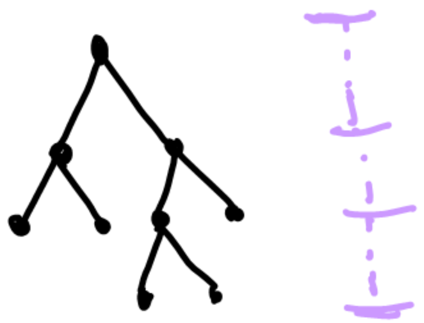
Does the $h(T)$ capture anything "meaningful"?

$h = 0$

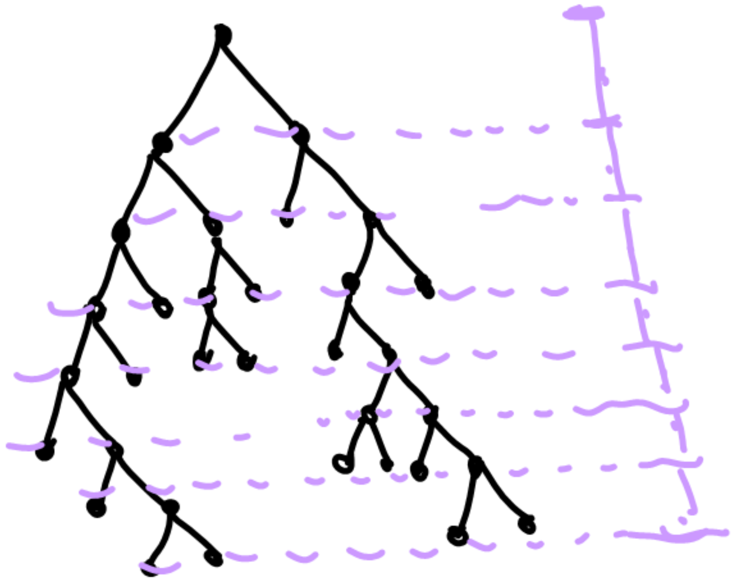
$h = 1$



$h = 2$



$h = 3$



$h = 7$

Example 4.4.

Proposition. *If T is a full binary tree with n vertices then $n \leq 2^{h(T)+1} - 1$*

Proof (By Structural Induction)

Base Case
The full binary tree that consists of one node (i.e. $n = 1$) has height $h = 0$, and so it follows that $n = 1 = 2 - 1 = 2^{0+1} - 1 = 2^{h(T)+1} - 1$.

Recursive Step Suppose T has n vertices and that it can be decomposed into two full binary trees T_1 and T_2, each with n_1 and n_2 vertices, and that the proposition is true for both T_1 and T_2. (We will show that the proposition is true for the full binary tree T.) It follows that

$$n = 1 + n_1 + n_2 \tag{30}$$

$$\leq 1 + 2^{h(T_1)+1} - 1 + 2^{h(T_2)+1} - 1 \tag{31}$$

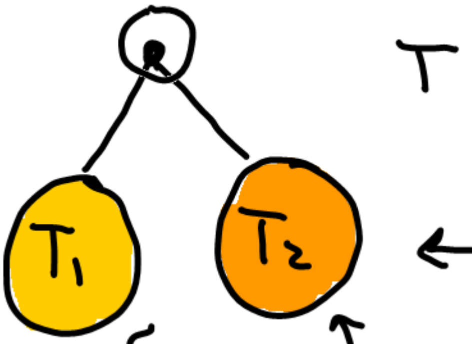
$$= 2^{h(T_1)+1} + 2^{h(T_2)+1} - 1 \tag{32}$$

$$\leq 2 \cdot \max\left\{2^{h(T_1)+1}, 2^{h(T_2)+1}\right\} - 1 \tag{33}$$

$$= 2 \cdot 2^{\max\{h(T_1), h(T_2)\}+1} - 1 \tag{34}$$

$$= 2 \cdot 2^{h(T)} - 1 = 2^{h(T)+1} - 1. \square \tag{35}$$

desired formula for T



assume the prop. is true for these "smaller" / "less compl." full binary trees

$n_1 \leq 2^{h(T_1)+1} - 1$

$n_2 \leq 2^{h(T_2)+1} - 1$

Structural Induction Outline

Proposition. $\forall x \in S, P(x)$

Proof by Structrual Induction

Base Case

Show that $P(s)$ is true for the "smallest" elements $s \in S$
(These are sometimes called "base elements.")

Recursive Step

Show being true at "smaller elements" implies being true at "larger ones"

Show $P(x) \Rightarrow P$(larger elements built from x)

Assume $P(x)$ (this is called the **inductive hypothesis**.)

Use S's recursive structure to relate x to "larger elements built from x"

Use recursive structure to conclude P(larger elements built from x) $\square$

Synonyms

easier	harder
simpler	complex
less complicated	more complicated
	built from