10/23/2024

Recursively defined Sets

Recursively defined Structures

We are going to look at this because we want to use induction on above structure or set. (Trying to extend induction beyond natural numbers)

A recursively defined structure/set is a structure/set S defined by:

1) Its smallest element(s) (base case(s))
2) Rule(s) that construct elements out of smaller elements.
3) The structure/set includes only elements that can be constructed from the base case and the recursive case.

$S = \{x : x \text{ is either case (1) or } x \text{ follows case (2)}\}$

Ex: The set of non-negative integers

1) 0
2) If k is a non-negative integer, $k+1$ is also a non-negative integer.

we can generate the smallest element

$\underline{\underline{0}}$

I can make 1 using 0 and case 2

$0 + 1 = 1$

1 is a non-negative integer

$1 + 1 = 2$

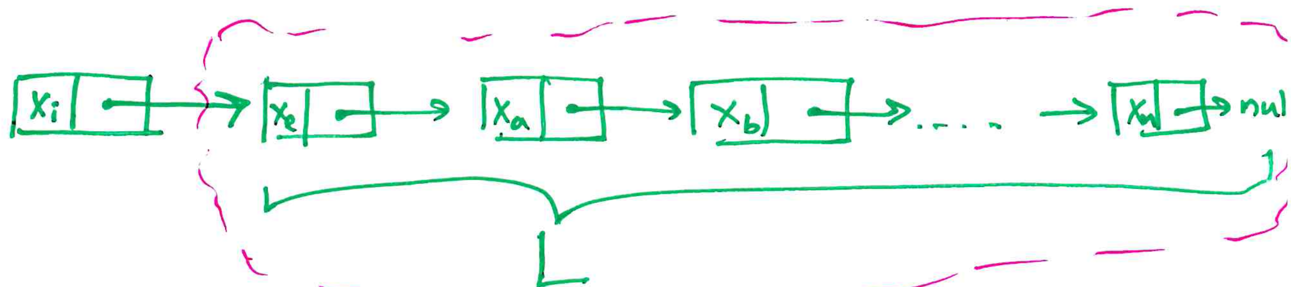
---

ex! A recursively defined structure.

Let's try to define a Linked List as a recursively defined structure.

1) Base case: Empty linked list [ ], null

2) Recursive case: $[x_i, L]$, where x_i is some data & L is a Linked List.

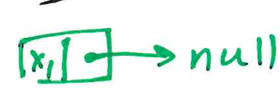


Smallest Linked List

[ ], null

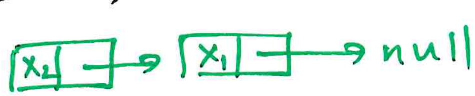
Linked List with 1- element

$[x_1, [\ ]]$



LL with 2- elements

$[x_2, [x_1, [\ ]]]$



Ex: set of ~~all~~ well formed statements of propositional logic, over a set of boolean variables X.

$X = \{\cancel{t}, q, r\}$

$t \Rightarrow q$, $t \vee q$, $r \vee q$, $r \Leftrightarrow q$

$(t \vee q) \Rightarrow (r \Leftrightarrow q)$

1. P, for some $P \in X$

2a) If P, S are well formed sentences of propositional logic;
$P \star S$ where $\star \in \{\wedge, \vee, \Rightarrow, \Leftrightarrow, \oplus\}$

2b) If p is WFS of propositional logic, $\neg p$ is a well formed statement of propositional logic

$X = \{a, b, c, d\}$

Base case elements

a, b, c, d

$a \vee b$ $\quad a \oplus d$ $\quad \neg a$ $\quad \neg b$

$(a \vee b) \Leftrightarrow (a \oplus d)$

$\neg((a \vee b) \Leftrightarrow (a \oplus d))$

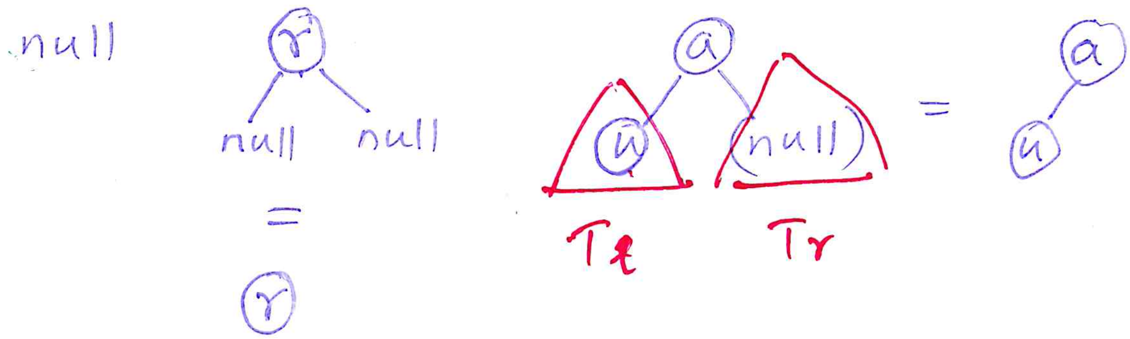
$S = \{a, b, d, c, a \vee b, a \oplus d, \neg a, \neg b, \ldots\}$

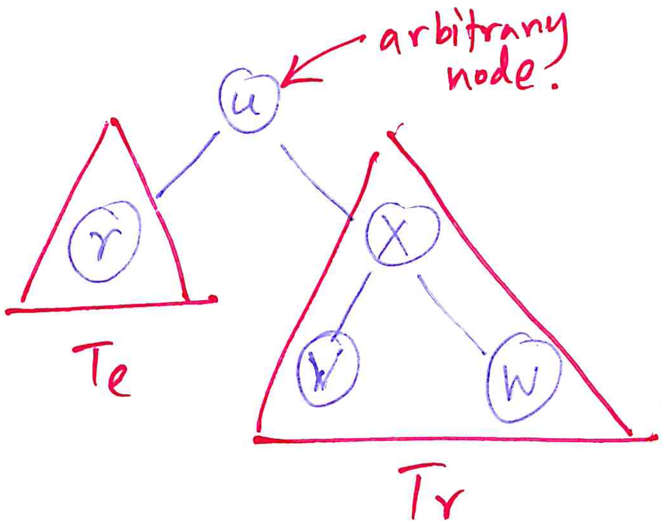
Ex: Binary Trees as a ~~rect~~ recursively defined structure.

1) null (empty tree) or _ ← nothing

2) root node r, left subtree T_ℓ, right subtree T_r

where r is an arbitrary value, & T_ℓ, T_r are both binary trees.





10/25/2024

Def: Suppose you want to prove P(n) holds for $n \geq 0$. To give a proof using mathematical induction we used to prove following:

1. Base case P(0) is true.
2. Inductive case: $\forall n \geq 1: [P(n-1) \Rightarrow P(n)]$

$[\forall n \geq 0: P(n)]$ ✓

Def: Strong Induction

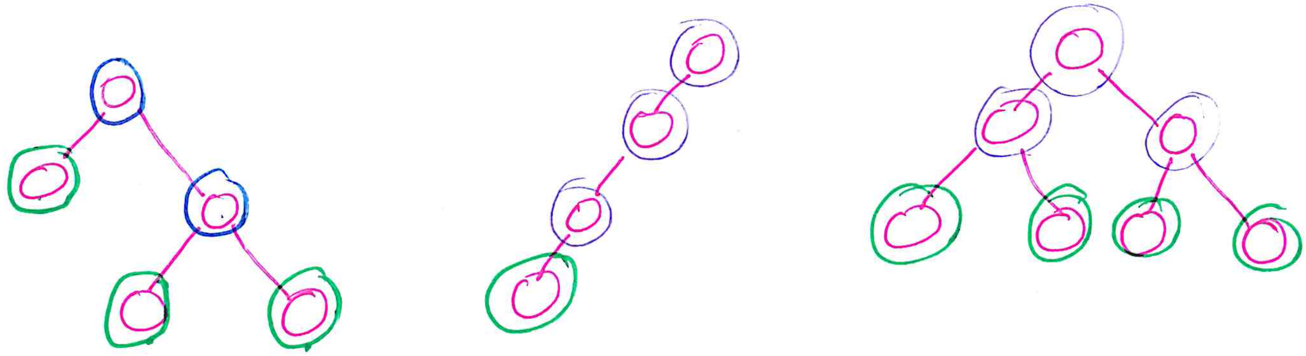
Suppose that we want to prove P(n) holds for $n \geq 0$. To give a proof using strong induction for $[\forall n \geq 0: P(n)]$, we need to prove:

1. Base case P(0) is true.
2. Inductive case: $\forall n \geq 1: [P(0) \wedge P(1) \wedge P(2) \wedge \ldots P(n-1) \Rightarrow P(n)]$

Both induction techniques are equivalent

Terms for binary trees.

- Binary tree is binary because each node has 2 children.
- Each edge connects pairs of nodes.
- Node is leaf node if it has null for both children.
- Node is an internal node, if it is not a leaf node.



Leaf nodes = 3
internal nodes = 2
$3 \leq 2+1$ ✓

leaf nodes = 1
internal nodes = 3
$1 \leq 3+1$ ✓

leaf nodes = 4
internal nodes = 3
$4 \leq 3+1$ ✓

Claim: In any binary tree T;

$$\text{leaves}(T) \leq \text{internals}(T) + 1$$

we will try to prove this using strong induction.

Step 01:

Let T be the binary tree created by using rule (2) n # of times.

$P: \mathbb{N} \longrightarrow \{T, F\}$, $P(n)$ is defined as follows

$$P(n) = \begin{cases} T, & \text{leaves}(T) \leq \text{internals}(T) + 1 \\ F, & \text{otherwise} \end{cases}$$

$P(n)$ is true, for any binary tree T constructed by n applications of rule (2).

Step 2: state the variable of which we run the induction on.

variable $n \in \mathbb{N}$

step 3: State the base case

$n = 0$

Step 4: prove the base case:

The binary tree T constructed ~~with~~ using rule (2) 0 number of times is the empty tree.

For empty tree T,

leaves$(T) = 0$ internals$(T) = 0$

$0 \leq 0 + 1$ ✓

Step 05: Inductive step

Since we are using strong induction

$$\left[\forall n \geq 1 : P(0) \wedge P(1) \wedge P(2) \wedge \dots P(n-2) \wedge P(n-1) \Rightarrow P(n) \right]$$

If any binary tree B generated using $k < n$ # rule(2) applicatios has leaves(B) $\leq$ internals(B)+1, then the binary tree T generated using n rule (2) applications must follow leaves(T) $\leq$ internals(T) + 1

Step 06: proof of inductive step.

we will consider 2 cases.

case 1 : $n = 1$

case 2 : $n \geq 2$

case 1 : $n = 1$

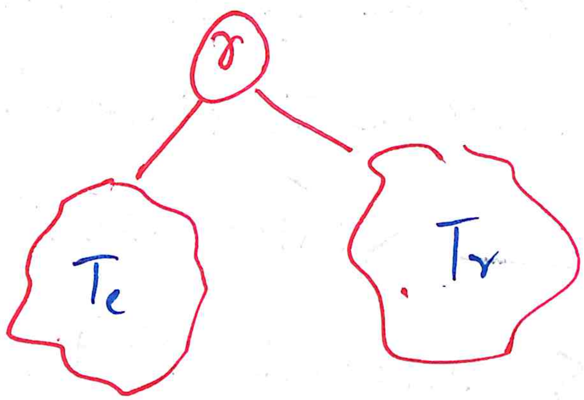
The only way to create binary Tree T with one application of rule (2) is to use rule(1) for both sub trees. So the tree T must only contain one node.

T contains 1 leaf node

T contains 0 internal nodes

$1 \leq 0 + 1$ ✓

Case 2: When $n \geq 2$, observe that tree must be generated using left subtree T_ℓ using ℓ # of rule (2) applications & right subtree T_r must be generated using r # of rule (2) applications. Then we need to use rule (2) once more to tie them together



Therefore, for T, $n = \ell + r + 1$, $\ell < n$, $r < n$

Now we are going to apply inductive hypothesis

$$\text{leaves}(T_\ell) \leq \text{internals}(T_\ell) + 1$$

$$\text{leaves}(T_r) \leq \text{internals}(T_r) + 1$$

Also, observe that either $T_\ell \neq \text{null}$ or $T_r \neq \text{null}$ or both is not equal to null.

Therefore leaves of T are the leaves of T_ℓ and T_r and internal nodes of T is $T_\ell + T_r + 1$.

(root cannot be leaf node because one of T_ℓ or T_r is not null.

$\text{leaves}(T) = \text{leaves}(T_l) + \text{leaves}(T_r)$

$\text{internals}(T) = \text{internals}(T_l) + \text{internals}(T_r) + 1$

putting them together.

$$\text{leaves}(T) = \text{leaves}(T_l) + \text{leaves}(T_r)$$

$$\leq \text{internals}(T_l) + 1 + \text{internals}(T_r) + 1$$

$$\leq \underbrace{\text{internals}(T_l) + \text{internals}(T_r) + 1}_{\text{internals}(T)} + 1$$

$$\underbrace{\text{leaves}(T) \leq \text{internals}(T) + 1}_{p(n)}$$

Step 07: we have proven the base case & strong inductive case.

$\therefore$ using principles of strong mathematical induction

$$\left[\forall n \geq 0 : P(n) \right]$$

$\blacksquare$