Recursively Defined Structures / Sets

A set S defined by

(1) its smallest element (base case)

(2) rules that construct compound elements out of smaller elts.

(recursive case)

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

<u>ex</u> A nonnegative integer

(1) 0

(2) $1+K$ for nonnegative int K

How do I make 1?

0 is a nonnegative int (1)

1 is $1+0$ for a nonneg. int, 0

<u>ex</u> A linked list 

(1) An empty list < >

(2) A list <X, L> where x is data and L is a linked list

(list/sequence/array/tuple: order matters, dupes allowed)

1-element linked list:

$\langle x, \langle\rangle\rangle$

2-element linked list:

$\langle x_1, \langle x_2, \langle\rangle\rangle\rangle$

pni

$\downarrow$

<u>ex</u> A well-formed proposition φ of propositional logic over propositional vars. X is:

1) p, for some $p \in X$ (base case)
2) $p \star q$ where $\star \in \{\wedge, \vee, \Rightarrow, \Leftrightarrow, \oplus\}$, p, q well-formed prop.
3) $\neg p$, p well-formed prop.

$p \Rightarrow q \wedge r \vee p \cdots$ $\qquad X = \{p, q, r\}$

Proof by Structural Induction.

Used to prove $\forall x \in S : P(x)$ for recursively defined set S.

How:

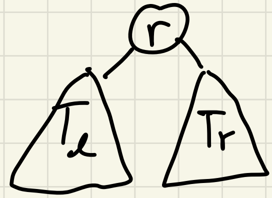
1) Prove $P(x)$ for all base cases of S
2) Prove that if $P(x)$ true for smaller elements of S, then true for larger elements.

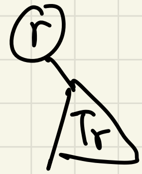
Def A binary tree T is either

(1) null (empty tree)

(null)

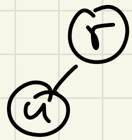
(2) root node r and T_ℓ, T_r, binary trees, attached to r with edges.





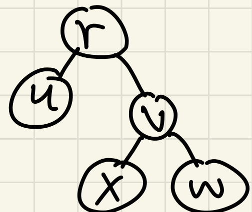
ex (null)

node/vertex



u is a child of r

r is the parent of u



w is a leaf

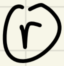
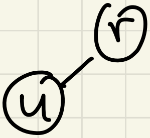
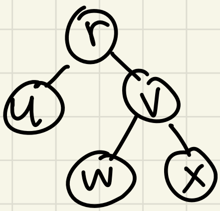
Terms:

- binary because each node has ≤ 2 children
- edges connect pairs of nodes
- node is a leaf if it has no children
- node is internal if it is not a leaf

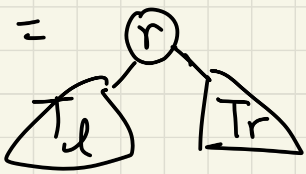
Claim: In any binary tree T,

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

ex

T	#leaves(T)	#int(T)	#int(T)+1	
(null)	0	0	1	0≤1 ✓
	1	0	1	✓
	1	1	2	1≤2 ✓
	3	2	3	✓

Let's build some intuition.

Suppose T =  and ≥1 of T_ℓ, T_r is not null

Then #leaves(T) = #leaves(T_ℓ) + #leaves(T_r)

because r is not a leaf and all leaves of T_ℓ and T_r are leaves of T.

$$\#\text{int}(T) = 1 + \#\text{int}(T_\ell) + \#\text{int}(T_r)$$

r

Claim: In any binary tree T,

$$\#leaves(T) \leq \#internals(T) + 1$$

$$\forall T \in \mathcal{T} : P(T) \quad P(T)$$

<u>Proof</u>: We use structural induction on the def. of binary tree.

Base case: WTS $P(null)$.
Since T is null, # leaves is 0
ints is 0. $0 \leq 0+1$, so $P(null)$ holds

Inductive case: We WTS

$\forall$ binary trees T, composed of root node r and binary trees T_ℓ, T_r

$$P(T_\ell) \wedge P(T_r) \Rightarrow P(T)$$

Suppose $P(T_\ell) \wedge P(T_r)$. That is,

$\#leaves(T_\ell) \leq int(T_\ell) + 1$ and

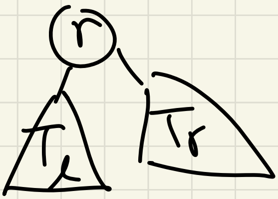
$\#leaves(T_r) \leq int(T_r) + 1$

WTS $P(T)$. That is, $\#leaves(T) \leq \#int(T)+1$

Case 1: T is just one node, a leaf.

r # leaves = 1 $1 \leq 0+1$ ✓
      # ints = 0

case 2: T has at least one of T_ℓ, T_r non-null.



So r is not a leaf.

r is an internal node.

$$\#int(T) = \#int(T_\ell) + \#int(T_r) + 1$$

$$\underline{\#leaves(T)} = \#leaves(T_\ell) + \#leaves(T_r)$$

$$\leq (\#int(T_\ell)+1) + (\#int(T_r)+1)$$

by inductive hypothesis

$$= \#int(T_\ell) + \#int(T_r) + 1 + 1$$

$$\leq \underline{\#int(T) + 1}$$

$$P(T_\ell) \wedge P(T_r) \Rightarrow P(T) \qquad \square$$