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)

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

ex A nonnegative integer

(1) 0

(2) $1+k$ for nonnegative integer k

how do I make 1?

0 is a nonnegative int. by (1)

1 is $1+0$ by (2)

now I can make 2, etc.

ex A linked list is either

(1) An empty list $\langle\ \rangle$ (list: order, dupes matter)

(2) A list $\langle x, L \rangle$ where x is a data element and L is another list

how do we define a 1-element linked list?

$\langle x, \langle\rangle\rangle$ because $\langle\rangle$ is a LL by (1)

2-element?

$\langle y, \langle x, \langle\rangle\rangle\rangle$ bc a list

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

1) p, for some $p \in X$
2) $p \circledast q$ where $\circledast \in \{\wedge, \vee, \Rightarrow, \Leftrightarrow, \oplus\}$
3) $\neg p$

for well-formed propositions p, q.

Say $X = \{p, q, r\}$. this def. allows us to generate any compound prop. over these props.

$$p \Rightarrow q \wedge r \vee p \Leftrightarrow q$$

Proof by structural induction is used to prove $\forall x \in S : P(x)$

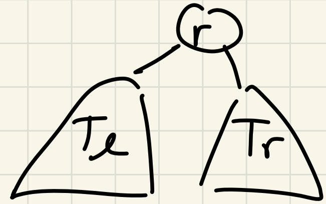
for recursively-defined sets S.

How:

1) Prove $P(x)$ true $\forall$ base cases of S

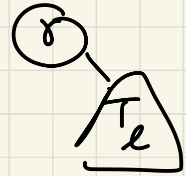
2) prove that if $P(x)$ true for smaller structures, then true for compound structures

Def A <u>binary tree</u> T is either



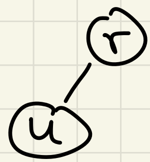
1) null (empty tree)

2) root node r and T_ℓ, T_r binary trees, attached to r w/ edge if non-empty

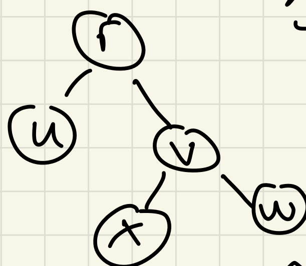


<u>ex</u> (null)

node/vertex



u is a child of r



w is a leaf

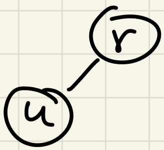
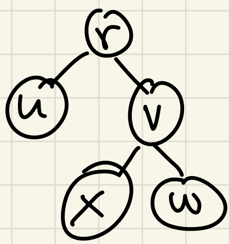
terms:

- <u>binary</u> bc each node has ≤ 2 children
- edges connect pairs of nodes
- node is a <u>leaf</u> if it has no children
- node is <u>internal</u> if it is not a leaf.

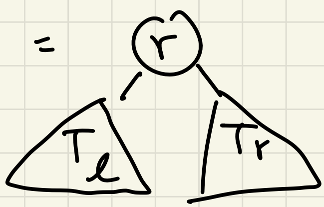
<u>claim</u> In any binary tree T,

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

ex

T	# leaves of T	# internals of T	# int(T) + 1	
(null)	0	0	1	$0 \leq 1$
r	1	0	1	$1 \leq 1$
	1	1	2	$1 \leq 2$
	3	2	3	$3 \leq 3$

let's get some intuition about why we would use induction to prove this.

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

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

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

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

r internal

all internals of T_ℓ and T_r internals of T.

Proof

① For any binary tree T, let $P(T)$ be the predicate that

$$\#leaves(T) \le \#internal(T) + 1$$

WTS $\forall$ binary trees $T : P(T)$. We use structural induction on the def. of binary tree.

② Base case: we WTS $P(null)$.
Since T is null, $\#$ leaves $= 0$ and $\#$ ints $= 0$. $0 \le 1 + 0$, so $P(null)$ holds.

③ Inductive case: We WTS

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

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

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

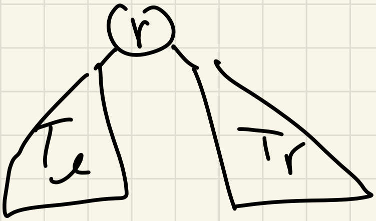
$\#leaves(T_\ell) \le internal(T_\ell) + 1$ and
$\#leaves(T_r) \le internal(T_r) + 1$.

WTS $P(T)$, that is, $\#leaves(T) \le \#int(T) + 1$.

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

r $\quad$ $\#$ leaves $= 1$ $\quad$ $1 \le 0 + 1$ ✓
$\#$ internal $= 0$

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.

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

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

$$\leq \big(\#\text{int}(T_\ell)+1\big) + \big(\#\text{int}(T_r)+1\big)$$

by inductive hypothesis $P(T_\ell) \wedge P(T_r)$

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

rewriting

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

substitution w/ $(*)$

So we have shown that $P(T_\ell) \wedge P(T_r) \Rightarrow P(T)$.

$\square$