Rundall Munroe: http://xkcd.com/244/

CS151 Fall 2014 Lecture 14 – 10/9

Recursive/Inductive Definitions

Prof. Tanya Berger-Wolf
http://www.cs.uic.edu/CS151

Adapted from Lap Chi Lau – The Chinese University of Hong Kong
and David Liben-Nowell – Carleton College

Strings

Let $\sum^*$ be the set of strings over alphabet $\sum$

Base Case: The empty string $\lambda \in \Sigma^*$
Recursive Step: If $w \in \Sigma^*$ and $x \in \Sigma$ then $wx \in \Sigma^*$

Example

$$\Sigma = \{a, b, c\}$$

Σ^*: λ
$\lambda a = a$ $\lambda b = b$ $\lambda c = c$
aa, ab, ac, ba, bb, bc, ca, cb, cc

aaa, aab, aac, aba, abb, abc, aca, acb, acc, ...

Strings

Let $\sum^*$ be the set of strings over alphabet $\sum$

The empty string $\lambda \in \Sigma^*$
If $w \in \Sigma^*$ and $x \in \Sigma$ then $wx \in \Sigma^*$

Example

$$\Sigma = \{1, 01\}$$

Σ^*: λ
$\lambda 1 = 1$ $\lambda 01 = 01$
11, 101, 011, 0101
111, 1101, 1011, 10101, 0111, 01101, 01011, 010101

For all w in $\sum^*$, w has no consecutive 0's and ends in 1

Let $\sum^*$ be the set of strings over alphabet $\sum$

The empty string $\lambda \in \Sigma^*$
If $w \in \Sigma^*$ and $x \in \Sigma$ then $wx \in \Sigma^*$

$$\Sigma = \{1, 01\}$$

For all w in $\sum^*$, w has no consecutive 0's and does not end in 0

Where is n? Need a notion of length, *L(w)*.
Recursively defined: *L(λ)=0, L(wx) = L(w) +1*

Any letter is only length 1

For all n >= 0, all w of length n in $\sum^*$ have no consecutive 0's and does not end in 0

P(n) := all w of length n in $\sum^*$ have no consecutive 0's and does not end in 0

Base Case: P(0) := "all w of length 0 in Σ^* have no consecutive 0's and does not end.."
The only string of length 1 are 1, and 01

Inductive Step: Assume P(n-1) and prove P(n)

u of length n is wx for some string w of length n-1 and letter x.
By inductive hypothesis, w has no consecutive 0's and does not end in 0.
x is either 1 or 01 which will not create consecutive 0's in wx and ends in 1 (not 0).

Let $\sum^*$ be the set of strings over alphabet $\sum$

Base Case: The empty string $\lambda \in \Sigma^*$
Recursive case : If $w \in \Sigma^*$ and $x \in \Sigma$ then $wx \in \Sigma^*$

$\Sigma = \{1, 01\}$

For all *w* in $\sum^*$, *w* has no consecutive 0's and does not end in 0

Or we could go directly from the recursive definition

Base Case: λ has o consecutive 0's and does not end in 0.

Inductive Step: Assume about w and prove about wx

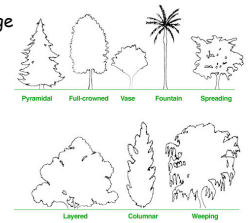
By inductive hypothesis, *w* has no consecutive 0's and does not end in 0.
x is either 1 or 01 which will not create consecutive 0's in wx and both end in 1.

Can follow the recursive (inductive) definition directly.
STRUCTURAL INDUCTION

Trees

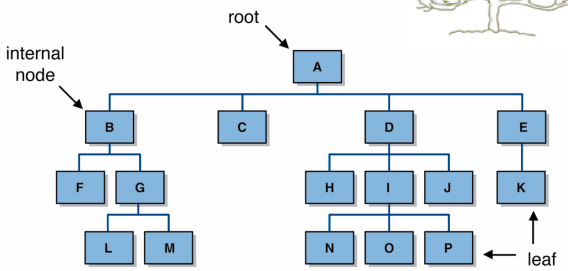
- **tree**: a set of linked vertices/nodes; a node may link to more than one other node
- a tree has a starting node called a **root** ; all other nodes are reachable from the root by the links between them
- a node in a tree that does not link to other nodes is called a **leaf**

A single node *r* is a rooted tree
Suppose *T1, T2, …, Tn* are disjoin rooted trees with roots *r1, r2, …, rn*
Then a new vertex *r* connected by a new edge to every root *r1, r2, …, rn* is a rooted tree



6

Tree diagram



7

Binary Trees

Base case: An empty tree NULL is a binary tree
A single node *r* is a binary tree
Recursive case: If *Tl, Tr* are disjoin binary trees with roots *r1, r2*
Then a new vertex *r* connected by a new edge to every root *r1, r2 is* a binary tree

Claim: In any binary tree T with at least one node, we have
(# of leaves in T) ≤ (# of internal nodes in T) +1.

Claim: *In any binary tree T with at least one node, we have*

$$(\#\text{ of leaves in } T) \leq (\#\text{ of internal nodes in } T) + 1.$$

Proof. We proceed by structural induction on T.

Base case ($T =$ `null`): Then T contains 0 leaves and 0 internal nodes. And indeed $0 \leq 0 + 1$.

Inductive case (T has root x, left subtree T_ℓ, and right subtree T_r): We assume the inductive hypotheses, namely

$$(\#\text{ of leaves in } T_\ell) \leq (\#\text{ of internal nodes in } T_\ell) + 1$$
$$(\#\text{ of leaves in } T_r) \leq (\#\text{ of internal nodes in } T_r) + 1$$

If x is itself a leaf, then $T_\ell = T_r =$ `null`, and therefore T contains 1 leaf and 0 internal nodes. Otherwise, the leaves of T are precisely the leaves of T_ℓ and T_r, and so

$$\begin{aligned} &(\#\text{ of leaves in } T) \\ &= (\#\text{ of leaves in } T_\ell) + (\#\text{ of leaves in } T_r) \\ &\leq (\#\text{ of internal nodes in } T_\ell) + 1 + (\#\text{ of internal nodes in } T_r) + 1 \end{aligned}$$

by the inductive hypotheses. Because $T_\ell \neq$ `null` or $T_r \neq$ `null`, the root node x is an internal node, and thus

$$\begin{aligned} &(\#\text{ of internal nodes in } T_\ell) + 1 + (\#\text{ of internal nodes in } T_r) + 1 \\ &= (\#\text{ of internal nodes in } T) + 1. \end{aligned}$$

□

Back to Binary Search
Extra credit with a written inductive proof

I have a number between 0 and 63. You ask a question, I'll tell you yes or no.

How long will it take you to find my secret number?

```
BinarySearch (0…n-1)
middle = floor((n-1)/2)
if (middle == "secret number")
     return (middle)
else if (middle > "secret number")
          BinarySearch(0..middle)
     else
          BinarySearch(middle+1..n-1)
```