CS180 – Trees

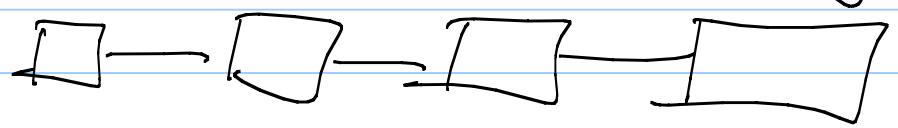
Note Title 10/28/2009

Announcements:

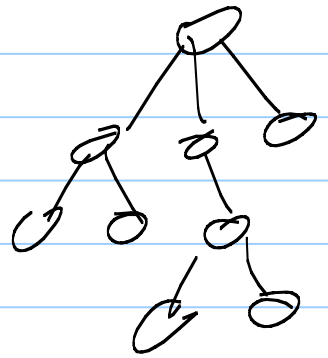
- HW due today (midnight)
- Program will be posted later today
  - Program 2 was graded & returned
  - Next exam in ~2 weeks

Ch 6 - Trees

All data structures so far have expressed linear orderings:

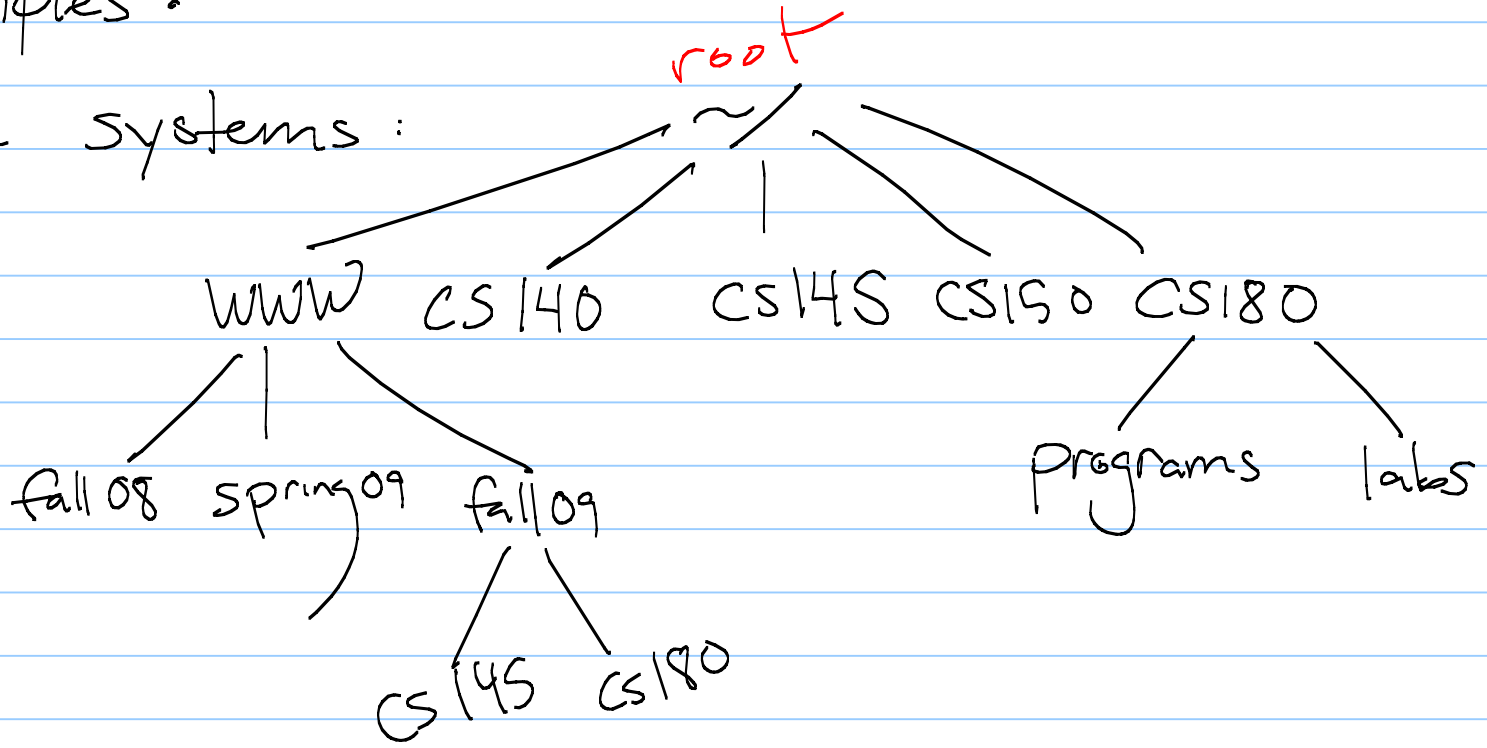


Some structures require more complex relations.



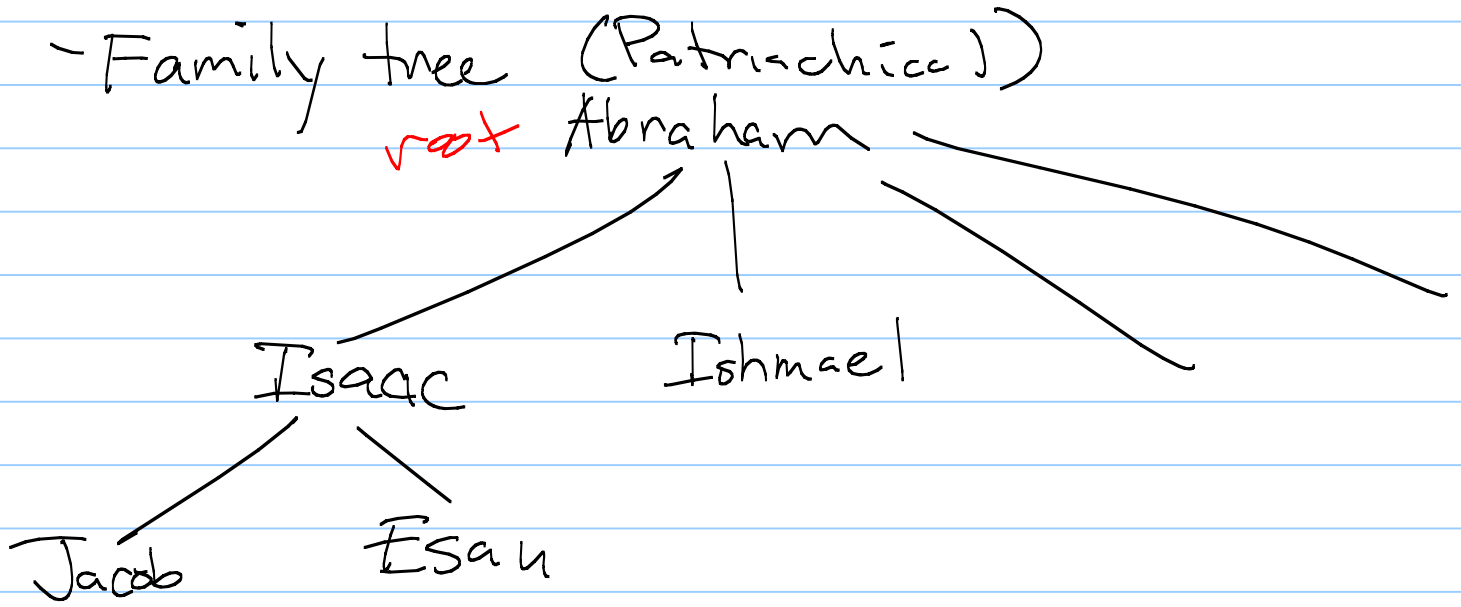
Examples:

- File systems:



Ex:

- Family tree (Patriachical)



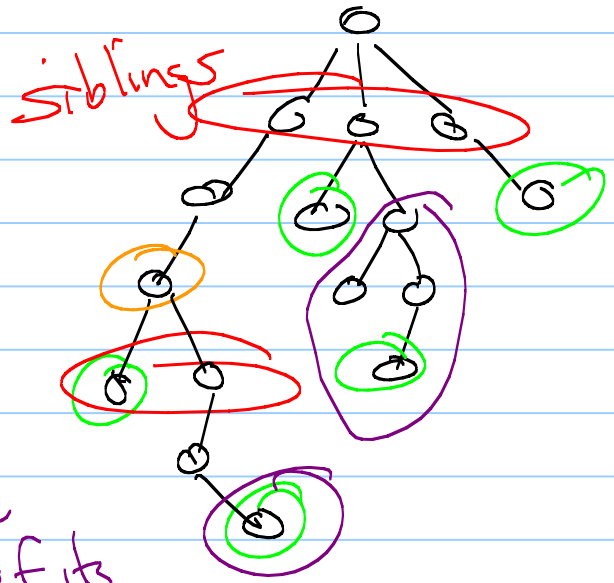
Definitions

A tree is set of nodes storing elements in a parent-child relationship.

- T has a special node, r, called the root
- Each node (except r) has a unique parent

More dfs

- child
- siblings - share common parent
- leaves — no children
- internal nodes - has parent and children
- rooted subtree — node v and all of its descendants
- ancestor
- descendant



Tree ADT (from book)

What sort of data might a tree class need?

Node structure:
- data or element
- parent pointer
- list or vector of child pointers
- counter for # of children

Private data:
- pointer to root or root node
- size

Tree functions:

O(1)

element(v): return element stored in node (or iterator) v.

root(): return iterator to root

parent(v): return the parent of v

O(# children)

children(v): return list of children

More functions:

- isInternal(v)
- isLeaf(v)
- isRoot(v)

$O(1)$

O(1)

size(): return # of nodes

swapElements(v, w): switch contents of nodes v & w

replaceElement(v, element)

replace WWW with public_html

Basic Algorithms

What is running time of previous methods?)

Most are O(1).

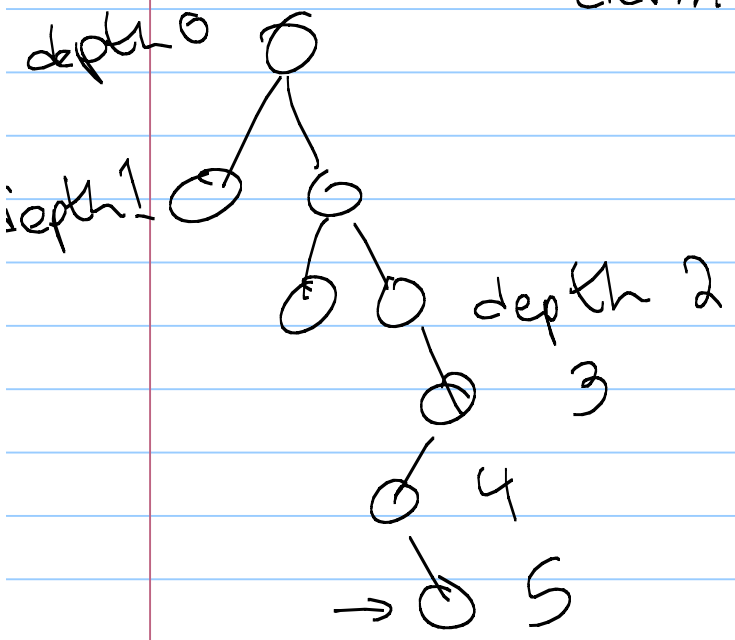
To actually traverse + print out contents of the tree would take linear time - O(n).

Computing Depth

depth: how deep a node is

defined recursively:
root has depth 0

every other node:
depth(v) = depth(parent(v)) + 1

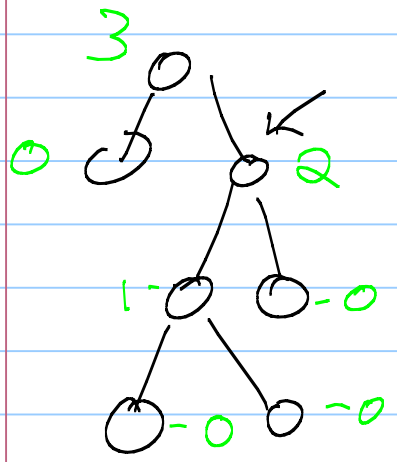


(Easy to give recursive algorithm.)

O(depth of tree)

Computing Height

Height of a leaf = 0



Height (v) = max (height of children) + 1

leads to recursive alg.

How long?

O(size of subtree rooted at v)