Outline

Last time:

- Finished up hashing
- Binary search, divide-and-conquer
- Recursive function and recurrence relation

Today:

- Tree Terminology
- Tree Traversal
- $N$-ary Tree
Divide and Conquer

Divide and conquer doesn’t work with linked list, unfortunately

So why use linked-list at all?

How can we speed up search and use dynamically allocated space?
Unsorted dictionary: Hash table
Sorted dictionary: Tree
Trees

Arboreal examples: birch, cherry, maple, oak, pine, plum, willow . . . .

Sugih Jamin (jamin@eecs.umich.edu)
Organizational Examples

Orgchart:

- CEO
  - VP Product Development
    - Bathroom
    - Bedroom
  - VP Marketing
    - N. America
  - VP Sales
    - Europe
    - Asia

Sugih Jamin (jamin@eecs.umich.edu)
Organizational Examples (cont)

Family tree:

- Grandparent
  - Uncle
  - Parent
    - Brother
    - You
    - Sister
      - Son
      - Daughter
    - Aunt
      - Niece
      - Son
      - Daughter
      - Nephew
    - 1st cousin
      - 2nd cousin
    - Grandchild
Organizational Examples (cont)

Object inheritance:

- array
  - deque
    - queue
    - stack
  - vector
  - tree
    - BST
    - Heap
    - Trie
    - k-d tree
    - B-Tree
      - BSP
      - Octree

Note: no multiple inheritance (or it won’t be a tree)
What is a Tree?

Tree: a set of nodes storing elements in a parent-child relationship such that:

- there is one root, the topmost node
- the root node has no parent
- all other nodes have exactly one parent
- parent-child relationship is denoted by direct link in tree
- there is a unique path from one node to another

Sugih Jamin (jamin@eecs.umich.edu)
Tree Terminology

$T$:

- $r$: root node of tree $T$
- $k$ is a child of $r$
- $k$ is a parent of $m$
- $m$ is a grandchild of $r$
- $j, x, z$ are siblings of $m$
- $j, l, n, x, z$ are leaf nodes
- degree of a tree: max number of children each node can have, this tree is of degree 2 (a binary tree)

Sugih Jamin (jamin@eecs.umich.edu)
Tree Terminology (cont)

- $r$: root node of tree $T$
- $T_r$: right subtree of $r$
- $T_l$: left subtree of $r$
- A path: the set of nodes visited to get from a node higher up on the tree to a node lower down
- A path from $r$ to $l$ is $\{k, m, l\}$
- The path length of $r$ to $l$ is 3 (hops)
- Path length may be 0, $\{i, i\}$ is a path

Sugih Jamin (jamin@eecs.umich.edu)
Tree Terminology (cont)

- ancestor: \( k \) and \( m \) are ancestors of \( n \), there is a path from \( k \) to \( m \) and \( m \) to \( n \)
- each node \( i \) is its own ancestor
- \( i \) is a proper ancestor of \( j \) if the path length from \( i \) to \( j \) is not 0
- descendant: \( m \) and \( n \) are descendants of \( k \), there is a path from \( k \) to \( m \) and \( k \) to \( n \)
- \( j \) is a proper descendant of \( i \) if the path length from \( j \) to \( i \) \( \neq 0 \)
Tree Terminology (cont)

- **depth** of node $i$ is the length of path from the root to $i$
- $\text{depth}(r) = 0$, $\text{depth}(l) = 3$
- all nodes on a **level** of the tree have the same depth, the root is at level 0
- the **depth of a tree** is the max depth of all nodes, this tree is of depth 3
- **height** of node $i$ is the longest path from $i$ to a leaf node
- $\text{height}(l) = \text{height}(z) = 0$, $\text{height}(m) = \text{height}(y) = 1$, $\text{height}(k) = 2$, $\text{height}(r) = 3$
Binary Tree

Binary tree: every node has 0, 1, or 2 children

Proper binary tree: every node has 0 or 2 children

Perfect binary tree: every level is fully populated

Complete binary tree: every level except the lowest is fully populated, the lowest level is populated left to right
Binary Tree Representation

A binary tree can be represented either as a linked structure:

- as an ordered set:
  \[ \{r, \{k, \{j\}, \{m, \{l\}, \{n\}\}\}, \{y, \{x\}, \{z\}\}\} \]

- or as an array (starting at index 1):
  \([-, r, k, y, j, m, x, z, -, -, l, n \]}

For a binary tree, a node at index \(i\) has its children at which indices?

A node at index \(i\) has its parent at which index?
Example Use of Binary Tree: Expression Tree

Encode $a/b + (c - d)e$ as an expression tree:

Tree traversals:

- depth first:
  - **pre-order**: visit node, $T_l$, $T_r$
  - **post-order**: visit $T_l$, $T_r$, node
  - **in-order**: visit $T_l$, node, $T_r$
- breadth first: level by level
Binary Tree Traversal: Expression Tree

\[ \frac{a}{b} + (c - d)e : \]

Print the expression tree in the “normal” order:
\[
(((a)/(b)) + (((c) - (d)) \times (e)))
\]

Which kind of tree traversal will you need?

Give me a recursive and an iterative implementation
$N$-ary Trees

A tree may be empty

**External** node: an empty node with no children

**Internal** node: a node with children

**Leaf** node: an internal node with only external nodes as children

How many external nodes does an $N$-ary tree with $n$ internal nodes have?
Characteristics of $N$-ary Trees

How many external nodes does an $N$-ary tree with $n$ internal nodes have?

binary tree:

- $n = 0$: 1
- $n = 1$: 2
- $n = 2$: 3

tertiary tree:

- $n = 0$: 1
- $n = 1$: 3
- $n = 2$: 5

4-ary tree:

- $n = 0$: 1
- $n = 1$: 4
- $n = 2$: 7

Every new internal node replenishes one external node and brings with it $N - 1$ new external nodes.

For $n$ internal nodes, we have $n(N - 1) + 1$ (original) external nodes.

For binary tree, $n$ internal nodes means $n + 1$ external nodes $\Rightarrow \max \left\lceil \frac{n}{2} \right\rceil$ leaf nodes.

How many internal nodes does an $N$-ary tree with $m$ external nodes have?
More Characteristics of $N$-ary Trees

How many links are there in an $N$-ary tree with $n$ internal nodes?

What is the maximum height of a binary tree of $n$ internal nodes?

How many internal nodes does it take to fully populate level $l$ of a binary tree?

What do you call a tree of $l$ levels that are fully populated?

How many internal nodes are there in a perfect binary tree of height $h$ ($h + 1$ levels)?

How many levels of a binary tree are needed to hold $n$ internal nodes?

What is the minimum height of a binary tree of $n$ internal nodes?