CS 61B Spring 2017

Heaps & Graphs

Discussion 10: March 22, 2017

1 Heaps

1.1 Is an array that is sorted in descending order also a max-oriented heap?

True, the heap invariant holds.

1.2 Are the values in an array-based min-heap sorted in ascending order?

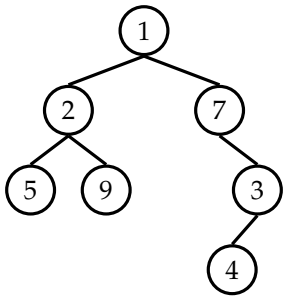
Not necessarily. We can have higher or lower priority items loaded on one branch of the tree.

1.3 The largest item in a heap must appear in position 1, and the second largest must appear in position 2 or 3. Give the list of positions in a heap where the kth largest can appear for $k \in \{2, 3, 4\}$. Assume values are distinct.

$k = 2$ can be in $\{2, 3\}$. $k = 3$ can be in $\{2 \dots 7\}$. $k = 4$ can be in $\{2 \dots 15\}$.

Consider complete binary trees with the largest values contained on one branch of the tree for a lower bound and consider how far the kth element can be from the root for an upper bound.

2 Traversals



Level-Order Traversals Nodes are visited top-to-bottom, left-to-right.

Depth-First Traversals Visit "deep nodes" before shallow ones.

2.1 Give the level-order traversal of the tree.

1 – 2 – 7 – 5 – 9 – 3 – 4

2.2 Give the depth-first traversal of the tree.

(a) Pre-Order

1 – 2 – 5 – 9 – 7 – 3 – 4

(b) In-Order

5 – 2 – 9 – 1 – 7 – 4 – 3

(c) Post-Order

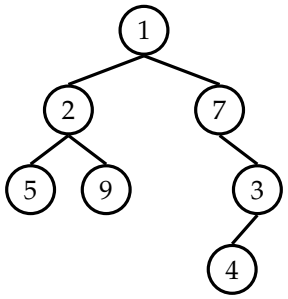
5 – 9 – 2 – 4 – 3 – 7 – 1

3 Searches

3.1 What is the difference between a *traversal* and a *search*?

A traversal normally iterates across all vertices in a graph while a search terminates once a goal state (or solution) is found.

```
function TREE-SEARCH(start)
    fringe ← java.util.Queue interface
    ADD(start, fringe)
    while fringe is not empty do
        node ← REMOVE(fringe)
        if node is the goal then return node
        for child in NEIGHBORS(node) do
            ADD(child, fringe)
    return failure
```



3.2 Give the order in which nodes are *visited* in a search of the tree if the fringe is a first-in, first-out Queue abstract data type.

1 – 2 – 7 – 5 – 9 – 3 – 4

3.3 Give the order in which nodes are *visited* in a search of the tree if the fringe is a last-in, first-out Stack abstract data type.

1 – 7 – 3 – 4 – 2 – 9 – 5

Output appears to explore the tree right to left because of the order in which elements are added to and removed from the stack.

```
function GRAPH-SEARCH(start)
    seen ← an empty set
    fringe ← java.util.Queue interface
    ADD(start, fringe)
    while fringe is not empty do
        node ← REMOVE(fringe)
        if node is the goal then return node
        if node is not in seen then
            ADD(node, seen)
            for child in NEIGHBORS(node) do
                ADD(child, fringe)
    return failure
```

3.4 In the graph search pseudocode, why is it necessary to keep track of a *seen* set?

To prevent an infinite loop when entering a cycle in the graph.

3.5 Give a tight asymptotic runtime bound for BFS and DFS on a graph $G = (V, E)$.

$O(|V| + |E|)$ for both searches.