CSE13S notes
Week of 2/24/23

Linked lists (LL)

- Can be thought of as a sequence of nodes, each containing a pointer to the next node
    - There are singly linked lists and doubly linked lists. A singly linked list just contains one pointer to the next node while a doubly linked list contains a pointer to the previous node as well
- Linked structures: structures where nodes are linked together (generally through the use of pointers)
- Advantages:
    - No fixed memory allocation
        - Can grow and shrink at run-time
- Disadvantages
    - Must allocate memory for each node
    - Arrays are friendlier to processor cache and are more memory efficient than linked lists
    - Traversal: cannot randomly access memory, must traverse all elements up the element we want to access
    - Reverse traversing is difficult in singly linked lists
        - Easy in doubly linked lists but those use more memory to store the extra pointer
- Doubly linked lists
    - Each node has a pointer to prev and next nodes
- Sentinel nodes
    - Designated dummy nodes used to mark specific points in a LL
    - In a doubly linked list there is a dummy node at the start and end

LL functions to implement

- Linked list destructor
    - Walks the linked list and delete each node
- Lookup:
    - Walk the list to look for key (O(n) time complexity)
- Inserting
    - Walk the list to check if element already present (for sets), else insert in front, move head
- Remove

    - Keep track of prev and curr nodes. When element to remove found, remove that (deallocate memory!) and change the pointers of the prev node to point to next node
- Poplist
    - For stack type linked lists
    - Disconnects and returns the head of the linked list
- Dropping
    - Disconnects and returns the tail of the linked list
- For doubly linked lists, having a sentinel head and tails nodes makes logic significantly easier, but costs more memory

2/24/23

Trees

- Tree is a type of direct acyclic graph, typically composed of nodes
- Exactly one path between 2 nodes

What's a node?

- Smallest entity in a tree
- Generally contains some value or key
- Binary tree:
    - Each node has up to 2 children
    - Generally implemented using structs, where pointers point to 2 children
    - Some implementations don't track the parents
- K-ary tree
    - Each node up to k children
    - A 2-ary tree is a binary tree

Terminology

- Generally, trees are visualized upside down
- Terms
    - root : origin node
    - Parent : higher level node of a specific node
    - Child : lower level node of a specific node
    - Subtree: a tree within a tree
    - Leaf : a 1-ary tree within a tree (verify def.)
    - Traversal : visiting each node exactly once

Traversal methods:

- preorder : key, left, right
- Inorder : left, key, right
- Postorder : left, right, key
- Level-order : same as BFS in a graph
    - Requires a queue

Binary search tree (ordered trees)

- Order is not necessary, but ordered trees are more useful
- A tree based on binary search
    - Key less than a node's value goes under left subtree
    - Key greater than goes in the right
    - Duplicates ignored (set)
- Balanced trees are more useful
    - A 1-ary tree isn't going to be very useful, linear time to search
    - A balanced binary tree emulates binary search
    - In a balanced tree, the height of the 2 subtrees from origin should not have a difference greater than 1

Partial order

- Not completely sorted
    - For ex: in a heap, the parent is greater than (or lesser than) both of their children, but the children themselves are not ordered

See slides for BST manipulation functions

- Removing a key:
    - Trickiest operation
    - Uses DFS to find node containing key
        - Three cases: node to remove missing left or right child, or node to remove has 2 children

Summary of trees:

- Widely used in CS

Data Compression

- Claude shannon (watch bit layer)
    - Father of modern information theory
    - Broke down communication into five parts:
        - Information source
        - Transmitter
        - Channel
        - Receiver
        - Destination
- Information source sends message -> transmitter -> a noise source applies noise to data -> receiver receives signal from transmitter ->message sent to destination

Information source

- Produces a message, or sequence of messages, to be communicated to the receiving terminal
- Messages take on various forms:

Channel

- The medium through which a signal a transmitted

Entropy

- Defined by shannon as the measure of uncertainty of occurrences of events

Run length coding

- Include number after an element such as char ‘A’ to indicate how many of that element occurred there

Huffman coding

- Developed by David A. Huffman
    - Distinguished member of UCSC

Building a huffman tree