CS280 Notes 3/23

By Christian Sagel

I am off by 25 probes on the stress test. I increment probes when I am.
The probe counts are one of the most difficult parts of the assignment to match right.
Mead sounded sad about this being his last class with us.

Graphs

Data structures to represent the graph: Adjacency matrix or adjacency list.
To traverse depth-first, we use a stack.
To traverse breadth-first, we use a queue.

We will be using an adjacency list for the assignment. We can use STL vector, list, etc.
The order n which we put the neighbors in the container is not arbitrary. If its weighted, we have to follow a specific way. (Most expensive, least expensive)

You can derive from queue and have front call top.
We will have a pointer to a linked list of all your neighbors.
Think on what we can use to keep track of the things to look for.

Spanning tree
A tree that is embedded in the graph. One of the fundamental differences is that a graph can have cycles, nodes can have multiple “parents”.
If we have 3 nodes in a binary tree, we know we are going to have exactly 2 links.
So we have n-1 edges.
If we have a graph we want to find the minimum number of edges in the graph for all nodes to be connected.
We want to know the set of edges in our graph that will have the least amount and keep them connected.
Spanning trees were invented for networking routing protocols. They wanted to find the cheapest route, removing redundancies.
Two-well known algorithms for finding minimum spanning trees from the graph: Prim’s algorithm and Kruskal’s algorithm.
Greedy algorithms: They will choose what seems to be the best choice at the time.

Prim’s algorithm using a tree: Choose any vertex in the graph, add it to an empty tree, until all nodes are in tree (choose the edge of least cost that emanates from a node in the tree. Add that edge and vertex to the tree)
To break ties for edges with same weights, pick arbitrarily. Though Mead suggests choosing by IDs, which are known to be unique.

Kruskal's algorithm using a forest: Construct a forest from the N nodes in the graph. Pt the sorted edges in a queue. Until there are N - 1 edges in the forest (a single tree), extract the "cheapest" edge from the queue.

To find whether there will be a cycle, there are Union-Find algorithms.
You pick representatives for components based on relationships; to find whether one of the nodes is a member of a set.

2-3 search trees:
2-3-4 trees: Radically different from BTrees.
A BST is a form of a 2-3-4 tree in that all nodes are always 2 nodes.
Red-Black trees are BSTs (all odes are 2-nodes). Red-black trees are used to represent (2-3-4) trees. Newly inserted nodes are marked as RED.
We don't speak about height. We know the tree will be balanced if the relationship between red and black nodes hold true.

"If purple were the coolest color on the color printer, we would be talking about Purple-Black trees right now" - Mead

A shortest path algorithm…

In most of our graph algorithms, we don't really implement the auxiliary data structures to support them, rather use library ones, etc.