# CS280 – More Graphs
# March 23, 2016

***http://azrael.digipen.edu/~mmead/www/Courses/CS280/Graphs-1.html***

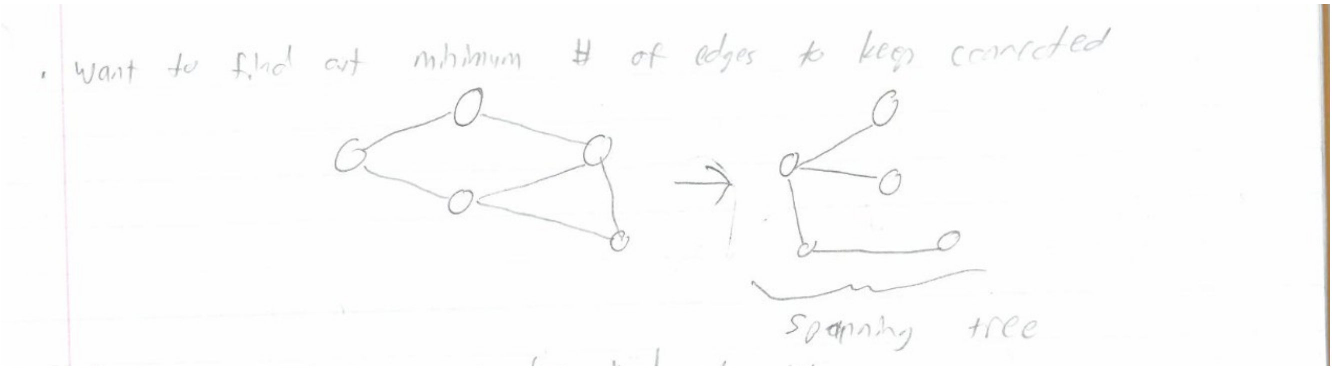
***Homework Questions***

- I'm off by a handful of probes and I don't know why
  - Only in stress test
  - It is ok to share output and drivers on the forum
  - Try dumping the table after every step and diffing so that you can find exactly where the problem is
- In the real world we are writing our own drivers
  - Spend a lot more time testing that writing code

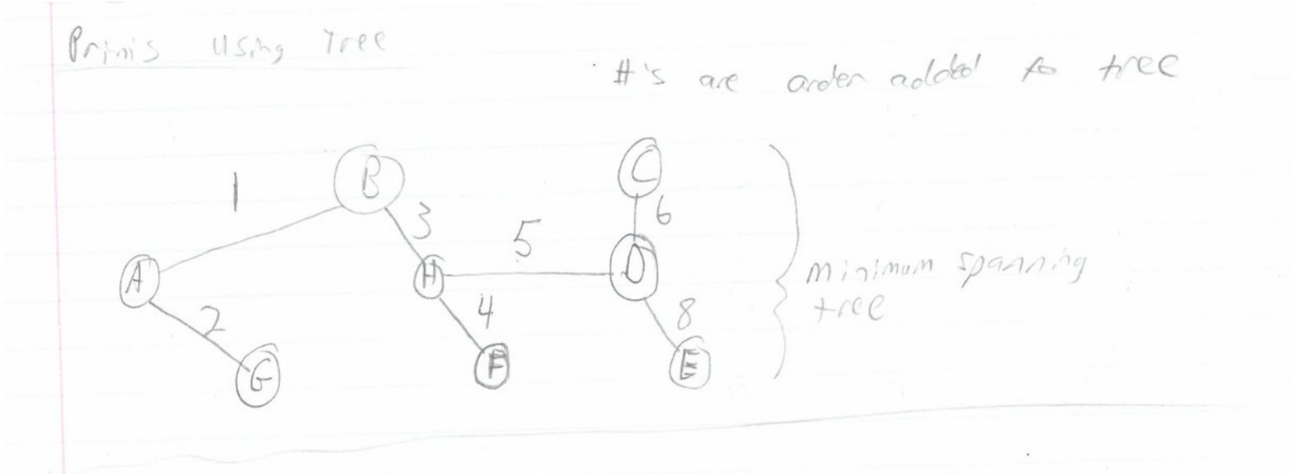
***Graphs***

- Adjacency matrix or adjacency list to represent graphs
- We will be using a list for the assigment
  - Use anything from the STL
- Everything will be weighted in the assignment
- Possible flaw in STL
  - Front() to get first element in queu
  - Top() to get first element in stack
  - To possibly fix make a myQueu class and top() function that just calls front()
- A spanning tree is a tree embedded in a graph
- If you have N nodes in a tree, you have n-1 links

- Prim's algorithm and Kruzkal's algorithm
  - Can't have cycles
- Complexity is mostly determined by the underlying data structure (array, list, etc)
- Greedy algorithm
  - Take best option **at that time**, not necessarily best overall
- How would you detect if you are forming a cycle?
  - Pick a node from each tree as a representative
  - Check representatives of each tree against each other for conflicts

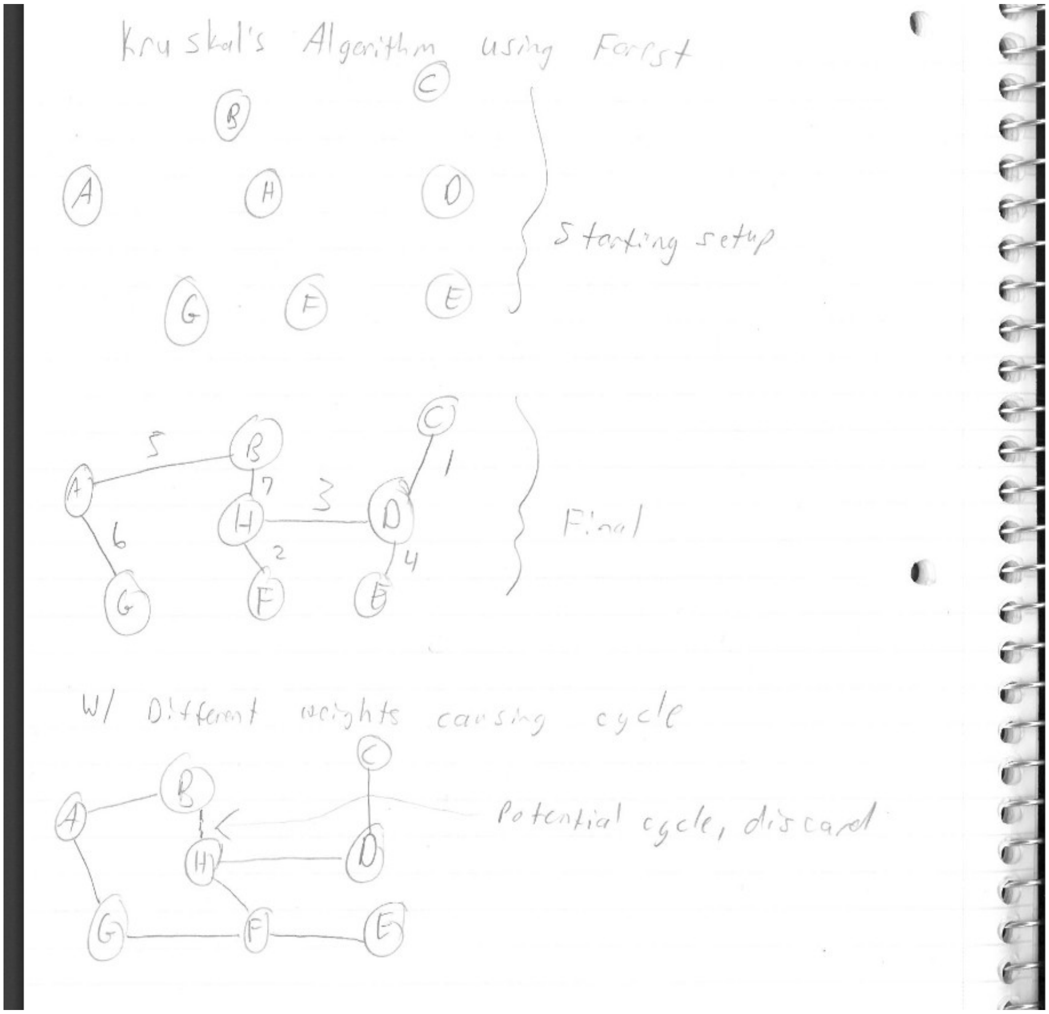


- want to find out minimum # of edges to keep connected



Prim's using Tree

- #'s are order added to tree


Kruskal's Algorithm using Forest
A
B
C
D
E
F
G
H
Starting setup
5
7
3
1
6
2
4
Final
W/ Different weights causing cycle
Potential cycle, discard

***Back to trees***

- Balancing for red-black trees as well as color
    - You can just store one extra bit and get red-black functionality from a BST!
- Tree is balanced as long as two constraints are met

- Don't have to check the height!

- Most operations done with a simple color swap