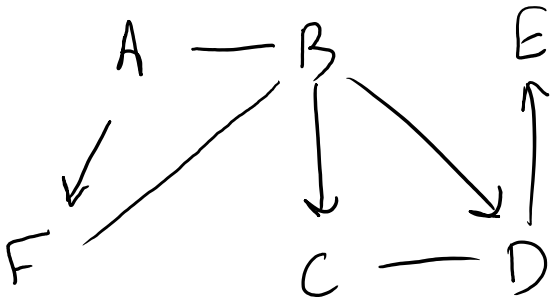
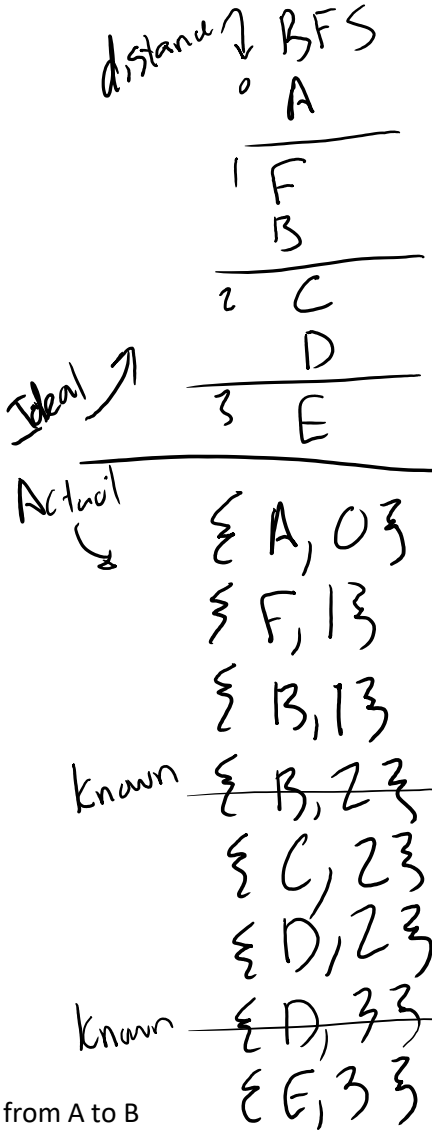
# 2018-11-06 Graph Search

Tuesday, November 6, 2018 3:14 PM




Question: What is the shortest path between A and E?

- Solving this requires a breadth-first search
  - Implies using a queue and a "distance" counter

BFS pseudocode:

1. Push{start, 0} into queue
2. While queue is not empty:
   a. Pop top KVP {node, distance}.
   b. Make top known
   c. For each outgoing edge:
      i. If Not known, push {node, distance + 1}

## Wrinkle: What if edges had weights?

- Example: google maps - what is the fastest time to get from A to B
- Consider edge weight instead of edge count
- In the above algorithm, simply replace the queue with a MIN Heap

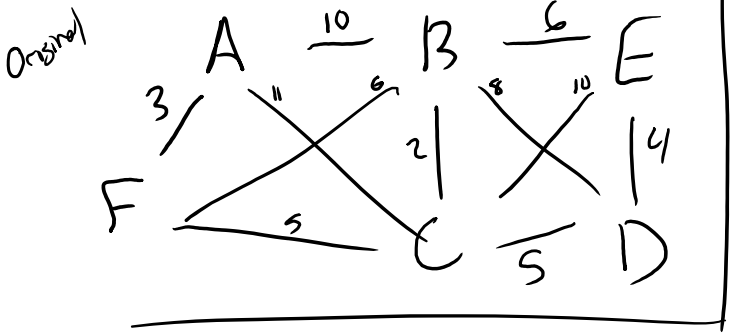
## Minimum Spanning Trees (MST)

- What are the essential edges such that:
  - The graph remains fully connected
  - The essentials have the least amount of weight
- Given an infinite way to run wire in a house, how can we do so such that all rooms have electricity and we spent the least amount of money to do this?
- MST does not guarantee that every node will be reachable using its shortest path

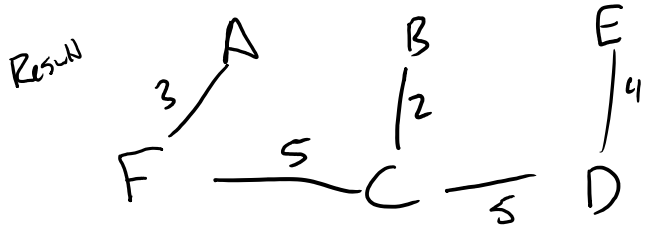
### Prim's MST Algorithm

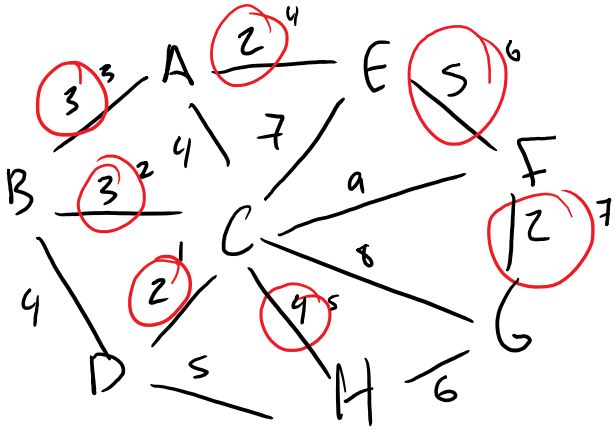
- We need to maintain a list of visited nodes
- Given some arbitrary starting location, push all outgoing edges into a min PQ
- While not all nodes have been visited:
  - Pop off least cost edge. If node has not been visited, push all of its outgoing edges onto the PQ. Mark edge as visited.

- Pop off least cost edge. If node has not been visited, push all of its outgoing edges onto the PQ. Mark edge as visited.




AF, 3 ✓
AB, 10
AC, 11
FC, 5 ✓
FB, 6
CB, 2 ✓
CD, 5 ✓
CE, 10
BE, 6
BD, 8
DE, 4 ✓




MST @ C