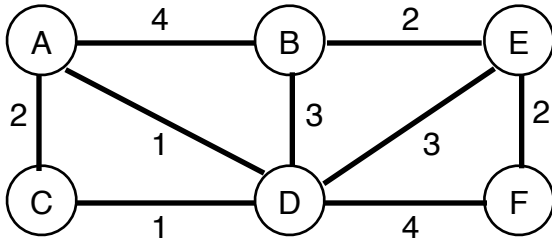


CS 312 - Algorithm Design

Midterm - March 13



1.
 - a. (6 pts.) Draw a Breadth-First Tree beginning at A for the graph above. Ignore the weights on the edges. Is that the only Breadth-First Tree rooted at A for this graph?
 - b. (6 pts.) Draw a Depth-First Tree beginning at A for the graph above. Ignore the weights on the edges. Is that the only Depth-First Tree rooted at A for this graph?
 - c. (6 pts.) Draw a Minimum Spanning Tree for the graph above. Is that the only Minimum Spanning Tree for this graph?
2. (6 pts.) Use induction to prove that a tree containing n nodes has $n-1$ edges.
3. Here is a greedy algorithm used to produce change:

```
giveChange (amount) {  
    numQuarters = amount / 25  
    amount = amount % 25  
    numDimes = amount / 10  
    amount = amount % 10  
    numNickels = amount / 5  
    amount = amount % 5  
    numPennies = amount  
}
```

- a. (6 pts.) True or false: The algorithm shown above will always give change using the fewest number of coins. If true, give a proof. If false, give a counterexample.
- b. (6 pts.) True or false: The algorithm shown above would produce an optimal answer for any possible currency system (real or imaginary). In particular, the coins would be given out from most valuable to least valuable. As many as possible of each coin would be given before considering the next lowest value. If true, give a proof. If false, give a counterexample.

4. Dijkstra's Algorithm for finding the shortest path only works if the weight on all the edges is non-negative. Below is Ford's algorithm which works even if edges have negative weights where G is the graph and S is the node to start from.

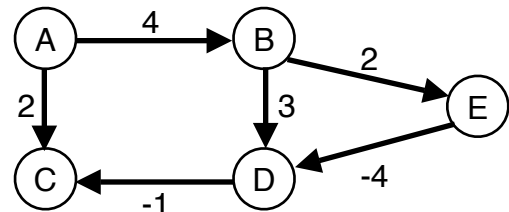
```

FordAlgorithm (G, S)
  for all vertices v {
    distance[v] =  $\infty$ 
  }
  distance[S] = 0

  for i = 1 to number of vertices {
    for each edge e = (u,v) {
      if distance[v] > distance[u] + weight[e]
        distance[v] = distance[u] + weight[e]
    }
  }
}

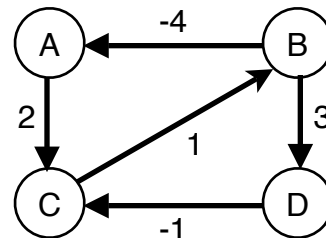
```

- a. (6 pts.) Show the result of the distance array when called with $S = A$ using the graph to the right.



- b. (2 pts.) What is the $O()$ cost of the algorithm shown above?

- c. (6 pts.) Ford's Algorithm does not work correctly if the graph contains a cycle whose total weight is negative. To the right is a graph with a negative cycle:



After completing the algorithm shown above, add another loop that makes one more pass through the edges. The job of this loop is to report if there are any negative weight cycles. So, the skeleton for this loop is:

```

for each edge {

  if (
    ) {
    error "Graph contains a negative cycle"
  }
}

```