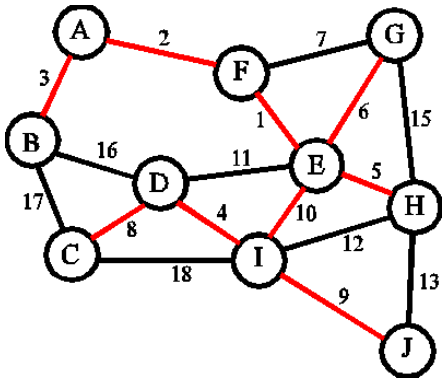
Section 8 Overview

Agenda

- Mini-Assignment
- Topological Sort
- Graph Algorithms
    - Dijkstra's
    - Prim-Jarnik
- Midterm Review

Mini Assignment

- Shortest Path: B,A,F,E,H,J. Could find using Dijkstra's

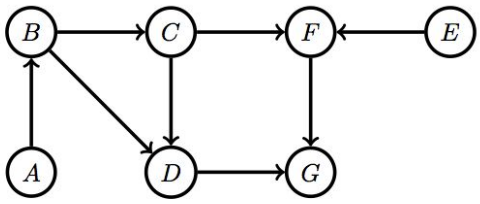


- MST: Could find using Prim-Jarnik's or Kruskal's.

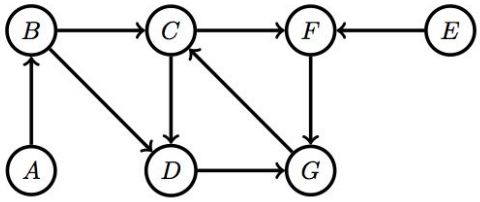
Topological Sort

Topological sort

1. In class, we implemented the topological sort algorithm using a stack. Consider the following DAG. If we replace the stack in the algorithm with a queue, what order does it produce? Did the algorithm still work? Why or why not? In general, what kinds of data structures can you use for topological sort?



2. The following graph is identical to the one from the first part except that an edge from G to C has been added. Hand-simulate topological sort on this graph. What's the output? Does anything go wrong? If so, what property of this new graph causes top sort to fail?



Solutions to above:

1. Example of sorted list using a queue: A, E, B, C, D, F, G (with a stack you'd get something like E, A, B, C, F, D, G)
    a. Yes, algorithm still works because nodes are only added to the given data structure when they are ready to be dequeued. In other words, a vertex would never get added to a given data structure without having all its prerequisites already fulfilled. Therefore, it doesn't matter in which order you dequeue a vertex.
    b. Other kinds of data structure: array, stack, queue, tree
2. There is no correct sorted list
    a. Since there is a cycle between C, D, and G, there is no possible topological sorted list for this graph. A cycle means that a given group of vertices will never have all of their prerequisites completed and therefore, can never be added to the data structure (to eventually be removed and added to the list).

Review of Graph Algorithms

Dijkstra's

- Shortest distance algorithm that uses a PQ
- O((|V| + |E|) * log(|V|))

- Doesn't work with negative weight edges
- Can be used to decorate the whole graph (distance from one point to all other points)

Prim Jarnik's

- Greedy algorithm to find MST using priority queue approach
- Start by decorating nodes with infinity cost, set start node to 0, keep removing the smallest cost node from the PQ and updating its neighbors accordingly (curr node cost + edge)
- O((|V| + |E|) * log(|V|))

Midterm Review

- Midterms handed back
- Questions about grades should be directed to the TA who graded the question (written on the front of the exam)
- Average can be found on the website (http://cs.brown.edu/courses/cs016/averages.html)

Optional Problems

Interview Question

Write an algorithm that takes in a list of numbers and a goal number, and returns whether or not a pair in the list sums to the goal number.

```
Solution:
def findNumSums(L, sum):
// inputs: list L of numbers, and integer sum
// output: whether or not there exists a pair in L that sums to sum
Hashset hash is an empty hashset of integers
for each number n in L:
      if hash contains sum - n:
            return true
      insert n into hash
return false
```