13 - Spanning Trees

Joseph Afework
CS 241

Dept. of Computer Science
California Polytechnic State University, Pomona, CA

Agenda

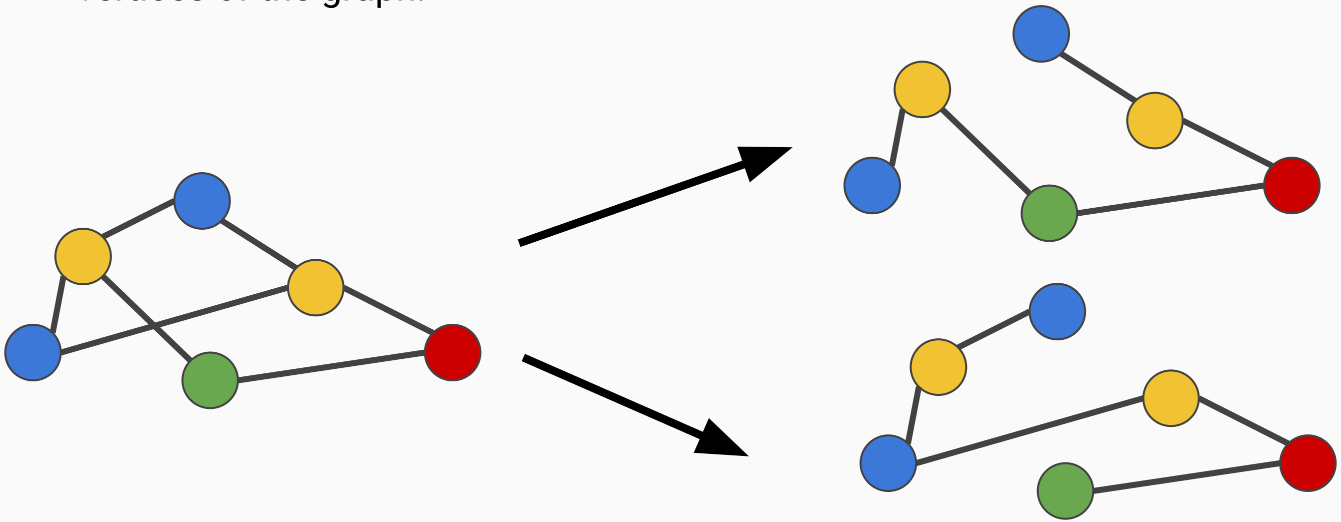
- Terms
- Prim’s Algorithm
- Kruskal’s Algorithm
- Analysis
- Challenge

Reading Assignment

- Read Chapter 28 - Graphs
    - Chapter 28 (Read about: **Spanning Trees, Prim’s and Kruskal’s Algorithm**)

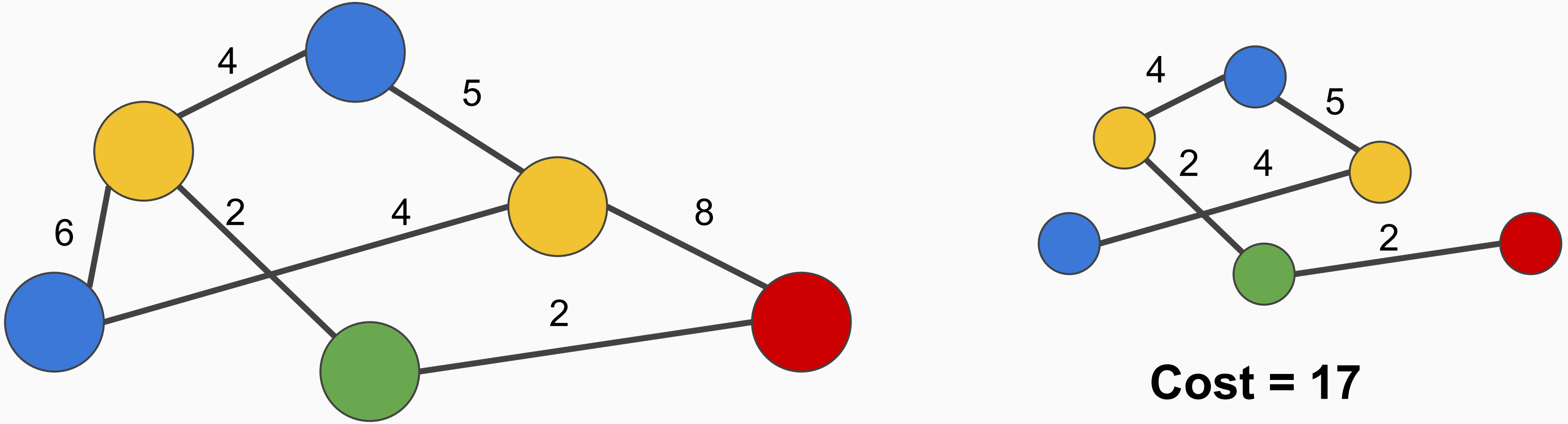
Spanning Tree

- A **spanning tree** is a subgraph of an undirected graph that forms a tree containing all the vertices of the graph.



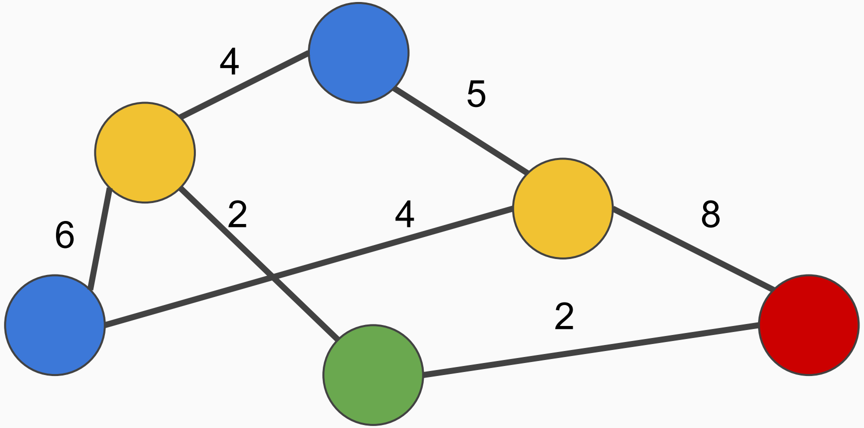
Minimum Spanning Trees (MST)

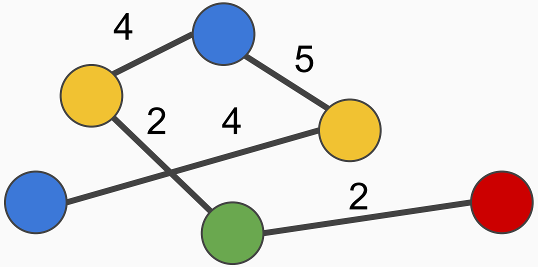
- A **minimum spanning tree** is a subgraph of an undirected ***weighted*** graph that forms a tree containing all the vertices of the graph such that the summation of the edge weights is minimized.



MST contd.

- Useful in networking
- Constructing a minimal path through a graph

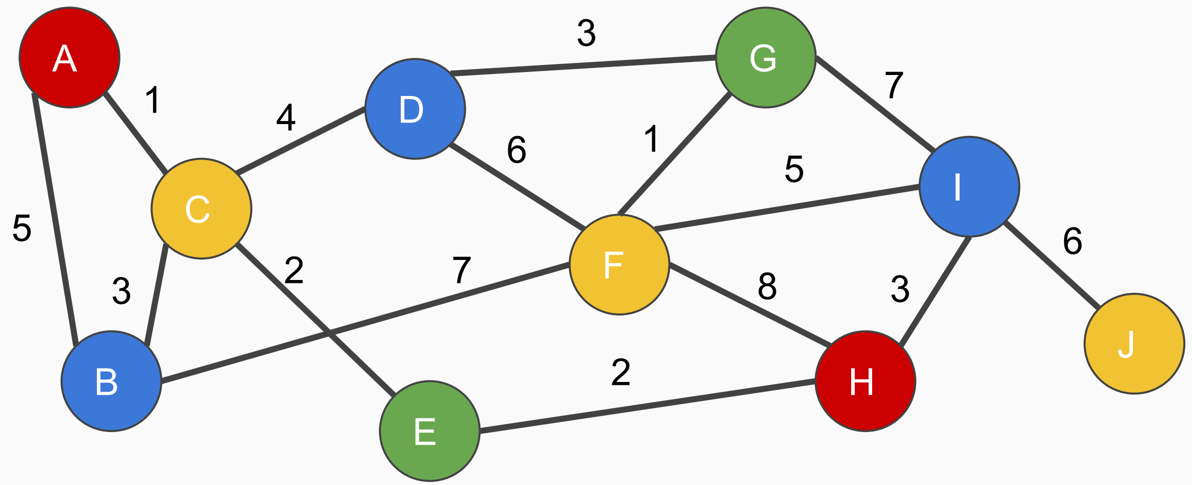




Cost = 17

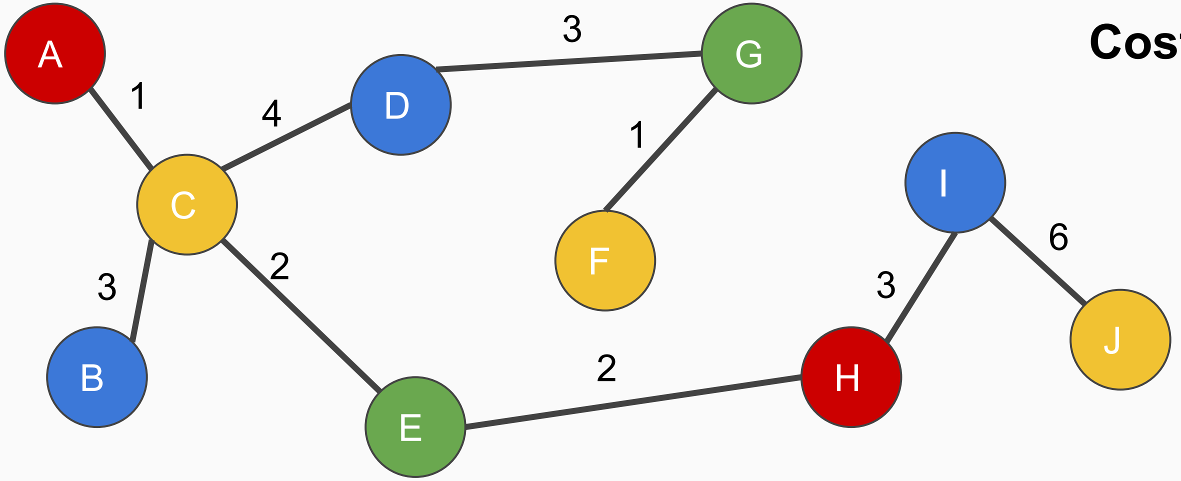
In Class Exercise

Construct a Minimum Spanning Tree for the following graph (however you can):



Solution

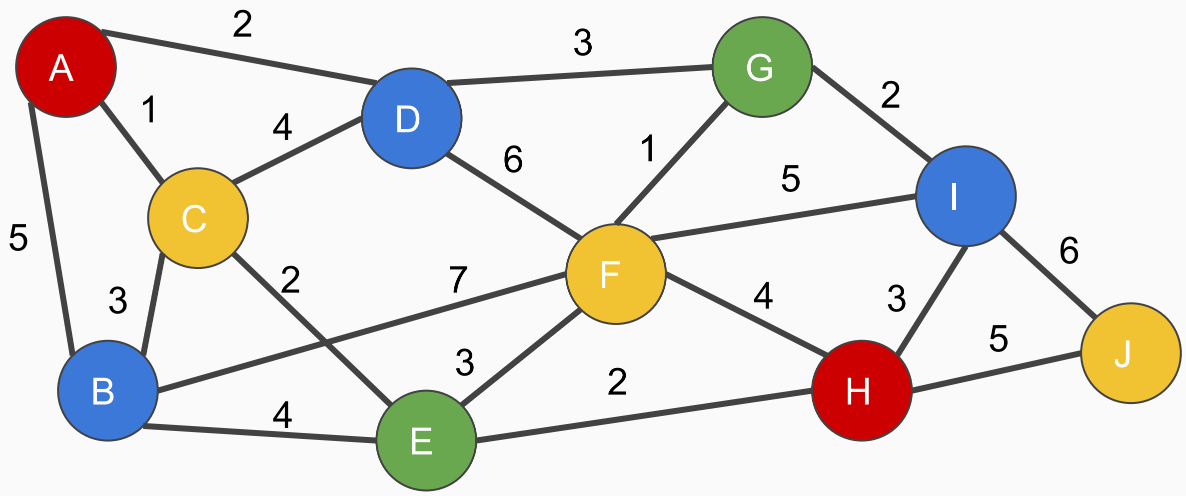
Construct a Minimum Spanning Tree for the following graph (however you can):



Cost = 25

MST contd.

- Does your method work well for large graphs?
- Can we do this a faster way?

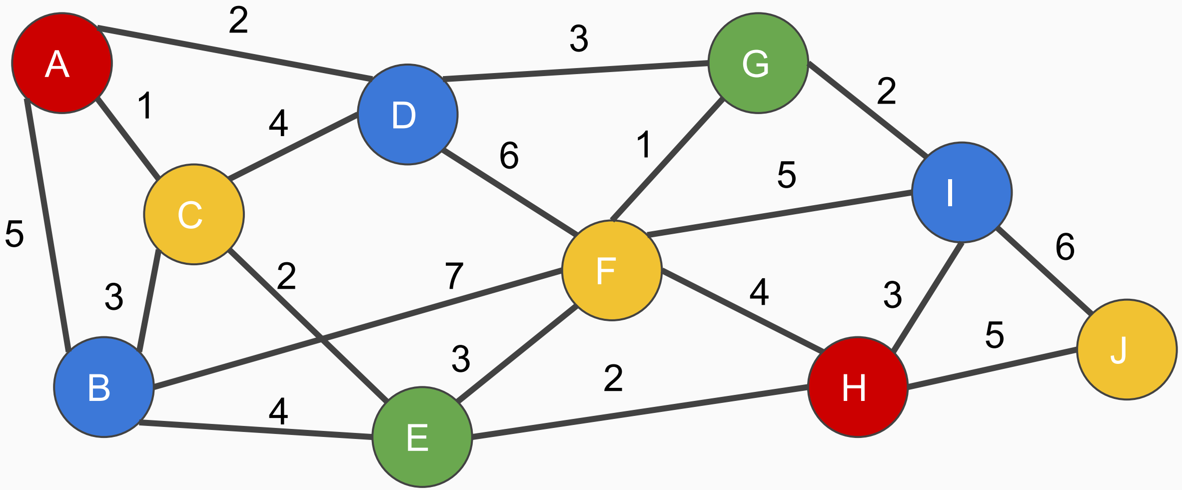


Prim’s Algorithm

1. Let V = all the nodes in the graph.
2. Let S = a set containing just an arbitrary starting node
3. Let A = an empty set that will contain all the edges in the final MST
4. While members of S != members of V
    a. Select an edge with a minimal weight such that the starting node of the selected edge is in S, and the terminating node of the edge is in V-S (which means that node hasn’t been added to S yet).
    b. If edges have a tie in weight, select arbitrarily
    c. Add the terminating node of the selected edge into S
    d. Add the selected edge to A
5. The set **A** now contains all the edges in the MST

In Class Exercise

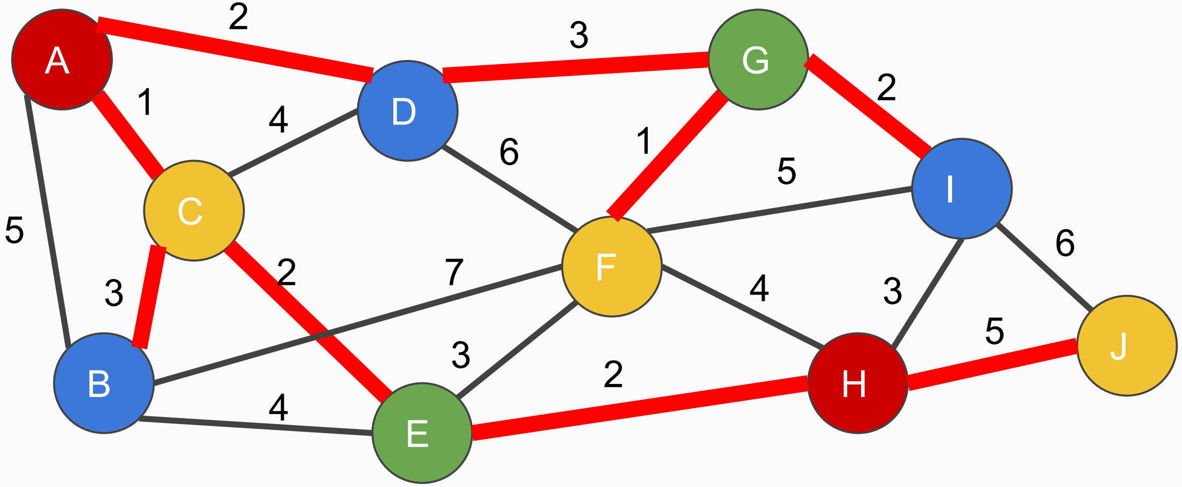
Construct a Minimum Spanning Tree using Prim's algorithm: **(Start at A)**



Prim Contd.

Solution

Cost = 21



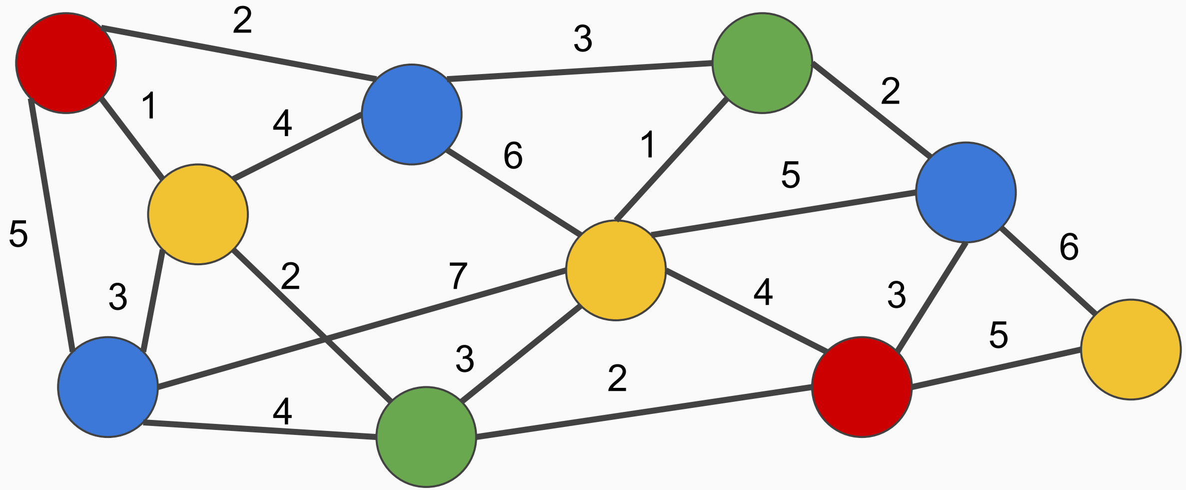
Kruskal’s Algorithm

1. Sort all the edges in non-decreasing order of their weight.
2. Pick the smallest edge (if tie in weight, pick edge arbitrarily).
    a. Check if it forms a cycle with the spanning tree formed so far.
        i. If cycle is not formed, include this edge.
        ii. Else, discard it.
3. Repeat step 2 until there are (V-1) edges in the spanning tree.

Note: A MST has V-1 edges, where V = # of vertices in the graph

In Class Exercise

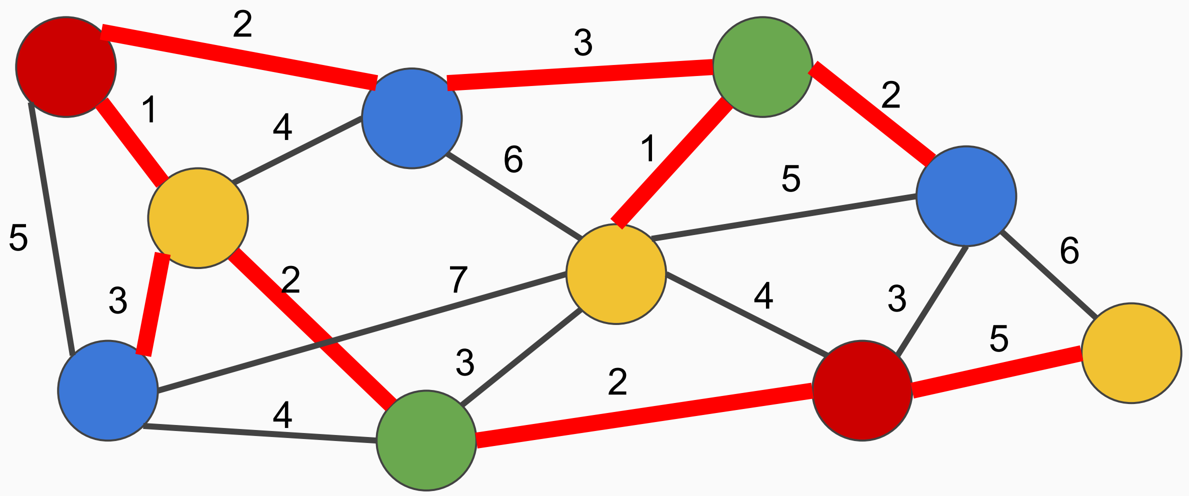
Construct a Minimum Spanning Tree using Kruskal's algorithm:



Kruskal Contd.

Solution

Cost = 21



Prim vs. Kruskal

- Prim’s algorithm begins with a starting node, Kruskal’s algorithm starts with an edge.
- Prim’s algorithm advances node by node, Kruskal's Algorithm select the next edge in a less structured but increasing order.

Performance

- Kruskal - faster in sparse graphs (limited density)
- Prim - faster in dense graphs with more edges than vertices.

Kruskal:

- **O(E log (E))** or O(E log(V))

Prim:

- Range from **O(V^2)** to O(E * log(V)) depending on the data structures used in implementation.

Challenge: Enron Email Dataset

Enron Email Data Set -> http://www.cs.cmu.edu/~enron/

- Contains 500k+ email messages
- Largest Open Data Set of Email Communication
- Great to use as test data

Neo4j -> https://neo4j.com/

- Graph Database
- Built in Graph Algorithms (shortest path, Prim, Kruskal.. etc)

Challenge Contd.

Instructions:

- Construct a graph where a node/vertex is a person (email address), and an edge represents an email communication between two nodes/vertices.
- Increment the edge value for each communication between two emails.
- Create a **maximum spanning tree** by inverting the values in the edges and applying kruskal's algorithm.

References

https://www.ics.uci.edu/~eppstein/161/960206.html

https://www.cpp.edu/~ftang/courses/CS241/notes/graph%20algorithms%202.htm

https://www.quora.com/What-is-the-difference-in-Kruskals-and-Prims-algorithm

https://snap.stanford.edu/data/email-Enron.html

https://neo4j.com/

http://web.cecs.pdx.edu/~sheard/course/Cs163/Doc/Graphs.html

http://mathworld.wolfram.com/MaximumSpanningTree.html

http://www.lighterra.com/papers/graphcoloring/