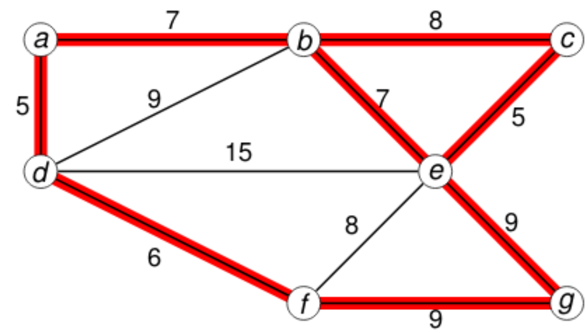
Minimum Spanning Trees

1 Connecting the vertices

Input: a graph G=(V,E) with a weight (or a cost) w(u,v) for each edge (u,v)



Objective: Choose a subset of the edges that connects the vertices. Find the solution that costs the least

1.1 Minimum spanning tree problem

Find a tree that spans the vertices and has minimum cost

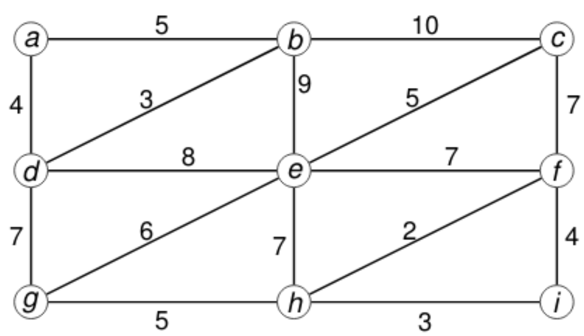
Basic properties of MSTs:

- have $|V| - 1$ edges
- Have no cycles
- might not be unique

2 Representations of weighted graphs

$$\begin{pmatrix} 0 & 5 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 10 & 3 & 9 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 5 & 7 & 0 & 0 & 0 \\ 4 & 3 & 0 & 0 & 8 & 0 & 7 & 0 & 0 \\ 0 & 9 & 5 & 8 & 0 & 7 & 6 & 7 & 0 \\ 0 & 0 & 7 & 0 & 7 & 0 & 0 & 2 & 4 \\ 0 & 0 & 0 & 7 & 6 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 7 & 2 & 5 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 & 3 & 0 \end{pmatrix}$$

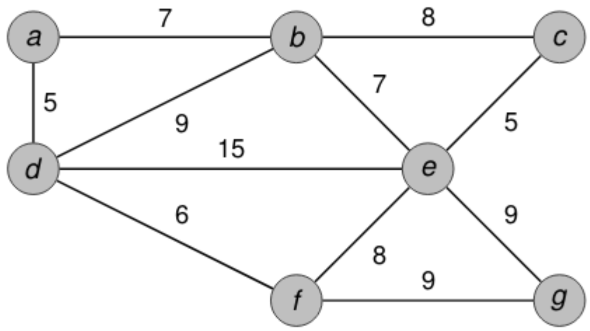
Note that the zeros represent the fact there is no edge between the two nodes, it could equally be ∞



3 Kruskal's Algorithm

1. Sort the edges by weight
2. Let A=∅
3. Consider edges in increasing order of weight. For each edge e, add e to A unless this would create a cycle (cycles are detected by running BFS between the two vertices before joining them, however this is a naive method)

Running time is $O(E \log V)$



3.1 Correctness

Claim - The set A is always a subtree of an MST
The claim implies the algorithm is correct since when it terminates, A is a spanning tree.

Proof of the claim - By induction
Base case

- $A = \emptyset$ so the claim is true in this case

Inductive step:

- Assume A is a subtree of a MST
- Must show that $A + e$ us a subtree of a MST when e is added to A.
- Let T be the MST that contains A
- If T contains e, we are done
- Suppose e is not in T. So $T + e$ contains a cycle
- Some of the edges in the cycle are not in $A + e$
- Let f be an edge in the cycle not in $A + e$
- Consider $T + e - f$. A tree that contains $A + e$
- $w(T + e - f) > w(T)$ since T is an MST
- $w(T) + w(e) - w(F) > w(T)$
- $w(e) > w(F)$
- This is a contradiction. The algorithm should pick F before e

4 Prim's Algorithm

1. Let $U = \{u\}$ where u is some vertex chosen arbitrarily
2. Let $A = \emptyset$
3. Until U contains all vertices: find the least weight edge e that joins a vertex v in U to a vertex w not in U and add e to A and w to U

Running time is $O(V \log V + E)$

4.1 Correctness

- Let T be the output
- Suppose that T is not a MST
- Let T^* be a MST with most edges in common with T
- Let e be the first edge that belongs to T but not T^*
- Consider the moment that e is chosen
  - U is the vertices chosen so far
  - W is the remaining vertices
  - Let e connect U to W
  - T^* must contain some other edge f from U to W
  - And $w(f) \geqslant w(e)$
- Notice that $T^* + e - f$ is a tree
- $w(T^* + e - f) \leqslant w(T^*)$
- So $w(T^* + e - f) = w(T^*)$ as no spanning trees can weigh less than T^* as it is an MST
- So $T^* + e - f$ is a MST with more edges in common with T than T^*
- A contradiction. BAM.