Minimal Spanning Trees

Vertices Edges

Let $G=(V,E)$ be a connected undirected graph with $E \subseteq V \times V$

Def. A ***weighted graph*** is a undirected graph $G=(V,E)$ with **weight function** $w : E \to \mathbb{R}$

Def. A ***spanning tree*** T of G is a subgraph of a undirected graph of G s.t.

1) T is a tree, i.e., a connected acyclic graph
2) V(T)=V(G)

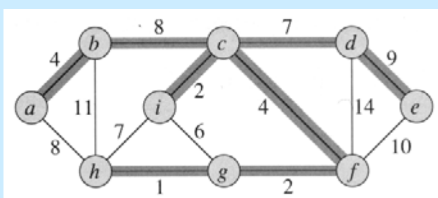
Minimal Spanning Trees

Def. A ***minimum spanning subtree*** of a weighted graph (G,w) is a spanning subtree of G of minimum weight

$$w(T)=\sum_{e\in T} w(e)$$

Minimum Spanning Subtree Problem: Given a weighted connected undirected graph (G,w), find a minimum spanning subtree

Example of a Minimal Spanning Tree



Minimal Spanning Trees

We will look at two Algorithms:

- Kruskal
- Prim ~ Dijkstra's Shortest paths algorithm

Both are $Greedy$ algorithms

The Generic Algorithm

```
GENERIC-MST(G, w)
  A ← Ø
  while A does not form a spanning tree
     do find an edge (u, v) that is safe for A
        A ← A ∪ {(u, v)}
  return A
```

This algorithm grows a set of edges A of G which upon termination of the algorithm becomes a minimal spanning tree of G.

- In Kruskal's algorithm, A is a coalescing forest
- In Prim's algoritm, A is an expanding tree

Property preserved by Loop: A is a subset of some minimum spanning tree

Safe Edges

Def. A ***safe edge*** for A is am edge (u,v) not in A s.t. A satisfies property P $\Rightarrow$ $A \cup \{(u,v)\}$ also satisfies property P

Note: If A is not a minimum spanning tree, then there exists a safe edge that can be added to A.

Problem: How do we find safe edges?

Terminology

Def. A ***cut*** (S,V-S) of an undirected graph G=(V,E) is a partion of the vertices V of G into two disjoint sets S and V-S, i.e., s.t.

- $V = S \bigcup (V - S)$
- $\varnothing = S \bigcap (V - S)$

An edge $(u,v) \in E$ ***crosses*** the cut (S,V-S) if one of its endpoints lies in S and the other lies in V-S.

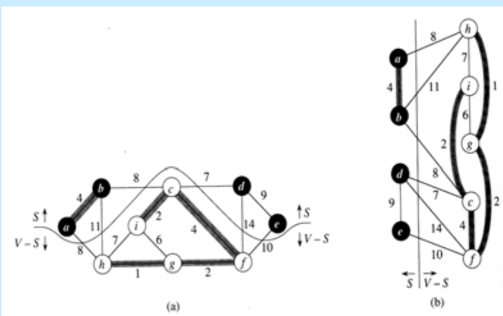
A cut ***respects*** a set of edges A of G if no edge of A crosses the cut.

More Terminology

Def. An edge crossing a cut (S,V-S) of a weighted undirected graph (G,w) is a ***light edge*** if its weight is minimum of the weight of any edge crossing the cut.

Note: An edge is a ***light edge satisfying property P*** if its weight is the minimum of any edge satisfying property P.

Example of a Cut



Minimal Spanning Trees

Theorem. Given

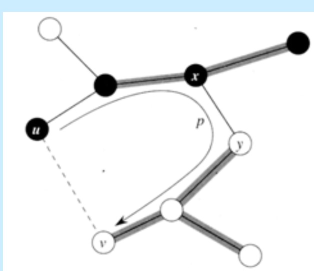
- G=(V,E) connected a undir graph
- w:E→R a weight function on G

Let

- $A \subseteq E$ be s.t. A lies in some minimum spanning tree
- (S,V-S) be a cut of G that respects A
- (u,v) be a light edge crossing (S,V-S)

Then (u,v) is safe!

Proof



A= Shaded Edges

Kruskal's Algorithm

Cor. Given (G=(V,E),w) is a connected undirected weighted graph.

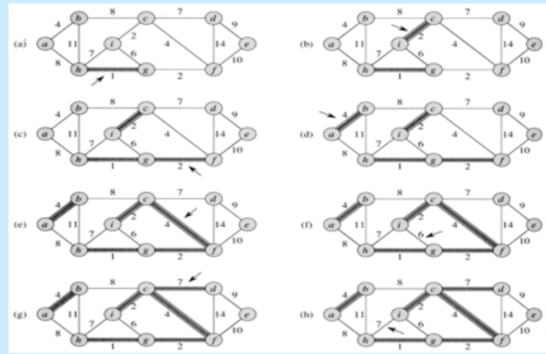
Let A be a subset of E lying in a minimum weight subtree of G, and let C be a connected component of a forest G_A=(V,A).

If (u,v) is a light edge connecting C to some other component of G_A, then (u,v) is safe for A

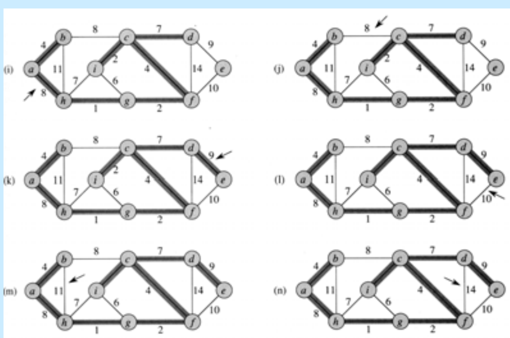
Kruskal's Algorithm

```
MST-KRUSKAL(G, w)
A ← Ø
for each vertex v ∈ V[G]
    do MAKE-SET(v)
sort the edges of E into nondecreasing order by weight w
for each edge (u, v) ∈ E, taken in nondecreasing order by weight
    do if FIND-SET(u) ≠ FIND-SET(v)
          then A ← A ∪ {(u, v)}
               UNION(u, v)
return A
```

Kruskal's Algorithm



Kruskal's Algorithm (Cont.)



Prim's Algorithm

```
MST-PRIM(G, w, r)
for each u ∈ V[G]
    do key[u] ← ∞
       π[u] ← NIL
key[r] ← 0
Q ← V[G]
while Q ≠ Ø
    do u ← EXTRACT-MIN(Q)
       for each v ∈ Adj[u]
           do if v ∈ Q and w(u, v) < key[v]
                 then π[v] ← u
                      key[v] ← w(u, v)
```