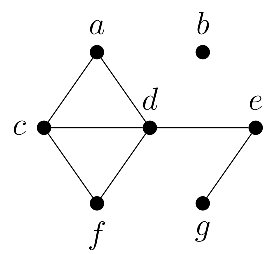
MSTs

Graphs

A graph $G = (V, E)$ is a set V of *nodes*, or *vertices*, and a set E of *edges*, where each edge is a set of two nodes.

For an edge $\{x, y\}$ of G, we say that x and y are *adjacent*, or *neighbours*. The *degree* of a node is its number of neighbours.



Exercise 1

For the above graph, give the node set and edge set. For each node, give its degree and set of neighbours.

Paths

In a graph, a *path* is a sequence $(v_1, \ldots, v_t)$ of distinct nodes in which each consecutive pair of nodes is adjacent. Here, we say that there is a path *from* v_1 *to* v_t, and v_1 and v_t are the *ends* of the path. For example, in the above graph (b) and (f, c, d, e) are paths, but (a, b) and (e, g, e) are not paths.

A path is *extendable* if it is a proper subsequence of another path. A path is *maximal* if it is not extendable. For example, (c, d) is extendable because (c, a, d) is a path; (c, a, d) is extendable because (f, c, a, d) is a path; (f, c, a, d) is extendable because (f, c, a, d, e) is a path; (f, c, a, d, e) is extendable because (f, c, a, d, e, g) is a path; and (f, c, a, d, e, g) is a maximal path.

Exercise 2

For the above graph, give all maximal paths.

Connectivity

Given a graph $G = (V, E)$ and a node subset S of V, $G[S]$ is the *subgraph induced by* S, namely the graph $G[S] = (S, E[S])$, where $E[S]$ is the set of edges of E with both nodes in S. For example, for the above graph, the subgraph induced by $S = \{a, b, c, d\}$ has node set S and edge set $\{\{a, c\}, \{a, d\}, \{c, d\}\}$.

A graph is *connected* if, for each ordered pair of nodes (x, y), there is a path from x to y. A *component* of a graph G is a induced subgraph $G[S]$ such that $G[S]$ is connected, and no superset T of S induces a connected subgraph. For example, the graph above has two components, with node sets $\{b\}$, and $\{a, c, d, e, f, g\}$.

Cycles

In a graph, a *cycle* is a path with at least three nodes whose ends are also adjacent. For example, in the above graph (a, d, c) and (a, d, f, c) are cycles but (a, d, c, f) is not.

Trees

A graph is *acyclic* if it has no cycles. A *forest* is a graph that is acyclic. A *tree* is a graph that it is acyclic and connected.

Minimum spanning subtrees

A subgraph of a graph is *spanning* if it includes all nodes (but not necessarily edges) of the graph. A *weighted graph* has weights on the edges. A *spanning subtree* of a graph is a spanning subgraph that is a tree (so, acyclic and connected).

Notice that a graph must be connected if it has a spanning tree, so whenever we ask for a spanning subtree of a graph, we assume that the graph is connected. The *weight* of a spanning subtree T is the sum of the weights of the edges of T. A *minimum spanning subtree* (MST) of a connected weighted graph G is a spanning subtree with minimum weight, among all spanning subtrees of G.

Tree properties

For a graph $G = (V, E)$ and an edge $e \notin E$, $G+e$ is the graph with node set V and edge set $E \cup \{e\}$.

Property 1 Let $G = (V, E)$ be a graph with a cycle C, and let e be an edge of C. Let $G' = G - e$ be the graph obtained from G by removing e from E. Let X be a component of G. Then X is a component of G'.

So, removing one edge of a cycle does not change any of the components.

Property T In a tree with at least 2 nodes, there is a path with at least 2 nodes, and the ends of a longest path each have degree 1.

Property 2 A tree with n nodes has exactly $n - 1$ edges.

Sketch of proof: use Property T. Argue by induction on n.

Property 3 A connected graph with n nodes and $n - 1$ edges is a tree.

Sketch of proof: For $n \geq 2$, there is a node with degree 1. (argue by contradiction). Show that removing it leaves a connected graph with $n - 1$ nodes and $n - 2$ edges. Argue by induction.

Property 4 A graph is a tree if and only if, between each pair of nodes, there is a unique path.

Sketch of proof: Assume G is a tree. So, between each pair of nodes, why is there a path? And why are there not two paths? Next assume that G is a graph with the above betweenness property. Why is G connected? Why is G acyclic?

Property 5 Let T be a spanning tree of a graph G, and let e be an edge not in T. Then $T + e$ has a cycle C, and for every edge c of C, $T + e - c$ is a spanning tree. Furthermore, if T is an MST of G, then $w(e) \geq w(c)$.

Exercise 3

Prove Property 5. First prove that $T+e$ has a cycle. Next prove that $T + e - c$ is connected. Next prove that $T + e - c$ is acyclic. Next prove that $T + e - c$ is spanning. Next prove that $w(e) \geq w(c)$.

Choice

As input to Kruskal's algorithm, we allow any weighted graph, so edge weights are not necessarily distinct. So, an execution of Kruskal's algorithm will have a choice whenever there is more than one unselected edge with minimum weight.

Kruskal is correct

Here we explain why Kruskal's algorithm is correct, i.e. why every tree returned is an mst. With respect to the input weighted graph, call a tree *Kruskal* if there is some execution of Kruskal's algorithm that returns that tree.

Theorem: for any weighted graph G, every Kruskal tree is an MST.

Exercise 4

Prove the above theorem. Use whichever of the tree properties are needed. Hint: see the webnotes.

Kruskal is complete

Here we show that Kruskal is complete, i.e. that every mst of a graph will be returned by some execution of Kruskal's algorithm.

Theorem: for any weighted graph G, every MST is Kruskal.

Sketch of proof. Let T be an arbitrary MST. For an execution of Kruskal's algorithm and resulting MST K, let $k_1, k_2, \ldots, k_{n-1}$ be the edges of K in the order they were selected. We say that k_1 was picked in step 1, k_2 in step 2, and so on.

If $T = K$ then our MST T is a Kruskal MST and we are done.

Now suppose $T \neq K$. So there is a first step q for which k_q is not in T. We will show how to find another Kruskal execution with edges $k'_1, k'_2, \ldots, k'_{n-1}$ and tree K' where the first step x for which k'_x is not in T satisfies $x > q$. By repeating this process at most $n-1$ times, we will end with a Kruskal execution whose tree is T.

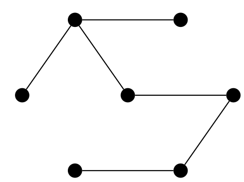
So, how do we find K'?

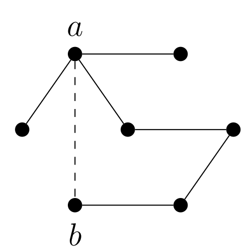
Let a, b be the nodes of edge k_q. Consider the graph $T + k_q$. T is a tree, so connected, so there is in T a path $P = (v_1, \ldots, v_t)$ with $a = v_1$ and $b = v_2$, so in $T + k_q$ the sequence P is a cycle.

If all edges of P are in K', then $P + k_q$ is a cycle of K', contradicting that K' is a tree. So there is some edge z of P which is not in K'. So, at Step q, when Kruskal picked k_q, it could have picked z but did not. So $w(k_q) \leq w(z)$.

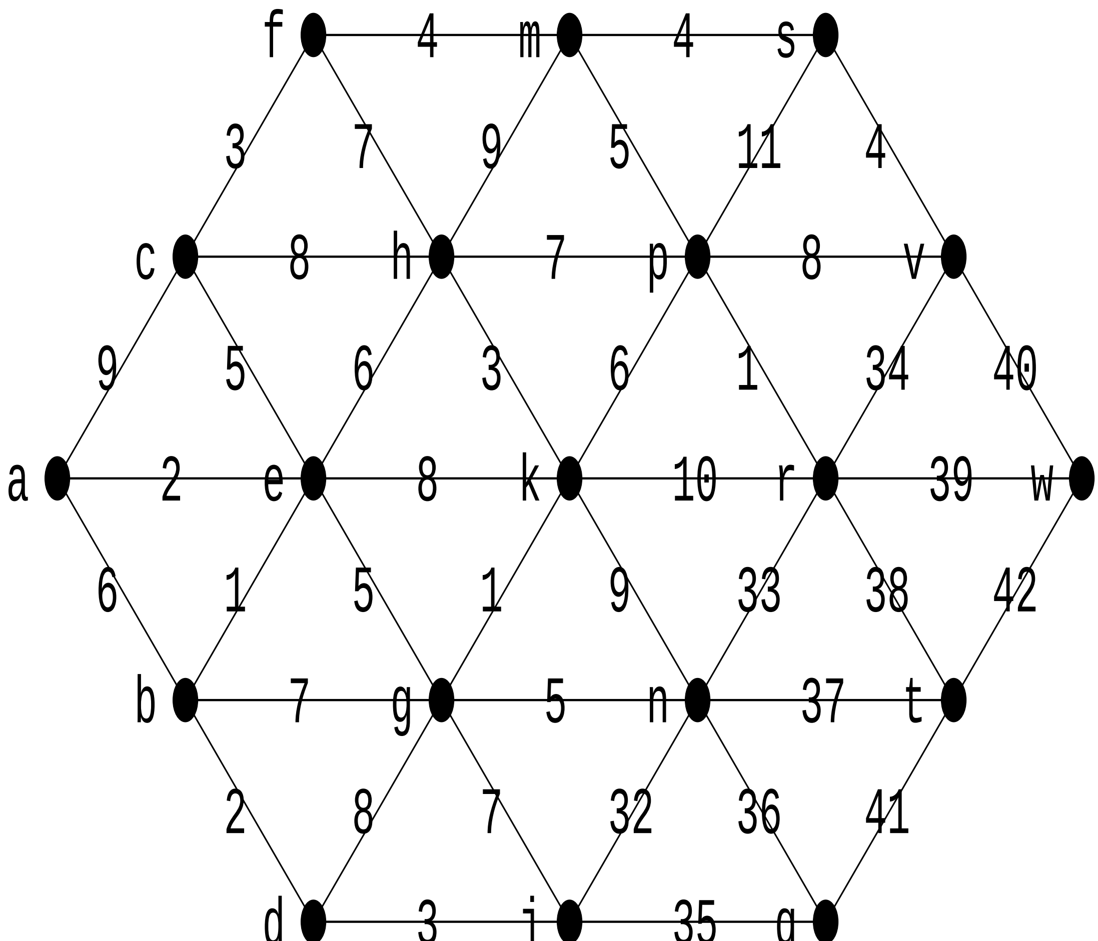
But by Property 5, $T + k_q - z$ is a spanning tree whose weight must be at least that of the weight of the MST T, so $w(k_q) \geq w(z)$.

So we have $w(k_q) \leq w(z)$ and $w(k_q) \geq w(z)$, so have $w(k_q) = w(z)$. Also, all edges $k_1, \ldots, k_{q-1}$ and also z are in T, which is acyclic. So, at step q, Kruskal can pick z instead of k_q, leading to a new execution of Kruskal, in which the first edge it picks that is not in T will be at some step x after step q. So we are done. □

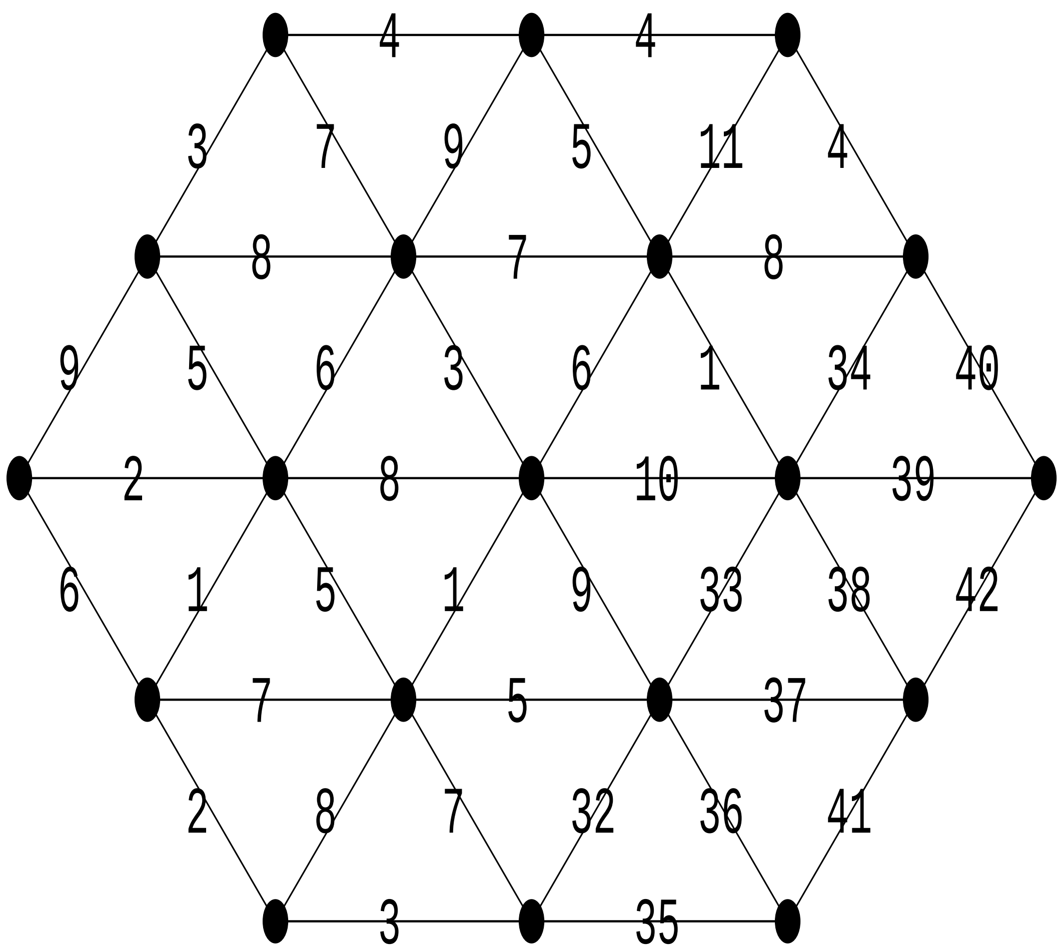




mst


f 4 m 4 s
3 7 9 5 11 4
c 8 h 7 p 8 v
9 5 6 3 6 1 34 40
a 2 e 8 k 10 r 39 w
6 1 5 1 9 33 38 42
b 7 g 5 n 37 t
2 8 7 32 36 41
d 3 j 35 q

mst


4
4
3
7
9
5
11
4
8
7
8
9
5
6
3
6
1
34
40
2
8
10
39
6
1
5
1
9
33
38
42
7
5
37
2
8
7
32
36
41
3
35

mst

```
        f●    4   m●    4   s●
    3     7     9     5     11    4
     c●    8   h●    7   p●    8   v●
  9     5     6     3     6     1     34    40
a●    2   e●    8   k●    10  r●    39  w●
  6     1     5     1     9     33    38    42
     b●    7   g●    5   n●    37  t●
    2     8     7     32    36    41
        d●    3   j●    35  q●
```

Kruskal mst