Weighted graph

Let $G=(N,E)$ be a simple graph. If a mapping $w:E\rightarrow R$ is given, then the triple $G=(N,E,w)$ is called a weighted graph. Each edge e of G is thus assigned a real number $w(e)$, called the weight of e.

If $G'=(N',E')$ is a subgraph of G, then $w(G')=\sum_{u\in E'} w(u)$ is called the weight of G'.

Minimal spanning tree

Let $G=(N,E,w)$ be a simple weighted graph. Let $S=(N,E')$ be a spanning tree of G. We say that S is a minimal spanning tree of G if $w(S)\leq w(T)$ for each spanning tree T of G.

Next we will describe two algorithms that can be used to find a minimal spanning tree of a simple weighted graph . Why such algorithms work is shown by the theorems that follow.

Theorem 7

Let $G=(N,E,w)$ be a simple weighted graph and let

$$C=(v,e_1,u_1,e_2,u_2,\dots,u_{p-1},e_p,v)$$

be a circle in *G*. If $c(e_1)>c(e_i), 2\le i\le p$, then edge e_1 is contained in no minimal spanning tree of *G*.

Proof:

Let $M=(N,E')$ be a minimal spanning tree of *G* and suppose $e_1\in E'$. Then we have a fundamental cut $D^M(e_1)$. By Theorem 5, in addition to e_1, there must be at least one more edge *f* that is both in *C* and $D^M(e_1)$. Since $f\in D^M(e_1)$, f is a chord in *M*. Thus, we can define $C^M(f)$. By Theorem 6 $e_1\in C^M(f)$. Now add *f* to *M* to create *M'*. The only circle in *M'* is $C^M(f)$ and, deleting e_1 from this circle, we get a spanning tree *M''* with a total weight smaller than *M*, which is a contradiction.

Corollary 8

Let $G=(N,E,c)$ be a simple weighted graph and let

$$C=(v,e_1,u_1,e_2,u_2,\ldots,u_{p-1},e_p,v)$$

be a circle in *G*. If $c(e_1)\geq c(e_i), 2\leq i\leq p$, then there is at least one minimal spanning tree not containing e_1.

This corollary affects graphs in which the weights of two different edges may be the same. It is not difficult to show that, provided that all the edge weights are distinct, there is a unique spanning tree of *G*.

Kruskal's algorithm

Let $G=(N,E,w)$ be a weighted simple connected graph with $E=\{e_1, e_2, \ldots, e_k\}$. Arrange the edges of E in a sequence $T_1=(t_1, t_2, \ldots, t_k)$ so that $w(t_i)\le w(t_j)$ for $i<j$. In subsequent steps construct graphs $S_1=(N,Q_1), S_2=(N,Q_2), \ldots$ Here $T_i(1)$ will denote the first term of T_i and T_i-1 the sequence T_i without its first term.

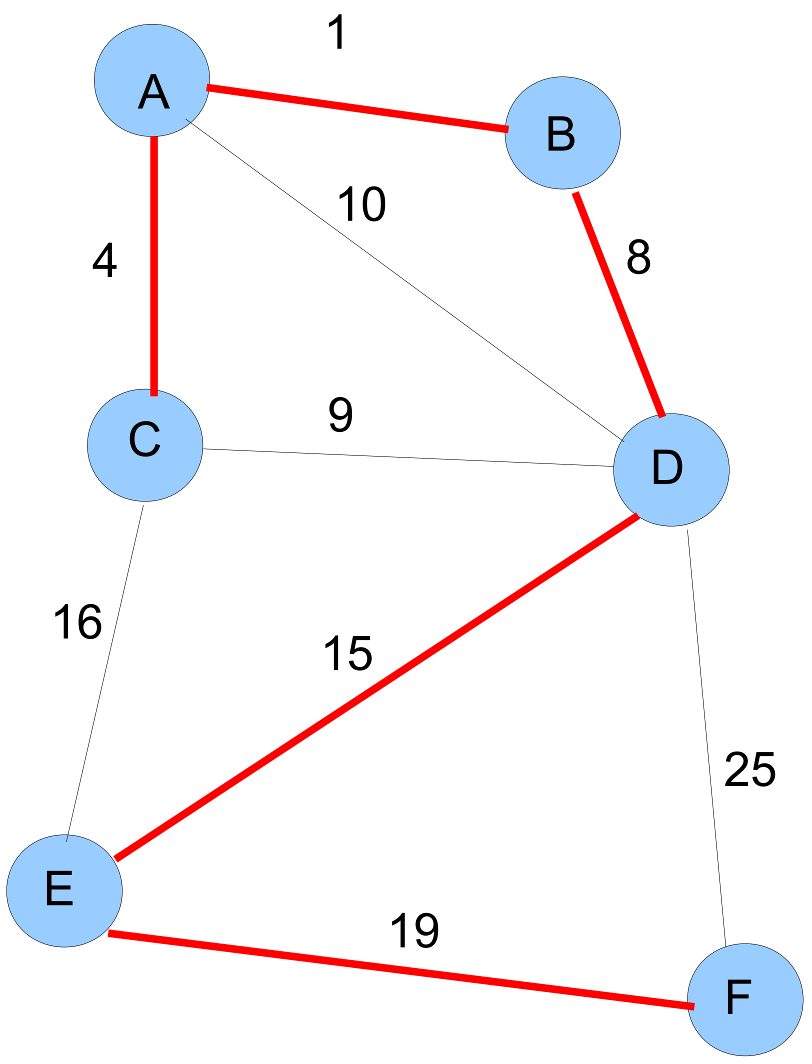
(a) $Q_1=\{t_1\}, T_2=T_1-1$

(b) If S_i is connected, it is a spanning tree and the algorithm stops

(c) Put $Q_{i+1}\overset{\text{def}}{=}Q_i\cup\{T_i(1)\}$, $T_{i+1}\overset{\text{def}}{=}T_i-1$ if adding edge $T_i(1)$ to Q_i does not give rise to a circle in S_i

(d) Put $Q_{i+1}\overset{\text{def}}{=}Q_i$, $T_{i+1}\overset{\text{def}}{=}T_i-1$ if adding $T_i(1)$ to Q_i does create a circle in S_i

Thus by Kruskal's algorithm, spanning tree is created step by step by adding edges from a sequence previously sorted in an ascending order by the edge weights. Should a circle be created by adding an edge, it is skipped.



w(A,B)=1

w(A,C)=4

w(B,D)=8

w(C,D)=9

w(A,D)=10

w(D,E)=15

w(C,E)=16

w(E,F)=19

w(D,F)=25

Prim's algorithm

Let $G=(N, E, w)$ be a weighted simple connected graph. For a subgraph $S=(V, J)$ of G not containing a circle, denote by $S^{+}=(V^{+}, J^{+})$ the graph created by adding a node v to V and and edge e to J such that

a) e is an edge between node v and a node in V

b) e has the least weight of all such edges.

In subsequent steps, create graphs S_1, S_2, ... as follows:

(a) $S_1=(\{u, v\}, \{e\})$ with $e=\{u, v\}$ and $w(e) \leq w(\{u, z\}) \forall z \in N$

(b) If, for an $S_i=(V_i, J_i)$, we have $|J_i|=|N|-1$, then S_i is a minimal spanning tree and the algorithm stops

(c) else $S_{i+1} \stackrel{\text{def}}{=} S_i^{+}$

Prim's algorithm starts with any node, always adding edges with minimum weights to extend the current graph to a connected graph without a circle.

As compared with Kruskal's algorithm, no previous sort of **all** the edges by their weights is necessary since Kruskal's algorithm mostly does not make use of edges with large weights at all.

Prim's algorithm is based on the following theorems:

Theorem 9

Let $G=(N,E,w)$ be a simple weighted graph and $\{u,v\}\in E$ an edge such that $w(\{u,v\})<w(\{u,z\})\forall z\in N$. Then $\{u,v\}$ is included in every minimal spanning tree of G

Corollary 10

Let $G=(N,E,w)$ be a simple weighted connected graph and $\{u,v\}\in E$ such an edge that $w(\{u,v\})\leq w(\{u,z\})\forall z\in N$. Then there is a minimal spanning tree of G including $\{u,v\}$

Corollary 11

Let $G=(N,E,w)$ be a simple weighted connected graph and $T=(N',E')$ a subtree of a minimal spanning tree of G. Then there is a spanning tree of G including T as a subgraph and, moreover, including an edge e with the least weight such that $e=\{u,v\},u\in N',v\in N-N'$

Prim's algorith in a matrix form

If the weights of the edges of a graph $G=(N, E)$ are defined as a matrix

$$\begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nn} \end{pmatrix}$$

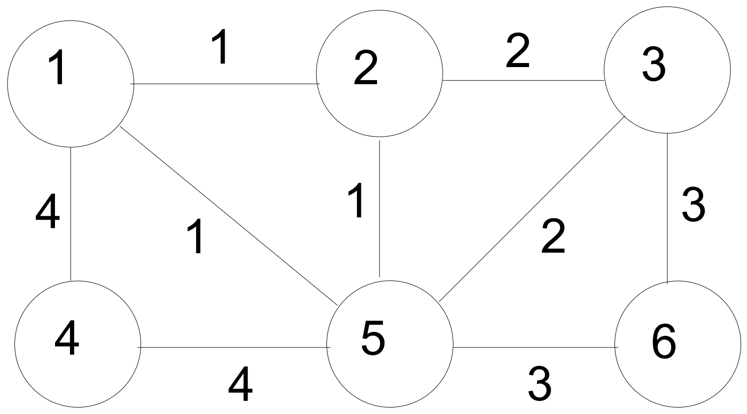
Prim's algorithm may be described as follows:

Step 1: Delete all column 1 entries and mark row 1

Step 2: If the marked rows only include underligned entries, the algorithm stops with the underlined entries indicating the edges of a minimal spanning tree. Else choose a minimal not underlined entry in marked rows.

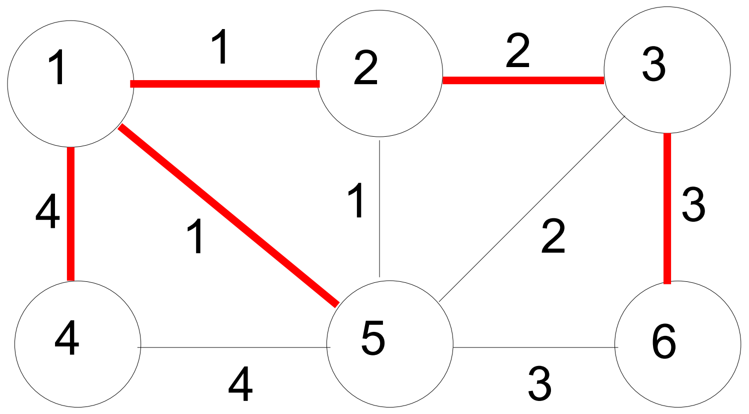
Step 3: If an entry w_{ij} is chosen , it is underlined, row j is marked and all the non-underlined entries in column j are deleted. Go to step 2.

Example



$$C_1 = \begin{pmatrix} - & 1 & - & 4 & 1 & - \\ - & - & 2 & - & 1 & - \\ - & 2 & - & - & 2 & 3 \\ - & - & - & - & 4 & - \\ - & 1 & 2 & 4 & - & 3 \\ - & - & 3 & - & 3 & - \end{pmatrix} \begin{array}{c} * \\ \\ \\ \\ \\ \\ \end{array}$$

$$C_2 = \begin{pmatrix} - & \underline{1} & - & 4 & 1 & - \\ - & - & 2 & - & 1 & - \\ - & - & - & - & 2 & 3 \\ - & - & - & - & 4 & - \\ - & - & 2 & 4 & - & 3 \\ - & - & 3 & - & 3 & - \end{pmatrix} \begin{array}{c} * \\ * \\ \\ \\ \\ \\ \end{array} \qquad C_3 = \begin{pmatrix} - & \underline{1} & - & 4 & \underline{1} & - \\ - & - & 2 & - & - & - \\ - & - & - & - & - & 3 \\ - & - & - & - & - & - \\ - & - & - & 4 & - & 3 \\ - & - & 3 & - & - & - \end{pmatrix} \begin{array}{c} * \\ * \\ \\ \\ * \\ \\ \end{array}$$



$$C_4=\left(\begin{array}{cccccc} - & \underline{1} & - & 4 & \underline{1} & - \\ - & - & \underline{2} & - & - & - \\ - & - & - & - & - & 3 \\ - & - & - & - & - & - \\ - & - & - & 4 & - & 3 \\ - & - & - & - & - & - \end{array}\right)\begin{array}{c} * \\ * \\ * \\ \\ * \\ \\ \end{array}$$

$$C_5=\left(\begin{array}{cccccc} - & \underline{1} & - & 4 & \underline{1} & - \\ - & - & \underline{2} & - & - & - \\ - & - & - & - & - & \underline{3} \\ - & - & - & - & - & - \\ - & - & - & 4 & - & - \\ - & - & - & - & - & - \end{array}\right)\begin{array}{c} * \\ * \\ * \\ \\ * \\ * \end{array}$$

$$C_6=\left(\begin{array}{cccccc} - & \underline{1} & - & \underline{4} & \underline{1} & - \\ - & - & \underline{2} & - & - & - \\ - & - & - & - & - & \underline{3} \\ - & - & - & - & - & - \\ - & - & - & - & - & - \\ - & - & - & - & - & - \end{array}\right)\begin{array}{c} * \\ * \\ * \\ \\ * \\ * \end{array}$$