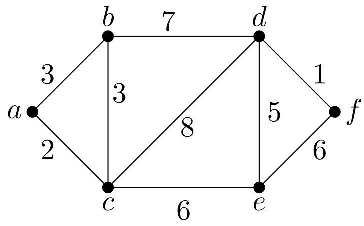
Weighted graph

A **weighted graph** G is a graph that have a number assigned to each edge e and that number is called the **weight of the edge** e and noted by $w(e)$. The **weight of the graph** G, $w(G)$, is the sum of the weights of all edges.

Example 1. The graph G pictured is a weighted graph with $W(G) = 41$.



The edge de has weight $w(de) = 5$.

Shortest path

A shortest path between two vertices in a weighted graph is a path of least weight.

Dijkstra's algorithm

To find a shortest path from vertex v_1 to vertex v_n in a weighted graph, carry out the following procedure.

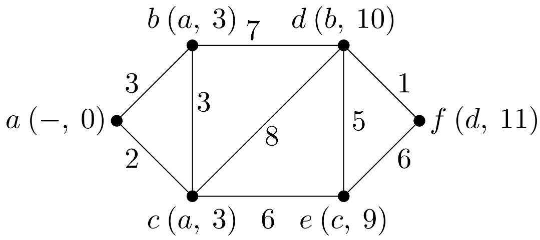
Step 1 Assign to v_1 the label $(-,\ 0)$.

Step 2 Until v_n is labeled or no further labels can be assigned, do the following:

(a) For each labeled vertex $u(x, d)$ and for each unlabeled vertex v adjacent to u, compute $d + w(e)$, where $e = uv$.

(b) For each labeled vertex u and adjacent unlabeled vertex v giving minimum $d' = d + w(e)$, assign to v the label (u, d'). If a vertex can be labeled (x, d') for various vertices x, make any choice.

Example 2. Applying the Dijkstra algorithm to determine the shortest path between the vertex a and the vertex f in the graph pictured:



Then, the shortest path between the vertex a and the vertex f is $acef$ with weight 11.

Minimum spanning tree

As we know, a **spanning tree** of a connected graph G is a subgraph which is a tree and which includes every vertex of G. A minimum spanning tree of a weighted graph is a spanning tree of least weight, that is, a spanning tree for which the sum of the weights of all its edges is least among all spanning trees.

Kruskal's algorithm

To find a minimum spanning tree in a connected weighted graph with $n > 1$ vertices, carry out the following procedure.

Step 1 Find an edge of least weight and call this e_1. Set $k = 1$.

Step 2 **While** $k < n$:

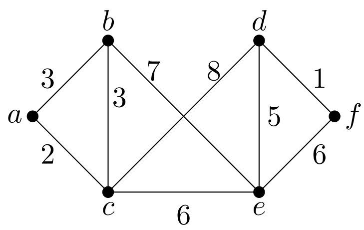
if there exists an edge e such that $\{e\} \cup \{e_1, e_2, \cdots, e_k\}$ does not contain a circuit

then let e_{k+1} be such an edge of least weight and replace k by $k+1$;

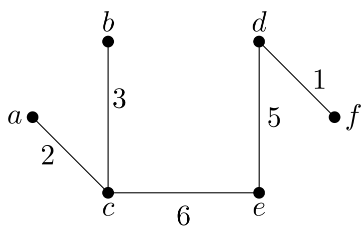
else output $e_1, e_2, \cdots, e_k$ and stop.

end while

Example 3. To determine the minimum spanning tree, applying the Kruskal's algorithm, in the connected weighted graph pictured:



the edge $e_1 = df$ because is the lowest weight, $w(df) = 1$. Then $e_2 = ac$ with $w(ac) = 2$; and know we can choose between ab or bc because they have the same weight $w(ab) = w(bc) = 3$. Let's consider $e_3 = bc$ and the next edge is $e_4 = de$ with $w(de) = 5$. We obtained two disconnected spanning graphs, but to be a tree we need a connected graph. So, add $e_6 = ce$ whitch $w(ce) = 6$. Our minimum spanning tree is



with weight $W(T) = 17$.

Prim's algorithm

To find a minimum spanning tree in a connected weighted graph with $n > 1$ vertices, proceed as follows.

Step 1 Choose any vertex v and let e_1 be an edge of least weight incident with v. Set $k = 1$.

Step 2 **While** $k < n$:

if there exists a vertex which is not in the subgraph T whose edges are $e_1, e_2, \cdots, e_k$

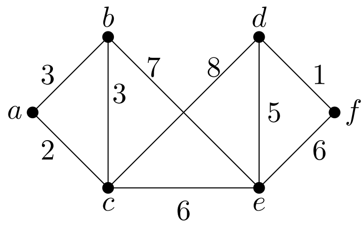
then • let e_{k+1} be an edge of least weight among all edges of the form ux, where u is a vertex of T and x is a vertex not in T;

• replace k by $k + 1$;

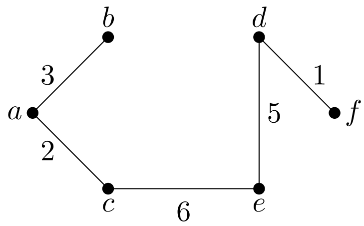
else output $e_l, e_2, \cdots, e_k$ and stop.

end while

Example 4. To determine the minimum spanning tree, applying the Prim's algorithm, in the connected weighted graph pictured:



consider for example $v = c$; the edge with least weight incident with v is ac with $w(ac) = 2$, then $e_1 = ac$; now we can choose ab or bc because both edges as the same weight. Let's consider $e_2 = ab$, and now because we can't have a circuit, $e_3 = ce$ because $w(ce) = 6$, $e_4 = ed$ because $w(ed) = 4$ and $e_5 = df$ because $w(df) = 1$. Our minimum spanning tree is



with weight $W(T) = 17$.

References

[1] Edgar Goodair and Michael Parmenter. *Discrete Mathematics with Graph Theory.* (3rd Ed.) Pearson, 2006.

Exercises in MathE platform