Practice 5: Trees

(Degree in Computer Science Engineering)
Departament de Matemàtica Aplicada
ETS Enginyeria Informàtica
Universitat Politècnica de València

Contents

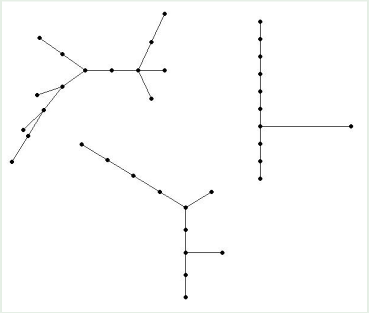
1. Trees
   - Definition of tree
   - Properties
2. Spanning trees of maximum/minimum cost
   - Spanning trees
   - Kruskal algorithms

Definition of tree

Definition (Tree)

A **cycle** is a closed path of positive length such that all its vertices are different (except the initial and final ones). A **tree** is a connected graph that has no cycle.
A **forest** is a graph whose connected components are trees.
The vertices of degree 1 in a tree are called **leaves**.

Example (The following graph is a forest. Every connected component is a tree)



Properties of trees

Properties of trees

A tree $G = (V, E, \psi)$ verifies the following properties:

1. A tree with n vertex has exactly $n-1$ edges. That is
$$|E| = |V| - 1.$$
2. Every tree has at least 2 leaves (vertex of degree 1).

Spanning trees

Definition (Spanning tree)

Given a connected graph G, in some cases it is interesting to find a subgraph T inside of G such that

- The sets of vertices of G and of T coincide.
- The set of edges of T is contained in the set of edges of G.
- T is a tree.

Such a graph T is called a **spanning tree** of G.

As we have seen, a spanning tree T keeps all the vertices of G connected, and it has less (or equal) edges than G.

Kruskal algorithms

In a connected graph, we can find several spanning trees.

If $G = (V, E, \psi)$ is a weighted graph, it will be of great interest to find trees such that the weights corresponding to the edges of the tree sum the maximum or minimum possible amount.

These special trees are very useful to solve optimization problems of benefits/costs in communications networks.

To find these special spanning trees we will use Kruskal's algorithms.

Kruskal's algorithm (maximum)

The following algorithm lets us compute the spanning tree of maximum (highest) weight.

Kruskal's algorithm (maximum)

Let $G = (V, E, \psi)$ be a weighted graph.

1. Make a list of all edges, ordered by their weights (**the first one, the one with maximum weight; and the last one, the one with the minimum**).
2. Create a void graph T, whose vertices are the same as in G, that is V.
3. Add the first edge in the list to T and remove it from the list.
4. Take the next edge in the list. If it does not close a cycle, add it to T. In any case, remove this edge from the list. Repeat the last step until you get an empty list.

Kruskal's algorithm (minimum)

The algorithm for the tree with minimum weight is quite similar:

Kruskal's algorithm (minimum)

Let $G = V(V, E, \psi)$ be a weighted graph.

1. Make a list of all edges, ordered by their weight (**the first one, the one with minimum weight; and the last one, the one with the maximum**).
2. Create a void graph T, whose vertices are the same as in G, that is, V.
3. Add the first edge in the list to T and remove it from the list.
4. Take the next edge in the list. If it does not close a cycle, add it to T. In any case, remove this edge from the list. Repeat the last step until you get an empty list.