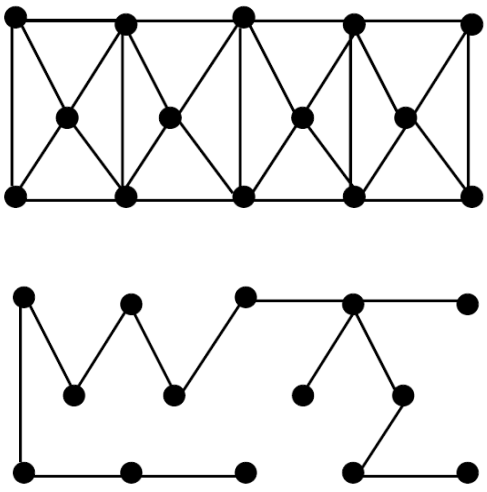
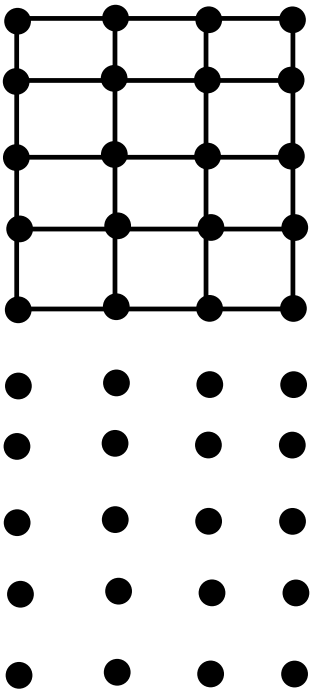
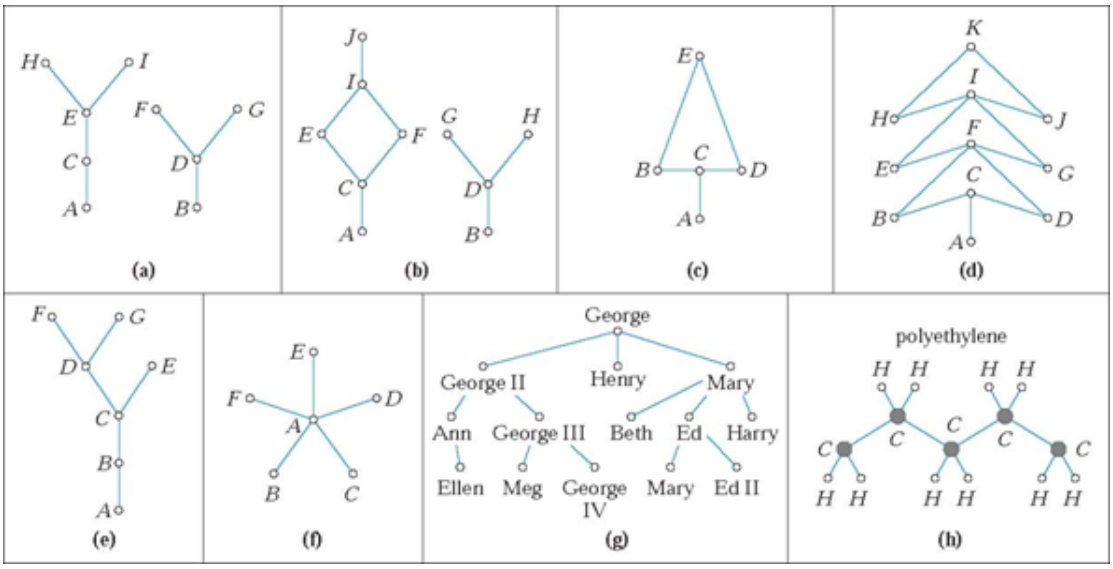
Networks, Spanning Trees and Steiner Points

- **Network**
  Another name for a connected graph.
- **Tree**
  A network with no circuits.
- **Spanning Tree**
  A *subgraph* that connects all the vertices of the network and has no circuits.
- **Minimum Spanning Tree (MST)**
  Among all spanning trees of a *weighted network*, one with the least total weight.







Properties of Trees

Property 1

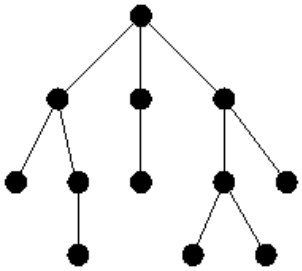
In a tree, there is one and only one path joining any two vertices.
If there is one and only one path joining any two vertices of a graph, then the graph must be a tree.

Property 2
In a tree, every edge is a bridge.
If every edge of a graph is a bridge, then the graph must be a tree.

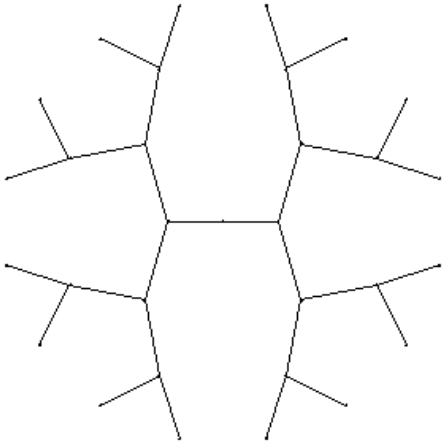
Here do we mean "graphs" or "networks"?

Property 3
A tree with N vertices has $N - 1$ edges.
If a network has N vertices and $N - 1$ edges, then it must be a tree.

(This second statement isn't quite true for all graphs. Just for networks. What is the difference?)



N= vertices
E= edges



N= vertices
E= edges

Note that every time you add a new vertex and an edge connecting it to the original tree, then you get a new tree and the number of edges is still one less than the number of vertices.

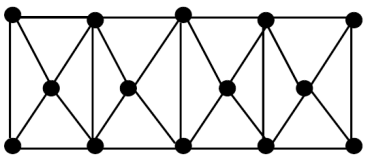
Spanning Trees

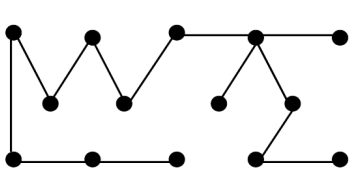
Property 4

If a network has N vertices and M edges, then $M \geq N - 1$.

$R = M - (N - 1)$ is the redundancy of the network. The number of edges you need to remove to make it a tree.

If M = N – 1, the network is a tree; if M > N –1, the network has circuits and is not a tree. (In other words, a tree is a network with zero redundancy and a network with positive redundancy is not a tree.

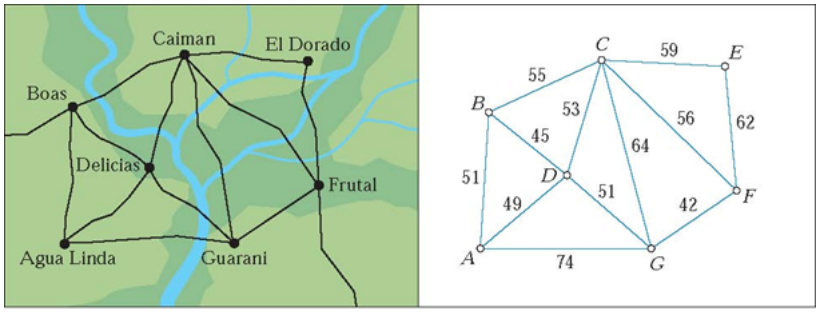




N= 14 Vertices
E= 29 edges
R= 29-(14-1)=16 extra edges

Consider the following map of seven Towns. Suppose your goal is to connect them all with a water irrigation pipeline. The graph at the right shows the "cost" of installing each of the possible pipes.

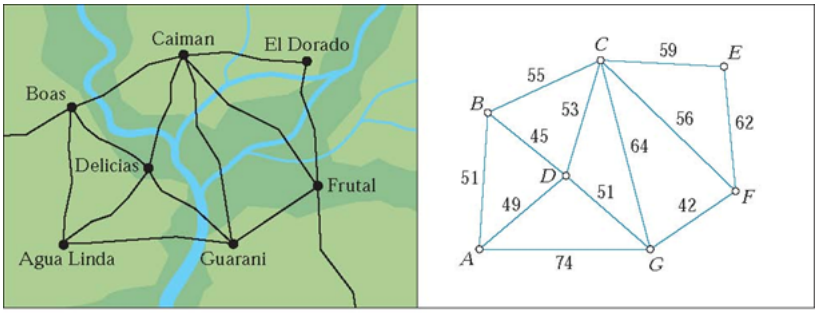
How would you do it?



(a) (b)

Total Cost:

try again.....

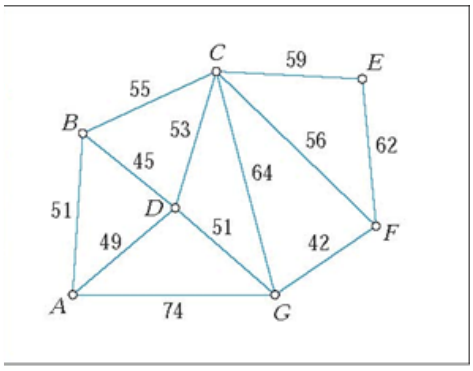


(a) (b)

Total Cost:

There are many spanning trees in a graph.
How could you find a Minimum Cost Spanning Tree?

Many ways to approach the problem.
We'll discuss "Kruskal's Algorithm".



(b)

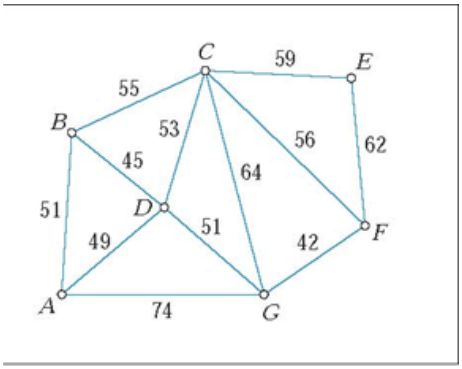
"Kruskal's Algorithm".

First Step. Among all the possible links, we choose the cheapest one,

Each Step: Pick Cheapest possible remaining link so that...

When do you finish?

How many edges do you have to add?



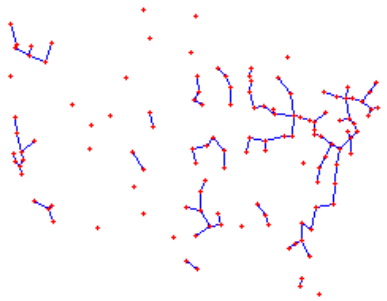
(b)

Two Examples of Kruskal's Algorithm

http://www.cut-the-knot.org/Curriculum/Combinatorics/WGraphs.shtml

Minimum Spanning Tree

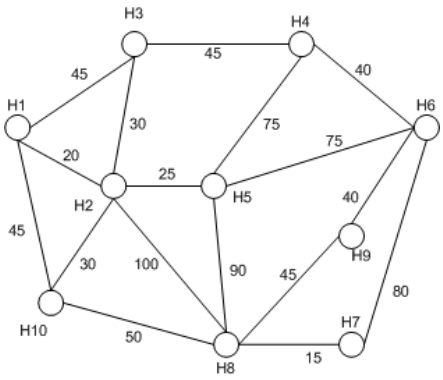
http://students.ceid.upatras.gr/~papagel/project/kruskal.htm

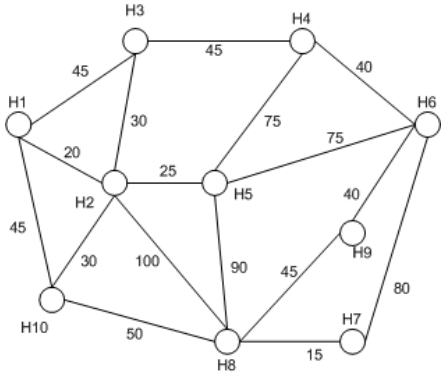


www.combinatorica.com

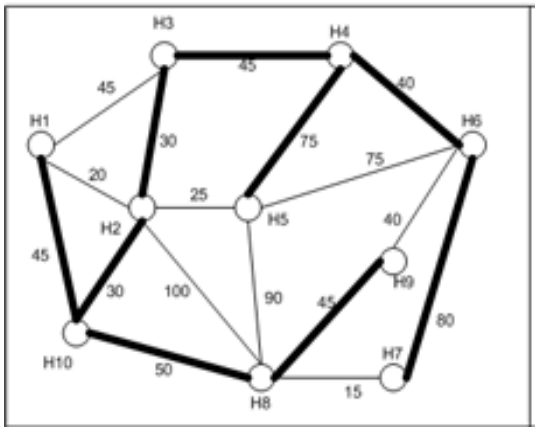
Is Kruskal's Algorithm Optimal?

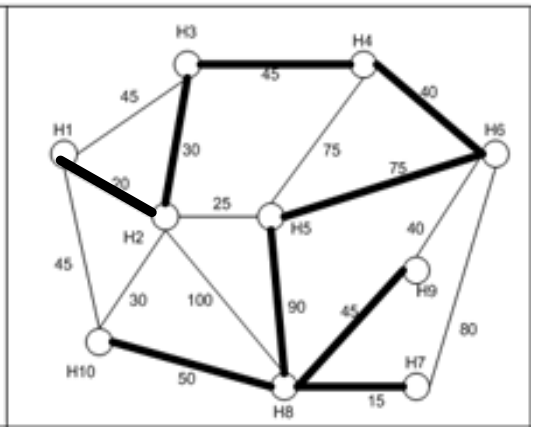
Does it always give the best possible spanning tree?





Two Spanning Trees





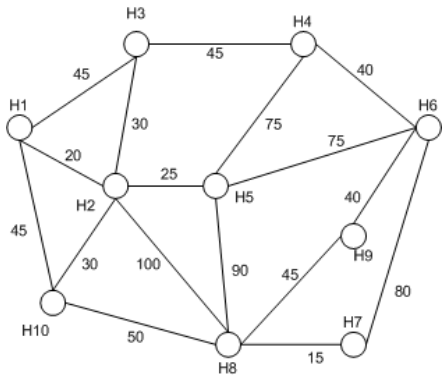
Cost = 45+30+50+30+45+75+40+80+45
= 440

Cost = 20+30+45+40+75+90+50+45+15
= 410

Optimal Spanning Tree has cost 295. How can you improve these spanning trees?

Look at a "cut set of edges" that breaks the graph into two parts.

Try Kruskal's Algorithm



Explanation that shows that Kruskal's Algorithm will always find the optimal spanning tree.

Every edge in the spanning tree is a bridge of the spanning tree.

Removing it breaks the tree into two disconnected parts. There are many edges from one part to the other.

Adding any of them will make a new spanning tree.

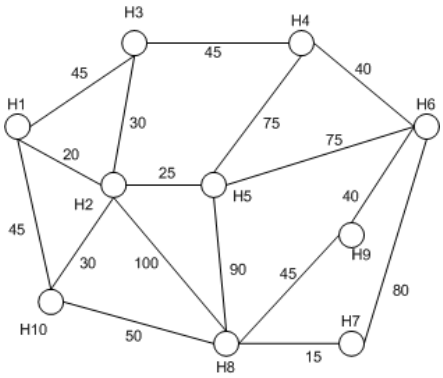
Picking the cheapest edge will make the cheapest of all those spanning trees.

Since Kruskal's algorithm adds the cheapest edges first, this assures that the resulting spanning tree will be the cheapest possible.

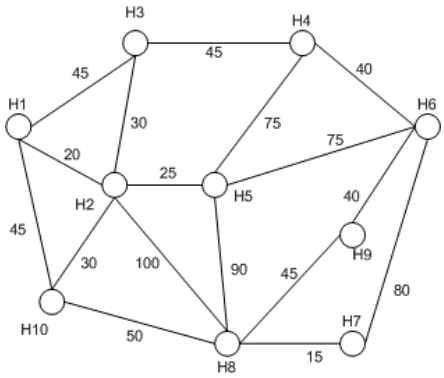
Another algorithm

Prim's Algorithm.

Informally:
1. Pick any point to start.
2. Chose the cheapest edge that connected your "tree-so-far" with a new vertex.
3. Repeat until all the vertices are included.



Start at H1.



Start at H7

Use Prim's Algorithm to generate a maze.

Start with a grid graph.
Give each edge a "random" weight.
Start at bottom right corner.
Add edges following Prim's Algorithm.

http://en.wikipedia.org/wiki/File:MAZE_30x20_Prim.ogv

Attachments

SingleSharePasswordProtectedVisualCryptographyViaCellularAut.cdf