Introduction to Graph Theory

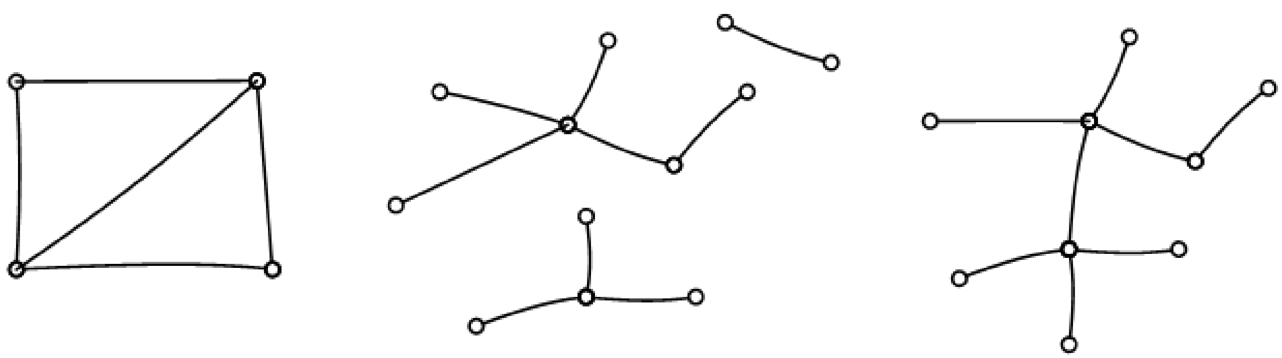
Lecture 26

Menu

- Trees and forests
- Spanning trees
- Minimum spanning tree
- Greedy algorithm for determining a minimum spanning tree
- Shortest path problem

Trees

- A **tree** is a connected graph with no cycles.
- A **forest** is a graph with no cycles and it may or may not be connected
- Example 1: Identify which of the following graphs are trees or forests.

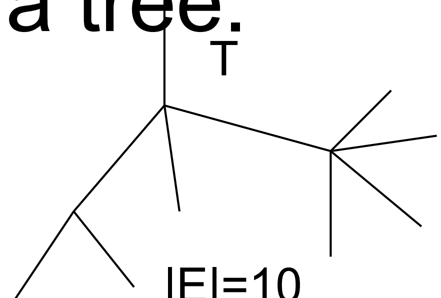


(a) (b) (c)

- Solution: (b) A forest (c) A tree

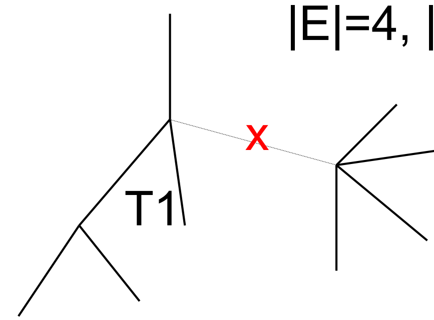
Tree properties

- If a tree T has at least two vertices then it has the following properties:
- There is exactly one path from any vertex V_i in T to any other vertex V_j
- The graph obtained from tree T by removing any edge has two components, each of which is a tree.
- $|E| = |V| - 1$



|E|=10
|V|=11

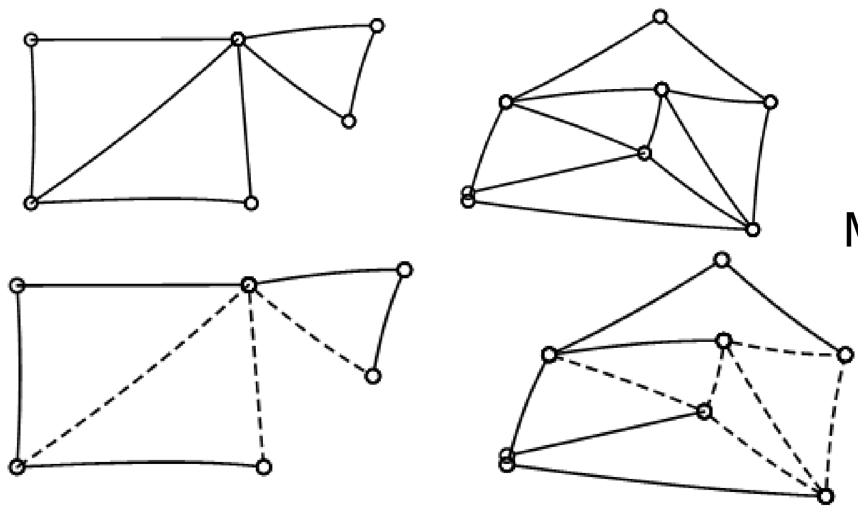
|E|=4, |V|=5



|E|=5, |V|=6

Spanning trees

- A **spanning tree** of a graph *G* is
  - a tree *T*
  - a spanning subgraph of *G*.
  - That is, *T* has the same vertex set as *G*.
- Example 2 Identify a spanning tree for each of the following graphs:



Hint: remove any edge which forms a cycle

Multiple spanning trees may exist

Given a graph G: How to draw a spanning tree?

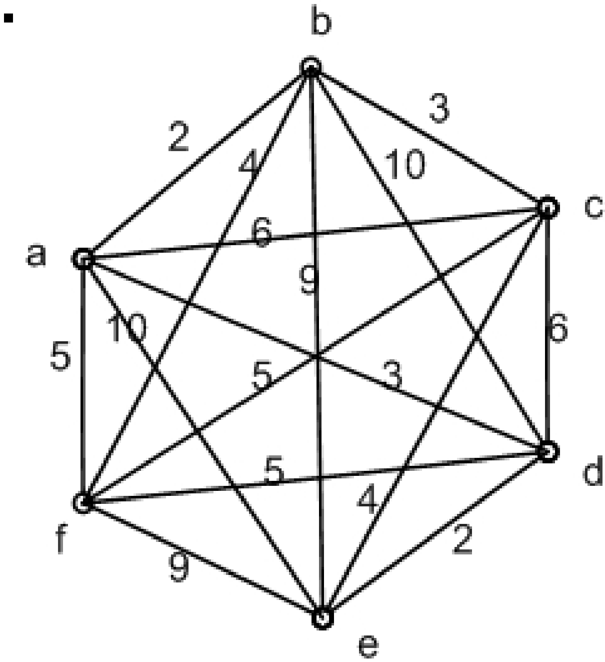
- Take any vertex of *G* as an initial partial tree.
- Add edges one by one so each new edge joins a new vertex to the partial tree.
- When to stop?
- If there are *n* vertices in the graph *G* then the spanning tree will have *n* vertices and *n*-1 edges.

Minimum spanning tree

- Suppose we have a group of offices which need to be connected by a network of communication lines.
- The offices may communicate with each other directly or through another office.
- Condition: there exists one path between any two vertices.
-
- In order to decide on which offices to build links between we firstly work out the cost of all possible connections.
- This will give us a weighted complete graph as shown next.
- The **minimum spanning tree** is then the spanning tree that has the <u>minimum cost among all spanning trees</u>.

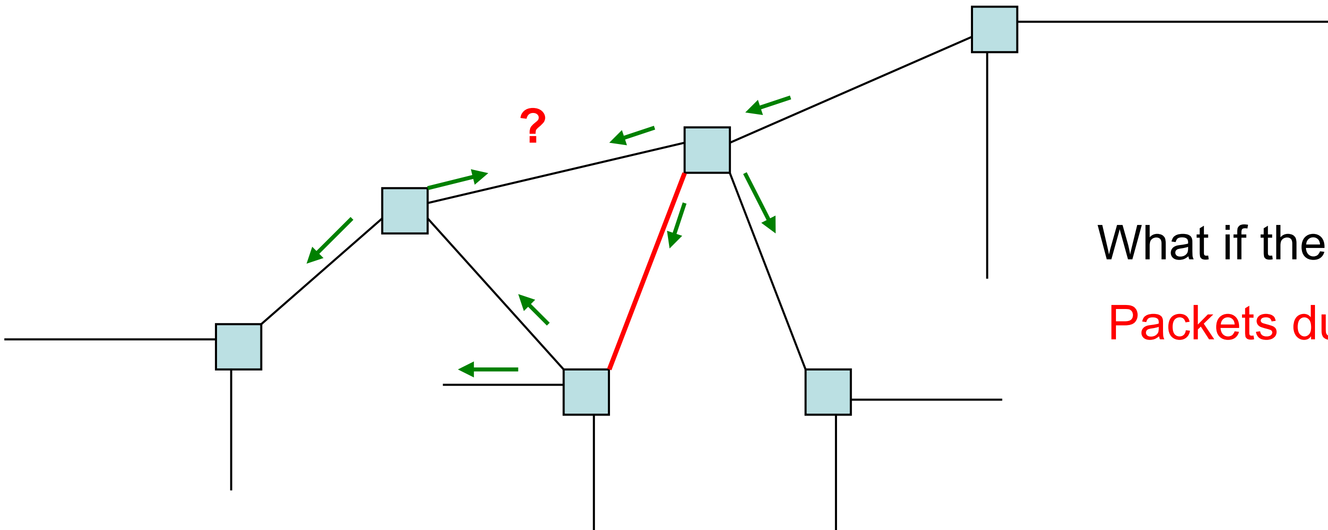
Minimum spanning tree

- A weighted complete graph.
- The vertices represent offices and the edges possible communication links.
- The weights on the edges represent the cost of construction of the link.



What is the use of minimum spanning tree?

- **City/town planning:** design a minimum-cost road layout connecting several cities
- Used in **communications**: Ethernet **bridge** layout **autoconfiguration** – avoid packets being sent over a network segment twice, so use of minimum spanning tree is required (no cycle)



What if there is a cycle?

Packets duplicated

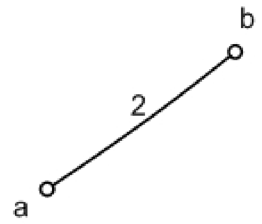
The gr$$dy algorithm for the minimum spanning tree

- Choose any start vertex to form the initial partial tree (*Vi*)
- Add the **chapst** edge, *Ei*, to a new vertex to form a new partial tree
- Repeat step 2 until all vertices have been included in the tree

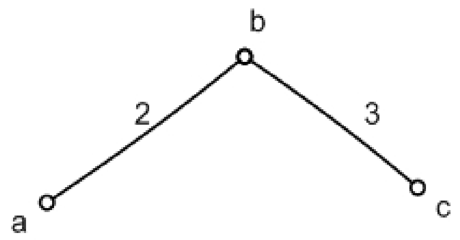
- Why is it ***greedy***?

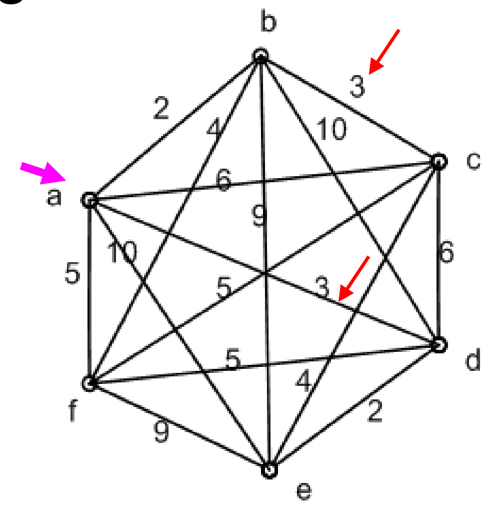
Find the minimum spanning tree for the graph representing communication links between offices as shown below.

- Start with any vertex, in this case choose the one marked *a*.
- Add the edge *ab* which is the cheapest edge of those adjacent to *a*.



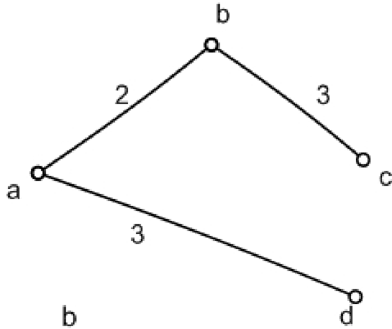
- Looking for the cheapest edge from among those incident to a or b, we find edges *bc* and *ad*, both costing 3, and that no other available edge costs less. We can choose either *bc* or *ad*. Arbitrarily we choose *bc*.

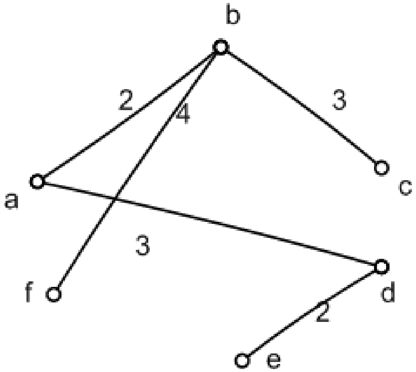




Find the minimum spanning tree for the graph representing communication links between offices as shown above.

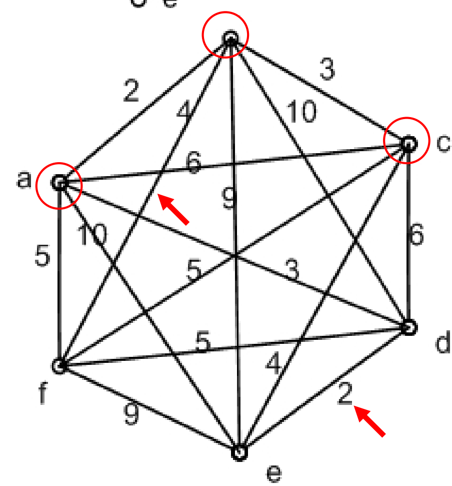
- We now look for the edge which is the cheapest remaining edge or those incident to *a* or *b* or *c* which forms a partial tree. This edge is *ad*.





- Continuing in this manner we find the minimum spanning tree shown:

- The total cost of our solution is found to be 2+3+3+2+4=14.



The shortest path problem

- The weights on a graph may represent *delays* in a communication network or *travel times* along roads.
- A practical problem to consider is to find the **shortest path** between any two vertices.
- **Shortest path → shortest delay**
- The algorithm to determine this will be demonstrated through an example.

Summary

- Definitions of **trees, forests** & **spanning trees**
- Shown **how to draw a spanning tree**
- Introduced the concept of a **minimum spanning tree**
- Presented the **gr$$dy algorithm** for determining a minimum spanning tree: shortest edge first
- Introduced **the shortest path problem**: to find the shortest path between any two vertices in a weighted graph

Readings

- [Mar07] Read 9.5
- [Mar13] Read 9.5