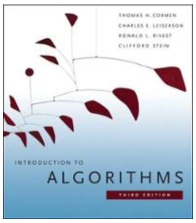
CS146 Data Structures and Algorithms



Chapter 23: Minimum Spanning Trees

Motivation: Minimum Spanning Trees

- To minimize the length of a connecting network, it never pays to have cycles.
- The resulting connection graph is connected, undirected, and acyclic, i.e., a *free tree* (sometimes called simply a *tree*).
- This is the *minimum spanning tree* or *MST* problem.

Formal Definition of MST

- Given a connected, undirected, graph $G = (V, E)$, a *spanning tree* is an *acyclic* subset of edges $T \subseteq E$ that connects all the vertices together.
- Assuming G is weighted, we define the *cost* of a spanning tree T to be the sum of edge weights in the spanning tree

$$w(T) = \sum_{(u,v)\in T} w(u,v)$$

- A *minimum spanning tree (MST)* is a spanning tree of minimum weight.

Figure1 : Examples of MST

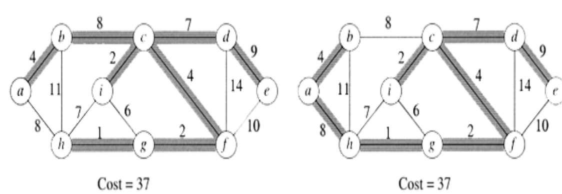
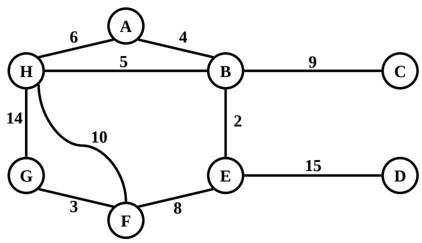


Figure 1: Minimum spanning tree.

- Not only do the edges sum to the same value, but the same set of edge weights appear in the two MSTs. NOTE: An MST may not be unique.

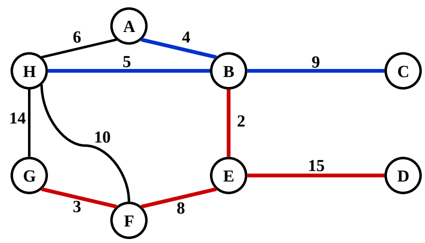
Minimum Spanning Tree

- *Which edges form the minimum spanning tree (MST) of the below graph?*



Minimum Spanning Tree

- Answer:



Minimum Spanning Tree

- MSTs satisfy the *optimal substructure* property: an optimal tree is composed of optimal subtrees
  - Let T be an MST of G with an edge (u,v) in the middle
  - Removing (u,v) partitions T into two trees T_1 and T_2
  - Claim: T_1 is an MST of $G_1 = (V_1,E_1)$, and T_2 is an MST of $G_2 = (V_2,E_2)$ (*Do V_1 and V_2 share vertices? Why?*)
  - Proof: $w(T) = w(u,v) + w(T_1) + w(T_2)$
    (There can't be a better tree than T_1 or T_2, or T would be suboptimal)