



IE 680 – Special Topics in Production Systems: Networks, Routing and Logistics*

Rakesh Nagi
Department of Industrial Engineering
University at Buffalo (SUNY)

**Lecture notes from Network Flows by Ahuja, Magnanti and Wong (1993)*



Minimum Spanning Trees

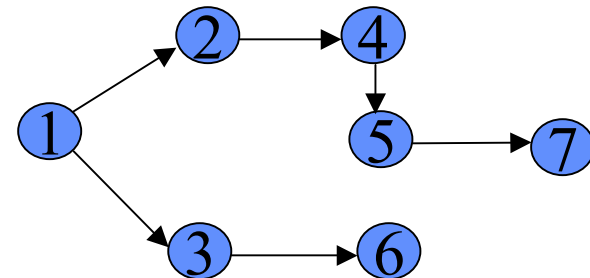
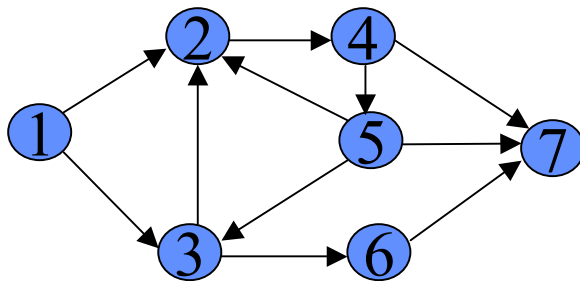
- Introduction
 - Definition of Spanning Tree
 - Applications
 - Optimality Conditions
- Minimum Spanning Tree Algorithms
 - Prim's Algorithm
 - Kruskal's Algorithm
 - Sollin's Algorithm
- Summary



Minimum Spanning Trees

- Spanning tree:

A tree T is a spanning tree of G if T is a spanning subgraph of G . Every spanning tree of a connected n -node graph G has $(n-1)$ arcs.



The arcs belonging to a spanning tree are called tree arcs and the rest are called nontree arcs.



Minimum Spanning Trees

- Applications
 - Designing Physical Systems
 - e.g.; Constructing a pipeline network to connect a number of towns using the smallest possible total length of pipeline.
 - Optimal Message Passing
 - Reducing Data Storage
 - Cluster Analysis
 - All-pairs Minimax Path Problem



Minimum Spanning Trees

Optimality Conditions

- **Cut Optimality Condition:** A Spanning Tree T^* is a minimum spanning tree if and only if it satisfies the following cut optimality conditions:
 - For every tree arc $(i,j) \in T^*$, $c_{ij} \leq c_{kl}$ for every arc (k,l) contained in the cut formed by deleting the arc (i,j) from T^* .
 - This implies that every arc in a minimum spanning tree is a minimum cost arc across the cut that is defined by removing it from the tree.
- **Path Optimality Condition:** A Spanning Tree T^* is a minimum spanning tree if and only if it satisfies the following path optimality conditions:
 - For every tree arc (k,l) of G , $c_{ij} \leq c_{kl}$ for every arc (i,j) contained in the path in T^* connecting nodes k and l .

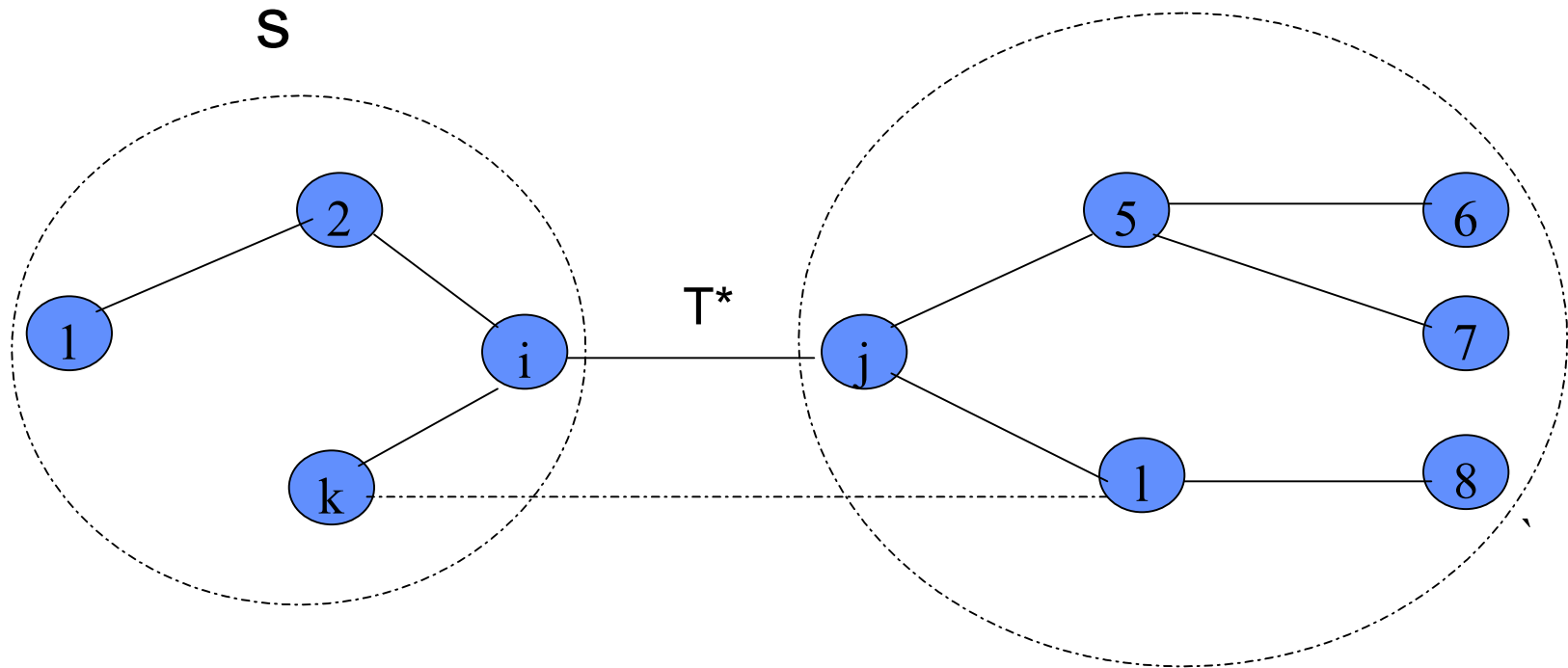


Minimum Spanning Trees

- Optimality Conditions

\bar{S}

S



Proving Cut and Path Optimality Conditions



Minimum-Spanning Tree Algorithm (Prim's Algorithm)

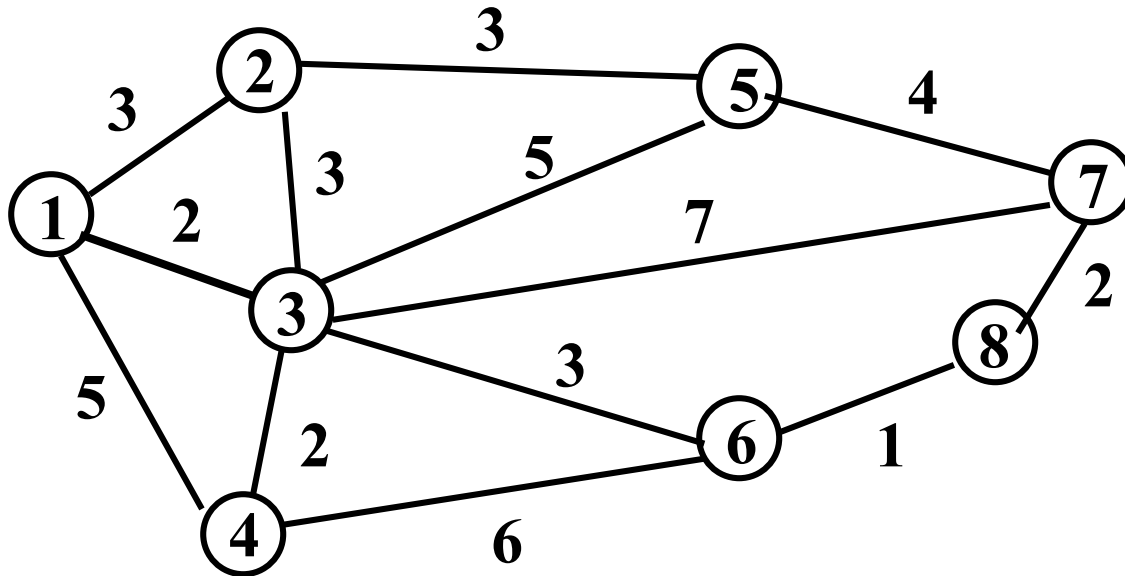
Based on cut optimality condition

1. Select any node in the network.
2. Connect this node to the nearest node that minimizes the total distance.
3. Considering all of the nodes that are now connected, find and connect the nearest node that is not connected.
4. Repeat the third step until all nodes are connected.
5. If there is a tie in the third step and two or more nodes that are not connected are equally near, arbitrarily select one and continue. A tie suggests that there might be more than one optimal solution.



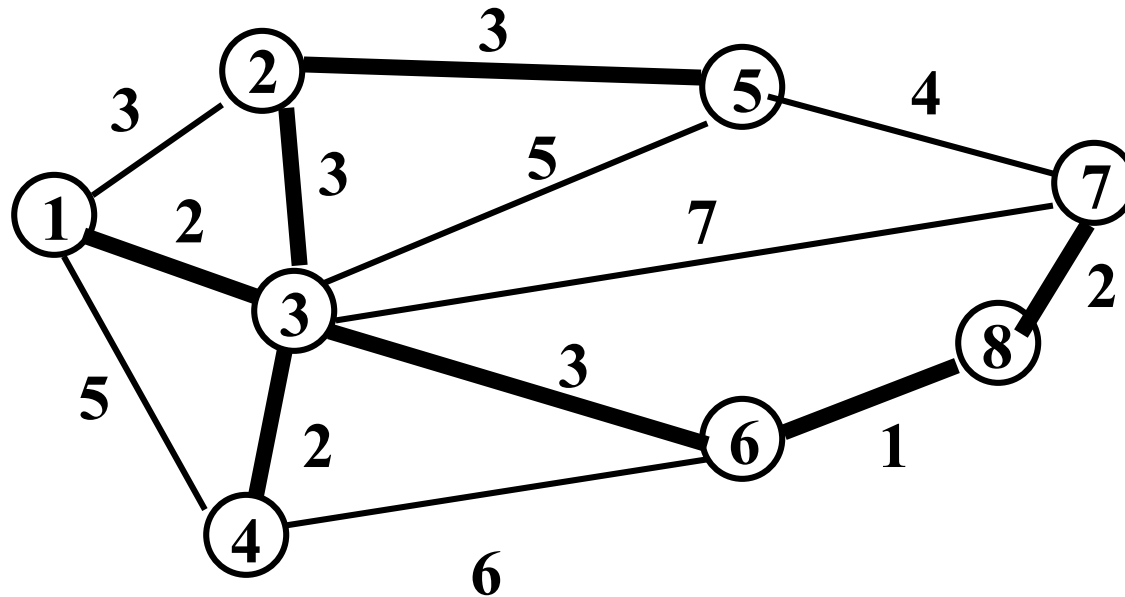
Minimum-Spanning Tree Algorithm (Prim's Algorithm)

- Given Network





Minimum-Spanning Tree Algorithm (Prim's Algorithm) Final Result



Cheapest/Minimum Tree has a total distance or cost of 16.



Minimum-Spanning Tree Algorithm (Kruskal's Greedy Algorithm)

Based on path optimality condition

Step 1: Begin with just the nodes and none of the arcs constructed.

Step 2: Of the remaining arcs that could be constructed, choose the cheapest/smallest. Break ties arbitrarily.

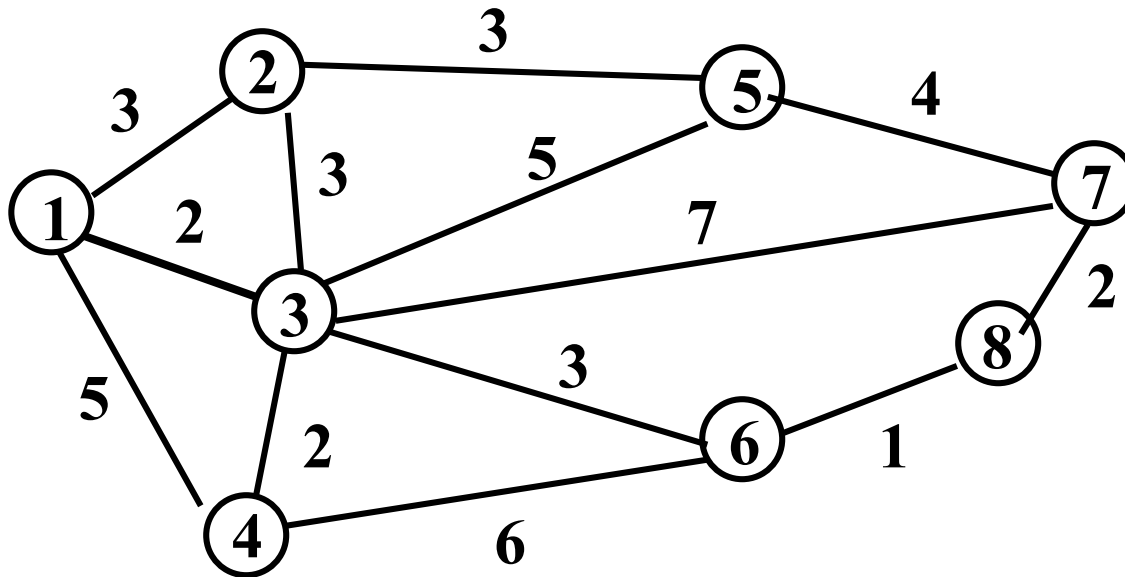
If constructing this arc does not create a cycle in the graph, construct it. Else discard this arc from further consideration.

Step 3: Repeat Step 2 until all nodes are connected.



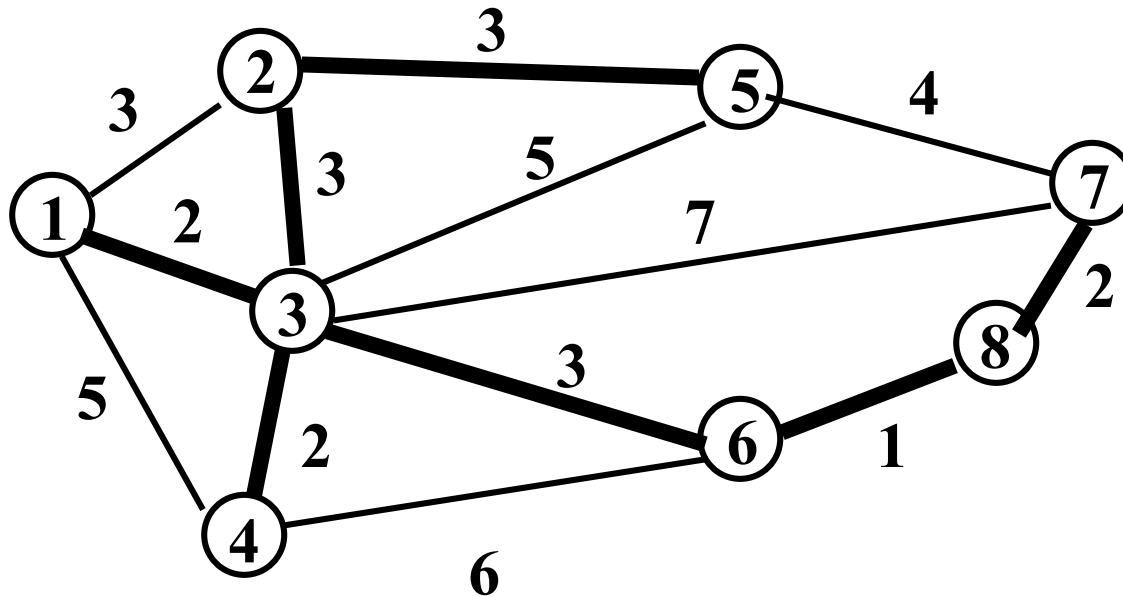
Minimum-Spanning Tree Algorithm (Kruskal's Greedy Algorithm)

- Given Network





Minimum-Spanning Tree Algorithm (Kruskal's Greedy Algorithm) Final Result



Cheapest/Minimum Tree has a total distance or cost of 16.



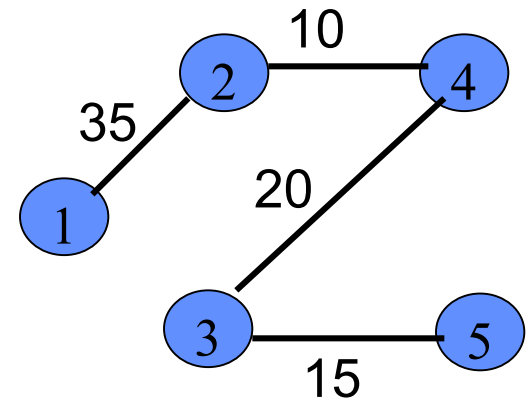
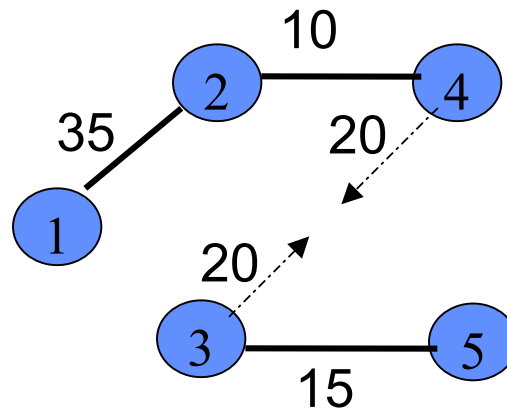
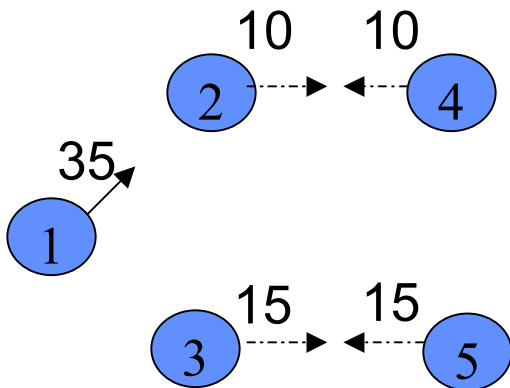
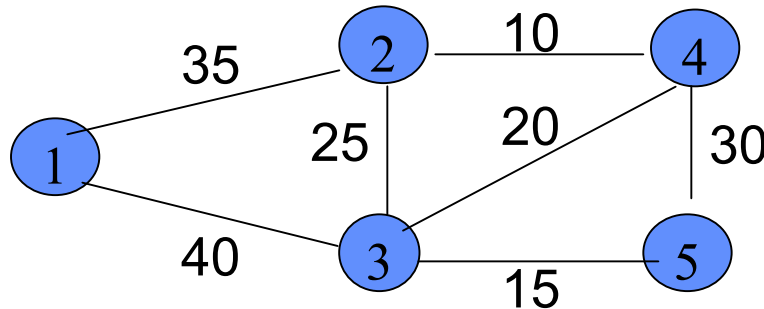
Minimum-Spanning Tree Algorithm (Sollin's Algorithm)

Hybrid version of Kruskal's and Prim's algorithm.

- It performs the two basic operations:
 - Nearest Neighbor – This operation takes as an input a tree spanning the nodes N_k and determines an arc (i_k, j_k) with the minimum cost among all arcs emanating from N_k .
 - Merge (i_k, j_k) – This operation takes as an input two nodes i_k and j_k , and if the two nodes belong to two different trees, then merge these two trees into a single tree.



Minimum-Spanning Tree Algorithm (Sollin's Algorithm)



Cheapest/Minimum Tree has a total distance or cost of

$$= 35 + 10 + 20 + 15 = 80$$