OPERATIONAL RESEARCH II

Agustina Eunike, ST., MT., MBA.
Industrial Engineering – University of Brawijaya

MINIMAL SPANNING TREE PROBLEM

Minimal Spanning Tree Problem

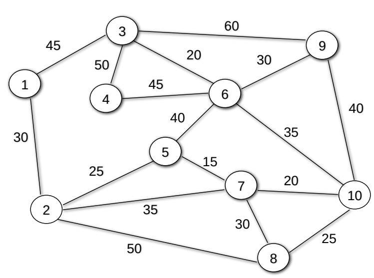
- A tree is a set of connected arcs that does not form a cycle.
- A spanning tree is a tree that connects all nodes of a network.
- The minimal spanning tree problem seeks to determine the minimum sum of arc lengths necessary to connect all nodes in a network.
- The criterion to be minimized in the minimal spanning tree problem is not limited to distance even though the term "closest" is used in describing the procedure. Other criteria include time and cost. (Neither time nor cost are necessarily linearly related to distance.)
- The problem involves *choosing* for the network the *links* that have the ***shortest total length***

Minimal Spanning Tree Problem

- Step 1: arbitrarily begin at any node and connect it to the closest node. The two nodes are referred to as connected nodes, and the remaining nodes are referred to as unconnected nodes.
- Step 2: identify the unconnected node that is closest to one of the connected nodes (break ties arbitrarily). Add this new node to the set of connected nodes. Repeat this step until all nodes have been connected.
- Note: A problem with ***n* nodes** to be connected will require ***n* - 1 iterations** of the above steps.

Example: Minimal Spanning Tree

- Find the minimal spanning tree:



Example: Minimal Spanning Tree

- Iteration 1: arbitrarily selecting node 1, we see that its closest node is node 2 (distance = 30). Therefore, initially we have:
  - Connected nodes: 1,2
  - Unconnected nodes: 3,4,5,6,7,8,9,10
  - Chosen arcs: 1-2
- Iteration 2: The closest unconnected node to a connected node is node 5 (distance = 25 to node 2). Node 5 becomes a connected node.
  - Connected nodes: 1,2,5
  - Unconnected nodes: 3,4,6,7,8,9,10
  - Chosen arcs: 1-2, 2-5

Example: Minimal Spanning Tree

- Iteration 3: The closest unconnected node to a connected node is node 7 (distance = 15 to node 5). Node 7 becomes a connected node.

  Connected nodes: 1,2,5,7

  Unconnected nodes: 3,4,6,8,9,10

  Chosen arcs: 1-2, 2-5, 5-7

- Iteration 4: The closest unconnected node to a connected node is node 10 (distance = 20 to node 7). Node 10 becomes a connected node.

  Connected nodes: 1,2,5,7,10

  Unconnected nodes: 3,4,6,8,9

  Chosen arcs: 1-2, 2-5, 5-7, 7-10

Example: Minimal Spanning Tree

- Iteration 5: The closest unconnected node to a connected node is node 8 (distance = 25 to node 10). Node 8 becomes a connected node.

  Connected nodes: 1,2,5,7,10,8

  Unconnected nodes: 3,4,6,9

  Chosen arcs: 1-2, 2-5, 5-7, 7-10, 10-8

- Iteration 6: The closest unconnected node to a connected node is node 6 (distance = 35 to node 10). Node 6 becomes a connected node.

  Connected nodes: 1,2,5,7,10,8,6

  Unconnected nodes: 3,4,9

  Chosen arcs: 1-2, 2-5, 5-7, 7-10, 10-8, 10-6

Example: Minimal Spanning Tree

- Iteration 7: The closest unconnected node to a connected node is node 3 (distance = 20 to node 6). Node 3 becomes a connected node.

  Connected nodes: 1,2,5,7,10,8,6,3

  Unconnected nodes: 4,9

  Chosen arcs: 1-2, 2-5, 5-7, 7-10, 10-8, 10-6, 6-3

- Iteration 8: The closest unconnected node to a connected node is node 9 (distance = 30 to node 6). Node 9 becomes a connected node.

  Connected nodes: 1,2,5,7,10,8,6,3,9

  Unconnected nodes: 4

  Chosen arcs: 1-2, 2-5, 5-7, 7-10, 10-8, 10-6, 6-3, 6-9

Example: Minimal Spanning Tree

- Iteration 9: The only remaining unconnected node is node 4. It is closest to connected node 6 (distance = 45).

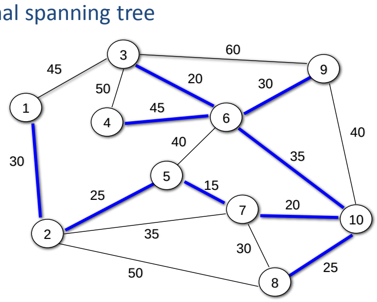
  Thus, the minimal spanning tree (displayed on the next slide) consists of:

  Arcs: 1-2, 2-5, 5-7, 7-10, 10-8, 10-6, 6-3, 6-9, 6-4

  Values: 30+25+15+20+25+35+20+30+45 = 245

Example: Minimal Spanning Tree

- Optimal spanning tree



Minimal Spanning Tree Problem

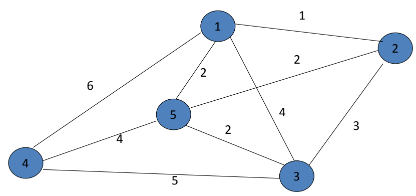
- Note:

  The minimum spanning tree problem is the broad category of *network design.* The objective is to design the most appropriate network for the given application rather than analyzing an already designed network.

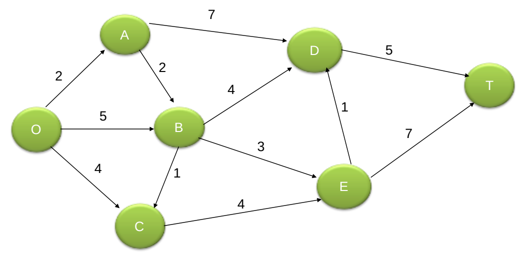
  A survey of this area:

  Magnanti, T. L., and R. T. Wong: "Network Design and Transportation Planning: Models and Algoritms," *Transportation Science,* **18**: 1-55, 1984.

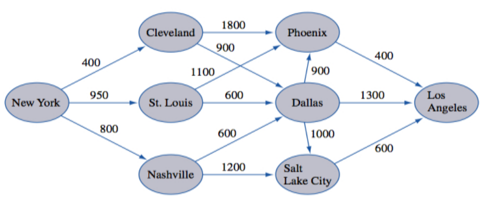
Example 2



Example 3 : Spanning Tree Problem for Servaada Park



- Following is a network representation of electricity system. What is the cheapest way to connect all cities in the system? Costs of connection are on arcs.



MAXIMUM FLOW PROBLEM

Maximum Flow Problem

- The maximum flow problem is concerned with determining the maximizes volume of flow from one node (called the source) to another node (called the sink). Only one node as supply node, and one node as demand node, the rest are transhipment nodes.
- The objective is to determine the feasible pattern of flows through the network that *maximizes the total flow* from the source to the sink. This amount is measured in either of two equivalent ways (the amount *leaving the source* or the amount *entering the sink*).
- In the maximal flow problem, each arc has a maximum arc flow capacity which limits the flow through the arc.
- It is possible that an arc, (i, j), may have a different flow capacity from i to j than from j to i.
- Maximum flow problem can be formulated as a *linear programming problem* and can be solved by the simplex method.

Maximum Flow Algorithm

- Step 1: find any path from the source node to the sink node that has positive flow capacities (in the direction of the flow) for all arcs on the path. If no path is available, then the optimal solution has been found.
- Step 2: find the smallest arc capacity, p_f, on the path selected in step 1. Increase the flow through the network by sending the amount, p_f, over this path.
- Step 3: For the path selected in step 1 reduce all arc flow capacities in the direction of the flow by p_f and increase all arc flows in the opposite direction of the flow by p_f. Go to step 1.

Maximum Flow Algorithm

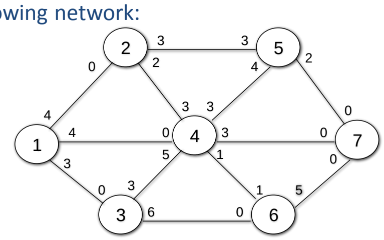
- Note:

Students often ask if it is necessary to increase the arc flow capacities in the opposite direction of the flow (later part of step 3), since it appears to be a wasted effort. The answer is YES, it is necessary.

This creation of fictitious capacity allows us to alter a previous flow assignment if we need to. Otherwise, we might have to terminate the algorithm before reaching an optimal solution.

Example: Maximum Flow

- Find the maximal flow from node 1 to node 7 in the following network:



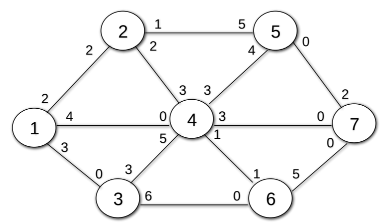
Example: Maximum Flow

- Iteration 1
  - Step 1: find a path from the source node, 1, to the sink node, 7, that has flow capacities greater than zero on all arcs of the path. One such path is 1-2-5-7.
  - Step 2: The smallest arc flow capacity on the path 1-2-5-7 is the minimum of {4, 3, 2} = 2.
  - Step 3: reduce all arc flows in the direction of the flow by 2 on this path and increase all arc flows in the reverse direction by 2:

(1-2) 4 - 2 = 2	(2-1) 0 + 2 = 2
(2-5) 3 - 2 = 1	(5-2) 3 + 2 = 5
(5-7) 2 - 2 = 0	(7-5) 0 + 2 = 2

Example: Maximum Flow

- Iteration 1 results



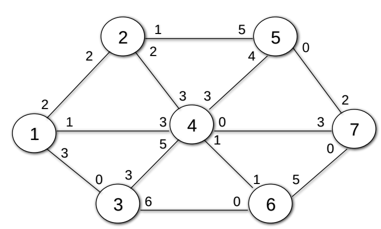
Example: Maximum Flow

- Iteration 2
  - Step 1: path 1-4-7 has flow capacity greater than zero on all arcs.
  - Step 2: The minimum arc flow capacity on 1-4-7 is 3.
  - Step 3: reduce the arc flow capacities on the path in the direction of the flow by 3, and increase these capacities in the reverse direction by 3:

(1-4) 4 - 3 = 1	(4-1) 0 + 3 = 3
(4-7) 3 - 3 = 0	(7-4) 0 + 3 = 3

Example: Maximum Flow

- Iteration 2 results



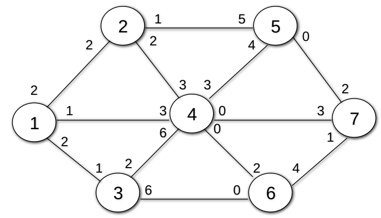
Example: Maximum Flow

- Iteration 3
  - Step 1: path 1-3-4-6-7 has flow capacity greater than zero on all arcs.
  - Step 2: The minimum arc capacity on 1-3-4-6-7 is 1.
  - Step 3: reduce the arc capacities on the path in the direction of the flow by 1 and increase the arc capacities in the reverse direction of the flow by 1:

(1-3) 3 - 1 = 2	(3-1) 0 + 1 = 1
(3-4) 3 - 1 = 2	(4-3) 5 + 1 = 6
(4-6) 1 - 1 = 0	(6-4) 1 + 1 = 2
(6-7) 5 - 1 = 4	(7-6) 0 + 1 = 1

Example: Maximum Flow

- Iteration 3 results



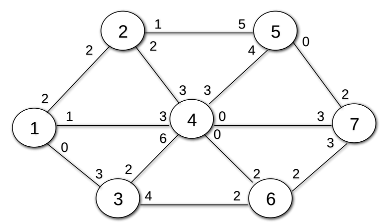
Example: Maximum Flow

- Iteration 4
  - Step 1: path 1-3-6-7 has flow capacity greater than zero on all arcs.
  - Step 2: The minimum arc capacity on 1-3-6-7 is 2.
  - Step 3: reduce all arc flow capacities on the path in the direction of the flow by 2 and increase the arc flow capacities in the reverse direction by 2:

(1-3) 2 - 2 = 0	(3-1) 1 + 2 = 3
(3-6) 6 - 2 = 4	(6-3) 0 + 2 = 2
(6-7) 4 - 2 = 2	(7-6) 1 + 2 = 3

Example: Maximum Flow

- Iteration 4 results



Example: Maximum Flow

- Iteration 5
  - Step 1: using the shortest route algorithm, the shortest route from node 1 to node 7 is 1-2-4-3-6-7.
  - Step 2: The smallest arc capacity on 1-2-4-3-6-7 is 2.
  - Step 3: reduce the arc flow capacities on the path in the direction of the flow by 2 and increase these capacities in the reverse direction of the flow by 2:

(1-2) 2 - 2 = 0	(2-1) 2 + 2 = 4
(2-4) 2 - 2 = 0	(4-2) 3 + 2 = 5
(4-3) 6 - 2 = 4	(3-4) 2 + 2 = 4
(3-6) 4 - 2 = 2	(6-3) 2 + 2 = 4
(6-7) 2 - 2 = 0	(7-6) 3 + 2 = 5

Example: Maximum Flow

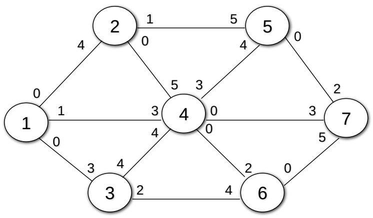
- Note:

Arc 3-4 is a case where in iteration 3, flow of 1 unit was directed from node 3 to node 4. In iteration 5 flow of 2 units was directed from node 4 to node 3.

By subtracting the assigned flow from the capacity of the "sending" end of the arc and adding it to the "receiving" end of the arc, the net effect of the oppositely directed flow assignments is readily known.

Example: Maximum Flow

- Iteration 5 results



Example: Maximum Flow

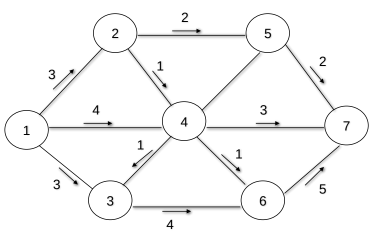
- Note

There are no arcs with positive flow into the sink node 7. Thus, the maximal possible flow from node 1 to node 7 has been found.

To identify the maximal flow amount and how it is to be achieved (directed), compare the original capacities with the adjusted capacities of each arc in both directions. If the adjusted capacity is less than the original capacity, the difference represents the flow amount for that arc.

Example: Maximum Flow

- Solution summary

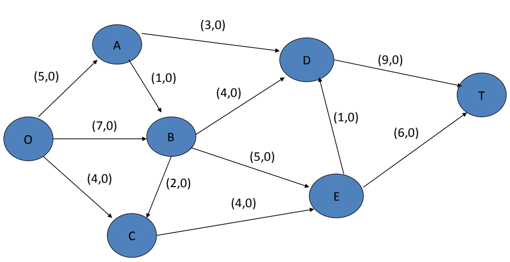


Example: Maximum Flow

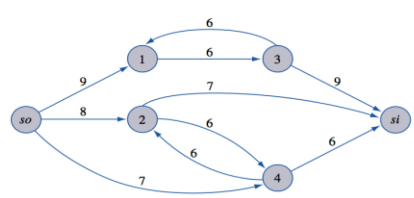
- Note

There is a degree of randomness to the maximal flow algorithm. (Recall that step 1 states "find any path ...") as long as you follow the algorithm you will reach an optimal solution, regardless of your path choice in each iteration. Two people solving the same problem might get the same optimal solution or their solutions might differ in regard to flow routings, but the maximal flows will be the same.

Example



- Find the maximum flow from source to sink



References

- Frederick Hillier and Gerald J.Lieberman, Introduction to Operations Research, Holden Day Ltd, San Fransisco, 1997
- Taha, Hamdy, Operation Research : An Introduction, Macmillan Publishing Company., New York, 1997