# Operations Research I

## Network Optimization Models

Ing. Lenka Skanderová, Ph.D.

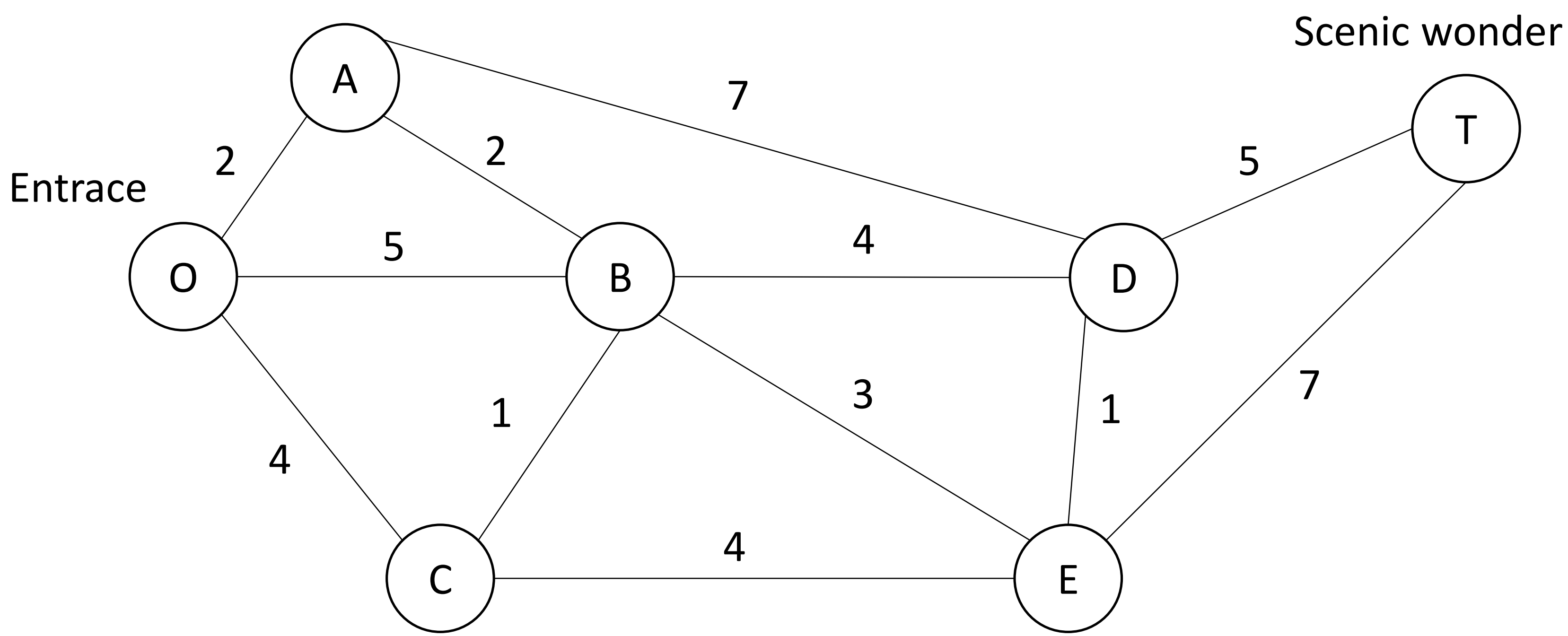
# Network Representation

- Widely used in different areas:
  - Financial planning
  - Resource management
  - Project planning
  - Distribution
  - Facilities location
  - Etc.

# Prototype Example
## Network Representation



SEERVADA PARK – has recently been set aside for a limited amount of sightseeing and backpack hiking. Cars are not allowed. There is only the road system for trams and rangers cars.




[1]

# Prototype Example
## Problems



SEERVADA PARK – a small number of trams is used to transport sightseers from the park entrace to station T and back.

Problems:

1. Which route from the park entrace O to station T has the smallest total distance for the operation of the trams? SHORTEST PATH PROBLEM

2. Telephone lines must be installed under the roads to establish telephone communication among all the stations. The installation is expensive and disruptive for natural environment ⇒ lines will ne installed under just enough roads to provide some connection between each pair of stations. Where the lines should be laid to accomplish this with a minimum total number of miles of line installed. MINIMUM SPANNING TREE

[1]

# Prototype Example

## Problems

SEERVADA PARK – a small number of trams is used to transport sightseers from the park entrace to station T and back.

Problems:

3. Strict ration has been placed on the number of trips that can be made on each of the roads per day ()because of ecology). How to various trips to maximize the number of trips that can be made per day without violating the limits on any individual road? MAXIMUM FLOW PROBLEM

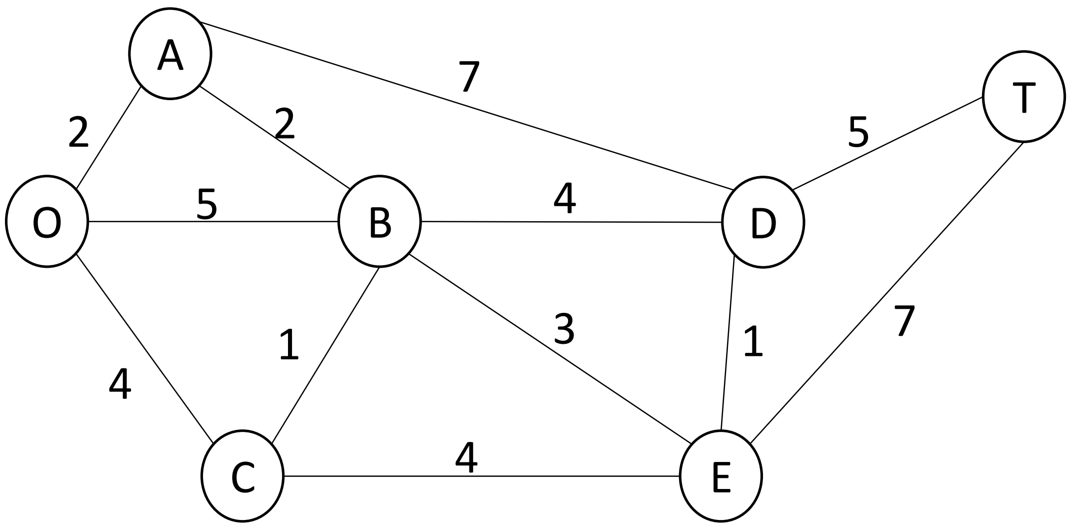
[1]

# Prototype Example
## Shortest Path



Which route from the park entrace O to station T has the smallest total distance for the operation of the trams?

- Dijkstra‘s algorithm
- Floyd-Warshall algorithm
- Algorithm based on the n-th nearest neighbor




[1]

# Shortest Path

## $N$-Nearest Neighbor Algorithm

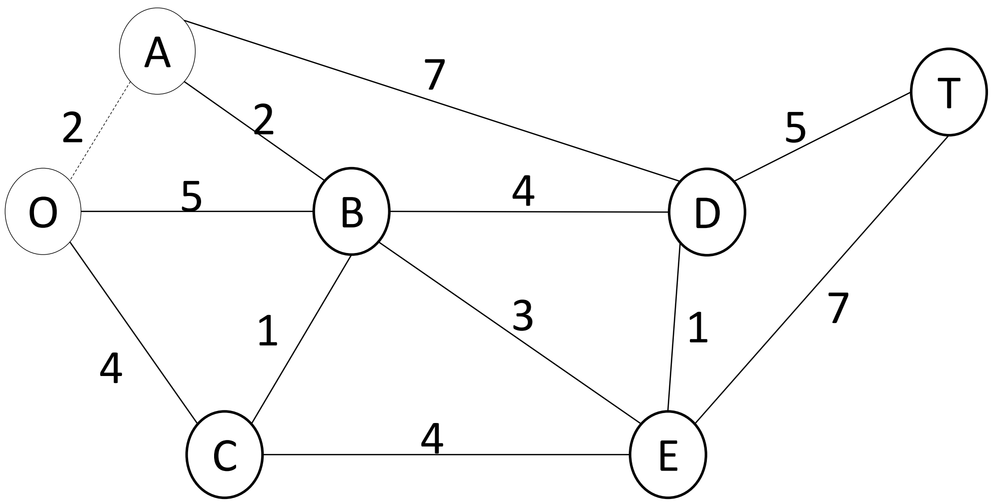


- Algorithm works with iterations:
  - At each iteration we will search for $n$th nearest neighbor to the origin, where $n = 1, 2, \ldots$ until $n$th nearest neighbor is the destination
  - The inputs at each iteration are the $n - 1$ nearest nodes to the origin including their shortest paths and distances from the origin. These nodes + origin are denoted as **solved nodes**
  - Each solved node directly connected to one or more unsolved nodes provide one candidate – the unsolved node with the shortest connecting link
  - For each such solved node and its candidate add distance between them and the distance of the shortest path from the origin to this solved node. The candidate with the smallest such total distance is the $n$th neaarest node [1]

# Shortest Path

## $N$-Nearest Neighbor Algorithm

- Prototype example
- **Shortest path**
- Minimum spanning tree
- Maximum flow problem
- Minimum cost flow problem



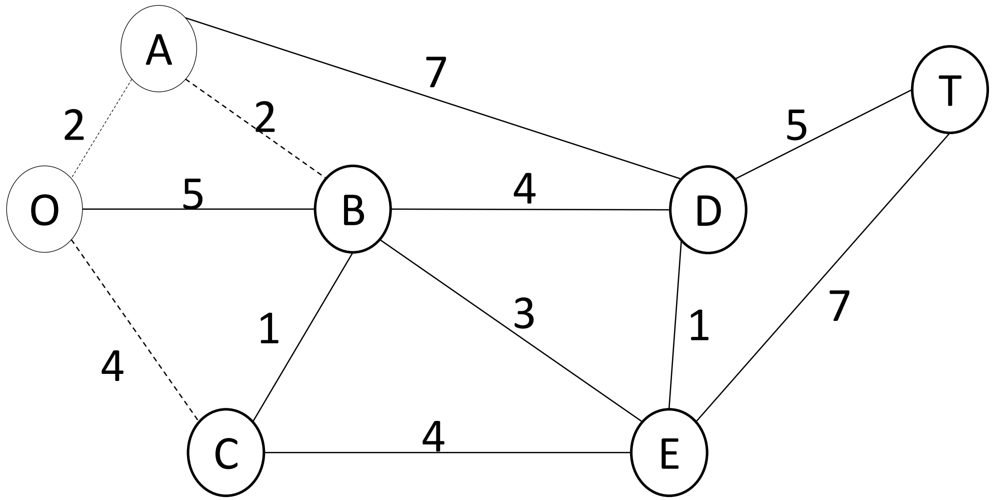

| $n$ | Solved nodes directly connected to unsolved nodes | Closest Connected Unsolved Node | Total distance involved | $n$th nearest node | Min. distance | Last Connection |
|---|---|---|---|---|---|---|
| 1 | O | A | 2 | A | 2 | OA |

[1]

# Shortest Path

## $N$-Nearest Neighbor Algorithm





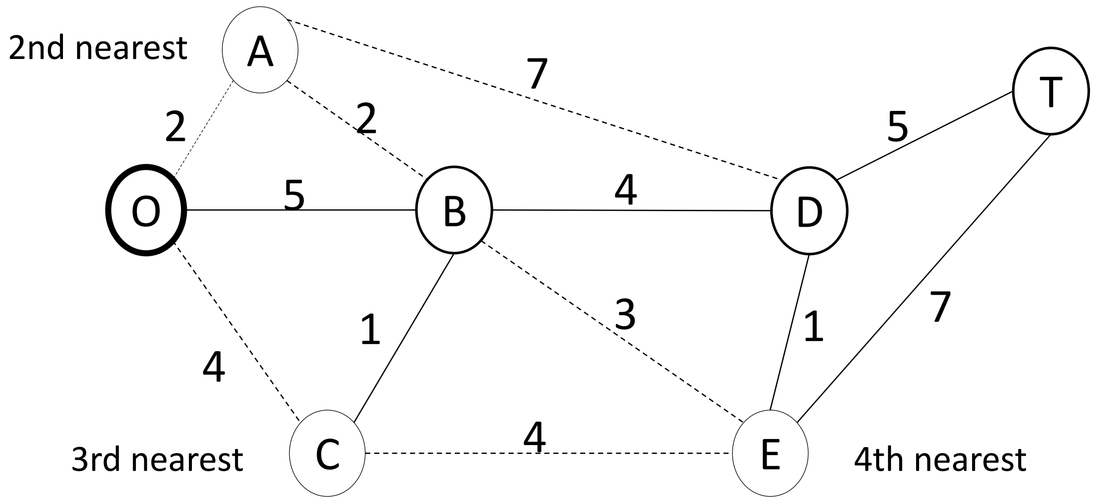

| $n$ | Solved nodes directly connected to unsolved nodes | Closest Connected Unsolved Node | Total distance involved | $n$th nearest node | Min. distance | Last Connection |
|---|---|---|---|---|---|---|
| 2 | O | C | 4 | C | 4 | OC |
| 3 | A | B | 2+2=4 | B | 4 | AB |

[1]

# Shortest Path

## $N$-Nearest Neighbor Algorithm





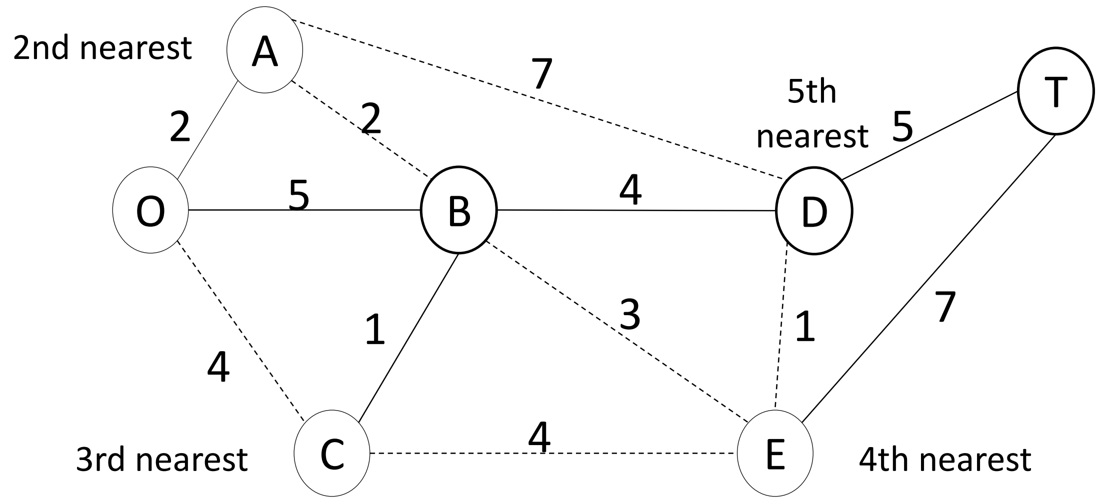

| $n$ | Solved nodes directly connected to unsolved nodes | Closest Connected Unsolved Node | Total distance involved | $n$th nearest node | Min. distance | Last Connection |
|---|---|---|---|---|---|---|
| 4 | A | D | 2+7=9 | E | 7 | BE |
| | B | E | 4+3=7 | | | |
| | C | E | 4+4=8 | | | |

[1]

# Shortest Path

## $N$-Nearest Neighbor Algorithm





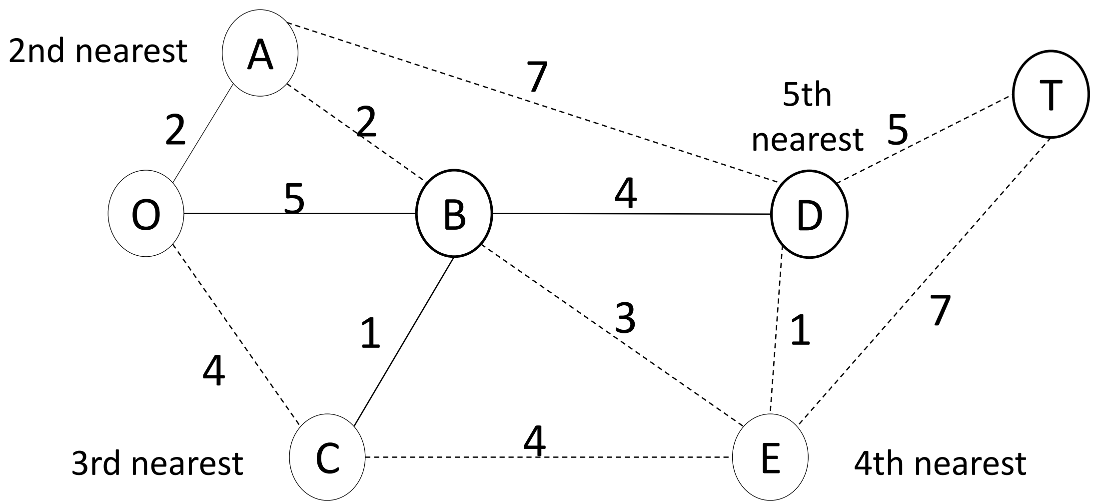

| $n$ | Solved nodes directly connected to unsolved nodes | Closest Connected Unsolved Node | Total distance involved | $n$th nearest node | Min. distance | Last Connection |
|---|---|---|---|---|---|---|
| 5 | A | D | 2+7=9 | **D** | 8 | BD<br>ED |
| | B | D | 4+4=8 | | | |
| | E | D | 7+1=8 | | | |

[1]

# Shortest Path

## $N$-Nearest Neighbor Algorithm



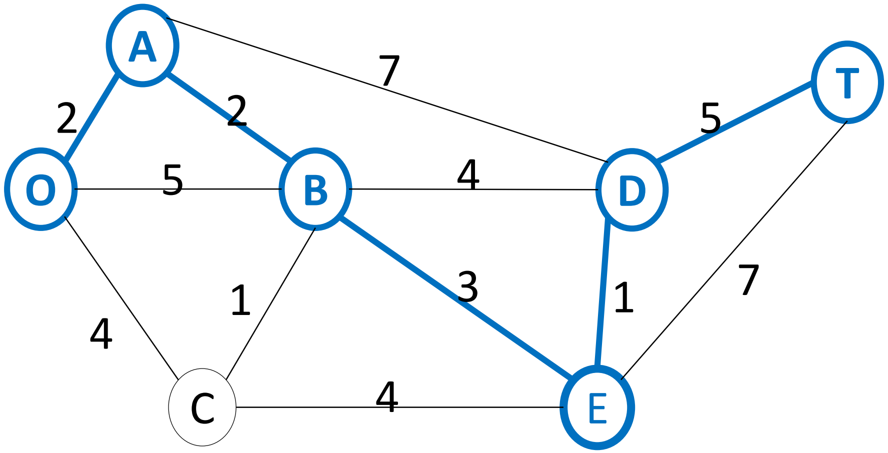


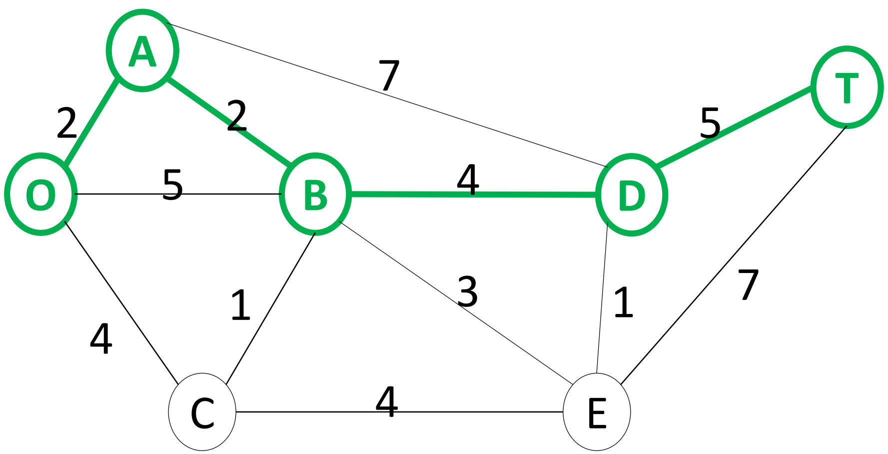

| $n$ | Solved nodes directly connected to unsolved nodes | Closest Connected Unsolved Node | Total distance involved | $n$th nearest node | Min. distance | Last Connection |
|---|---|---|---|---|---|---|
| 6 | D | T | 8+5=13 | **T** | 13 | DT |
| | E | T | 7+7=14 | | | |

# Shortest Path

## $N$-Nearest Neighbor Algorithm






Based on the table, there are two alternatives:

O -> A -> B -> E -> D -> T

O -> A -> B -> D -> T

[1]

# Minimum Spanning Tree

2. Telephone lines must be installed under the roads to establish telephone communication among all the stations. The installation is expensive and disruptive for natural environment ⇒ lines will ne installed under just enough roads to provide some connection between each pair of stations. Where the lines should be laid to accomplish this with a minimum total number of miles of line installed. MINIMUM SPANNING TREE

Shortest path vs. minimum spanning tree

- Shortest path: the shortest path between the origin and destination
- Minimum spanning tree: there must be a path between each pair of nodes

[1]

# Minimum Spanning Tree

## Definition



- The goal is to minimize the total length of the links connecting each pair of nodes, i.e. to find out the spanning tree with the minimum total length of the links

- For network with $n$ nodes we need $(n-1)$ nodes to connect each pair of nodes

- Applications:
  - Telecommunication networks design
  - Design of a network of wiring on electrical equipment
  - Design of a network pipelines to connect a number of locations
  - Etc.

[1]

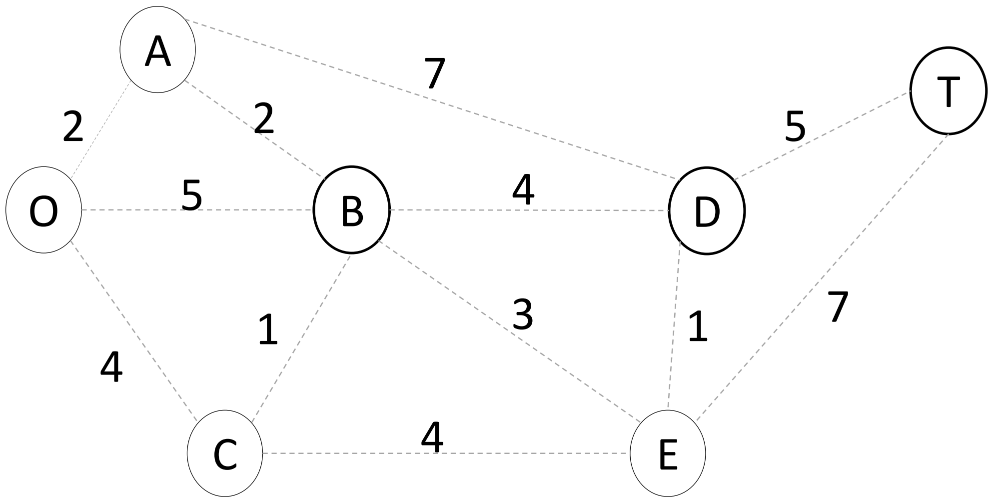
# Minimum Spanning Tree
## Algorithm



1. Begin with any node in the graph.

2. Identify the closest unconnected node to the selected (connected) nodes.

3. Thre resulting network is the minimum spanning tree

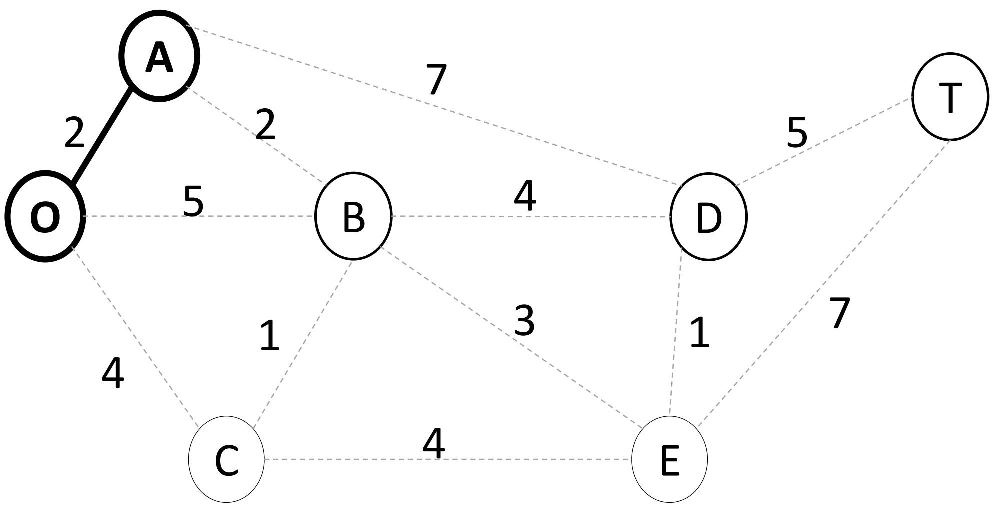



[1]

# Minimum Spanning Tree

## Algorithm

1. We can start at the node O (this is selected arbitrarily), the closest unconnected node to the node O is A

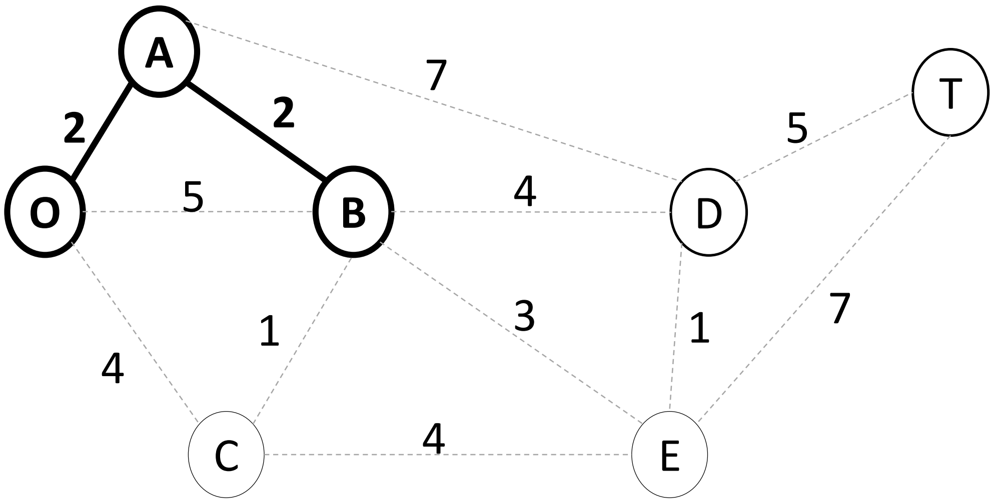



[1]

# Minimum Spanning Tree

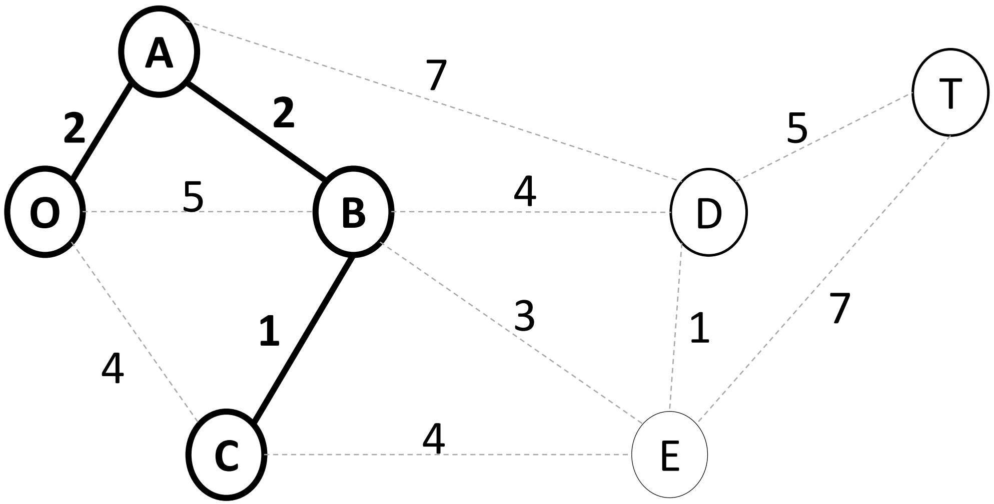
## Algorithm

2. We have two possibilities – node O and node A, the closest node to node A is the node B.




[1]

3. Now, we have three possibilities – nodes O, A, and B. The closest unconnected node to nodes O, A, or B is the node C.

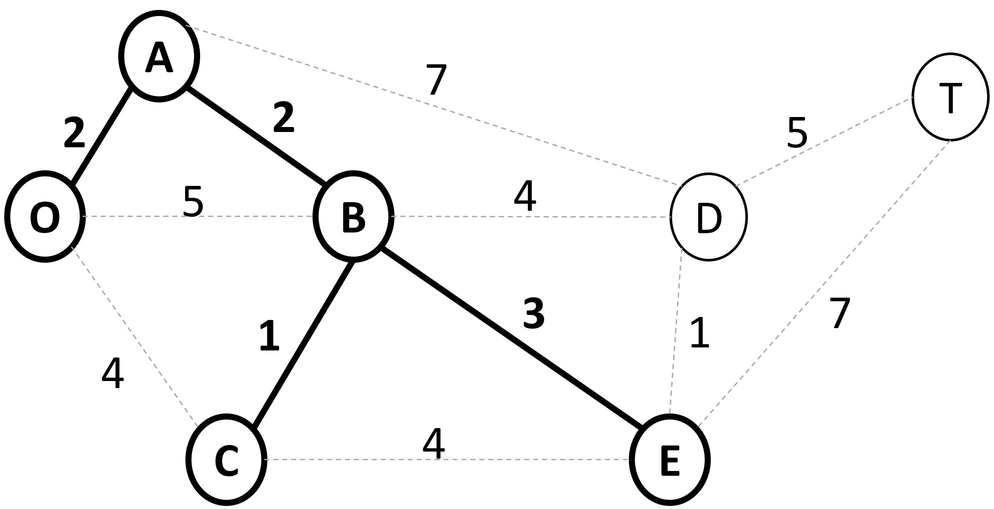



[1]

# Minimum Spanning Tree
## Algorithm



4. Now, we have four possibilities – nodes O, A, B, and C. The closest unconnected node to nodes O, A, B, or C is the node E (closest to the node B).



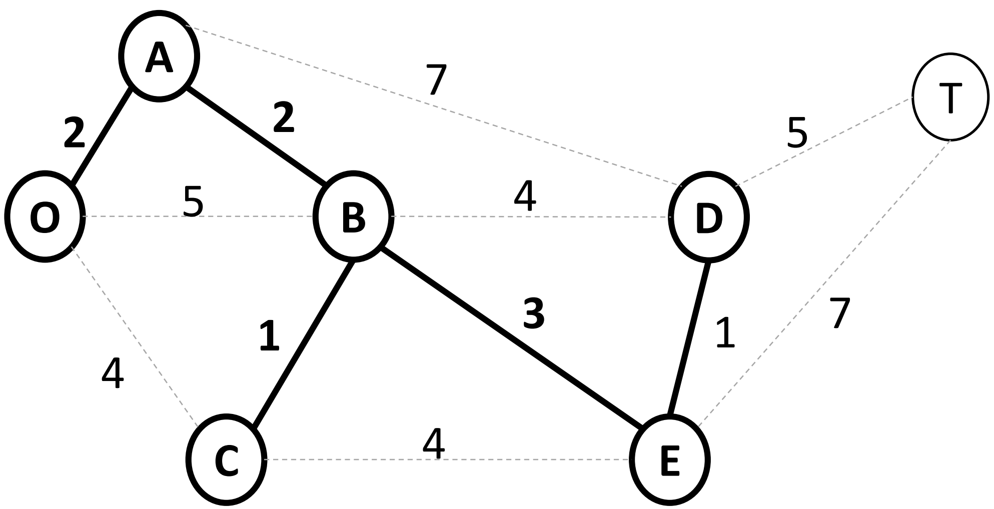

[1]

# Minimum Spanning Tree

## Algorithm



5. Now, we have five possibilities – nodes O, A, B, C, and E. The closest unconnected node to nodes O, A, B, C, or E is the node D (closest to the node E).



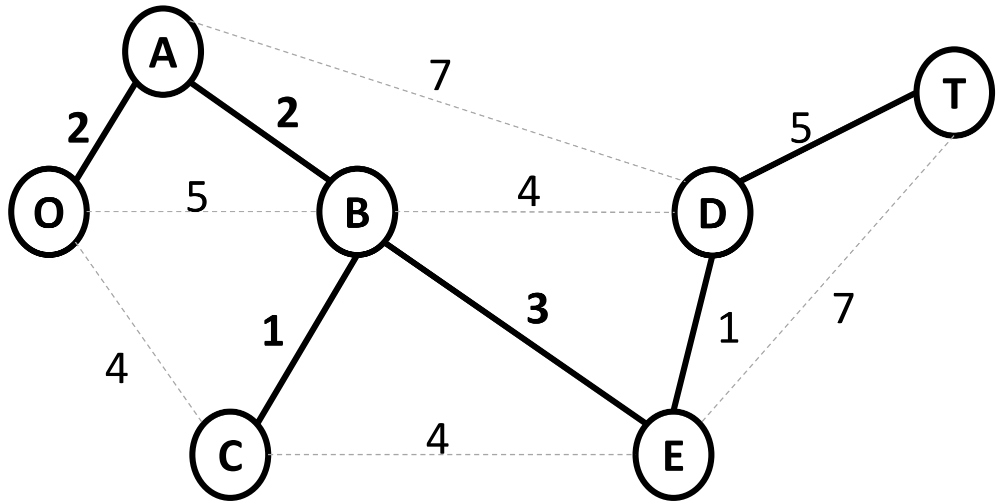

[1]

# Minimum Spanning Tree

## Algorithm



6. Now, we have six possibilities – nodes O, A, B, C, E, and D. The closest unconnected node to nodes O, A, B, C, E, or D is the node T (closest to the node D).




[1]

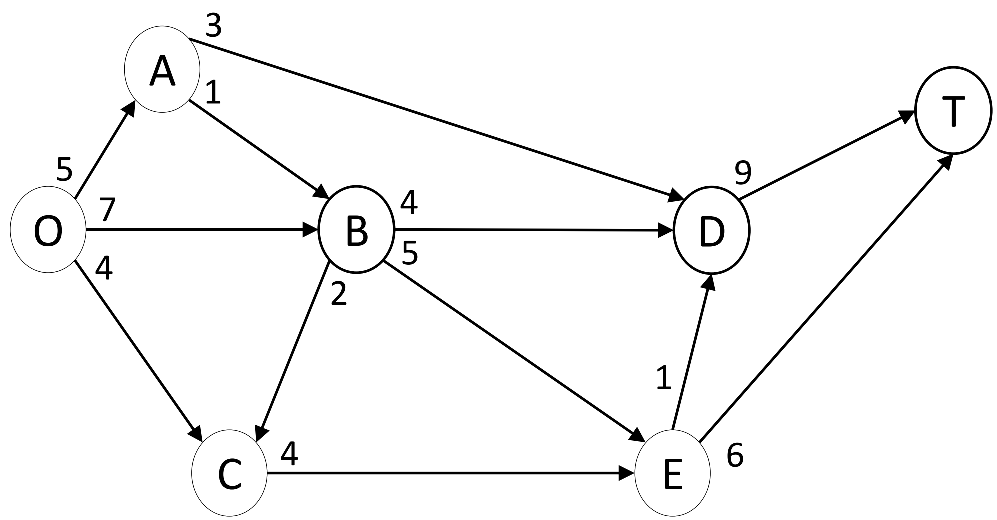
# Maximum Flow Problem

## Description



3. Strict ration has been placed on the number of trips that can be made on each of the roads per day (because of ecology). How to various trips to maximize the number of trips that can be made per day without violating the limits on any individual road? MAXIMUM FLOW PROBLEM

The undirected graph has been changed to the directed one. At each node, we can see the max. capacity of that road (link) → maximum number of trips using this way




[1]

# Maximum Flow Problem

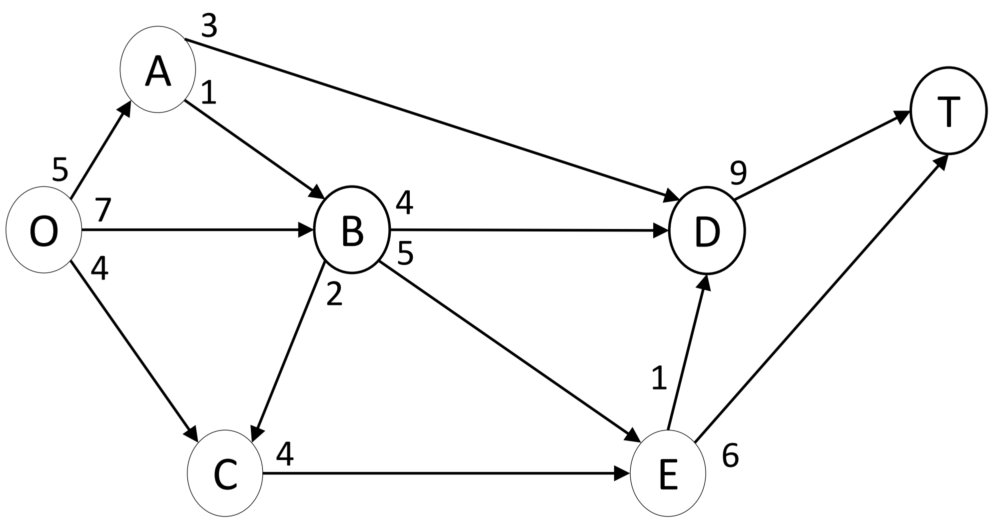
## Characteristics



1. All flows through a directed and connected network originates at one node – **source** and end in another node – **sink** . Other nodes – **transhipment nodes**
2. We always respect the arrowhead – this is the direction of the flow
3. The goal: To maximize the total amount  of flow fro  the source to the sink.

Applications:

- Maximize the flow of the oil in the pipelines
- Maximize the flow of vehicles throught transportation network.
- Etc.

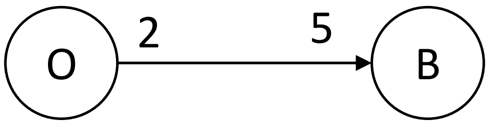



[1]

# Maximum Flow Problem
## Algorithm



- Algorithm based on the residual network and augmenting path
- When some flows have been assigned to the arsc, the residual network shows the remaining arc capacities – residual capacities.
- For example we choose the flow through the arc O -> B with the arc capacity 7. The assigned flows include a flow of 5 through this arc leaving the residual capacity of 2 (7-5=2)




[1]

# Maximum Flow Problem
## Algorithm



- Before beginning, all directed arec are changed to the undirected links. The arc capacity in the original direction remains the same! The arc capacity in the opposite direction is zero.

- Always when some amount of flow is assigned to an arc, this amount is subtracted from the residual capacity in the same direction and added to the residual capacity in the oppositr direction.

- Augmenting path = directed path from the source to the sink in the residual network such that every arc on this path has strictly positive residual capacity.

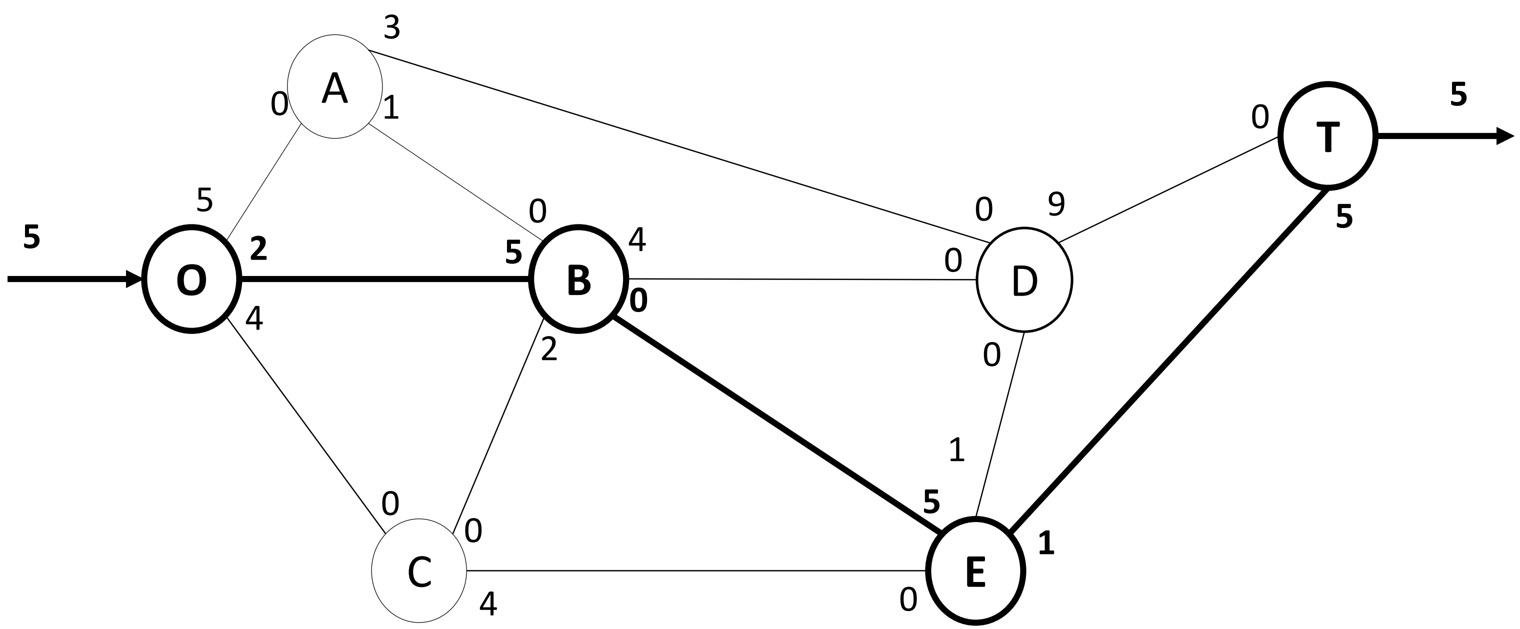
- Augmenting path algorithm repeatedly selects some augmenting path and adds a flow equal to its residual capacity to that path in the original network (until there are no more augmenting paths).

[1]

# Maximum Flow Problem

## Step 1

- One of the several augmenting paths is O -> B -> E -> T. Its residual capacity is $\min\{7, 5, 6\} = 5$

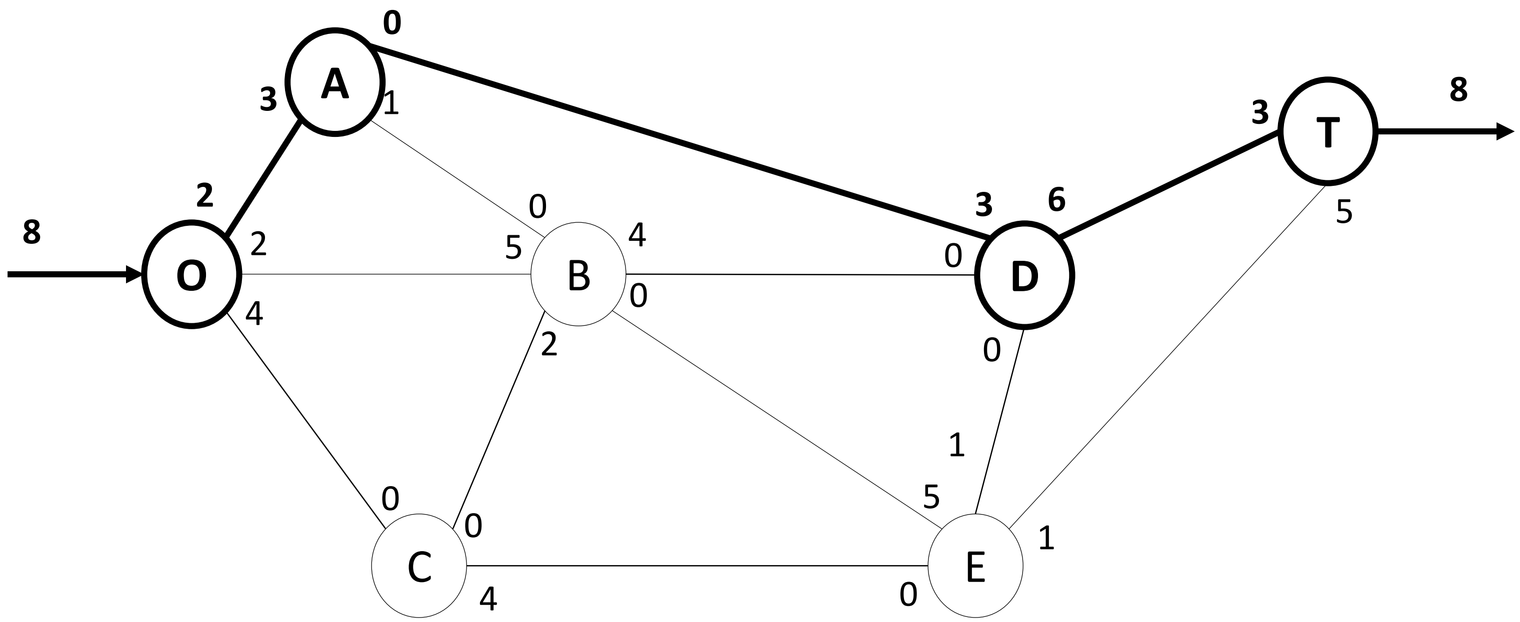



[1]

# Maximum Flow Problem

## Step 2

- The next of the several augmenting paths is O -> A -> D -> T. Its residual capacity is $\min\{5, 3, 9\} = 3$



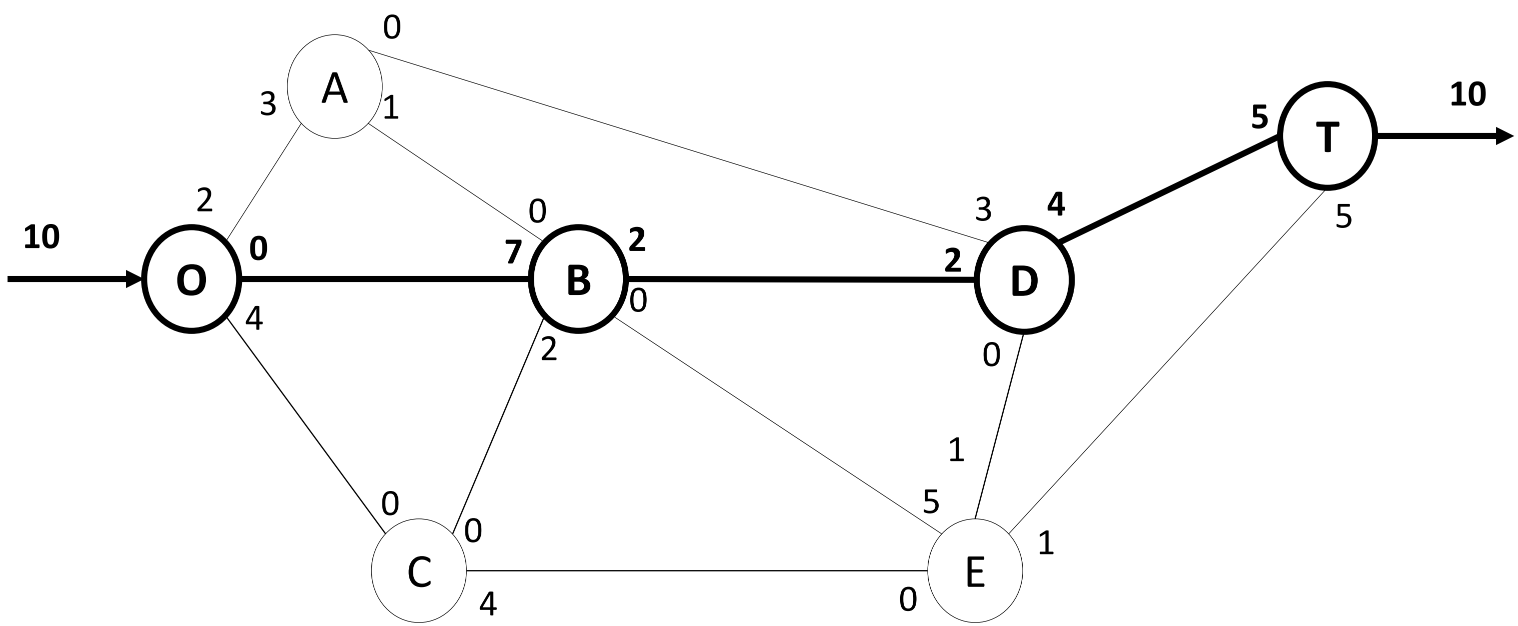

[1]

# Maximum Flow Problem

## Step 3



- The next of the several augmenting paths is O -> B -> D -> T. Its residual capacity is $\min\{2, 4, 6\} = 2$



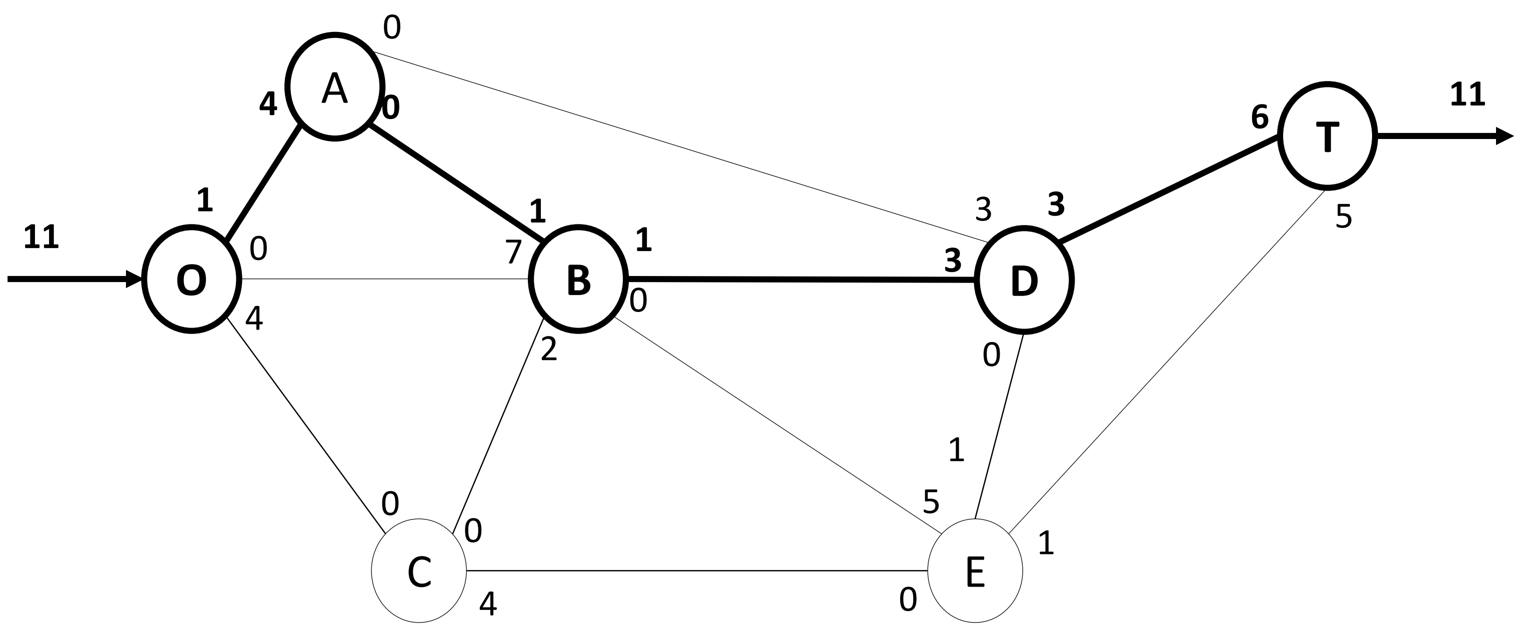

[1]

# Maximum Flow Problem

## Step 4



- The next of the several augmenting paths is O -> A -> B -> D -> T. Its residual capacity is $\min\{2, 1, 2, 4\} = 1$



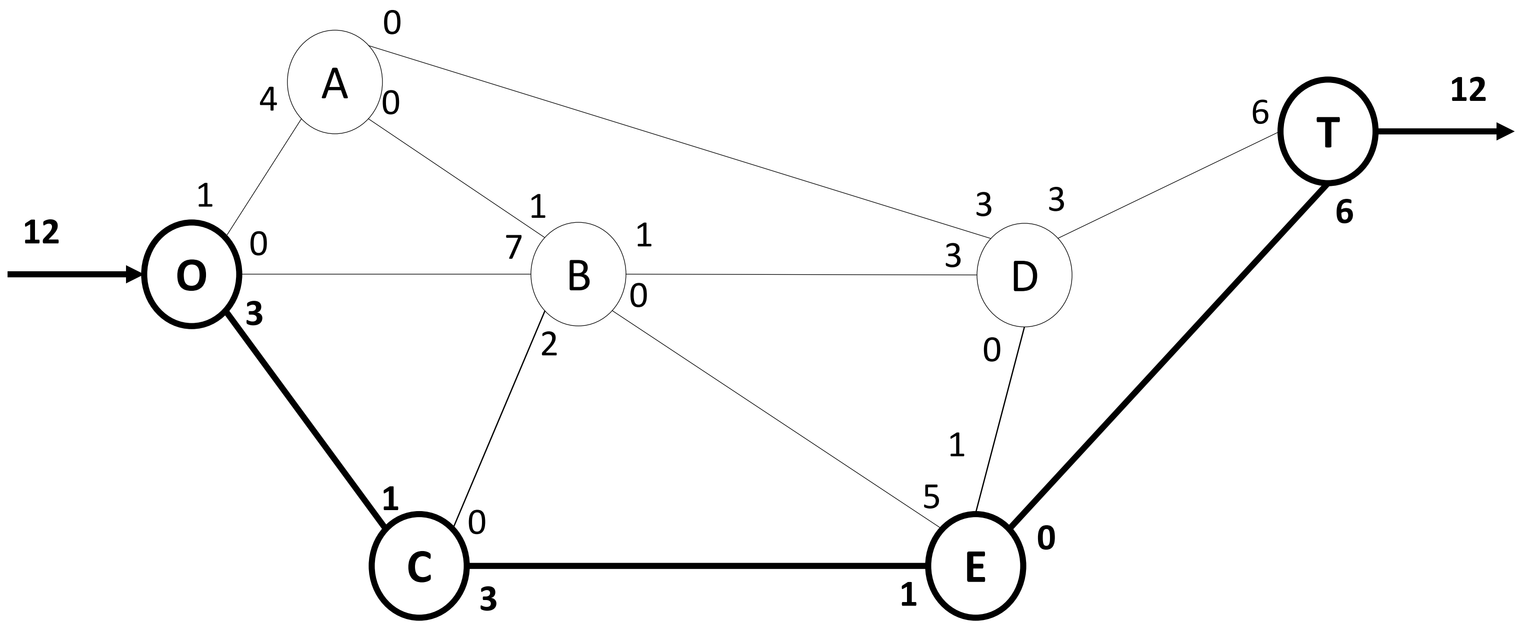

[1]

# Maximum Flow Problem

## Step 5



- The next of the several augmenting paths is O -> C -> E -> T. Its residual capacity is $\min\{4, 4, 1\} = 1$



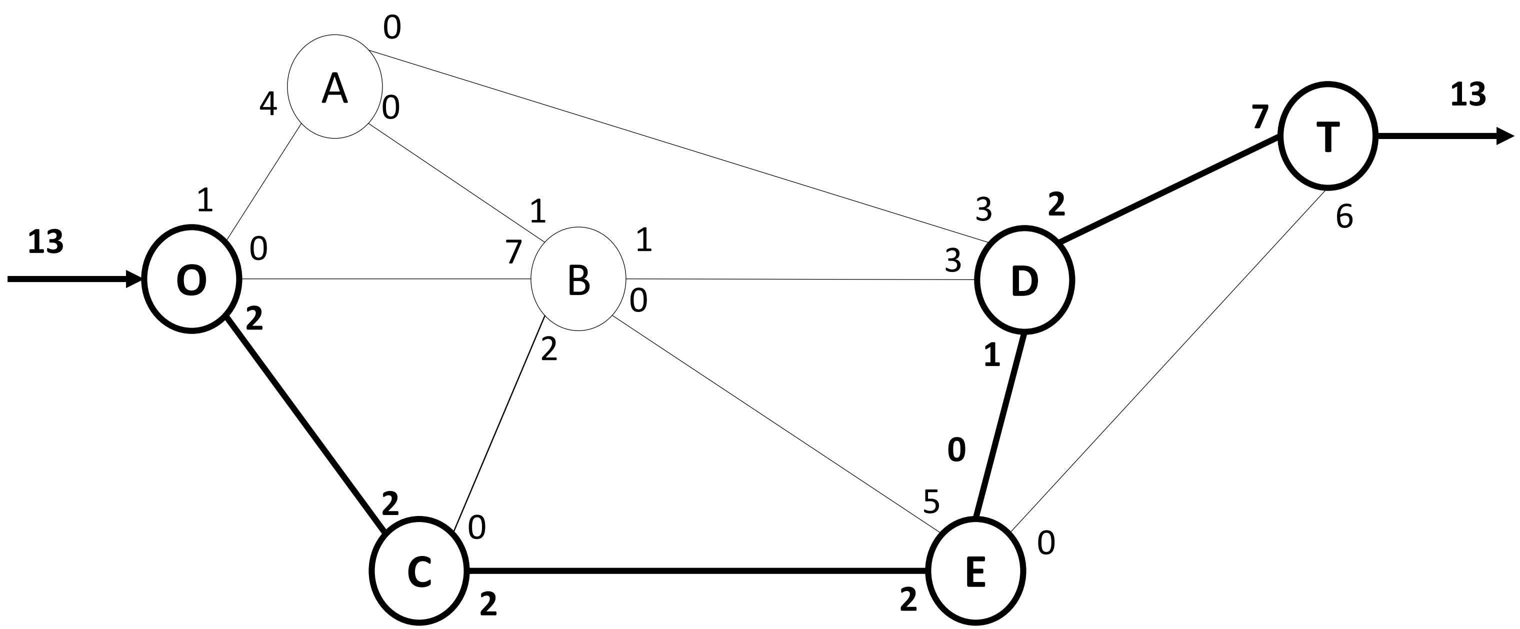

[1]

# Maximum Flow Problem

## Step 6



- The next of the several augmenting paths is O -> C -> E -> D-> T. Its residual capacity is $\min\{3, 3, 1, 3\} = 1$



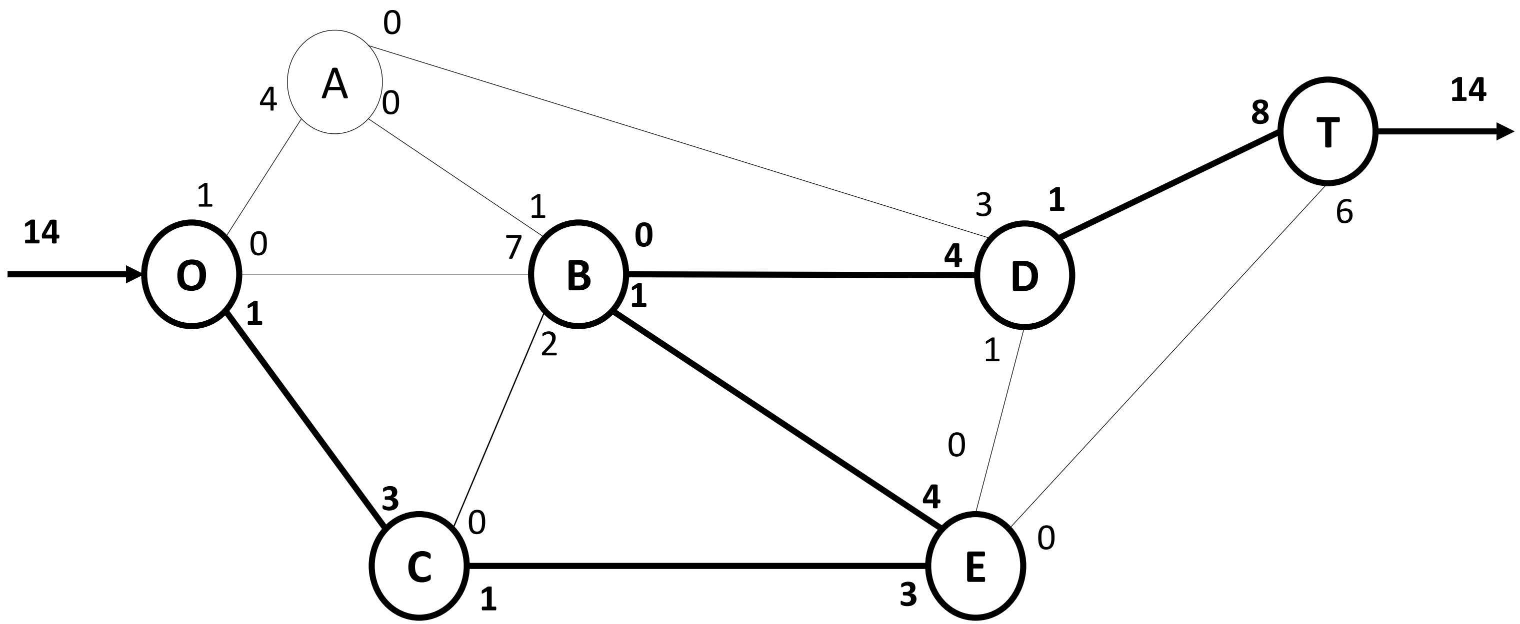

[1]

# Maximum Flow Problem

## Step 7



- The next of the several augmenting paths is O -> C -> E -> B -> D -> T. Its residual capacity is $\min\{2, 2,5,1, 2\} = 1$



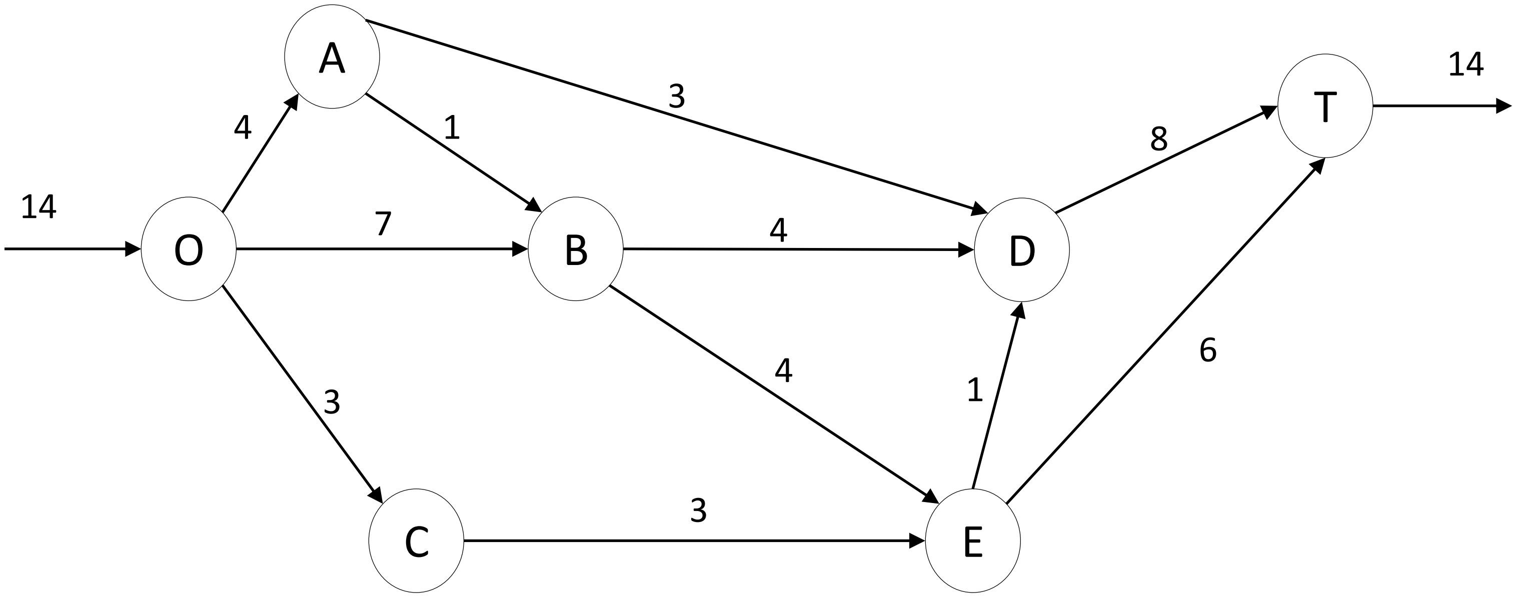

[1]

# Maximum Flow Problem

## Finish



- There is no other augmenting path, therefore the found pattern of the flow is maximal (optimal)




- The arc C -> B does not exists in the original graph, therefore it does not exist in the result graph

[1]

# Maximum Flow Problem

## Applications



- Maximize the flow through company‘s distribution network from its vendors to its factory
- Maximize the flow of oil through a system of pipelines
- Maximize the flow of vehocles through transportation network
- Maximize the flow of water through a systém of aqueducts
- Etc.

[1]

# Minimum Cost Flow Problem
## Description



- It takes into consideration:
  - Flow through network with the limited arc capacities (like maximum flow problem)
  - Cost (or distance) for flow through an arc (like shortest-path problem)
  - Multiple sources (supply nodes) and multiple desctinations for the flow (like transportation or assignment problem)

It can be formulated as a linear programming problem and it can be solved very effectively using the **network simplex method**

[1]

# Minimum Cost Flow Problem
## Description



- Description of the problem:
  - The network is a directed and connected network
  - At least one node is a supply node
  - At least one node is a demand node
  - All remaining nodes are transshipment nodes
  - Flow through an arc is allowed only in the direction indicated by the arrowhead, where the maximum amount of flow is given by the capacity of that arc.
  - The network has enough arcs with sufficient capacity to enable all the flow, where the cost per unit flow is known.
  - The objective is to minimize the total cost of sending the available supply through the network to satisfy the given demand. We can also maximize the total profit

[1]

# Minimum Cost Flow Problem

## Model Formulation

$x_{ij}$ = flow through arc $i \rightarrow j$

This information includes:

- $c\_ij$ … cost per unit flow through arc $i \rightarrow j$
- $u_{ij}$ … arc capacity for arc $i \rightarrow j$
- $b_i$ … net flow generated at node $i$

The value of $b_i$ depends on the characteristic of node $i$:

- $b_i > 0$ … if node $i$ is a supply node
- $b_i < 0$ … if node $i$ is a demand node
- $b_i = 0$ … if node $i$ is a transshipment node

The objective is to minimize the total cost of sending the available supply through the network to satisfy the given demand

[1]

# Minimum Cost Flow Problem

## Model Formulation

Minimize $Z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$

Subject to

Total flow out of node $i$

$$\sum_{j=1}^{n} x_{ij} - \sum_{j=1}^{n} x_{ji} = b_i \qquad \text{for each node } i$$

Total flow out into node $i$

and

$$0 \leq x_{ij} \leq u_{ij} \qquad \text{for each arc } i \rightarrow j$$

**Feasible solution property:** A necessary condition for a minimum cost flow problem to have any feasible solutions is that

$$\sum_{i=1}^{n} b_i = 0$$

The total flow generated at the supply nodes = total flow absorbed at the demand nodes

[1]

# Minimum Cost Flow Problem

## Example



This example is mentioned in the lecture Linear programming

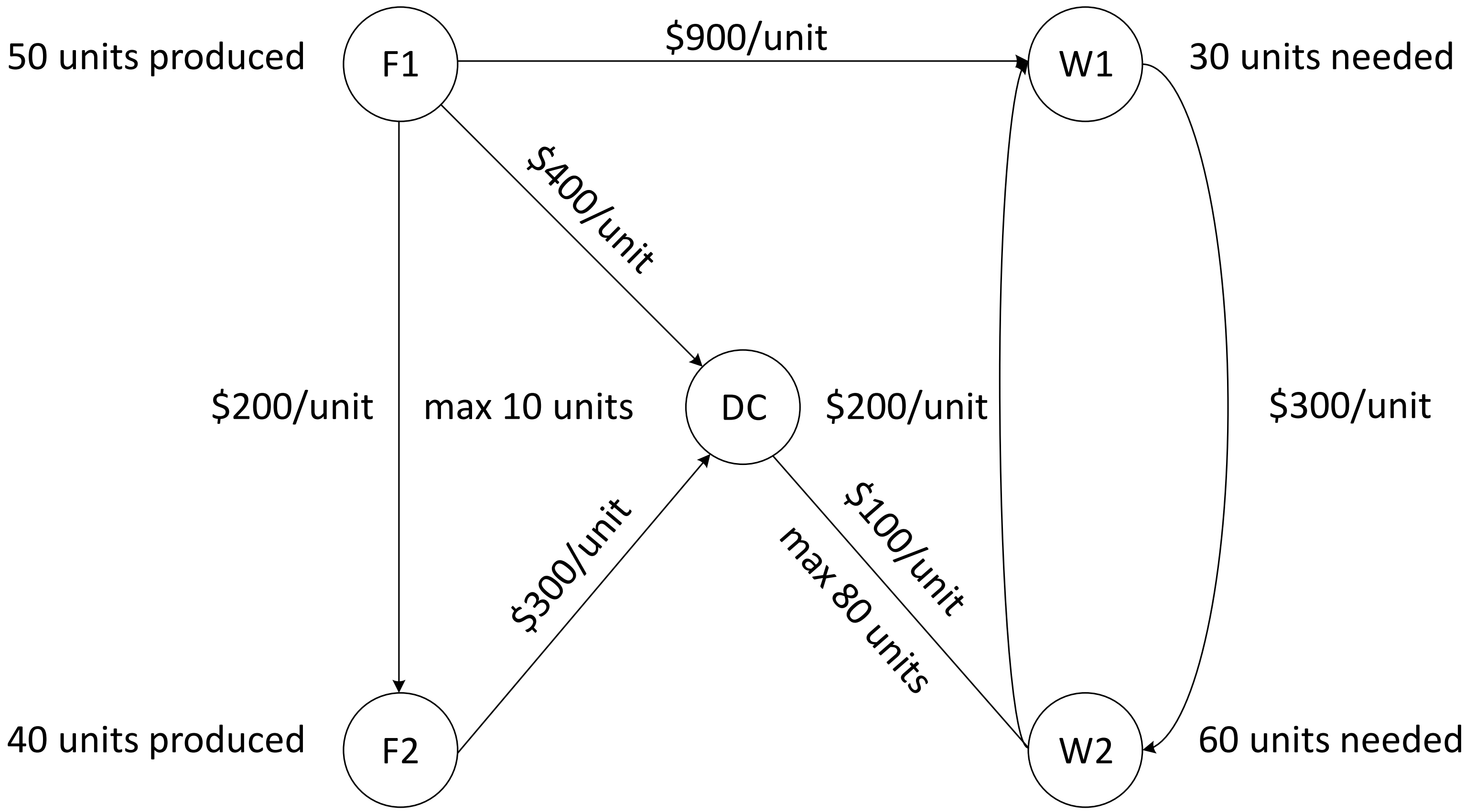
- Distribution of the product at 2 different factories
- The product must be shipped to 2 warehouses, either factory can supply either warehouse
- Possible distribution trajectories:
  - Factory 1 → Distribution center → Warehouse 2
  - Factory 1 → Factory 2 → Distribution center → Warehouse 2
  - Factory 1 → Warehouse 1 → Warehouse 2
  - Factory 2 → Distribution center → Warehouse 2
  - Factory 2 → Distribution center → Warehouse 2 → Warehouse 1
- The objective is to minimize the total shipping cost.

[1]

# Minimum Cost Flow Problem

## Example






[1]

# Minimum Cost Flow Problem
## Example



- 7 shipping lanes → 7 decision variables:
  - $x_{F1-F2}, x_{F1-DC}, x_{F1-W1}, x_{F2-DC}, x_{DC-W2}, x_{W1-W2}, x_{W2-W1}$
- Two upper-bound constraints:
  - $x_{F1-F2} \leq 10$
  - $x_{DC-W2} \leq 80$

  limited shipping capacities
- Flow constraint for each location:
  - $Amount\ ship\ out\ - amount\ ship\ in = required\ amount$
  - For $F1$ it is 50
  - For $F2$ it is 40
  - For $W1$ it is -30
  - For $W2$ it is -60
  - For $DC$: the total amount shipped from $F1$ and $F2 =$ the total amount shipped from $DC$ to $W1$ and $W2$

[1]

# Minimum Cost Flow Problem
## Example



- Minimize

$$Z = 2x_{F1-F2} + 4\,x_{F1-DC} + 9x_{F1-W1} + 3x_{F2-DC} + x_{DC-W2} + 3x_{W1-W2} + 2x_{W1-W1}$$

subject to

$$\begin{aligned}
x_{F1-F2} + x_{F1-DC} + x_{F1-W1} &= 50 \text{ (F1)}\\
-x_{F1-F2} + x_{F2-DC} &= 40 \text{ (F2)}\\
-x_{F1-DC} - x_{F2-DC} + x_{DC-W2} &= 0 \text{ (DC)}\\
x_{F1-W1} + x_{W1-W2} - x_{W2-W1} &= \text{-30 (W1)}\\
-x_{DC-W2} - x_{W1-W2} + x_{W2-W1} &= \text{-60 (W2)}
\end{aligned}$$

- Upper-bound constraints:
  - $x_{F1-F2} \leq 10$ $\quad x_{DC-W2} \leq 80$

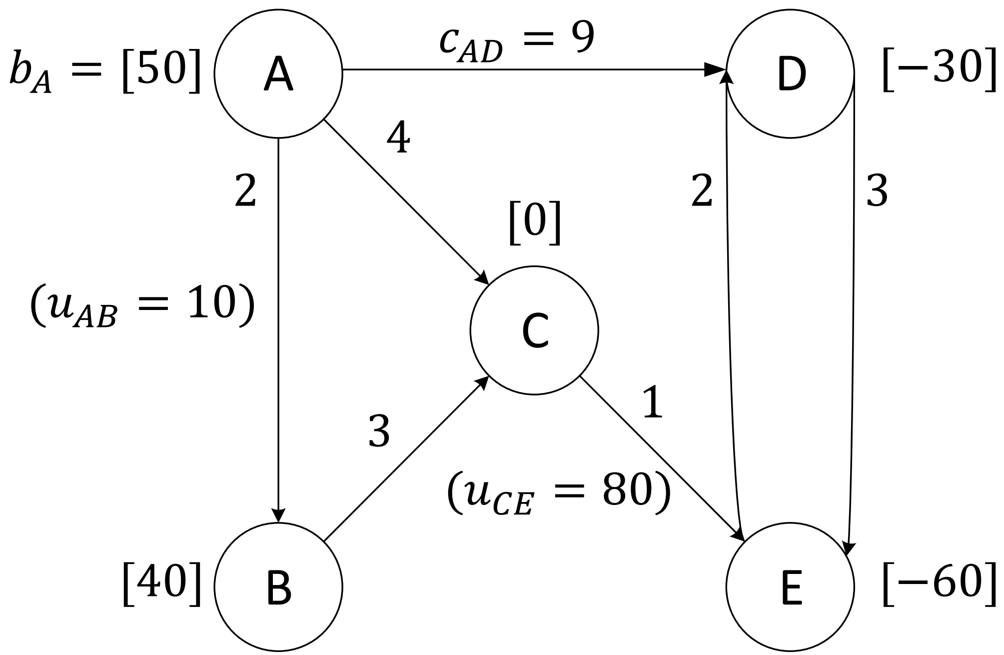
- Don‘t forget the nonnegativity constraints!

[1]

# Minimum Cost Flow Problem
## Example



- The network can be also drawn as follows:



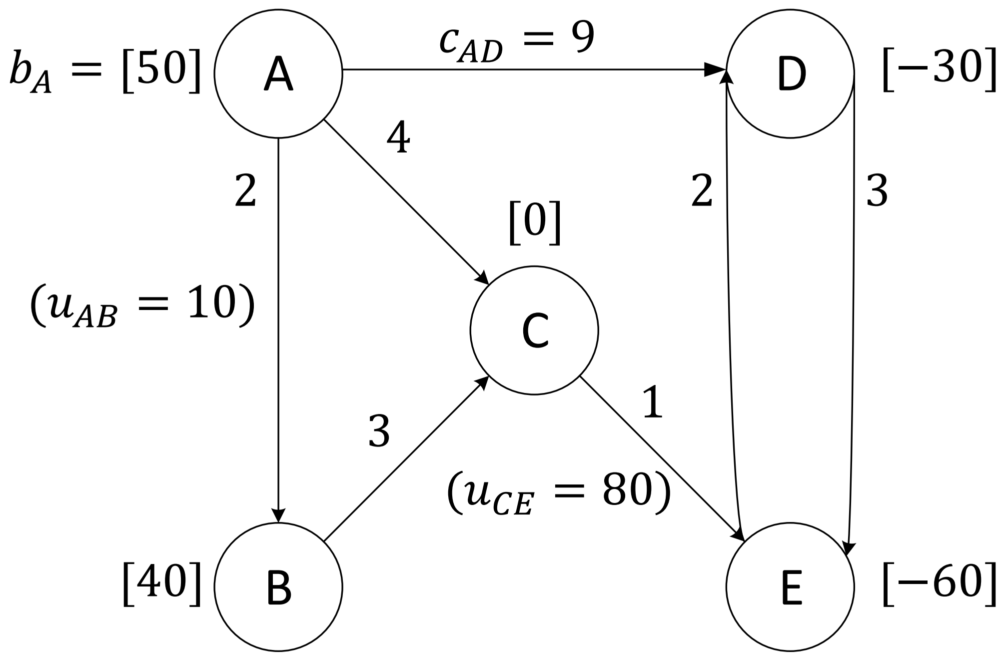

[1]

# Minimum Cost Flow Problem

## Incorporation the Upper Bound Technique



- The network can be also drawn as follows:




[1]

# Minimum cost flow problem

## Applications

- The typical application is to the operation of a company‘s distribution network.

- Example: International Paper Company – the world‘s largest manufacturer of pulp, paper, and paper products. The supply nodes in the distribution network are the woodlands in their various locations. However, before the company‘s good can reach the demand nodes (custumers), the wood must pass through a long sequence of transshipment nodes.

  Woodlands → woodyards → sawmills → paper mills → converting plants → warehouses → customers

[1]

# Literature




[1] Hillier and Lieberman: Introduction to Operations Research, 8th edition, 2005