Unit 5. Network Analysis

Jordi Villà i Freixa

Universitat de Vic - Universitat Central de Catalunya
Study Abroad. Operations Research

jordi.villa@uvic.cat
https://mon.uvic.cat/cbbl

last updated: November 12, 2025

November 12-19th, 2025

Preliminary

This course is strongly based on the monography on Operations Research by Carter, Price and Rabadi [1], and in material obtained from different sources (quoted when needed through the slides).

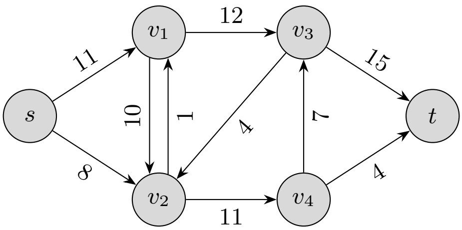
Introduction: Summary

Introduction: Learning outcomes

- Getting familiar with the use of network analysis in OR
- Understanding network flow in graphs
- Understanding minimum cost network flow problems
- Applying network connectivity to LP problems
- Solving shortest path problems
- Understanding and applying dynamic programming

Introduction: The concept

- Network analysis provides a framework for the study of a special class of linear programming problems that can be modeled as network programs.
- Some of these problems correspond to a physical or geographical network of elements within a system, while others correspond more abstractly to a graphical approach to planning or grouping or arranging the elements of a system.

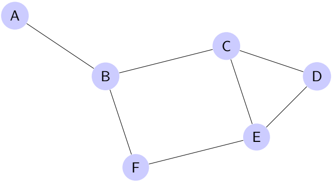


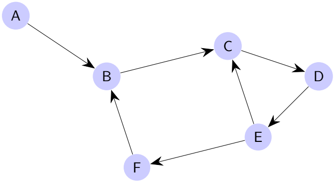
Introduction: Examples

- Systems of highways, railroads, shipping lanes, or aviation patterns, where some supply of a commodity is transported or distributed to satisfy a demand;
- pipeline systems or utility grids can be viewed as fluid flow or power flow networks;
- computer communication networks represent the flow of information;
- an economic system may represent the flow of wealth;
- routing a vehicle or a commodity between certain specified points in the network;
- assigning jobs to machines, or matching workers with jobs for maximum efficiency;
- project planning and project management, where various activities must be scheduled in order to minimize the duration of a project or to meet specified completion dates, subject to the availability of resources;
- and many, many others.

Definitions: Summary

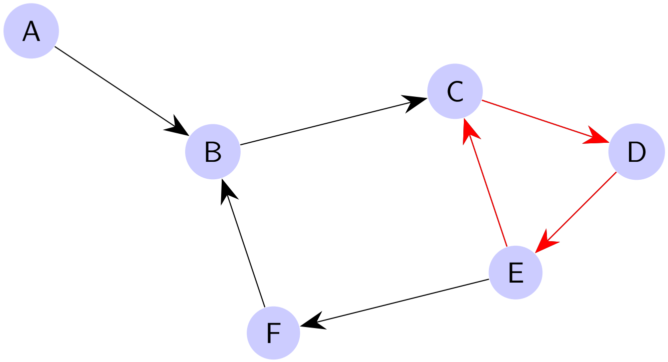
Graphs and networks: definitions





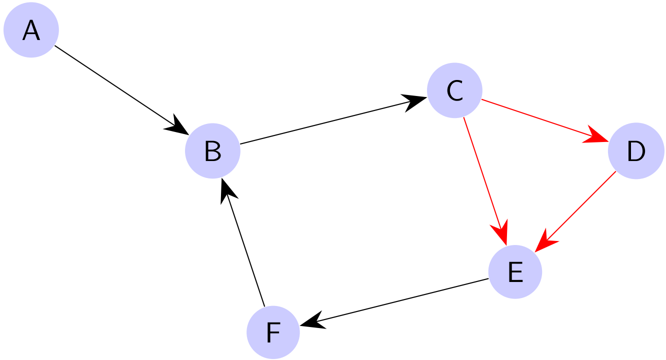
- A graph consists of a set of nodes V (vertices, points, or junctions) and a set of connections called arcs A (edges, links, or branches).
- Each connection is associated with a pair of nodes and is usually drawn as a line joining two points. The graph can be defined as *directed* or *undirected*.
- The **degree of a node** is the number of arcs attached to it. An isolated node is of degree zero.
- In a directed graph, or digraph, the arc is often designated by the ordered pair (A, B). In digraphs, the direction of the flow matters.

Paths



$$A, (A, B), B, (B, C), \underbrace{C, (C, D), D, (D, E), E, (E, C), C}_{\text{cyclic path and chain}}$$

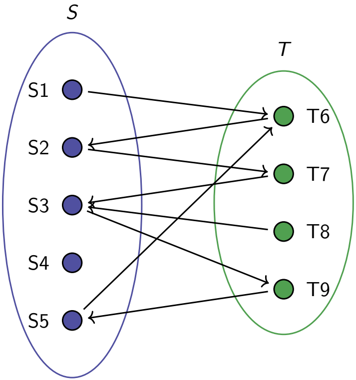
FACULTAT DE CIÈNCIES, TECNOLOGIA I ENGINYERIES UVIC | UVIC-UCC

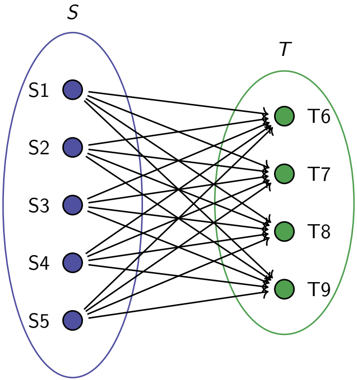


$$A, (A, B), B, (B, C), \underbrace{C, (C, D), D, (D, E), E, (C, E), C}_{\text{cyclic path, non cyclic chain}}$$

- If all the arcs in a path are forward arcs, the path is called **directed chain** or simply **chain**.
- **path** and **chain** are synonimous if the graph is undirected.
- In the second example above we saw a cyclic path but not a cyclic chain, as it included the backward arc (C, E).
- A **connected graph** has at least one path connecting every pair of nodes.
- In a **bipartite graph** the nodes can be partitioned into two subsets S and T, such that each node is in exactly one of the subsets, and every arc in the graph connects a node in set S with a node in set T.
- Such a graph is **complete bipartite** if each node in S is connected to every node in T.

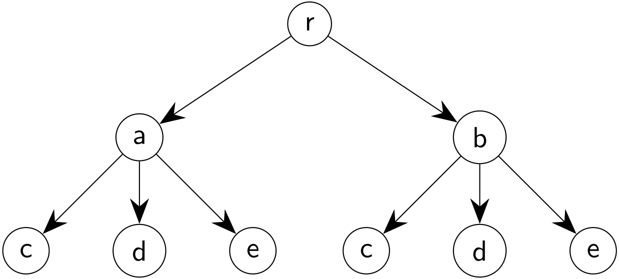
Bipartite vs complete bipartite graphs





FACULTAT DE CIÈNCIES, TECNOLOGIA I ENGINYERIES
UVIC | UVIC-UCC

- A **tree** is a directed connected graph in which each node has at most on predecessor, and one node (the root node) has none. In an undirected graph, we have a tree if the graph is connected and contains no cycles.



- A **network** is a directed connected graph that is used to model/represent a system/process. The arcs are typically assigned weights representing cost, value or capacity corresponding to each link.

- Nodes in networks can be designated as **sources**, **sinks** or **transshipments**. A **cut set** is any set of arcs which, if removed from the network, would disconnet the sources(s) from the sink(s).
- **Flow** can be thought of as the total amount of an entity that originates at the source, makes it through the different nodes and reaches the sink.

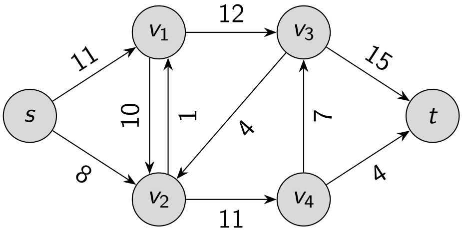
Maximum flow: Summary

FACULTAT DE CIÈNCIES, TECNOLOGIA I ENGINYERIES
UVIC | UVIC-UCC

Maximum flow I

Maximum flow in networks

Determine the maximum possible flow that can be routed through the various network links, from source (s) to sink (t), without violating the capacity constraints.
Important! the commodity is only generated at the source and consumed at the sink.



Maximum flow II

The **maximum flow problem** can be stated as a LP formulation.

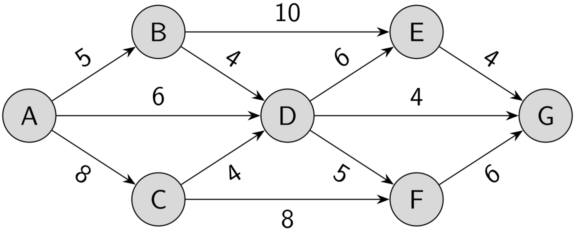
$$
\begin{array}{lrll}
\text{maximize} & z = f & & \\
\text{subject to} & \sum_{i=2}^{n} x_{1i} = f & & \\
 & \sum_{i=1}^{n-1} x_{in} = f & & \\
 & \sum_{i=1}^{n} x_{ij} = \sum_{k=1}^{n} x_{jk}, & \text{for} & j = 2, 3, \ldots, n-1 \\
 & x_{ij} \leq u_{ij}, & \text{for all} & i, j = 1, 2, \ldots, n
\end{array}
$$

Maximum flow algorithm

All network problems here can be solved using the Simplex method, but the network structure can help us solving it more efficiently. In the **Ford-Fulkerson labelling algorithm**:

1. Use a labelling procedure to look for a flow augmenting path. If none can be found, stop; the current flow is optimal;
2. Increase the current flow as much as possible in the flow augmenting path, until reaching capacity of some arc. Come back to step 1.

Exercise 1 Find the maximum flow in this network using the Ford-Fulkerson labelling algorithm:



We aim to find the **maximum flow** from A (source) to G (sink) in the given network.

Step-by-Step Ford-Fulkerson Algorithm The capacities of each edge are shown in the graph. We start with an initial flow of 0 and iteratively find augmenting paths until no more can be found.

An augmenting path is $A \to C \to F \to G$ with a bottleneck capacity of:

$$\min(8, 8, 6) = 6.$$

We augment this flow along the path and update the residual capacities:

- $A \to C : 8 - 6 = 2$ (residual capacity).
- $C \to F : 8 - 6 = 2.$
- $F \to G : 6 - 6 = 0.$

The current total flow is now:

$$\text{Total flow} = 6.$$

Step 2: Augmenting Path 2 Next, we find the augmenting path $A \to B \to E \to G$ with a bottleneck capacity of:

$$\min(5, 10, 4) = 4.$$

Augmenting this flow and updating residual capacities:

- $A \to B : 5 - 4 = 1.$
- $B \to E : 10 - 4 = 6.$
- $E \to G : 4 - 4 = 0.$

FACULTAT DE CIÈNCIES, TECNOLOGIA I ENGINYERIES

UVIC | UVIC-UCC

The current total flow is now:

$$\text{Total flow} = 6 + 4 = 10.$$

Step 3: Augmenting Path 3 We find another augmenting path $A \to C \to D \to E \to G$ with a bottleneck capacity of:

$$\min(2, 4, 6, 4) = 2.$$

Augmenting this flow and updating residual capacities:

- $A \to C : 2 - 2 = 0.$
- $C \to D : 4 - 2 = 2.$
- $D \to E : 6 - 2 = 4.$
- $E \to G : 4 - 2 = 2.$

The current total flow is now:

$$\text{Total flow} = 10 + 2 = 12.$$

Step 4: No More Augmenting Paths At this point, no more augmenting paths exist in the residual network. The algorithm terminates.
Maximum Flow The maximum flow from A to G is:

$$\boxed{12}.$$

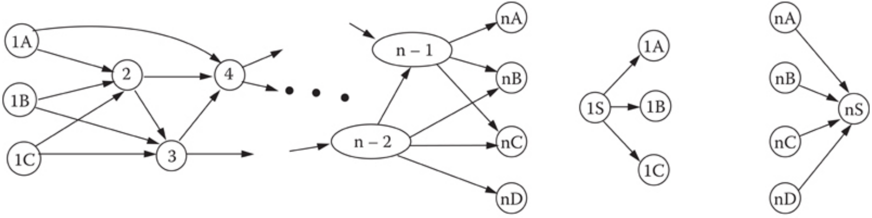
Verification: Flow Conservation and Saturation We verify that:

- **Flow conservation** holds at all intermediate nodes (B, C, D, E, F).
- The flows on edges do not exceed their capacities.

- In any network, there is always a bottleneck that in some sense impedes the flow through the network.
- The total capacity of the bottleneck is an upper bound on the total flow in the network.
- Cut sets are, by definition, essential in order for there to be a flow from source to sink, since removal of the cut set links would render the sink unreachable from the source.
- The capacities on the links in any cut set potentially limit the total flow.
- The minimum cut (i.e., the cut set with minimum total capacity) is in fact the bottleneck that precisely determines the maximum possible flow in the network (Max-Flow Min-Cut Theorem): the capacity of the cut is precisely equal to the current flow and this flow is optimal. In other words, a saturated cut defines the maximum flow.

Multiple sinks and sources

We can generate a supersource or a supersink node with unlimited capacity and repeat the process of optimization as above:[1]



(a) (b) (c)

Minimum Cost Network Flow: Summary

FACULTAT DE CIÈNCIES, TECNOLOGIA I ENGINYERIES
UVIC | UVIC-UCC

General formulation of the MCF Problem I

Problem Statement:

- Given a directed graph $G = (N, A)$, where N is the set of nodes and A is the set of arcs.
- Each arc $(i, j) \in A$ has:
  - A cost c_{ij}: cost per unit flow.
  - A capacity u_{ij}: maximum flow allowed.
- Each node $i \in N$ has a supply/demand value b_i, where:

$$\begin{aligned} b_i &> 0 \quad \text{(supply)} \\ b_i &< 0 \quad \text{(demand)} \\ b_i &= 0 \quad \text{(transshipment node).} \end{aligned}$$

General formulation of the MCF Problem II

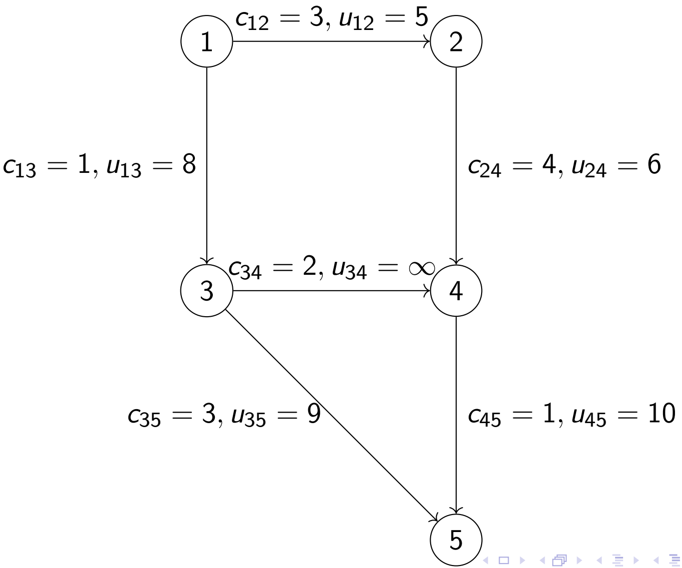
Objective: Minimize the total cost of flow:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij}$$

Subject to:

- Capacity constraints: $0 \leq x_{ij} \leq u_{ij}$ for all $(i,j) \in A$.
- Flow balance equations: $\sum_{j\in N} x_{ij} - \sum_{j\in N} x_{ji} = b_i$ for all $i \in N$.

Example


1
2
3
4
5
$c_{12} = 3, u_{12} = 5$
$c_{13} = 1, u_{13} = 8$
$c_{24} = 4, u_{24} = 6$
$c_{34} = 2, u_{34} = \infty$
$c_{35} = 3, u_{35} = 9$
$c_{45} = 1, u_{45} = 10$

Formulation

Formulation:

$$\min 3x_{12} + 1x_{13} + 4x_{24} + 2x_{34} + 3x_{35} + 1x_{45}$$

Subject to:

$$\begin{aligned}
x_{12} + x_{13} &= b_1 = 5 \quad \text{(supply at node 1)} \\
x_{12} - x_{24} &= b_2 = 0 \quad \text{(transshipment at node 2)} \\
x_{13} - x_{34} - x_{35} &= b_3 = 0 \quad \text{(transshipment at node 3)} \\
x_{24} + x_{34} - x_{45} &= b_4 = 0 \quad \text{(transshipment at node 4)} \\
x_{35} + x_{45} &= b_5 = -5 \quad \text{(demand at node 5)}
\end{aligned}$$

Capacity constraints:

$$0 \leq x_{ij} \leq u_{ij} \quad \forall (i,j) \in A$$

Node-Arc Incidence Matrix I

Node-Arc Matrix:

Node	$(1,2)$	$(1,3)$	$(2,4)$	$(3,4)$	$(3,5)$	$(4,5)$	RHS (b)
1	1	1	0	0	0	0	5
2	-1	0	1	0	0	0	0
3	0	-1	0	1	1	0	0
4	0	0	-1	-1	0	1	0
5	0	0	0	0	-1	-1	-5
Capacities (u)	5	8	6	7	9	10	–
Costs (c)	3	1	4	2	3	1	–

Node-Arc Incidence Matrix II

Flow Balance Equations:

$$A \cdot \mathbf{x} = \mathbf{b}, \quad 0 \leq \mathbf{x} \leq \mathbf{u}$$

where:

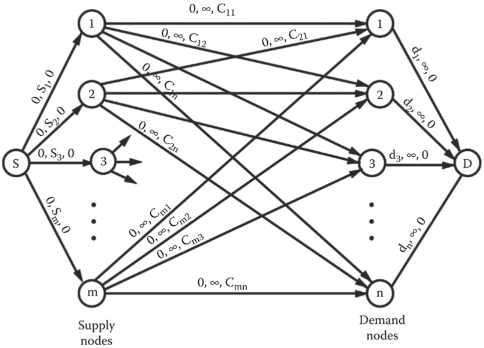
- $\mathbf{x} = [x_{12}, x_{13}, x_{24}, x_{34}, x_{35}, x_{45}]^T$ (flows).
- $\mathbf{b} = [5, 0, 0, 0, -5]^T$ (supplies/demands).

Transportation problem

- Useful when there are costs associated with the flow, given a link capacity.
- Let us assume that every node is a source (supply) and a sink (demand). Imagine a distributor with several warehouses and a group of costumers. Serving each customer from a given warehouse has an associated cost. **Bipartite graph**.
- For m supply nodes, each providing s_i suplly, and n demand nodes, each demanding d_j. Assuming that the total demand equals the total supply: $\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j$ we aim at satisfying the demand using the available supply minimizing cost routes.

FACULTAT DE CIÈNCIES, TECNOLOGIA I ENGINYERIES
UVIC | UVIC-UCC

$$
\begin{array}{llll}
\text{minimize} & z = \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} & & \\
 & \sum_{j=1}^{n} x_{ij} = s_i & \text{for} & i = 1,\ldots,m \\
\text{subject to} & \sum_{i=1}^{m} x_{ij} = d_j & \text{for} & j = 1,\ldots,n \\
 & x_{ij} \geq 0 & \text{for all} & i,j
\end{array}
$$



Exercise 2 Find the minimum cost in this trasportation problem:[1]

Sources	Sinks (Customers)					
(Warehouses)	1	2	3	4	5	Supply
1	28	7	16	2	30	20
2	18	8	14	4	20	20
3	10	12	13	5	28	25
Demand	12	14	12	18	9	65

NOTE: The Simplex method says that we should first find any basic feasible solution and then look for a simple pivot to improve the solution. repeat until the optimal solution is found.

Optimizing

1. Finding initial solution. Note that the network structure simplifies the solution by Simplex, as we do not need artificial variables nor, thus, a two-phase solution.
    - Northwest corner rule. Do not pay atention to cost. Always check $m + n - 1$ cells (independent variables $\equiv$ non basic variables) for a FS.
    - Minimum cost method. Start by minimum cost cell at every step.
    - Minimum "row" cost method. Start in the minimal cost cell in the first row.
    - Vogel's method (opportunity cost)
2. Transportation simplex

Transportation simplex

Once we have any feasible solution, we aim at finding the optimal one. Consider:

Minimum Row Cost Final Solution

Sources	Sinks (Customers)					
(Warehouses)	1	2	3	4	5	Supply
1	28	7 2	16	2 18	30	20
2	18 8	8 12	14	4	20	20
3	10 4	12	13 12	5	28 9	25
Demand	12	14	12	18	9	65

Transportation simplex

We can reduce the total cost by reducing the individual costs in every row i by u_i and in every column j by v_j:

$$c'_{ij} = c_{ij} - u_i - v_j$$

Check that, now:

- $\sum_i \sum_j x_{ij} c'_{ij} = 0$
- Check how some costs are now negative in non-basic cells.
- If we increase the number of units in those non-basic cells from 0 to some value, reducing at the same time the number of units in the basic cells, we can reduce the overall cost $\sum_i \sum_j x_{ij} c_{ij}$
- Reduced costs are now zero in all basic variables in the reduced system.

Transportation simplex

In practice:

1. We find first an initial feasible solution as explained above
2. Calculate the u_i and v_j, taking into account that $c_{ij} = u_i + v_j$, for all basic variables (used squares in the table). We start by assigning $u_1 = 0$.
3. We calculate the *improvement index* by $I_{ij} = c_{ij} - u_i - v_j$ for all non-used squares in the table.
4. If all I_{ij} are positive, the solution is already optimal and we are done.
5. If some $I_{ij} < 0$, then build a loop with such value in the corner and alternative $\pm$ signs in all vertex.
6. Use the above $\pm$ to increase the number of units in the position that had $I_{ij} < 0$
7. We return to step 2.

References

[1] Michael W. Carter, Camille C. Price, and Ghaith Rabadi. Operations Research, 2nd Edition. CRC Press.

[2] David Harel, with Yishai Feldman. Algorithmics: the spirit of computing, 3rd Edition. Addison-Wesley.

[3] Ronald L. Rardin. Optimization in Operations Research, 2nd Edition. Pearson.

[4] J. Hefferon. Linear algebra (4th Ed).

[5] K.F. Riley, M.P. Hobson, S.J. Bence. Mathematical Methods for Physics and Engineering (2nd Ed). McGraw Hill.

[6] J. Nocedal, S. J. Wright. Numerical Optimization (2nd Ed). Springer.

[7] Kenneth J. Beers. Numerical methods for chemical engineering: applications in Matlab. Cambridge University Press.

[8] D. Barber. Bayesian reasoning and machine learning. Cambridge University Press.