Unit 4. Network Analysis

Jordi Villà i Freixa

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

jordi.villa@uvic.cat

9-16/05, 2023

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

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).

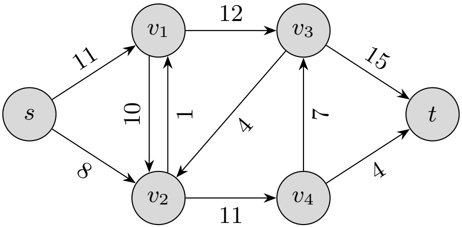
Summary

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.

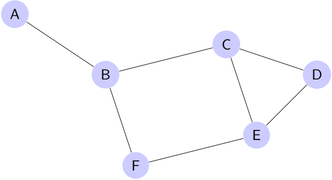


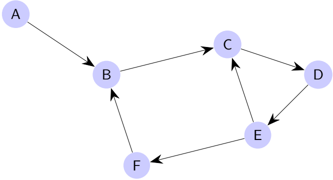
Introduction: Examples

- 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.

Summary

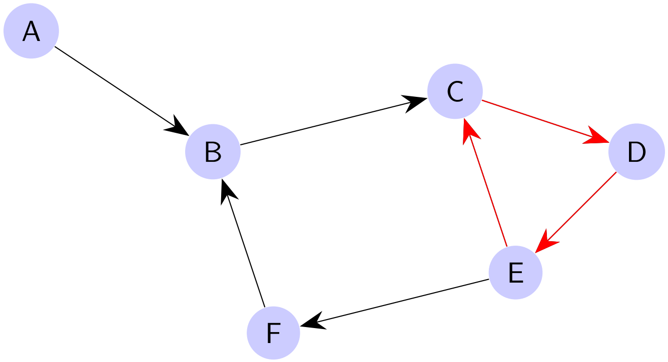
Graphs and networks: definitions





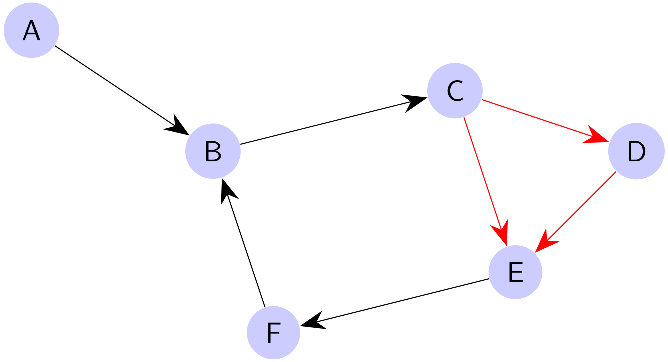
- A graph consists of a set of nodes V (vertices, points, or junctions) and a set of connections called arcs A (edges, links, or branches).
- Each connection is associated with a pair of nodes and is usually drawn as a line joining two points. The graph can be defined as directed or undirected.
- The 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



$$A, (A, B), B, (B, C), \underbrace{C, (C, D), D, (D, E), E, (E, C), C}_{\textit{cyclic}}$$

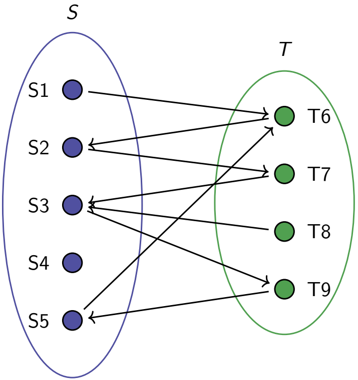
FACULTAT DE CIÈNCIES, TECNOLOGIA I ENGINYERIES

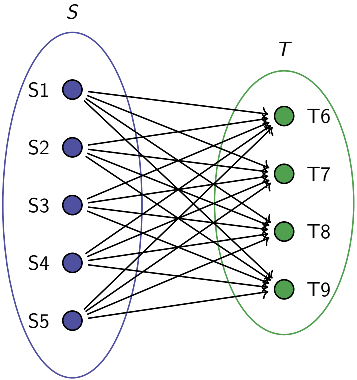


$$A, (A, B), B, (B, C), \underbrace{C, (C, D), D, (D, E), E, (C, E), C}_{noncyclic}$$

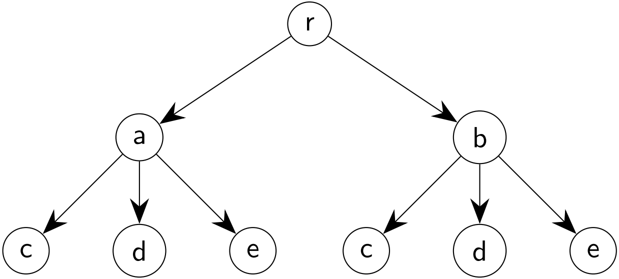
- If all the arcs in a path are forward arcs, the path is called **directed chain** or simply **chain**.
- **path** and **chain** are synonimous if the graph is undirected.
- In the second example above we saw a cyclic path but not a cyclic chain, as it included the backward arc (C, E).
- A **connected 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





- 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.

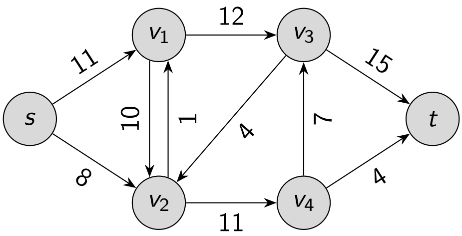
- Nodes in networks can be designated as **sources**, **sinks** or **transshipments**. A **cut set** is any set of arcs which, if removed from the network, would disconnet the sources(s) from the sink(s).
- **Flow** can be thought of as the total amount of an entity that originates at the source, makes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

Summary

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.

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

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

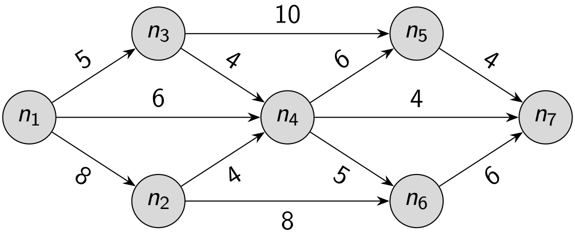
Maximum flow algorithm

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

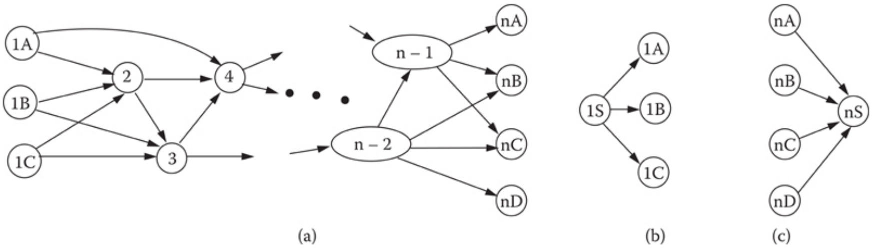
Find the maximum flow in this network using the Ford-Fulkerson labelling algorithm:



- 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]



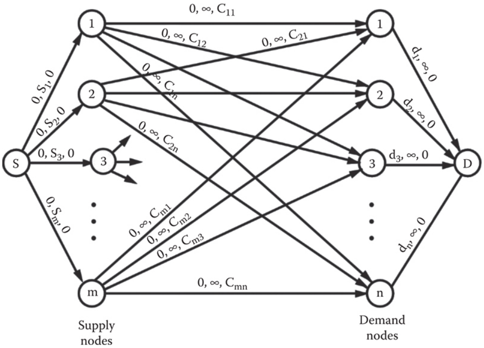
Summary

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.
- For m supply nodes, each providing s_i suplly, and n demand nodes, each demanding d_j. Assuming that the total demand eaquals 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.

$$\text{minimize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

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



Exercise 2

Find the minimum cost in this trasportation problem:

Sources (Warehouses)	Sinks (Customers) 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.
   - Northwest corner rule
   - Minimum cost method
   - Minimum "row" cost method
   - Vogel's method
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	$\boxed{28}$	$\boxed{7}$ 2	$\boxed{16}$	$\boxed{2}$ 18	$\boxed{30}$	20
2	$\boxed{18}$ 8	$\boxed{8}$ 12	$\boxed{14}$	$\boxed{4}$	$\boxed{20}$	20
3	$\boxed{10}$ 4	$\boxed{12}$	$\boxed{13}$ 12	$\boxed{5}$	$\boxed{28}$ 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 $\sum_i \sum_j x_{ij} c_{ij}$

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.

Summary

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

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.