Foundations of Operations Research
Introduction to AMPL and Modellisation

Pierre Hosteins
hosteins@di.unito.it

Politecnico di Milano

November 14th 2013

- Transportation Problem and Integrality Property
- Graph Models

Exercise: Transportation Problem

A company has to ship empty containers stocked in the depots of Verona, Perugia, Roma, Pescara, Taranto, and Lamezia, to the main national harbors of Genova, Venezia, Ancona, Napoli, and Bari. The number of empty containers available in each depot, and the demand of containers in each harbor are:

	Availability
Verona	10
Perugia	12
Roma	20
Pescara	24
Taranto	18
Lamezia	40

	Demand
Genova	20
Venezia	15
Ancona	25
Napoli	33
Bari	21

Exercise: Transportation Problem

The transportation cost is of **1 Euro per km**. The distances from the depots to the harbors, in km, are the following:

	Genova	Venezia	Ancona	Napoli	Bari
Verona	290	115	355	715	810
Perugia	380	340	165	380	610
Roma	505	530	285	220	450
Pescara	655	450	155	240	315
Taranto	1010	840	550	305	95
Lamezia	1072	1097	747	372	333

Give a linear programming formulation for the problem of minimizing the total transportation cost while fulfilling the demand (use integer and continuous variables and compare the results.

Network Optimisation

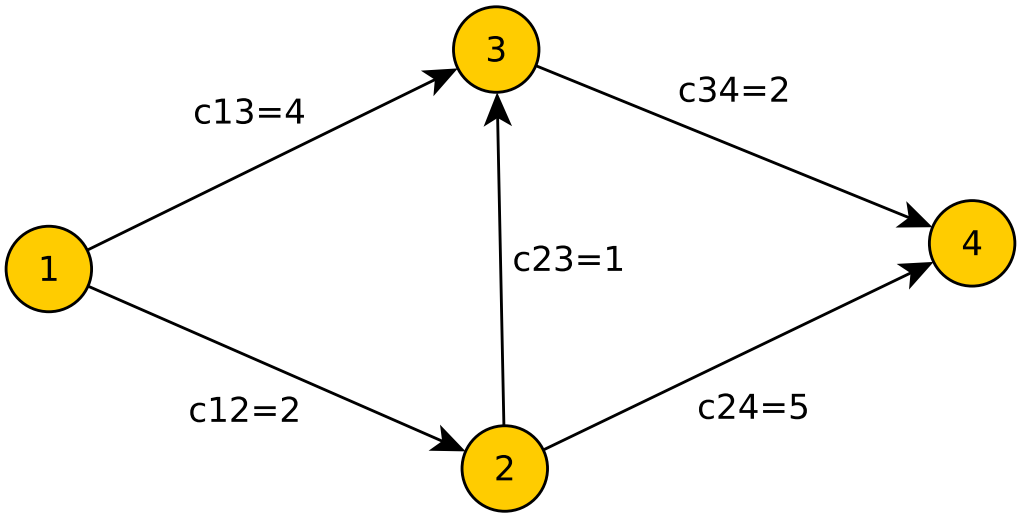
A graph is a mathematical structure composed of a **set of vertices or nodes** V and a **set of edges or arcs** E that link the nodes $i \in V$ to each other. Thus a graph G is usually defined as a couple $G = (V, E)$.
There are two basic types of graphs:

- unoriented: arcs do not have any orientation and an arc between two nodes i and j can be travelled from to i to j as well as from j to i. Example: a connection between two computers where the information can travel both ways.
- oriented: an arc (i, j) is a relation from i to j only and does not imply anything on a relation from j to i. Example: roads can be travelled only one way or can have a traffic flow that differs between both directions.

Usually the arcs $(i, j) \in E$ are associated to a **weight** c_{ij}. The arcs can be seen as a couple of nodes (i, j) with i and $j \in V$, thus E is a subset of $V \times V$: $E \subset V \times V$.

Network Optimisation

Illustration on the **Shortest Path Model**: we must find the shortest path P between two nodes r and s from a graph G.



The weights on the arc represent the distance between nodes. Find the shortest path from node 1 to node 4. A path is a set of arcs.

Network Optimisation

- **Variables**: we define a decision variable for each arc $\Rightarrow x_{ij} = 1$ if arc (i, j) is in the shortest path and 0 otherwise.
- **Objective**: the cost is the length of the path

$$\text{Min} \quad \sum_{(i,j)\in E} c_{ij}\, x_{ij}$$

- **Constraints**: The solution must represent a full path in the network, starting at node $r = 1$ and ending at node $s = 4$. Thus if an arc from P enters a node i (except for r and s), one (and only one) arc from P must leave it:

$$\forall i \in V : i \neq r \text{ and } i \neq s, \quad \sum_{(h,i)\in E} x_{hi} - \sum_{(i,j)\in E} x_{ij} = 0$$

Network Optimisation

- For the starting and ending there must be only one arc leaving and entering (respectively):

$$\sum_{(h,r)\in E} x_{hr} - \sum_{(r,j)\in E} x_{rj} = -1$$

$$\sum_{(h,s)\in E} x_{hs} - \sum_{(s,j)\in E} x_{sj} = 1$$

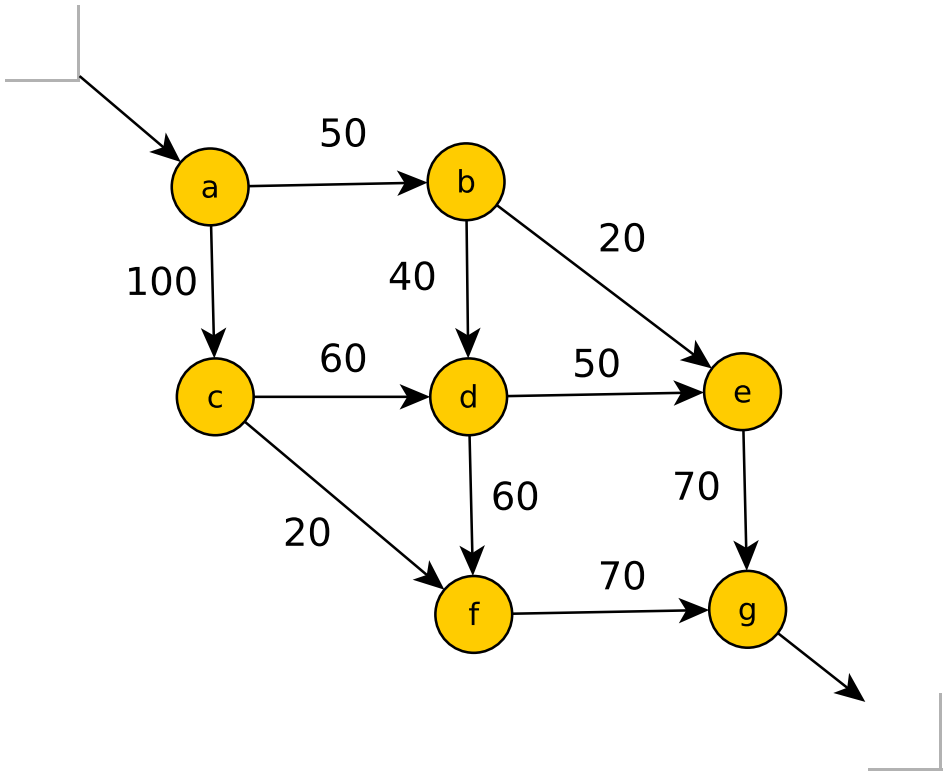
Network Optimisation

```
#Model (shortestpath.mod)
set Nodes; #Vertex set
set Arcs within {Nodes,Nodes}; #Arcs set
param c{(i,j) in Arcs}; #costs for each arc
param r; #Starting node
param s; #Ending node
var x{(i,j) in Arcs} binary;

minimize Length:  sum{(i,j) in Arcs} c[i,j] * x[i,j];

subject to flow{i in Nodes:  i != r and i != s}:
sum{(h,i) in Arcs} x[h,i] - sum{(i,j) in Arcs} x[i,j] = 0;
subject to flow_r:
sum{(h,r) in Arcs} x[h,r] - sum{(r,j) in Arcs} x[r,j] = -1;
subject to flow_s:
sum{(h,s) in Arcs} x[h,s] - sum{(s,j) in Arcs} x[s,j] = 1;
```

Network Optimisation

```
#Model file (shortestpath.dat)

set Nodes := 1 2 3 4;
set Arcs := (1,2) (1,3) (2,3) (2,4) (3,4);
param r := 1;
param s := 4;
param c :=
[1,2] 2
[1,3] 4
[2,3] 1
[2,4] 5
[3,4] 2
;
```

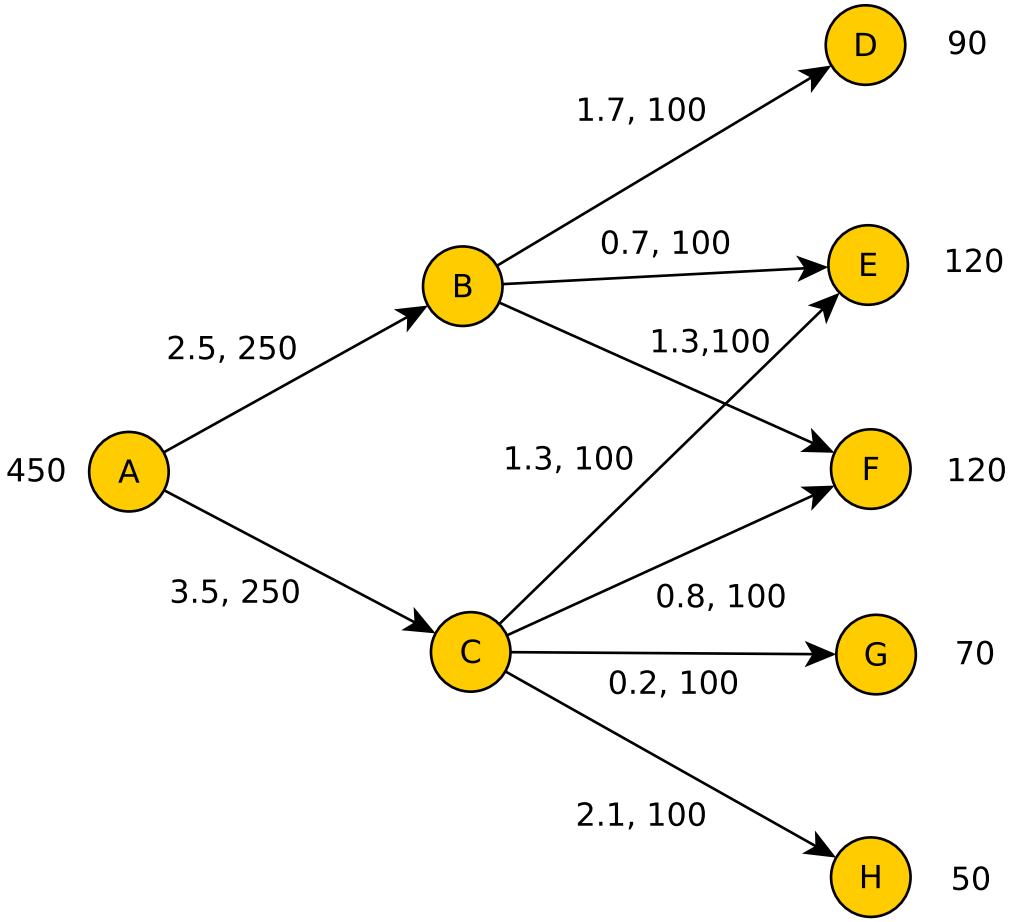
Maximum Flow Model



The Figure presents a diagram of a **simple traffic network**. The nodes and arcs represent **intersections and roads**: **capacities**, shown as numbers next to the roads, are in cars per hour. We want to find the maximum traffic flow that can enter the network at a and leave at g.

Minimum-cost transshipment model

We want to represent the cities and intercity transportation links through the following graph:



Minimum-cost transshipment model

A manufacturing plant at the city A will make 450,000 packages of a certain product in the next week, as indicated by the 450 at the left of the diagram (the natural unit is 1000 packages).
The cities B and C are distribution centers, which receive packages from the plant and transship them to warehouses at the cities D, E, F, G and H.
These warehouses require 90, 120, 120, 70 and 50 thousand packages, respectively, as indicated by the numbers at the right. For each intercity link there is a shipping cost per thousand packages and an upper limit on the packages that can be shipped, indicated by the two numbers next to the corresponding arrow in the diagram.
Find the lowest-cost plan of shipments that uses only the available links, respects the specified capacities, and meets the requirements at the warehouses.