LP. Lesson 7. Chapter 13: Network flow problems

Will study mathematical models for flows in networks. This has applications of the type

- physical networks: internet, roads, railroads, airplanes, telecommunication, computer networks
- logical networks: many possibilities, f.ex. economical investments, optimal allocation of jobs for computers

Note: we shall look at linear algebraic models, not nonlinear models or differential equations.

The models we will look are are special cases of linear programming (LP).

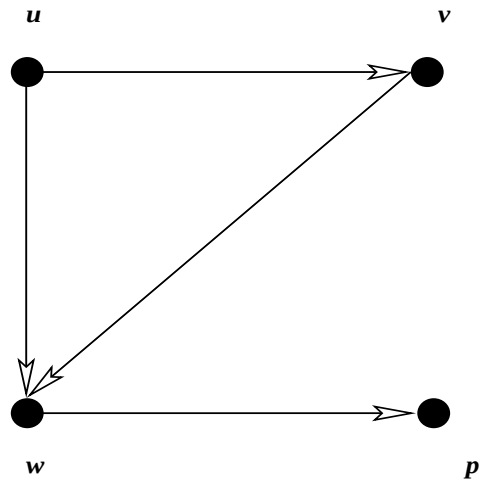
1. The network flow problem

To present the models we need to introduce the term directed graph.

- A directed graph is an ordered pair $D = (V, E)$ where ....
- V is a finite set; the elements are called nodes (or vertices, points)
- E is a finite set which consists of certain ordered pairs of nodes; these elements are called edges (or lines or arcs)

We can represent any directed graph by a drawing in the plane.

Example:
$V = \{u, v, w, p\}$, $E = \{(u, v), (u, w), (v, w), (w, p)\}$.



When $e = (u, v)$ we say that u is a initial node for the edge e, while v is an terminal node for e. And u and v are called the end nodes of e.

We imagine a flow in a directed graph by picturing that goods (materials, data, oil, water,...) flow along every edge in the direction that the edges shows (from the initial node to the terminal node).

We assume that in each node there is a certain supply or demand for goods, the supply in node v is denoted by b_v.

- $b_v > 0$: v is the supply node
- $b_v < 0$: v is the demand node
- $b_v = 0$: v is the transshipment node

For a flow to exist we must have that $\sum_{v \in V} b_v = 0$ since the network does not receive or send goods to the rest of the world. Now let

- x_{uv} be the amount of goods that flows along the edge (u, v) (from u to v along the edge).

We usually have a constraint saying that the flow x_{uv} in each edge (u, v) should be nonnegative and not above a certain given capacity a_{uv}, so

$$0 \leq x_{uv} \leq a_{uv} \quad ((u, v) \in E).$$

We assume that there are costs for shipping goods and that this cost is linear. For each edge (u, v) there is a given cost c_{uv} so that the cost for sending x_{uv} along this edge is $c_{uv}x_{uv}$. The total cost is then

$$\sum_{(u,v)\in E} c_{uv}x_{uv}.$$

We require that the flow $x = (x_{uv} : (u, v) \in E)$ fulfills the following equations called flow balance

$$\sum_{u:(u,v)\in E} x_{uv} - \sum_{w:(v,w)\in E} x_{v,w} = -b_v \quad (v \in V).$$

The first sum equals the total inflow to node v while the other sum is the total outflow from node v. The left-hand side is then the net flow into v and this should equal $-b_v$. The minus in front of b_v is caused by the definition of b_v in front. (We here follow the notation of Vanderbei.)

By multiplying the equation for v by -1 it says that outflow minus inflow equals b_v; this might be easier to remember. But the form above will be used further on (and affects the dual).

We now have all the ingredients in the model and can write the minimum cost network flow problem (MCF):

$$\begin{aligned} \min \quad & \sum_{(u,v)\in E} c_{uv} x_{uv} \\ \text{s.t.} \quad & \\ & \sum_{u:(u,v)\in E} x_{uv} - \sum_{w:(v,w)\in E} x_{v,w} = -b_v \quad (v \in V) \\ & 0 \le x_{uv} \le a_{uv} \qquad\qquad\qquad ((u,v) \in E). \end{aligned}$$

So, this is an LP problem. But it is an LP problem with a special structure!!

Our further work is mainly about understanding this structure better. This will also lead to an efficient method for solving the problem based on a special version of the simplex method.

We will simplify the presentation a bit by assuming that the capacities are infinite so that we don't have to consider these bounds. However, it is possible to treat the general case with a small modification of the method presented under (quickly described later).

It is practical to write the problem in matrix form. Let $n = |V|$ and $m = |E|$. We introduce column vectors x, c and b that correspond to flow, (unity)costs and supply/demand. We have here chosen an ordering of V and E, and x and c have length m while b has length n. Further on we let O signify the zero vector (here of length m). (MCF) the problem is then on matrix form

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & \\ & Ax = -b \\ & x \geq O. \end{aligned}$$

The matrix A is a $n \times m$ matrix which is chosen so that $Ax = -b$ represents the equations for flow balance. This means that A has a column for each edge (u, v) and this column has a component 1 in row v (which means the row that corresponds to node v), -1 in row u and apart from that, all the components are 0.

If we instead consider the rows in A, there is one row for each node v. And row v contains a 1 for each edge which has v as terminal node and -1 for each edge that has v as initial node; the rest of the components are 0. We therefore observe that $Ax = -b$ really corresponds to the flow balance equations.

The matrix A represents the graph $D = (V, E)$ and is called (node edge) the incidence matrix of the graph.

Duality.

As usual when one has an LP problem it is a good idea to look at the dual problem. We will see that it has a very interesting structure.
By using known techniques one can write the (MCF) in standard form $\max\{h^T x : Wx \geq q, x \geq O\}$ and find the dual problem which is

$$\begin{aligned} \max \quad & -b^T y \\ \text{s.t.} \quad & \\ & A^T y + z = c \\ & z \geq O. \end{aligned}$$

Here the vector y has a component for each node and the vector z has a component for each edge.

On component form this dual problem becomes (Dual-MCF) the following

$$\begin{aligned} \max \quad & -\textstyle\sum_{v \in V} b_v y_v \\ \text{s.t.} \quad & \\ & y_v - y_u + z_{uv} = c_{uv} \quad ((u,v) \in E) \\ & z_{uv} \geq 0 \qquad\qquad ((u,v) \in E). \end{aligned}$$

So:

- Two types of dual variables.
- The node variable y_v is tied to the balance equation for the node v.
- y_v is free, while the edge variable z_{uv} is nonnegative.
- Note that the equation determines z_{uv} uniquely when y is given.

From the last observation we can see that the (Dual-MCF) is equivalent to the LP problem

$$-\min \sum_{v \in V} b_v y_v \ \text{ s.t } \ y_v \leq y_u + c_{uv} \ \ ((u,v) \in E).$$

- Here we have eliminated the z_{uv}-variables (they are not a part of the objective function and each of these variables are only a part of one equation).
- If y fulfils these constraints, y is called an feasible potential. A y like this is related to the distances in the graph: if we think of c_{uv} as the length of the edge (u, v), and let y_v be the distance from a certain node r to node v (defined as the length of a shortest path from r to v), then y will be an feasible potential. More about potentials later.

We will now apply the general LP theory to the (MCF). We know that

- we have the optimal solution precisely when we have a feasible primal solution and an feasible dual solution and these together fulfil the complimentary slack demands..

For the (MCF) the complimentary slack conditions become

$$x_{uv} z_{uv} = 0 \quad ((u, v) \in E).$$

Or equivalently:

- if $y_v < y_u + c_{uv}$ (which means that $z_{uv} > 0$), then $x_{uv} = 0$.

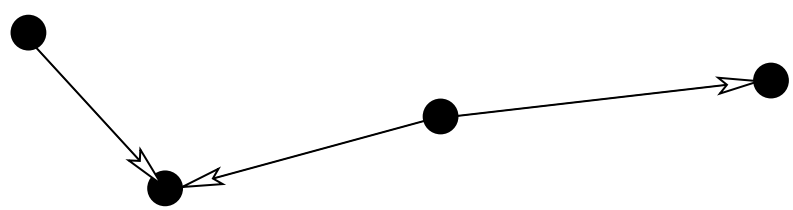
This plays a central part for the algorithm we'll look at later on!

2. Spanning trees and bases

(MCF) can be solved far more efficiently than general LP problems because the bases in the problem have a special structure. This is caused by the structure of the node edge incidence matrix which is the coefficient matrix. We'll take a closer look at this:

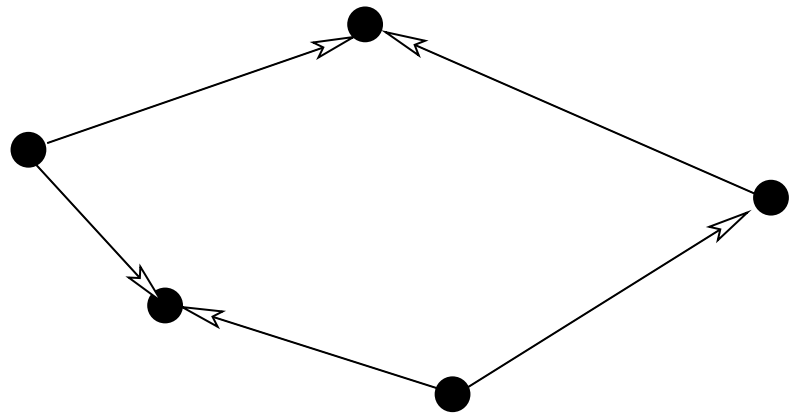
First, we need some graph terms.

Path: a sequence $v_1, v_2, \ldots, v_t$ where (v_i, v_{i+1}) or (v_{i+1}, v_i) belongs to E (is a edge in the graph) for all i. We say that the path is between v_1 and v_t (the end nodes of the path). Here is an example of a path with 3 edges (and thereby 4 nodes):



We will from now on assume that the graph is connected, which means that there is a path between each pair of nodes.

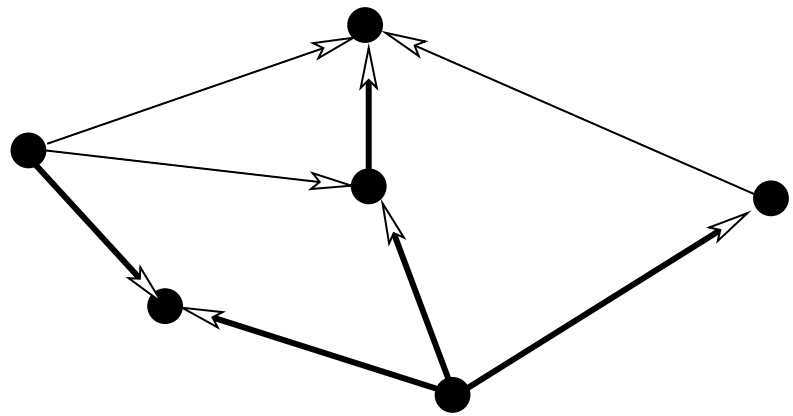
Cycle: like a path apart from that the end nodes coincide, f.ex.



Subgraph: new derived graph $D' = (V', E')$ where $V' \subseteq V$ and $E' \subseteq E$ and where each edge in E' has both end nodes in V'.

Tree: a connected graph T which does not contain any cycle. (Then the number of edges of T = the number of nodes -1).

Spanning tree: a connected subgraph $T = (V, E')$ where T doesn't contain a cycle; note that T contains all the nodes in V (a spanning subgraph). Here is an example of a graph and a spanning tree (denoted by thick edges):



The important thing for us now is that

1. Spanning trees correspond to LP bases.
2. Calculations of corresponding basic solutions can be done by simple operations in the spanning tree.
3. The pivot corresponds to a simple modification of the spanning tree (into a new spanning tree).

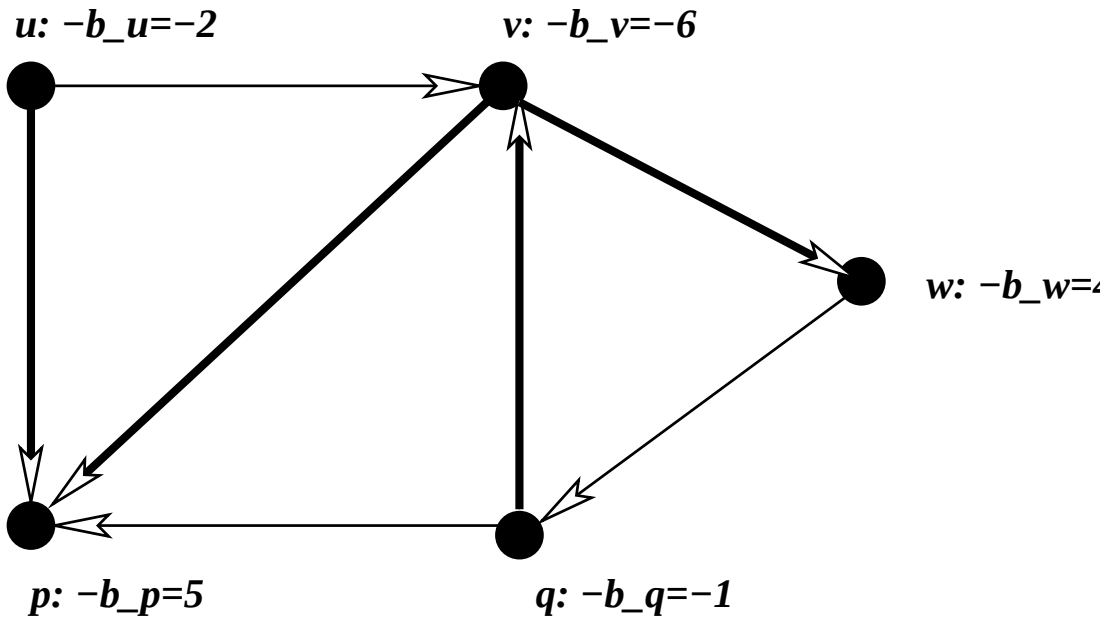
We should mention that the number of spanning tree can be very high (it depends on the graph), men we still get an efficient method (because only some of the spanning trees are "visited").

Let us first study how to calculate the flow x for a given spanning tree T in the graph.

We shall calculate the corresponding so called tree solution x.

- We require that there is no flow outside the tree, which means that $x_e = 0$ for all $e \notin E(T)$ (here $E(T)$ denotes the edges of the spanning tree T).
- It turns out that this is enough to determine all the variables of the edges in the tree uniquely! So, each x_e for $e \in E(T)$ then becomes uniquely determined.
- Note that this is the same phenomenon we know from general LP: when the nonbasic variables are chosen, the basic variables are uniquely determined.

We will illustrate the principle through the following example:



1. $x = 0$ outside T: $x_{uv} = x_{qp} = x_{wq} = 0$
2. Balance in u, outflow-inflow=b_u gives $x_{up} = 2$.
3. Balance in w gives $x_{vw} = 4$.
4. Balance in q gives $x_{qv} = 1$.
5. Balance in v gives $x_{vp} = 6 + 1 - 4 = 3$.

(In node 4 we could alternatively have used the balance in node p.)

The important part was that by choosing a clever ordering of the x-variables, the calculations became simple: in each case the new variable x_e was determined by an equation where only x_e was unknown. F.ex. in step 5 above x_{vp} was determined simply because we knew all the other variables for the neighboring edges of node v.

Algorithm for calculation of x for a given T:

1. Choose a leaf in the spanning tree T that is a node that is next to exactly one edge e in T.
2. Calculate x_e for this edge e.
3. "Remove" e from T, and go back to step 1 above until all the variables are determined.

It is obvious that this procedure works. Here, we use that any tree has at least one leaf (actually at least two), why?

So:

- We have a simple and efficient algorithm for calculating the tree solution x for a spanning tree T.

Our next project is to explain

- why spanning trees correspond to LP basics and
- how the other calculations of a pivot can be done.

These questions will be answered during the next lecture.