Network Models

Simple Network Models

Consider the network described in MCFNetwork.pptx. Assume a cost is specified for each arc. We wish to inject units of flow into the nodes on the left, flow it through the network and extract it from the nodes on the right. The cost of flow on an arc is equal to the volume of flow multiplied by the cost of the arc. The Minimum Cost Flow problem is to select the flows so the total cost is minimised.

More formally, we have a network with a set of nodes N and a set of arcs A. For each node $n \in N$ we define s_n the "supply" of node n. In the example given:

$$s_1 = 50, s_2 = 95, \dots, s_5 = 0, s_6 = 0, \dots, s_{11} = -70, s_{12} = -65$$

Nodes with $s_n > 0$ are known as "supply nodes" or "sources". Nodes with $s_n < 0$ are known as "demand nodes" or "sinks". Nodes with $s_n = 0$ are known as pure transhipment nodes. (A transhipment node is any node that can have flow through it. A pure transhipment node is also neither a source or a sink.)

For each arc $a \in A$ we define f_a as the node where arc a starts (from node), t_a as the node where arc a ends (to node) and c_a as the cost per unit flow. We also define a decision variable x_a for each arc – the units of flow on arc a.

We can then define the Minimum Cost Flow problem as:

$$minimise \sum_{a \in A} c_a x_a$$

Subject to:

$$\sum_{a \in A, f_a = n} x_a - \sum_{a \in A, t_a = n} x_a = s_n \quad \forall\, n$$

$$x_a \geq 0 \quad \forall\, a$$

This formulation is straightforward to implement. See network.py.

Some Observations

The constraints of MCF are linearly dependent. If we sum them all up we get $0 = 0$. We can choose any constraint to omit.

There must exist an optimal solution in which the arcs with flow greater than 0 do not form any undirected cycles. If such a cycle existed we could traverse the cycle (in any direction) and sum up the costs of the arcs, treating an arc cost as negative if it is contrary to the direction of the cycle. If the cycle has net negative cost, we can increase the flow on all arcs in the cycle (decrease if they are in the contrary direction). This will not change the flow balance at any node but will decrease the objective value. If the cycle has net positive cost, we can push the flow in the other direction. And if it the cycle has cost 0 we can eliminate at least one arc in the cycle by pushing flow around the cycle.

In general, the arcs representing any basic feasible solution to MCF form a tree on the network. This observation has lead to the development of highly specialised solution techniques for MCF and similar problems. See whiteboard for summary. If time permits we may return to these algorithms in more detail later in the course.

MCF can be extended in a straightforward way to include arc capacities.

A surprising number of real world problems can be expressed using MCF or its variants. For example, spoil pile reclamation.

Any problem where each variable appears in exactly two constraints – one with a +1 coefficient and the other with a -1 coefficient – is a network flow problem.

Multi-period Network Flow Models

Imagine that we want to keep track of not only the locations involved in flow, but the also the time dimension. This may happen because the production (supply) at a nodes varies (say week by week) in a different way to which the demand varies. In between the supply and demand nodes we may place warehouse nodes, perhaps even with fixed capacity.

In such a network we replicate the nodes – once for each time period – and arcs move between time periods (forward in time) as well as between nodes. Additionally, we add "holding" arcs which connect a node in one time period to the same node in the next time period. These holding arcs represent storage at a node and may have an associated cost and capacity.

See whiteboard for diagram.

To formulate such a problem we need to extend the supply data to have a time dimension (s_{nt}), the arc data to have a time dimension (x_{at}) and add holding variables at each node (h_{nt}) with holding cost at each node d_n. We assume that the arc represented by x_{at} goes from f_a at time t and arrives at t_a at time $t+1$. Assume that time is indexed over $T = \{0,1,\dots,t_{max}\}$.

Then we can write the Multi-period minimum cost flow problem (MPMCF) as:

$$minimise \sum_{t\in T, a\in A} c_a x_{at} + \sum_{t\in T, n\in N} d_n h_{nt}$$

Subject to:

$$\sum_{a\in A, f_a=n} x_{a0} + h_{n0} = s_{n0} \qquad \forall\, n$$

$$\sum_{a\in A, f_a=n} x_{at} + h_{nt} - \sum_{a\in A, t_a=n} x_{a(t-1)} - h_{n(t-1)} = s_{nt} \qquad \forall\, n, \forall\, 0 < t < t_{max}$$

$$- \sum_{a\in A, t_a=n} x_{a(t_{max}-1)} - h_{n(t_{max}-1)} = s_{nt_{max}} \qquad \forall\, n$$

$$x_{at} \geq 0 \qquad \forall\, a, t$$

$$h_{nt} \geq 0 \qquad \forall\, n, t$$

The objective now includes holding costs and the balance constraint are specialised to cope with $t = 0$, when there is no inflow to any node except for supply in that period, and $t = t_{max}$ when there is no outflow from any node except for demand in that period.

Observations

With an appropriate redefinition of nodes and arcs then the MPMCF can be written as an MCF. Each node at each point in time in MPMCF will correspond to a node in the resultant MCF. This is easily seen by observing that each variable appears in exactly two constraints – one with a +1 coefficient and one with a -1 coefficient.

MPMCF can easily be generalised in several ways:

- Upper bounds on holding variables;
- All costs and bounds can be time dependent;
- An arc can take longer than one time period to move from its from node to its to node.

As with all network models, very large scale MPMCF models can be solved quickly.

See in class commercial example.

Multi-Commodity Network Flow

Network flow formulations can be generalised by assuming there are multiple different types of commodities, or products, flowing through the network. Supplies and demands are expressed independently for each product, but (some) arcs have shared upper bounds. That is, the total amount of flow on the arc over all commodities is bounded. If there are no shared upper bounds on arcs then the problem decomposes into separate MCF problems for each commodity. If we introduce a set of P product types, indexed by p, extend x, s and c in the obvious way and introduce shared upper bounds on all arcs (u_a) then the Multi-Commodity Network Flow problem (MCNF) can be written as:

$$minimise \sum_{a \in A, p \in P} c_{ap} x_{ap}$$

Subject to:

$$\sum_{a \in A, f_a = n} x_{ap} - \sum_{a \in A, t_a = n} x_{ap} = s_{np} \quad \forall\, n, p$$

$$\sum_{p} x_{ap} \leq u_a \quad \forall\, a$$

$$x_a \geq 0 \quad \forall\, a$$

Observations

The concept of a “commodity” can be used to model anything where we need to keep track of the “type” of the flow. For example, in some vehicle flow models it can be vehicle type, whereas in mail flow models it represents the destination mail centre.

Generalised Network Flow

Generalized network flow problems generalize traditional network flow problems by specifying a gain factor $\lambda_a > 0$ for each arc a. For each unit of flow that enters the arc λ_a units exit. For traditional network flows, the gain factor of every arc is one. Generalized flows satisfy capacity constraints and node conservation constraints just like traditional network flows.

Generalized flows have many applications. Gain factors can represent physical transformations of a commodity due to leakage, evaporation, breeding, theft or interest rates. They can also represent transformations from one commodity into another as a result of manufacturing, scheduling, or currency exchange: converting dollars into euros, raw materials into processed materials into finished products, acres into feed into fattened cattle, crude oil into processed oil, and machine time into completed orders.

Modern LP solvers can exploit special techniques to get significant speed ups on problems which are expressed as generalised network flow problems, with or without additional side constraints.