AIMMS Modeling Guide - Network Flow Models

This file contains only one chapter of the book. For a free download of the complete book in pdf format, please visit www.aimms.com.

AIMMS B.V.
Diakenhuisweg 29-35
2033 AP Haarlem
The Netherlands
Tel.: +31 23 5511512

AIMMS Inc.
11711 SE 8th Street
Suite 303
Bellevue, WA 98005
USA
Tel.: +1 425 458 4024

AIMMS Pte. Ltd.
55 Market Street #10-00
Singapore 048941
Tel.: +65 6521 2827

AIMMS
SOHO Fuxing Plaza No.388
Building D-71, Level 3
Madang Road, Huangpu District
Shanghai 200025
China
Tel.: ++86 21 5309 8733

Email: info@aimms.com
WWW: www.aimms.com

Chapter 5

Network Flow Models

This chapter

Here, network flow models are introduced as a special class of linear programming models. The AIMMS network formulation is also introduced, and some sensitivity analysis is performed. Furthermore, several classes of network flow models are described.

References

Overviews of network algorithms can be found in [Go77], [Ke80] and [Or93]. An overview of applications of network flow models can be found in [Gl92] and [Or93].

5.1 Introduction

What is a network?

A *network* is a schematic diagram, consisting of points which are connected by lines or arrows. An example is given in Figure 5.1. The points are referred to as *nodes* and the lines are called *arcs*. A *flow* may occur between two nodes, via an arc. When the flow is restricted to one direction, then the arcs are pointed and the network is referred to as a *directed network*.

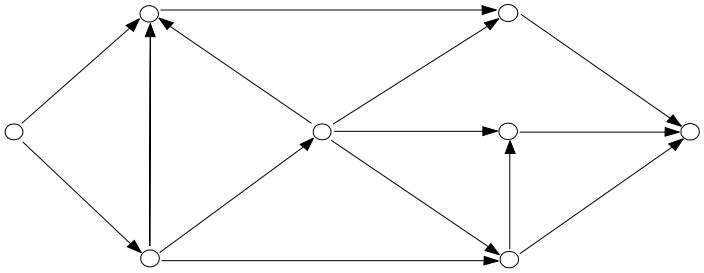

Figure 5.1: A directed network

What is a network flow model?

Network flow models form a class by themselves. They are linear programming models, and can be formulated and solved as such. In practice, however, network flow models are modeled more naturally in terms of nodes and arcs, and are solved quicker by special network algorithms. Therefore, a special type of AIMMS formulation is available for network problems.

The next section

In the following section, an example of a network flow model is given. This example concerns the shipment of goods from factories to customers. The nodes of the network are the factories and the customers, while the arcs represent the possible routes over which the goods can be shipped. The amounts of goods actually shipped form the flows along the various arcs.

5.2 Example of a network flow model

Verbal description

A Dutch company has two factories, one located at Arnhem and one located at Gouda. The company sells its products to six customers, located in London, Berlin, Maastricht, Amsterdam, Utrecht and The Hague. For reasons of efficiency, deliveries abroad are only made by one factory: Arnhem delivers to Berlin, while Gouda ships goods to London. Figure 5.2 illustrates the situation.

Figure 5.2: Factories and customers

Model data

Each factory has a limited supply of goods: Arnhem has 550 tons, and Gouda has 650 tons available. Customer demands and transportation costs from factory to customer are specified in Table 5.1. The goal is to satisfy the customers' demand while minimizing transportation costs.

from to	Arnhem [1000 $/ton]	Gouda [1000 $/ton]	Demand [tons]
London		2.5	125
Berlin	2.5		175
Maastricht	1.6	2.0	225
Amsterdam	1.4	1.0	250
Utrecht	0.8	1.0	225
The Hague	1.4	0.8	200

Table 5.1: Transportation costs and customer demands

Index sets

The index sets are the sets of factories and customers. These two quantities determine the size of the underlying optimization model.

Decision variables

Since the goal of the model is to answer the question, "How many tons of goods should be shipped from the various factories to the various customers?", the decision variables are the numbers of items to be shipped from each factory to each customer, and are measured in tons. Notice that there are ten decision variables, one for each factory-customer pair drawn in Figure 5.2. Models like this one illustrate the usefulness of index notation, since the number of variables increases rapidly when the number of factories or customers grows.

Objective

The objective is to minimize the transportation costs of transporting goods from factories to customers. The costs are measured in thousands of dollars per ton.

Constraints

The constraints follow logically from the problem statement. First, the amount of goods shipped from a factory must be less than or equal to the supply at that factory. Second, the amount of goods shipped to customers must meet their demand. So there are two groups of constraints: supply and demand constraints, both measured in tons.

The verbal formulation

Minimize: *The total transportation costs,*
Subject to:

- *for all factories: Total shipment from a factory can at most be the supply, and*
- *for all customers: Total shipment to a customer must at least be the demand.*

Modeling non-permitted combinations ...

In the verbal formulation, no attention has been paid to the fact that not all combinations of factories and customers are permitted. There are several ways to model this. The first one is to incorporate non-permitted combinations with high cost, so that they will never be used. This option is not recommended for efficiency reasons, since the model contains unwanted variables that unnecessarily increase the size of the problem. The second, and recommended, alternative is to restrict the domain over which the variables are defined, thereby eliminating unwanted variables. For example, one could replace the first constraint as follows:

- *for all factories: Total shipment from a factory* to permitted customers *can at most be the supply.*

... using nonzero costs

In this example, the nonzero transportation costs from Table 5.1 can be used as a basis to restrict the index domain of the variables. These costs give an unambiguous indication of which combinations are permitted. When no cost figure is supplied, then a particular combination factory-customer is to be ignored.

5.3 Network formulation

Two formulation options

There are two possible formulations for network problems. One is the version of the linear programming model stated above. Another option is to take advantage of the special structure of the network. Network flow models can be formulated in a natural way in terms of nodes and arcs. In addition, AIMMS provides a special network flow algorithm, that solves these problems faster than would a standard linear programming algorithm.

Network interpretation

Before formulating the example as a network flow model, some comments are made on the network interpretation of this problem. The basic concepts are *supply* and *demand*. The factories are considered supply nodes. The flow (of goods) out of the various supply nodes must not exceed the amount of goods available. This is again the supply constraint. The customers are considered demand nodes. The flow into the demand nodes must at least match the amount of goods required. Again, the demand constraint appears. Finally, the flows must be such that the costs of transportation are minimized.

Network formulation and AIMMS

In AIMMS it is possible to specify a problem by creating a model using arcs and nodes. `ARC` and `NODE` declarations have taken the place of `VARIABLES` and `CONSTRAINTS`, respectively. Furthermore, the keyword `NetInflow` indicates the flow into a node minus the flow out of it, whereas the keyword `NetOutflow` has the opposite interpretation. These keywords enable one to specify the *balance constraints* on each node. For each arc, the associated pair of nodes is specified, as well as costs attached to it, using the attributes `FROM`, `TO`, and `COST`. Capacities on arcs are specified using its `RANGE` attribute.

Model declarations

The following symbols are used in the mathematical description of the network example of the previous section.

Indices:

f	*factories*
c	*customers*

Parameters:

S_f	*supply at factory f*
D_c	*demand by customer c*
T_{fc}	*unit transport cost between f and c*

Nodes:

FN_f	*factory supply node for f*
CN_c	*customer demand node for c*

Arcs:

$Flow_{fc}$	*transport between f and c*

Nodes and arcs in AIMMS

The following declarations mimic the declarations typically found in AIMMS network models. They take the place of the usual algebraic notation to describe constraints in constraint-based models.

```
NODES:
  identifier   : FN
  index domain : f
  definition   : NetOutflow <= S(f)
  text         : factory supply node for f ;

  identifier   : CN
  index domain : c
  definition   : NetInflow >= D(c)
  text         : customer demand node for c ;

ARC:
  identifier   : Flow
  index domain : (f,c) | T(f,c)
  range        : nonnegative
  from         : FN(f)
  to           : CN(c)
  cost         : T(f,c) ;
```

No standard notation

Network models form a special class of mathematical programs for which there is no generally accepted notation other than the standard flow balances. This is an instance in which modeling languages such as AIMMS have introduced their own keywords to facilitate the description of large-scale symbolic network models.

5.4 Solution and sensitivity analysis

The optimal solution

The optimal solution of the model and data described in the previous sections could be given in a table, but it is more clearly presented as a picture. The optimal deliveries are given in Figure 5.3. The optimal total transportation cost is $1,715,000.

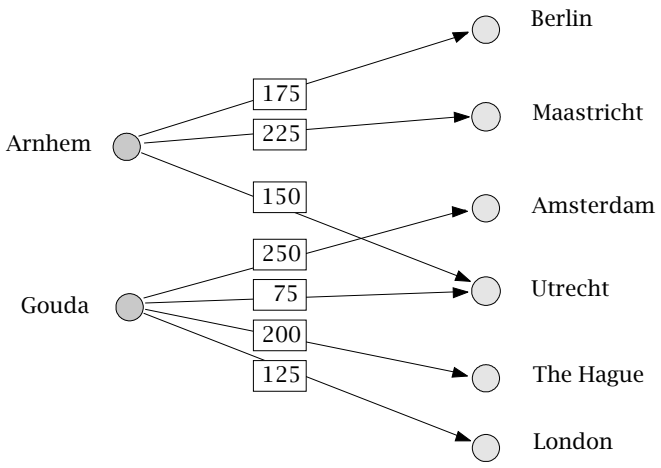

Figure 5.3: The optimal network solution

Reduced costs

In Table 5.2, the reduced costs are given for those routes that were not included in the optimal solution.

from factory	to customer	reduced costs [1000 $/ton]
Arnhem	Amsterdam	0.6
	The Hague	0.8
Gouda	Maastricht	0.2

Table 5.2: Reduced costs

The modified network model

From this table, it is likely that shipments from Gouda to Maastricht would be included in the optimal program if the transportation costs were reduced by approximately $200/ton (from $2000/ton to $1800/ton). Solving the modified model confirms this prediction . Another optimal solution exists and is given in Figure 5.4. The total costs are still $1,715,000.

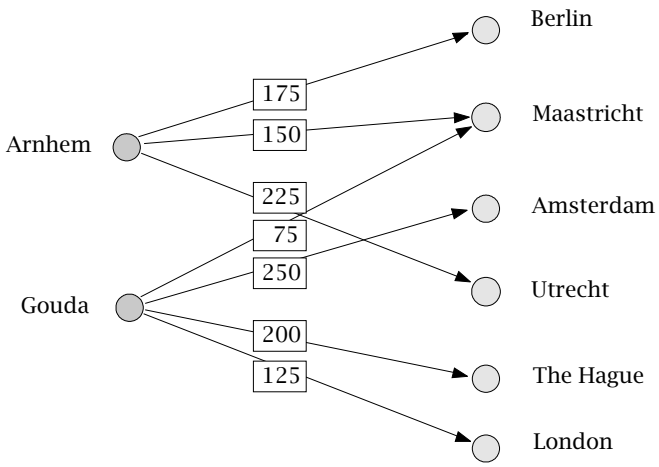

Figure 5.4: An optimal solution of the modified network model

Shadow prices

In Table 5.3, the shadow prices corresponding to the demand constraints for the original network model are given. Adjusting the demand in Berlin downwards reduces the objective the most. There is a reduction of $2,500 per unit transport, but there is the extra benefit that Arnhem is then free to supply Utrecht at a reduction of $200 per unit over Gouda. This gives an overall reduction of $2,700 per unit.

	shadow price [$1000/ton]
London	2.5
Berlin	2.7
Maastricht	1.8
Amsterdam	1.0
Utrecht	1.0
The Hague	0.8

Table 5.3: The shadow prices for the demand constraint

5.5 Pure network flow models

This section

In this section several generic examples of a pure network flow model are presented.

The transportation problem

The example from the previous sections is generally referred to as a *transportation problem*. Transportation problems are characterized by the fact that the nodes are divided into two distinct sets of supply nodes and demand nodes. Supply nodes are often referred to as *sources*, and demand nodes are known as *sinks*. All the arcs in the network go from a source to a sink.

The assignment problem

The *assignment problem* is a special case of the transportation problem. It has the same *bipartite* structure as the transportation problem, but the supply or demand of each node is exactly one. Several practical problems can be modeled as an assignment problem. Examples are the assignment of personnel to tasks and the assignment of jobs to machines.

The transshipment problem

A general pure network flow model may also contain intermediate (or transshipment) nodes. These nodes can have both a flow into the node and a flow out of the node. This type of problem is often referred to as the *transshipment problem*. For instance, adding nodes for distribution centers, with arcs from the factories and arcs to the customers, turns the transportation problem into a transshipment problem. Transshipment models are used in a wide range of practical problems, such as distribution problems, scheduling inventory and production problems, and land allocation problems.

Adding capacities on the arcs

In most practical situations, the flow along an arc is not unlimited, but restricted to some finite capacity. Upper bounds on the flow along arcs are easily handled by the network algorithm. Similarly, lower bounds on the flow along arcs can also be included.

The general pure network flow model

Assuming that the objective is to minimize the total costs associated with the flow along the arcs, the general pure network flow model can be summarized as follows.

Minimize: *Total costs,*
Subject to:

- *for each supply node: the net outflow must be (less than or) equal to the available supply,*
- *for each demand node: the net inflow must be (greater than or) equal to the required demand,*
- *for each transshipment node: the net inflow must be equal to the net outflow, and*
- *for each arc: the flow must be within its bounds.*

Integer solutions

Pure network flow problems have a surprising feature. When all demands are integer-valued, and all lower and upper bounds on the flows along the arcs are integer-valued, then the following is true:

If the network flow model has at least one feasible solution, it has an integer-valued feasible solution, furthermore if it has an optimal solution, it has an integer-valued optimal solution.

In this situation, the network simplex algorithm is guaranteed to find such a solution, and you do not have to resort to an integer programming algorithm. The gains in computation time are considerable.

5.6 Other network models

This section

This section describes two types of problems that can be formulated as pure network flow models, and also some types of network problems that cannot be formulated as pure network models. All of these models can be represented within the AIMMS modeling language.

The shortest path problem

The *shortest path problem* is a problem that can be formulated as a transshipment model. As the name suggests, the goal is to find the shortest path between a single origin and a single destination. All one has to do is to place a supply of one at the origin node and a demand of one at the destination node. All the intermediate nodes have zero demand and supply. The lengths of the arcs are used as costs. The shortest path from the origin to the destination is then determined by the arcs that carry a nonzero flow in the optimal solution of the transshipment model.

The maximum flow problem

The objective in a *maximum flow problem* is to maximize the flow through the network from a single source to a single sink, while the arcs can only carry a limited flow. This problem can be stated as a capacitated transshipment problem by introducing one additional arc from the sink to the source with infinite capacity. The cost attached to this new arc is -1.0, while the cost attached to all the original arcs is zero. All nodes in the network (including the source and sink) have zero demand and supply. By minimizing the total cost, the maximum flow through the network is found. An example of the maximum flow problem can be found in a traffic network, where the arcs, representing roads, have limited traffic capacity. The traffic flows are measured in, for example, number of cars per hour. There are other examples in which the flows represent either messages in a telephone network, or cash in a currency trading network, or water in a pipe transport network.

Generalized network problems

In a pure network flow model, the flow along an arc is conserved, while in *generalized network problems* gains or losses can be specified for each arc. Gains and losses can be due to conversion of units, waste, loss in quality, etc. Generalized network models cannot be solved with a pure network flow algorithm. There are special codes, however, that solve generalized network models, but

these codes, just as with linear programming solvers, do not necessarily terminate with an integer-valued optimal solution.

Multi-commodity network flow problems are just like capacitated transshipment or transportation problems, except that there is more than one commodity to be shipped. In most applications, the arc capacity restrictions do not apply to just a single commodity, but to several commodities together. In this case, the multi-commodity network model cannot be solved with a pure network flow algorithm. The comments made for generalized network models apply here as well. There are specialized solvers, but they too do not necessarily terminate with an integer-valued optimal solution. In practice, linear programming solvers and constraint generation techniques are frequently used for the solution of large-scale multi-commodity network models.

Multi-commodity network problems

5.7 Summary

In this chapter, a transportation problem has been formulated as an optimization model. Transportation problems belong to a special class of network flow problems. Although these problems can be formulated as linear programming models, it is much more natural to formulate them in terms of nodes and arcs, taking advantage of the special structure of the problem. Moreover, solution algorithms exist that take advantage of the network structure of the problem. These algorithms often reach an optimal solution much faster than would linear programming solvers. AIMMS provides facilities for both formulating and solving network models. A special property of many network flow models is that the optimal solution is integer-valued as long as the supplies and demands attached to the sources and sinks are integers. Some examples of different well-known classes of network flow problems were given

Bibliography

[Gl92] F. Glover, D. Klingman, and N.V. Phillips, *Network models in optimization and their applications in practice*, John Wiley & Sons, New York, 1992.

[Go77] B. Golden and T. Magnanti, *Deterministic network optimizations: A bibliography*, Networks **7** (1977), 149–183.

[Ke80] J. Kennington and R. Helgason, *Algorithms for network programming*, John Wiley & Sons, New York, 1980.

[Or93] J.B. Orlin, R.K Ahuja, and T.L. Magnanti, *Network flows, theory, algorithms and applications*, Prentice Hall, New Jersey, 1993.