CHAPTER 99
Network Optimization

RICHARD T. WONG
Telcordia Technologies

1. NETWORK MODELS: INTRODUCTION

Network flow models constitute a special class of linear programming problems. Historically, the useful special structure of networks sparked much of the initial interest and enabled the design of special-purpose algorithms that made network models much easier to solve than general linear programming problems. In addition, networks have a number of representational features that enhance their attractiveness for application in areas of industrial engineering such as manufacturing, production, and supply chain management or logistics and distribution systems. Network models can be especially useful in analyzing systems with spatial or temporal relationships. In network models, the nodes can represent entities such as geographic locations or space-time locations and the arcs can represent allowable ways of going from one node to another. One application scenario might be where the goal is to find the shortest travel route (set of roads) between a dispatching depot and a customer in need of a special delivery. The nodes correspond to intersections of roads, and the arcs correspond to sections of roads connecting the intersections. Another example of networks arises in modeling a flexible manufacturing system (Kumar and Kroll 1987) where a node might represent the location of a machine or a storage area. The arcs would correspond to connections between various system locations via a conveyor system or an automated guided vehicle system.

Network models offer a number of desirable characteristics, including:

Flexibility: Network models can accommodate a large number of different situations.

Versatility as a subproblem: Even in situations where linear network flow models do not apply, such as when there are nonlinear elements in the system, network models often arise usefully as subproblems for the more general model.

 Handbook of Industrial Engineering: Technology and Operations Management, Third Edition.
Edited by Gavriel Salvendy

Ease of explanation: Since many network models can be depicted in a very easy-to-understand way, network models and their results are generally easy to explain and motivate to nontechnical managers.

Ease of Solution: There are a variety of computer codes (running on computers ranging from mainframes to personal computers) capable of solving large-scale models that might arise in practice; these limited computational requirements also facilitate "what-if" analysis that can be useful in performing sensitivity and scenario analyses on a model.

The remainder of this chapter gives more information concerning the formulation, solution and application of network flow models. Section 2 details some of the principal network models. Sections 3 and 4 discuss solution procedure concepts and give information concerning available software for solving network flow models. Section 5 discusses some examples of network applications.

2. CLASSES OF NETWORK MODELS

All of the network flow models presented in this section can be viewed in terms of a single general model—the minimum cost flow problem. However, there are many important problem subclasses whose structures are particularly suggestive for potential applications and that can be solved with special-purpose solution algorithms. This section gives an overview of various types of network flow models.

2.1. Minimum Cost Flow Problem

The minimum cost flow problem is the most general network flow model. It arises in applications such as the efficient distribution of a single product from a set of source (production) sites to a set of demand sites over a given distribution network. The product may be shipped directly from a source site to a demand site, or it may pass through one or more intermediate points in the distribution network before reaching its final destination.

In the network model, the nodes represent source sites, demand sites, or other (intermediate) nodes that are neither source nor demand sites. The net supply at node i is denoted as b_i (for source sites b_i is positive, for demand sites b_i is negative, and for all other nodes b_i is zero). To simplify our discussion, it is assumed that the total amount produced at the source sites is exactly equal to the total amount required at the demand sites. An arc connecting two nodes represents a transportation link. Associated with each arc (i, j) is an upper bound (arc capacity) u_{ij} limiting the total amount of goods that can flow on it. (In more complex models, there may also be a specific lower bound ij on the total flow on the arc. However, to simplify the discussion in this chapter, it is assumed that all lower bounds are zero.) An uncapacitated arc is one that has no upper bound on the amount of flow it can carry. There is a unit transportation flow cost c_{ij}. The minimum cost flow problem can be represented as the following linear program:

$$\text{Minimize} \sum_{(i,j)} c_{ij}x_{ij} \tag{1}$$

$$\text{Subject to:} \sum_{j} x_{ij} - \sum_{j} x_{ji} = b_i \text{ for all nodes } i \tag{2}$$

$$x_{ij} \leq u_{ij} \tag{3}$$

$$x_{ij} \geq 0 \text{ for all arcs } (i, j) \tag{4}$$

The objective is to find the minimum cost routing of the flows through the network such that all source and demand node constraints (2) and all arc capacity constraints (3) are satisfied.

Figure 1 gives an example of an eight node network that represents a distribution system between factories and retailers. Nodes 1 and 2 are factories whose production capacities (net supplies) are 6 and 9. Nodes 3, 4, and 5 are warehouses (intermediate nodes) whose net supplies are zero. Nodes 6, 7, and 8 correspond to the retailers, whose net supplies are -3, -5, and -7. Each arc is labeled with its unit transportation cost and its upper bound. In this example, there are only arcs between source nodes and intermediate nodes and between intermediate nodes and sink nodes. In a general minimum cost flow model there can be arcs between any two nodes in the network.

In many application situations we may wish to restrict the arc flows to be integer valued. The minimum cost flow problem has the interesting property that if all of the net supplies b_i, and arc upper bounds u_{ij} are integer valued, then whenever there is an optimal solution, there is always an integer-valued optimal solution.

The minimum cost flow problem, besides having many useful applications, also contains a number of special cases that arise in a variety of application scenarios. We will now discuss some of these special cases.

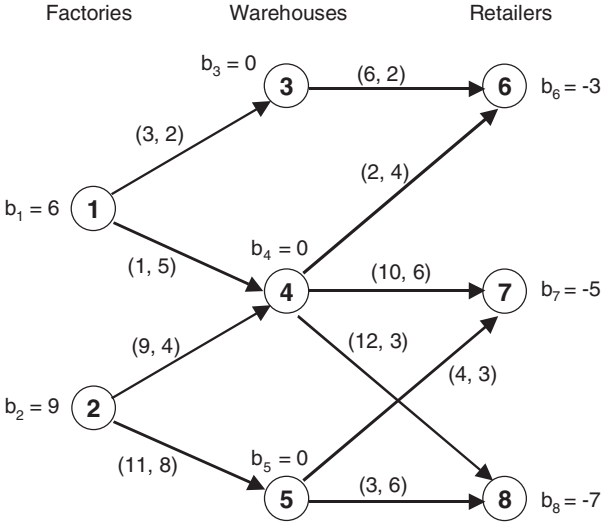

Figure 1 Example of Minimum Cost Flow Problem.

2.2. Transportation Problem

In this subclass of problems there are only source and demand sites (no intermediate nodes are included), and shipments can only be made directly between source and demand sites. The source node i has a supply of s_i and demand node j has demand of d_j (or a net supply of $-d_j$). The transportation problem can be formulated as the following linear programming model:

$$\text{Minimize} \quad \sum_{(i,j)} c_{ij}x_{ij} \tag{5}$$

$$\text{Subject to} \quad \sum_{j} x_{ij} = s_i \text{ for all source nodes } i \tag{6}$$

$$\sum_{i} - x_{ij} = d_j \text{ for all destination nodes } j \tag{7}$$

$$x_{ij} \leq u_{ij} \tag{8}$$

$$x_{ij} \geq 0 \text{ for all arcs } (i,j). \tag{9}$$

Constraints (6) and (7) regulate the net supply for the nodes, and constraint (8) enforces the upper bounds on arc flows. Transportation problems can arise in areas such as logistics inventory control and production planning.

Consider the following example of a production planning problem (Bowman 1956). Production must be scheduled over the next three quarters where the demands are predicted to be 40, 30, and 60, respectively. In each quarter, items may be produced on a regular shift at a cost of \$10 per item. During the regular shift there is a capacity restriction limiting production to 45 items per quarter. A maximum of 20 items can also be produced during an overtime shift, at a cost of \$12 per item. Finally, items can be held in inventory at a cost of \$0.50 per unit per month.

This type of planning problem can be formulated as a transportation model, and the corresponding network is given in Figure 2. Each production shift corresponds to a source node, and the node supply represents the production shift capacity. There are six source nodes since each quarter i can have a regular shift (denoted by node r_i) and an overtime shift (denoted by o_i). For each quarter j there is a demand node j, and finally, there is a dummy demand node DUMMY, which is included to ensure that the total demand equals the total supply. The cost coefficient for arc (g, h) represents the unit cost of supplying demand node h from supply node g. For example, the unit cost of supplying demand node 3 from node o_1, the overtime shift in quarter 1, is (\$12.00 + (2 × \$0.50) = \$13.00). Note that the costs of all arcs incident to the DUMMY node are zero since these flows do not represent any actual production.

This transportation model can also be formulated as the following linear programming problem:

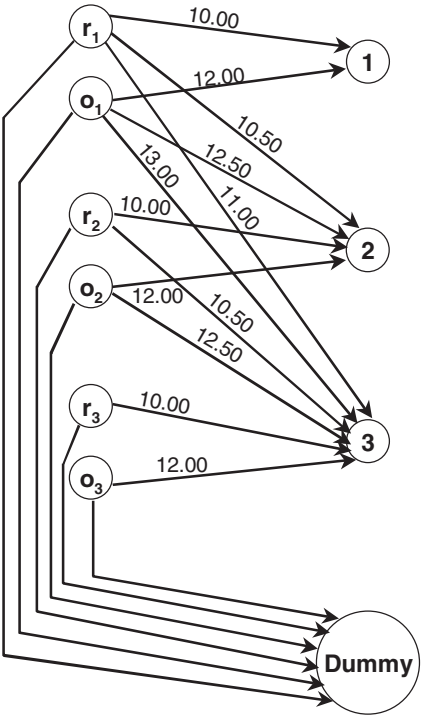

Figure 2 Transportation Problem Representation of Production Planning Example.

$$\begin{aligned}
\text{Minimize } & 10.00\,x_{r1,1} + 10.50\,x_{r1,2} + 11.00\,x_{r1,3} + 12.00\,x_{o1,1} + 12.50\,x_{o1,2} \\
& + 13.00\,x_{o1,3} + 10.00\,x_{r2,2} + 10.50\,x_{r2,3} + 12.00\,x_{o2,2} \\
& + 12.50\,x_{o2,3} + 10.00\,x_{r3,3} + 12.00\,x_{o3,3}
\end{aligned}$$

$$\begin{aligned}
\text{Subject to: } & x_{r1,1} + x_{r1,2} + x_{r1,3} + x_{r1,dummy} = 45 \\
& x_{o1,1} + x_{o1,2} + x_{o1,3} + x_{o1,dummy} = 20 \\
& x_{r2,2} + x_{r2,3} + x_{r2,dummy} = 45 \\
& x_{o2,2} + x_{o2,3} + x_{o2,dummy} = 20 \\
& x_{r3,3} + x_{r3,dummy} = 45 \\
& x_{o3,3} + x_{o3,dummy} = 20 \\
& -x_{r1,1} - x_{o1,1} = -40 \\
& -x_{r1,2} - x_{r2,2} - x_{o1,2} - x_{o2,2} = -30 \\
& -x_{r1,3} - x_{r2,3} - x_{r3,3} - x_{o1,3} - x_{o2,3} - x_{o3,3} = -60 \\
& -x_{r1,dummy} - x_{o1,dummy} - x_{r2,dummy} - x_{o2,dummy} - x_{r3,dummy} - x_{o3,dummy} = -65 \\
& x_{ij} \geq 0 \text{ for all arcs } (i,j).
\end{aligned}$$

Notice that every variable appears in exactly two equations—once with a coefficient of $+1$ and once with a coefficient of -1. For example, the variable $x_{r1,1}$ appears in the first equation with a $+1$ coefficient and in the seventh equation with a -1 coefficient. This special property for the variable coefficients holds for any transportation problem and more generally for ANY general minimum cost flow problem. This special structure is also the key to many of the specialized solution approaches for network flow problems.

2.3. Assignment Problem

A special case of the transportation problem is the assignment problem where there are exactly n source nodes that all have a supply of 1 and n demand nodes that all have a demand of 1 (net supply of -1). Thus, each source node must be assigned to a unique demand node. The cost of assigning source node i to demand node j is c_{ij} and all arcs are uncapacitated. The assignment problem can be represented in the following way:

$$\text{minimize} \quad \sum_{(i,j)} c_{ij}x_{ij} \tag{10}$$

$$\text{subject to:} \quad \sum_{j} x_{ij} = 1 \text{ for all sources nodes } i \tag{11}$$

$$\sum_{i} x_{ij} = 1 \text{ for all demand nodes } j \tag{12}$$

$$x_{ij} \geq 0 \text{ for all arcs } (i,j) \tag{13}$$

Constraints (11) and (12) enforce the supply and demand restrictions.

The assignment problem can arise in a manufacturing system when we view the source nodes as jobs and the demand nodes as machines that can perform the jobs. In the next time cycle, each job must be assigned to a unique machine. The coefficient c_{ij} represents the cost of assigning job i to machine j during this time cycle. The optimal solution assigns the jobs to machines so that the total cost during the next time cycle is minimized.

2.4. Shortest Path Problem

Another special case of the uncapacitated minimum cost flow problem is when there is a single source site and a single demand site and all arcs are uncapacitated. The problem reduces to finding the shortest (minimum cost) path from the source node s to the sink node (demand site) t. As discussed in Section 1, one application of this model might be where we wish to find the best travel route for a truck traveling between a depot and a special customer. Another important application is the routing of information packets in the Internet. A shortest path-routing model is used to determine the path traversed by information packets sent over the Internet (Sackett and Metz 1997). The shortest path problem can be formulated as the following linear programming model:

$$\text{minimize} \quad \sum_{(i,j)} c_{ij}x_{ij} \tag{14}$$

$$\text{subject to:} \quad \sum_{j} x_{ij} - \sum_{j} x_{ji} = \begin{cases} 1 & i = s \\ -1 & i = t \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

$$x_{ij} \geq 0 \text{ for all arcs } (i,j) \tag{16}$$

The objective function minimizes the travel costs and constraint (14) ensures that there is a path from node s to node t.

2.5. Longest Path Problem

The longest path problem is identical to the shortest path problem except that the goal is to find the longest path from a source node to a sink node. An important application of this problem is in the area of project scheduling (Elmaghraby 1977), where there are precedence constraints that specify some tasks cannot be completed until other tasks are done. Each arc represents a task and its length is the task duration. Each node indicates a precedence relationship. The tasks corresponding to outgoing nodes cannot be started until all of the tasks corresponding to the incoming nodes are completed. The source and sink nodes represent the overall start and completion of the entire project. The length of the longest path between the source and sink nodes represents the time required to complete the entire project even if unlimited resources are available to perform tasks that could be performed in parallel. The precedence constraints would generally restrict what tasks could be performed in parallel. The arcs on the longest path indicate the ''bottleneck'' tasks whose reduction in task duration would result in an overall reduction of the project duration.

The linear programming formulation of the longest path problem is identical to that of the shortest path problem except that the objective function must be maximized instead of minimized.

2.6. Maximum Flow Problem

The typical maximum flow problem has a network where there are arc upper bounds and a designated source node s and a designated sink node t. The objective is to find the flow pattern that maximizes

the total flow v from the source to the sink node. The maximum flow problem can be formulated as the following linear programming model:

$$\text{minimize} \quad v \tag{17}$$

$$\text{subject to:} \quad \sum_j x_{ij} - \sum_j x_{ji} = \begin{cases} v & i = s \\ -v & i = t \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

$$x_{ij} \leq u_{ij} \tag{19}$$

$$x_{ij} \geq 0 \text{ for all arcs } (i,j) \tag{20}$$

One application of this model is in the area of emergency evacuation of a facility (such as a building or subway station). The source node represents the location of workers in the facility and the sink node represents a safety area. The arcs can correspond to the various links from one part of the facility to another (stairways, corridors, etc.) and the arc capacity indicates the maximum number of people who could traverse a link per unit time. The maximum flow represents the maximum rate at which people could be evacuated from the facility (see Chalmet et al. 1982 for a more elaborate model).

2.7. Additional Models

There are a number of more advanced models that are beyond the scope of this chapter. For example, there are many network flow models that have additional side constraints. The general linear constraints could correspond to tarriff constraints or other restrictions that cannot be directly incorporated into the basic network flow model. There are some advanced network-based solution procedures that can accommodate linear side constraints (Glover and Klingman 1985). These procedures have been incorporated into available computer software. Another useful generalization of the minimum cost flow problem is when there is more than one type of flow in the network. These situations arise frequently in applications where there may be more than one type of product that is being processed by a distribution network or more than one type of material that is flowing through a materials handling facility or supply chain system. Multicommodity flow models, although they can model a wide range of situations, are usually considerably more complicated to solve than comparably sized (single-commodity) network flow models (McBride 1998; Ali et al. 1984).

Other network models incorporate features such as fixed setup costs or nonlinear flow costs. For example, in a production scheduling environment, the amount of flow on an arc could correspond to the number of units assigned to be processed on a machine. There could be a setup cost incurred if there is a nonzero number of units scheduled on the machine. In a transportation network, there could be a fixed construction cost for opening up a transportation link and a variable operating cost that would depend on the traffic flow through the link. The usual modeling procedure for setup and fixed costs is to introduce 0–1 (binary integer) variables and to create a mixed integer programming model.

Another advanced mixed integer programming network model is the traveling salesman problem, which determines the minimum cost or length tour that passes through each node of the network exactly once. Note that such a tour can also be viewed as the sequence of nodes to visit. From this perspective, the traveling salesman problem has applications in machine scheduling where the sequencing of a set of jobs to be run on a machine must be determined. The arc cost for arc (i,j), c_{ij}, corresponds to the setup or changeover cost on the machine from job i to job j. The tour can also be viewed as the travel route of a delivery vehicle that starts out at a depot and then must visit every other node to make deliveries and then return to the depot. In this context, the cost c_{ij} represents the travel cost of going from the customer at node i to the customer at node j. Finally, the traveling salesman problem also arises in the manufacture of printed circuit boards with a robotic assembly unit (Chan and Mercier 1989). There is a given set of locations at which components must be inserted. The term c_{ij} now corresponds to the cost of inserting the component at location j immediately after inserting the component at location i.

Nonlinear costs could arise when the production cost function exhibits economies of scale and is concave. Another example of nonlinear flow costs is in the area of transportation and distribution planning. Models for analyzing the traffic congestion on a road network usually have arc travel costs that are convex functions of the arc flow in order to represent congestion effects on the network. These models also generally have multiple commodities to represent the various origin–destination characteristics of the network users.

3. NETWORK MODEL SOLUTION TECHNIQUES

The current solution technology for network models is quite powerful and is capable of solving large-scale models that can arise in practice. Also, the continually increasing computational power of

computers guarantees that the size of network models that can be solved will continue to grow. Specialized solution procedures are available on a wide spectrum of machines, ranging from personal computers and workstations to mainframe machines. Some tests have indicated that specialized network codes can be one to two orders of magnitude faster than general linear programming codes in solving network flow models. This section gives a brief overview of some basic concepts for network flow problem-solution procedures. The next section discusses some available computer software for solving network flow models.

Over the past 50 years, there have been several periods of evolutionary development in the field of network algorithms. In the 1950s and 1960s, many classical results and algorithms were developed (Ford and Fulkerson 1962). In the 1970s, many researchers began to recognize and emphasize the role of computer data structures in the efficient implementation of algorithms. With these new and powerful implementations, a number of large-scale applications of network models were successfully completed.

Starting in the 1970s and continuing into the 1980s and 1990s, researchers began to design and evaluate algorithms according to a worst-case computation time criterion (Ahuja et al 1993). That is, the speed of a solution procedure was evaluated according to the maximum number of solution procedure steps required as a function of the problem size. For example, a shortest path algorithm A with worst-case computation time of $4n^2$ is a procedure that when applied to a problem with n nodes requires no more than $4n^2$ steps. It is widely believed that the rate of worst-case computation growth (for algorithm A, the n^2 term) is more important in evaluating algorithms than the constant (for algorithm A, the factor of 4). So the worst-case computation time of algorithm A is usually described as $O(n^2)$ (pronounced ''order n squared''). That is, the worst-case computation time grows as the square of the number of nodes in the network and the exact rate of growth with the constant is not specified. Although the worst-case criterion is mainly a theoretical measure, there is some empirical evidence that indicates that some of the recently developed network solution methods with good worst-case computational performance also perform well relative to other solution procedures.

The following subsections highlight some useful concepts for solving various classes of network flow problems.

3.1. Minimum Cost Flow and Transportation Problems

One of the most effective algorithms for these classes of network problems has been a specialized implementation of the simplex algorithm for linear programming. This type of approach uses special data structures to exploit the special properties of the network models and accelerate the steps of the simplex algorithm. For example, for these network flow models (and all of the other related subclasses discussed in this chapter), the set of basic variables corresponds to a set of arcs that form a spanning tree for the underlying network. Computing such items as the current values for the dual multipliers is easily done with a specialized procedure that exploits the basis tree structure (Ahuja et al. 1993).

3.2. Shortest Path Problem

Dynamic programming and a specialized algorithm due to Dijkstra (Ahuja et al. 1993) are the two most widely used shortest path procedures. Dijkstra's algorithm iteratively labels the nodes with their shortest path distance from the origin node. All nodes are initially unlabeled except for the origin node, which has a label of 0. At each iteration the algorithm labels the unlabeled node that has the minimum shortest path distance from the origin node. The procedure terminates when the destination node is labeled. Dijkstra's algorithm is an $O(n^2)$ algorithm, where n is the number of nodes in the network.

The fastest shortest path codes use either dynamic programming or Dijkstra's algorithm in conjunction with sophisticated data structures that reduce the amount of time spent searching for required problem information.

3.3. Maximum Flow Problem

For the maximum flow problem, the classical solution concept has been the augmenting path. Given a feasible flow pattern, an augmenting path procedure then attempts to find a path from the source node to the sink node such that we can increase the flow along this path and thus increase the total flow from the source node to the sink node. The general augmenting path algorithm iteratively performs a series of path augmentations where each augmentation increases the flow on the selected path as much as possible subject to the arc capacity constraints.

The example in Figure 3*a* shows how a judicious selection of augmenting paths can accelerate the performance of the augmenting path algorithm. In the example, the arc labels are the arc capacities. Nodes s and t are the source and sink nodes, respectively. The initial solution starts with zero flow on all of the arcs. The procedure starts by augmenting the flow on path s–1–2–t by one unit to obtain the flow pattern in Figure 3*b*. Next the procedure augments the flow on path s–2–1–t by one unit to obtain the flow pattern in Figure 3*c*. Iteratively augmenting the flows on these two paths would require a total of 1000 path augmentations. Notice that the optimal solution can be reached

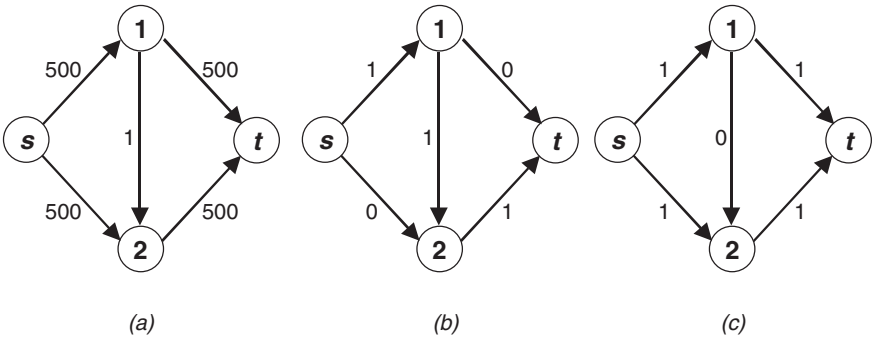

Figure 3 (*a*) Four-Node Example of Maximum Flow Problem (*b*) Example Flow Pattern after One Flow Augmentation (*c*) Example Flow Pattern after Two Flow Augmentations.

from the initial solution in just two augmentations (augment paths *s*–1–*t* and *s*–2–*t* by 500 units each)! Research in the 1970s highlighted the value of judiciously selecting the augmenting paths used and demonstrated that the augmenting path with the minimum number of arcs, the shortest augmenting path, should be chosen (in our example, the paths *s*–1–*t* and *s*–2–*t* would be selected before the path *s*-1-2-*t* since they have fewer arcs). Thus, the maximum flow problem can be approached by solving a series of shortest path problems. This procedure can be improved by saving information about the shortest path computation from one iteration to the next in order to reduce the work required to solve each shortest path problem (Goldfarb and Grigoriadis 1988).

4. COMPUTER SOFTWARE FOR NETWORK FLOW MODELS

A considerable amount of computer software is available for solving network flow problems. During the latter part of the 1990s, the use of the World Wide Web on the Internet has greatly facilitated the dissemination of information and software for solving network flow models.

For personal computers, a number of packages are available which have modules for solving various classes of network flow problems (see Oberstone 1990, chaps. 7, 8 for additional information). The CPLEX (www.cplex.com), OSL (www.6.software.ibm.com/es/oslvZ/features/lib.htm), SAS/OR (www.sas.com), and LINDO (www.lindo.com) commercial systems have network flow modules and run on machines ranging from mainframes to workstations and personal computers. Also, Michael Trick (mat.gsia.cmu.edu/companies.html) has assembled a set of World Wide Web links to a variety of companies that offer OR software including network optimization routines.

There are also a number of codes available for solving network flow models. See Kennington and Helgason (1980), Simeone et al. (1988), and Bertsekas (1991), which all give listings of some network flow codes. More and Wright (1993) contains some discussion of network optimization and available software routines. Also, Michael Trick (see mat.gsia.cmu.edu/resource.html, under Software Packages and Descriptions) has assembled a set of World Wide Web links to a variety of OR software packages (including network optimization routines) that are available for little or no cost.

The field of project management, which includes applications of the longest path problem, has spawned a number of software packages for both commercial and educational use. See Oberstone (1990, chaps. 13, 16) and Wasil and Assad (1988) for a discussion and comparison of some of these packages. It has been estimated that at one time there were over 200 different products on the market (see Scheduling Software, www.buildersnet.org/cpmtutor/html/schedulingsoftware.html). See Microsoft Project (www.microsoft.com/office/project/default.htm) and Job Boss (engineering.software-directory.com/software-2.cdprod1/009/683.Job.Boss.sthml) for two examples of such systems.

5. ADDITIONAL NETWORK FLOW EXAMPLES

This section gives further examples of applications that can be cast as network flow models. These examples also help illustrate the types of situations that can be formulated as network flow problems. In many other applications, the network model structure is often hidden and can require considerable ingenuity to identify and formulate.

5.1. An Example of Dynamic Network Flow: A Material-Handling System

Network flow models can represent temporal as well as spatial relationships. Consider the four-node dynamic maximum flow network example depicted in Figure 4. For each arc the two labels represent

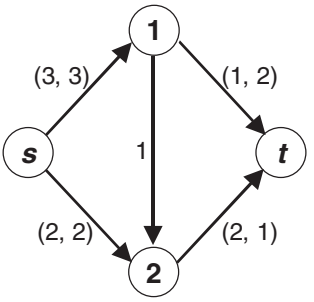

Figure 4 Four-Node Example of Dynamic Network Flow Model.

the arc capacity and the time required to traverse the arc. The dynamic network model can be expanded and represented by an equivalent static model as depicted in Figure 5. In this expanded network, each node represents a specific location at a particular point in time. Such a network could be used, for example, to compute the maximum flow possible between the source and sink node within a given interval of time. By adding arc costs and node demands, the network model could be reformulated as a minimum cost flow problem.

This type of dynamic network flow model can be used to model certain types of material-handling systems (Maxwell and Wilson 1981). The necessary modeling assumptions include that both time and space can be represented by discrete points and that there be only a single type of material which flows through the system. These dynamic network flow models could be useful in evaluating and optimizing preliminary material-handling system configurations and analyzing where system bottlenecks could occur due to storage or processing rate limitations. Maxwell and Wilson (1981) also discuss the role of other methodologies such as simulation (see Chapters 93–96 of this Handbook) in the overall design and evaluation of material-handling systems.

5.2. Personnel Assignment

The assignment of personnel to jobs is an often-cited application area for network flow models. Consider the classical assignment problem where the model represents the assignment of n people to n jobs. There is one source node for each person available and one demand node for each job. One possible objective function would be to assign people to jobs in order to minimize the relocation costs.

More elaborate network flow models for the personnel assignment problem are also possible (Liang and Thompson 1987). Suppose there are more people available then there are jobs but each

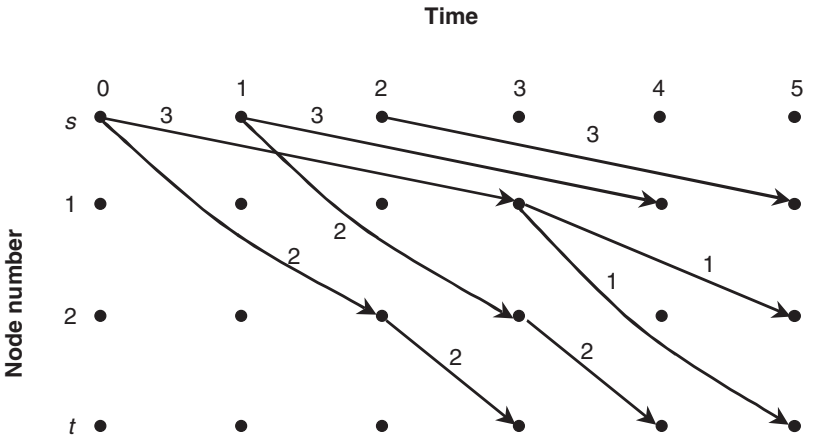

Figure 5 Static Equivalent of Four-Node Dynamic Network Flow Model.

job can be filled by only one person. In particular, consider an example where six people and four different jobs are available. This type of personnel assignment problem can be represented by the minimum cost network flow model in Figure 6. Every node in the layer of nodes (P_1, P_2, . . . , P_6) represents an available person and has a supply of 1. Node t has a net supply of -6. Every node in the layer of nodes (J_1, J_2, J_3, J_4) corresponds to an available job and along with node s are all intermediate nodes whose net supply is zero. All arcs in the network have a capacity of one. In the network flow problem solution, person P_i is assigned to job J_j if there is a flow of one unit on the arcs (P_i, J_j) and (J_j, t). Since arc (J_j, t) has a flow capacity of one, the total flow into node J_j can be at most one and so job j can be assigned to at most one person. Person P_i is not assigned to a job if there is a flow on the arcs (P_i, s) and (s, t). Note that for each personnel node P_i, all of the possible arcs to the job nodes are not included, since a person may only be qualified for a subset of the available jobs.

This type of network flow model can be further expanded by adding another layer of nodes, as depicted in Figure 7. Every job is associated with a particular branch location. In this example there are two branch locations represented by the nodes B_1 and B_2. The total flow between a location node and node t is the total number of jobs at that location that are filled. This information can be useful in enforcing preferences or restrictions concerning the staffing levels at various locations. The staffing preferences can be incorporated by attaching the appropriate arc costs to the arcs incident to node t. As for the restrictions, suppose that company policy specifies a restriction that at most k new jobs at location B_1 will be filled. The model can incorporate this constraint by setting the arc flow capacity on arc (B_1, t) to be k.

This additional layer of nodes illustrates how other preferences or constraints can be added to a network flow model without destroying the special network flow structure. Such modeling techniques are quite useful since network models can be solved much more efficiently than general linear programs.

There is another personnel assignment question for which a network model may be useful. There is a set of jobs and a set of people where for each person there is a list of the jobs that the person is qualified for. The objective is to Maximize the total number of jobs that can be filled with qualified people. This question can be modeled as a maximum flow problem, as shown in Figure 8. All arcs in the network have a capacity of one. The objective function is to maximize the flow between nodes s and t. An arc connecting nodes P_i and J_j indicates that person i is qualified for job j. In the optimal flow pattern, person i is assigned to job j if the flows on arcs (P_i, J_j), (s, P_i) and (J_j, t) are all one. Since every arc (J_j, t) has a flow capacity of one, the total flow into node J_j can be at most one and so job j can be assigned to at most one person. For this model, there are no direct considerations of cost but instead the objective is to maximize the number of jobs filled by qualified people.

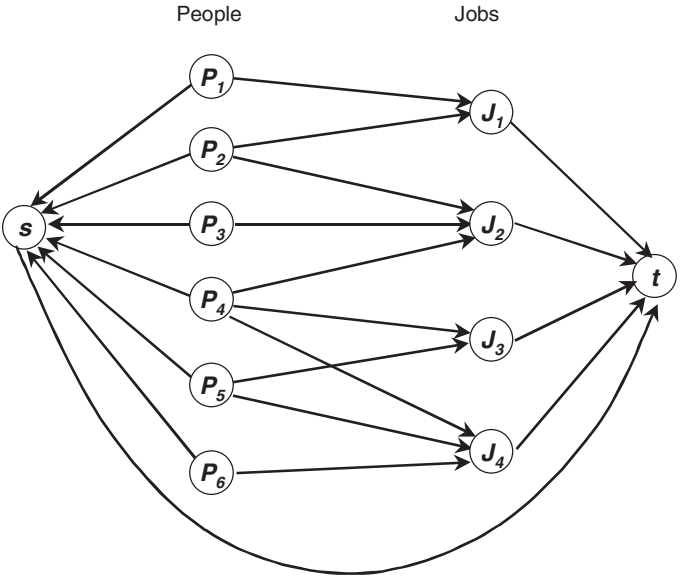

Figure 6 Minimum Cost Network Flow Model of Personnel Assignment Example.

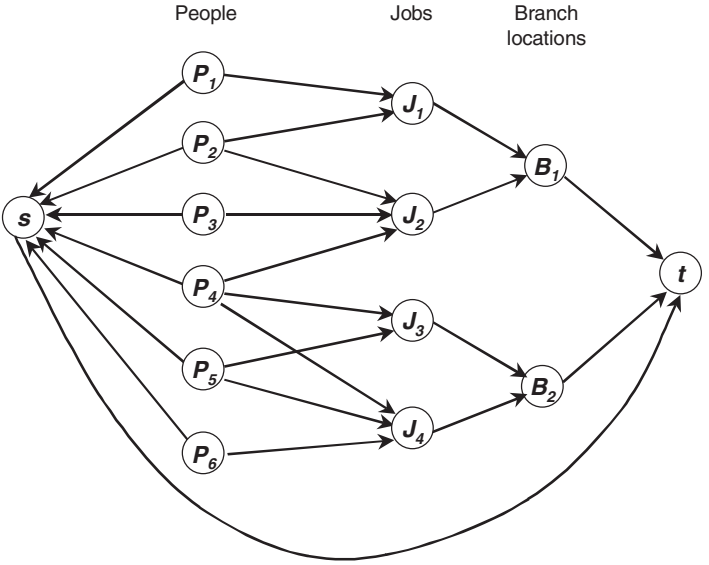

Figure 7 Expanded Minimum Cost Network Flow Model of Personnel Assignment Example.

5.3. Equipment Replacement

Network models can also be useful for the decisions related to replacing equipment over a fixed time horizon. For example, a manufacturing center has just installed a new $20,000 cutting machine at the start of year 1. At the start of each of the next four years, the firm has the option to continue operating the current machine for an additional year or to trade in the current machine for a new machine. The operating costs for the machine grow with the age of the machine, while the trade-in value of a machine decreases with age. The operating costs and trade in values are given below:

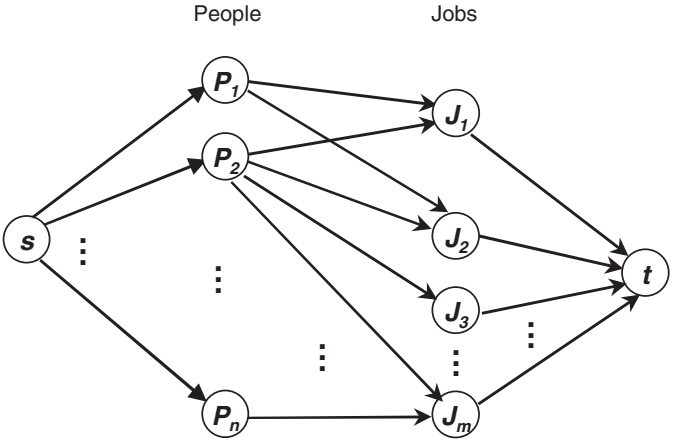

Figure 8 Maximum Flow Model of Alternative Version of the Personnel Assignment Example.

Number of Years Used	Salvage Value	Operating Cost During Last Year
1	$15,000	$1,000
2	$10,000	$2,000
3	$8,000	$4,000
4	$7,000	$8,000

The firm must decide the optimal equipment replacement strategy over the next four years. Figure 9 gives a shortest path network model for determining the optimal replacement strategy. Each of the first five layers of nodes represents the start of a new year. The last layer, which consists of the sink node t, represents the end of the planning horizon. The nodes within each layer represent the status (the age or the number of years the machine has been in use) of the current machine at the start of the year. Each arc represents a decision that can be made at the start of a year. For example, the arc from node Year 3–Age 2 to node Year 4–Age 1 corresponds to having a two-year-old cutting machine at the start of year 3 and then trading it in to buy a new machine so that at the start of year 4 the current cutting machine is one year old. The arc cost of $11,000 corresponds to the cost of buying the new cutting machine ($20,000) minus the trade in value of the two-year-old machine ($10,000) plus the operating cost during year 3 ($1,000). The arcs incident to node t correspond to the final action of trading in the current machine at the end of the planning horizon. The negative arc cost corresponds to the trade-in (salvage) value of the cutting machine. The arcs on the shortest path from nodes s to node t will represent the least-cost machine-replacement strategy over the four-year time horizon.

5.4. Reliability

In many engineering systems for areas such as telecommunications and transportation, the issue of system reliability can arise. A telecommunications system can be modeled as a network where the arcs correspond to communication links and the nodes correspond to switching machines. Figure 10 gives an example of such a representation. The system is designed to send messages from node s to node t where failures may remove an arc (communication link) from service. Assume that the nodes (switching machines) do not fail (or are much less likely to fail than the links). One question of interest is to compute the number of arc disjoint paths between nodes s and t. The answer provides some measure of the redundancy of the network and its resilience to communication link failures. This question can be answered with a maximum flow network model where each arc has a capacity

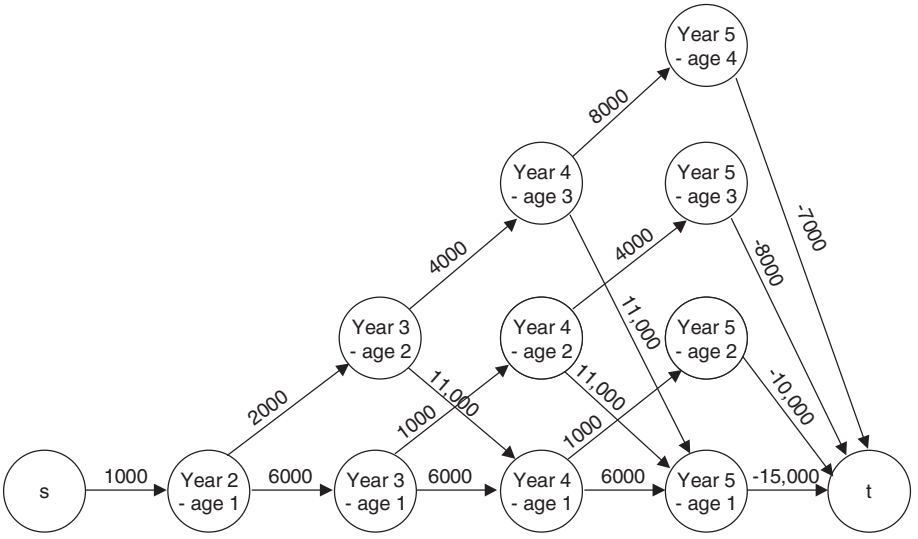

Figure 9 Shortest Path Model of Equipment Replacement Example.

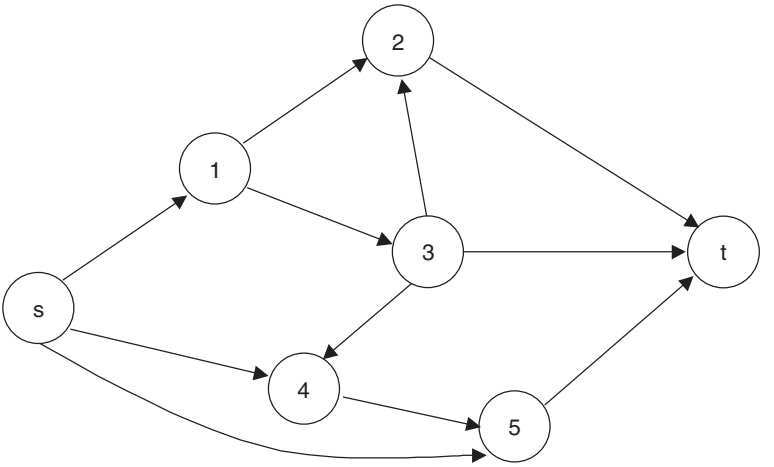

Figure 10 Network Reliability Example.

of one and the maximum flow between s and t is the number of arc disjoint paths between s and t. Since each arc has a capacity of one, every arc disjoint path between s and t can carry one unit of flow and every arc can only belong to at most one flow carrying path, so the maximum flow between these two nodes will equal the (maximum) number of arc disjoint paths between s and t.

Acknowledgement

Much of the work for the original version of this chapter for the second edition of this volume was performed while the author was at Purdue University. Much of the work done in revising this chapter for the third edition of this volume was performed while the author was at AT&T Labs, Middletown, NJ.

REFERENCES

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1993), *Network Flows: Theory, Algorithms, and Applications,* Prentice Hall, Upper Saddle River, NJ.

Ali, I., Barnett, D., Farhangian, K., Kennington, J., Patty, B. Shetty, B., McCarl, B., and Wong, P. (1984), "Multicommodity Network Problems: Applications and Computations," *AIIE Trans*., Vol. 16, pp. 127–134.

Bertsekas, D. P. (1991), *Linear Network Optimization: Algorithms and Codes,* MIT Press, Cambridge, MA.

Bowman, E. H. (1956), "Production Scheduling by the Transportation Method of Linear Programming," *Operations Research,* Vol. 3, pp. 100–103.

Chalmet, L. G., Francis, R. L., and Saunders, P. B. (1982), "Network Models for Building Evacuation," *Management Science,* Vol. 28, pp. 86–105.

Chan, D., and Mercier, D. (1989), "IC Insertion: An Application of the Travelling Salesman Problem," *International Journal of Production Research,* Vol. 27, pp. 1837–1841.

Elmaghraby, S. E. (1977), *Activity Networks,* Wiley-Interscience, New York.

Ford, L. R., and Fulkerson, D. R. (1962), *Flows in Networks,* Princeton University Press, Princeton, NJ.

Glover, F., and Klingman, D. (1985), "Basis Exchange Characterizations for the Simplex SON Algorithm for LP/Embedded Networks," *Mathematical Programming Study,* Vol. 24, pp. 141–157.

Goldfarb, D., and Grigoriadis, M. D. (1988), "A Comparison of the Dinic and Network Simplex Methods for Maximum Flow," in *Fortran Codes for Network Optimization,* B. Simeone, P. Toth, G. Gallo, F. Maffioli, and S. Pallantino, Eds., J. C. Baltzer, Basel also published as Annals of Operations Research, Vol. 13, pp. 83–123.

Hillier, F. S., and Lieberman, G. J. (1980), *Introduction to Operations Research,* Holden-Day, San Francisco.

Kennington, J. L., and Helgason, R. L. (1980), *Algorithms for Network Programming,* Wiley-Interscience, New York.

Kumar, K. R. and Kroll, D. E. (1987), "Dynamic Network Modeling of an FMS," in *Modern Production Management Systems,* A. Kusiak, Ed., North-Holland, Amsterdam, pp. 19–30.

Liang, T. J., and Thompson, T. J. (1987), "A Large-Scale Personnel Assignment for the Navy," *Decision Sciences,* Vol. 18, pp. 234-249.

Maxwell, W. L., and Wilson, R. C. (1981), "Dynamic Network Flow Modeling of Fixed Path Material Handling Systems," *AIIE Transactions,* Vol. 13, pp. 12–21.

McBride, R. D. (1998), "Advances in Solving the Multicommodity-Flow Problem," *Interfaces,* Vol. 28, pp. 32–41.

More, J. J., and Wright, S. J. (1993), *Optimization Software Guide,* SIAM, Philadelphia.

Oberstone, J. (1990), *Management Science: Concepts, Insights and Applications,* West, St. Paul, MN.

Sackett, G. C., and Metz, C. (1997), *ATM and Multiprotocol Networking* (*Computer Communications*), McGraw-Hill, New York.

Simeone, B., Toth, P., Gallo, G., Maffioli, F., and Pallantino, S., Eds. (1988), *Fortran Codes for Network Optimization,* Annals of Operations Research, Vol. 13.

Syslo, M. M., Deo, N., and Kawalik, J. S. (1983), *Discrete Optimization Algorithms,* Prentice Hall, Englewood Cliffs, NJ.

Wasil, E. A., and Assad, A. A. (1988), "Project Management on the PC: Software, Applications and Trends," *Interfaces,* Vol. 18, pp. 75–84.

ADDITIONAL READING

Bradley, S. P., Hax, A. C., and Magnanti, T. L., *Applied Mathematical Programming,* Addison-Wesley, Reading, MA, 1977.

Ozan, T. M., *Applied Mathematical Programming for Production and Engineering Management,* Prentice-Hall, Englewood Cliffs, NJ, 1986.

Phillips, D. T., and Garcia-Diaz, A., *Fundamentals of Network Analysis,* Prentice-Hall, Englewood Cliffs, NJ, 1981.