Applied Network Optimization

CHRISTOPH MANDL

Applied Network Optimization

CHRISTOPH MANDL

Institute for Advanced Studies,
Stumpergasse 56, 1060 Wien, Austria

ACADEMIC PRESS

London · New York · Toronto · Sydney · San Francisco

A Subsidiary of Harcourt Brace Jovanovich, Publishers

ACADEMIC PRESS INC. (LONDON) LTD
24–28 Oval Road,
London NW1

U.S. Edition published by
ACADEMIC PRESS INC.
111 Fifth Avenue
New York, New York 10003

British Library Cataloguing in Publication Data

Mandl, Christoph
Applied network optimization.
1. Network analysis (Planning) 2. Mathematical optimization
I. Title
658.4'032 T57.85 79-40808

ISBN 0-12-468350-9

Filmset in Northern Ireland
at The Universities Press (Belfast) Limited
and printed in Great Britain by
John Wright & Sons Limited,
at The Stonebridge Press, Bristol

Preface

In recent years network optimization has become an important field in operational research. It includes such areas as shortest paths, network flows, traffic equilibrium, Chinese postman and travelling salesman problems, vehicle routing and location on a network, as well as design of an optimal network. Since the work of Ford and Fulkerson in 1962 on flows in networks, this field has been rapidly expanding, as, for example, the bibliography of Golden and Magnanti (1977) demonstrates. Perhaps one reason for the great interest in network optimization is its broad applicability to problems in private enterprise as well as to those in public systems. Telephone networks, rail and road networks, water and wastewater canal systems, and airline networks—all have to be regularly adapted or expanded to meet changing demands, and this leads to decision problems that can be supported by network optimization. Especially promising are network models in transportation systems, urban and regional planning, and civil engineering, and from these areas models have been chosen for presentation in this book.

As in many fields of operational research, the theory of network optimization is better developed and better represented in the literature than the applications. Therefore this book is orientated towards practical application of network optimization, and is an attempt to present applications which are general enough to be of broader use. A solution algorithm is presented for each problem, enabling the reader to implement the algorithm and solve his problem. A book like this obviously cannot cover all applications of network optimization, and it reflects, rather, the author's own experience and interests.

In Chapter 2 network flow problems are discussed. After introducing shortest path and maximum flow algorithms, the traffic assignment problem is presented. Two models, one with flow-dependent the other with flow-independent travel time, are discussed. For both, the differences between descriptive and normative traffic assignment are shown and solution methods described.

Chapter 3 deals with the design of an optimal network and location problems on a network. Algorithms are presented for the following problems: optimal expansion of a waste-water canal system; an optimal waste-water canal system and optimal filter plant location to minimize construction and operating costs; optimal location of emergency service facilities to minimize the number of locations for a given service level in terms of the response time; an optimal network of an offshore natural-gas pipeline system to minimize construction and operating costs; optimal expansion of a road and rail network, under investment constraints, to maximize reduction of transportation time; and the optimal design of a network for air transportation.

In Chapter 4 the sequential construction of a waste-water canal system is analysed, with the objective of minimizing the amount of uncleaned waste water during the construction period of the total canal network. This problem turns out to be a special scheduling problem under precedence constraints.

Chapter 5 is concerned with some vehicle routing problems. Starting with street cleaning routes, the Chinese postman problem is presented and an algorithm for limited route lengths is discussed. Waste collection is the second topic in this chapter, including the presentation of the travelling salesman problem. An heuristic solution procedure is then discussed for school bus routing, the object being to minimize the number of buses necessary.

Finally, all of Chapter 6 deals with the problem of computing optimal routes of buses (or trams) in an urban public transportation network. For a given number of buses the routes are chosen in such a way as to minimize the sum of transportation times of all passengers, thus improving the service of the system.

Most of the algorithms explained in this book were implemented by the author in a FORTRAN version. This Interactive Network Optimization System (IANOS) was written to supply the user with an easy-to-handle computer program in an interactive mode for modest-sized problems. This program package is available from the author (see Mandl, 1979).

Earlier versions of the manuscript have been used as a text for a graduate course at the Department of Industrial Engineering at Universidade Catolica in Rio de Janeiro, and also at the Department of Operational Research at the Institute of Technology in Vienna. The many constructive criticisms from students attending these courses have helped to form the text, and for this help the author would like to express his thanks.

In preparing this book I have benefited from many people. Especially, I would like to thank my colleagues Andrés Polyméris and Hans-Jakob Lüthi, members of the Department of Operational Research at the ETH Zürich, with whom I started to explore new areas of application of operational research in public systems, as well as Professor Franz Weinberg, director of the Department, who first introduced me to the possibilities of operational research.

I am also grateful to Klaus Plasser, Department of Computer Science at the Institute for Advanced Studies, Vienna, for his great help when implementing and testing the computer programs. For typing the manuscript I am indebted to the patience of Mrs Irene Krizsanits. Finally, I would like to express my thanks to Arthur Bourne, Academic Press, for his fruitful suggestions for improving the manuscript.

September, 1979 Christoph Mandl

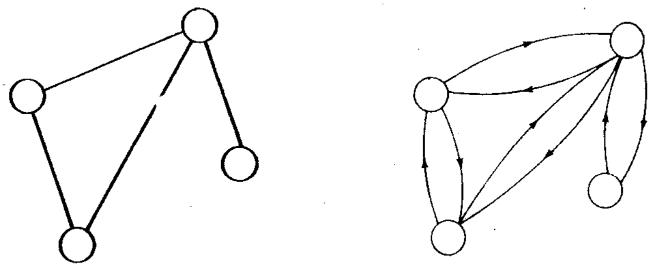
Contents

Chapter 1

Networks

Quite a number of systems may be viewed as and are called networks, although their physical appearance is quite different in nature. Most of these systems are results of human civilization and are of great importance for its functioning. In particular, transportation networks, like road, rail and airline networks, canal networks for transporting water, waste water, oil and natural gas, and communication networks of telephones and computers will be discussed here. However, even natural systems such as caves and rivers can be viewed as networks.

To see the common structure behind these different systems, they must be abstracted from their physical appearance. Then their underlying structure may be seen as a collection of points, which might be road crossings, railway stations, airports, pumping stations and so on, and a collection of lines, which might be roads, canals, telephone cables, etc. connecting all or some of the points. In short we denote a network N by $N=(X, A)$, where X is the set of points, usually called nodes or vertices, and A is the set of lines, called arcs or links. In this abstract formulation of a network, the differences between networks are not in their physical appearance but in their structure, which is described by the sets X and A, and by denoting the nodes of X which are directly connected by some arc of A. In many practical situations the connections between two nodes are directed, in the sense that although node a may be connected with node b, the reverse need not be true. Networks of this type are called directed networks and the associated arcs are known as directed arcs. Examples of such arcs are one-way streets, where cars may only drive in one direction, and also canals or rivers, where water can only flow from higher to lower points. A non-directed network may easily be transformed into a directed



(a) (b)

Fig. 1.1. (a) Non-directed network; (b) directed network.

network. Instead of one non-directed arc one only has to define two directed arcs, connecting two vertices in different directions, as can be seen in Fig. 1.1. In this book we will, for simplicity, call a directed network simply a network.

Before giving some necessary definitions, we have to discuss how to describe fully the structure of a network in a convenient form. There are two common ways for doing so, both of which give a matrix formulation of a network structure. The more compact form is the adjacency matrix. The elements a_{ij} of this matrix are defined as

$$a_{ij} = \begin{cases} 1 & \text{if there is an arc from node } i \text{ to } j \\ 0 & \text{if no arc from node } i \text{ to } j \text{ exists.} \end{cases}$$

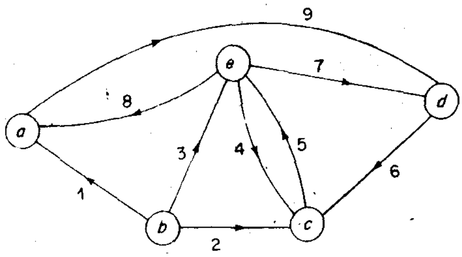
The other notation is the incidence matrix, where the elements b_{ij} of this matrix are defined as

$$b_{ij} = \begin{cases} -1 & \text{if arc } j \text{ starts at vertex } i \\ 0 & \text{if arc } j \text{ neither starts nor ends at vertex } i \\ -1 & \text{if arc } j \text{ ends at vertex } i. \end{cases}$$

Because a network normally consists of more arcs than nodes, the incidence matrix usually contains more columns than the adjacency matrix for the same network. In Fig. 1.2 an example of the graphical and the matrix representation of a network is given.

Of course, for large networks, both matrix representations require much data storage in a computer. Therefore, only the non-zero elements of these matrices, which include all the information, are stored.

In all the problems presented in this book, some value c_j will be associated with an arc $j \in A$. These values can denote the length of the arc, the construction cost, the capacity, the amount of cars (water, oil) passing through the arc per hour, etc. The c_j values can either be stored



(a)

	a	b	c	d	e
a	0	0	0	1	0
b	1	0	1	0	1
c	0	0	0	0	1
d	0	0	1	0	0
e	1	0	1	1	0

(b)

	1	2	3	4	5	6	7	8	9
a	−1	0	0	0	0	0	0	−1	1
b	1	1	1	0	0	0	0	0	0
c	0	−1	0	−1	1	−1	0	0	0
d	0	0	0	0	0	1	−1	0	−1
e	0	0	−1	1	−1	0	1	1	0

(c)

Fig. 1.2. (a) Graphical representation of a network; (b) adjacency matrix; (c) incidence matrix.

for each arc when using the incidence matrix notation, or an extended adjacency matrix can be defined, with the following elements d_{ij}:

$$d_{ij} = \begin{cases} c_l & \text{if arc } l \text{ connects node } i \text{ with node } j \\ Z & \text{if no arc between node } i \text{ and } j \text{ exists.} \end{cases}$$

The value Z must then be some number $\neq c_l$ for all arcs $l \in A$. Usually Z is chosen to be either negative or very large.

A path or route in a network is any sequence of arcs where the final node of one arc is the initial node of the next. Thus in Fig. 1.2a the sequences of arcs

$$1-9-6-5$$

$$3-7-6$$

$$4-5$$

are all routes. A path is called a circuit, if the initial and the final node of this path are identical. If costs are associated with the arcs of a network, then the cost of a route or a circuit is the sum of the costs of all arcs belonging to this route. Thus, if a path consists of the set of arcs P the cost $l(P)$ of the path is defined as

$$l(P) = \sum_{j \in P} c_j$$

where c_j denotes the cost of arc j.

If not all arcs of a given network $N = (X, A)$ are relevant, some of them may be ignored by building a partial network $N_p = (X, A_p)$, with $A_p \subset A$. If not all nodes of a given network $N = (X, A)$ are of interest one can construct a subnetwork $N_S = (X_S, A_S)$ of N, where $X_S \subset X$ and $A_S \subset A$ excludes all arcs the initial and final node of which is not a member of X_S.

If a network shows a railway system, with the nodes being railway stations and the arcs the rails, then the network representing only the main connections is a partial network, the network which represents only the railway system of a special region is a subnetwork, and the network which represents the main connections of the special region is a partial subnetwork (Fig. 1.3).

Some network structures are of particular interest, three of which are discussed below.

If all pairs of nodes are directly connected by an arc, the network is called complete. A good example of this is an airline network. If the nodes denote airports and the arcs denote direct flight connections

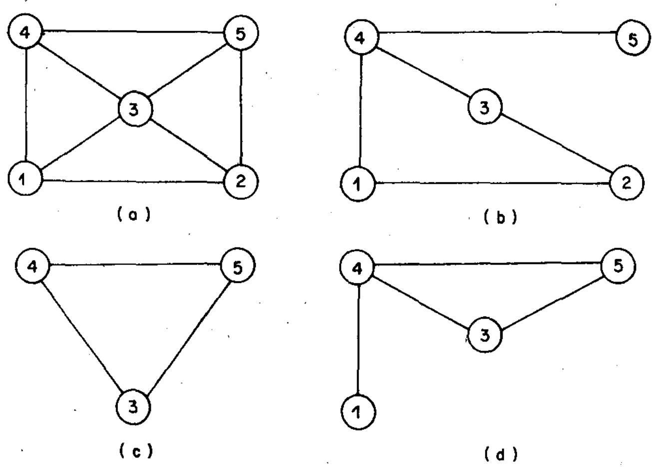


Fig. 1.3. (a) Network; (b) partial network; (c) subnetwork; (d) partial subnetwork.

between two airports, then the network of all possible direct flight connections within a certain region is usually a complete network.

A network $N=(X, A)$ is said to be bipartite if the set of its nodes can be partitioned into two subsets Y and Z, such that all arcs have the initial or terminal vertex in Y and the other in Z.

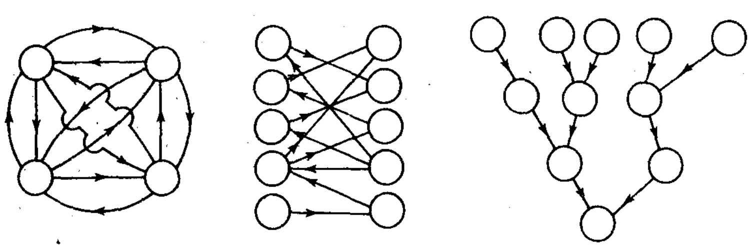
The opposite to a complete network is the tree network or tree. While a complete network contains as many arcs as possible, without having any arc more than once, a tree contains as few arcs as possible such that every node is the initial or terminal vertex of at least one arc. To be more precise, a tree is a network which contains no circuit, but in which there is exactly one path from every node to one particular node. Of all network structures with a given set of nodes X, a tree has the advantage of requiring the minimum number of arcs to connect all nodes with one particular node. This is the reason why canal systems for water or waste water tend to be tree networks, since the construction cost for trees is lower than for arbitrary networks. Also, topographical constraints can lead to tree structures, as, for example, with rivers.

Fig. 1.4 shows examples of all the three types of network structure.

In a general network, it might well happen that it is not possible to find a path between two nodes. Such a network is called non-connected; for example, a tree is a non-connected network. Conversely, a netwc.k in which a path exists between any pair of nodes is called connected. Obviously, road networks, rail networks and communication networks should be connected. Another example of a non-connected network is that shown in Fig. 1.2, where there is no path to node b. Of course, a complete network is always connected.

Finally, a vertex i is said to be adjacent to a vertex j if an arc exists with its initial vertex being i or j and its terminal vertex being j or i.

Now, many network optimization problems can be formulated in linear integer optimization form. Therefore, network problems are strongly connected with integer programming. As is well known, such problems



(a) (b) (c)

Fig. 1.4. (a) Complete network; (b) bipartite network; (c) tree.

tend to be difficult compared to say linear programming. Karp (1975) analysed in detail the difficulty or computational complexity of combinatorial problems. He claims that in general such problems can be divided into two classes. In the first class are those problems which can be solved in polynomial time. This means that if a network has n vertices, the computational time to solve the problem defined on this network is in the worst case growing with $O(n^k)$, where k is some fixed integer number $1, 2, \ldots$ depending on the problem. Such problems, which are quite easy, are said to belong to the P-class and include shortest paths and some network-flow problems.

In the second class, the so-called NP-class, are those problems which today can only be solved in exponential time, i.e. the computational time is in the worst case growing with $O(k^n)$, where n is the number of vertices in the network and k is some fixed integer. Clearly, the solution of NP-problems causes a lot of trouble, but, as we shall see in this book, many practical problems like general integer programming, "travelling salesmen" routes, setcovering and others are of the NP-type.

Thus, for large and difficult problems, approximation techniques and especially heuristic algorithms are of great importance and often run with great success, although the reason for this is still unknown; as Karp (1975) believes: "The ultimate explanation of this phenomenon will undoubtedly have to be probabilistic".

Besides the above-mentioned approaches, branch-and-bound methods and dynamic programming should also be recognized as useful tools for solving combinatorial problems.

Chapter 2
Network Flows

In this chapter problems arising within a given network in which there are flows from vertex to vertex will be discussed. Such flows may be cars within a road network, where the roads are represented by the arcs and the vertices represent cities. The same model applies to public transportation systems, like those of railways and buses, and one can also think of flows in connection with a waste-water canal system or an oil pipeline system.

The easiest problem in this context is the computation of shortest distances and paths between two nodes of a network. With the mathematical formulation of such problems we will introduce the special constraints of network flow problems, the conservation equations. We then discuss how to find the maximum flow between two nodes, when the flow on each arc is restricted. The background of this problem will be the search for bottlenecks in an urban waste-water canal system. Finally, the traffic assignment or traffic equilibrium problem, which consists of finding the flow on each arc within a road network, will be presented. Two models will be discussed: firstly, when arc costs are independent of the arc flow; in the more realistic model, however, the arc costs will increase with increasing flow, thus modelling the common experience that travel time by car increases when traffic density is high.

2.1. Shortest Paths

Anyone who wishes to travel from his present location to any other point, either by foot, car, train, aeroplane or some other mode of transportation, is solving his personal shortest path problem. Although the objectives will be quite different, the type of problem will be the same. Business men will, according to the rule that time is money, try to minimize transportation time, students might prefer to find the cheapest route, car drivers who want to save petrol might choose the shortest path, while passengers on trains with lots of luggage would perhaps prefer a route with the minimum number of changes of trains. All these problems can be formulated as finding the shortest path in a network. The only differences will be the meaning of the cost of an arc. For the first-mentioned objective the cost will be travel time, for the second it will be the price of a ticket, and for the car driver the cost will be denoted by the length of a road.

Although by experience most people solve their problem quite well and therefore will not need any support by a computer, shortest path problems are important for planning and analysing networks, as will be seen later in the book. In this section we will first discuss the problem of finding the shortest distances and routes from one node to one or all other nodes. To gain some insight into the mathematical structure of the problem, we will first formulate it as a linear optimization model and then present an efficient solution procedure. For the more general problem of finding the shortest distances and routes between all pairs of nodes of a network, another algorithm will be presented.

Assume a network $N=(X, A)$, where the cost of an arc $j \in A$ is called c_j. If we want to find the shortest distance and route from node $s \in X$ to node $t \in X$ this can be formulated as a linear optimization model as follows: let x_j be a variable which is one if a person travelling from s to t uses arc j, and is otherwise zero. Then

$$\text{minimize} \sum_{j \in A} c_j x_j \tag{2.1}$$

subject to

$$\sum_{\substack{j \in A \text{ with initial} \\ \text{vertex } k \in X}} x_j - \sum_{\substack{j \in A \text{ with terminal} \\ \text{vertex } k \in X}} x_j = \begin{cases} 1 & \text{for } k = s \in X \\ 0 & \text{for all other } k \in X \\ -1 & \text{for } k = t \in X \end{cases} \tag{2.2}$$

$$x_j \geq 0 \quad \text{for all } j \in A. \tag{2.3}$$

One convenient property of (2.1–2.3) is that their solution will be integer without explicitly stating an integrality condition. Equations (2.2) are called conservation equations and simply state that if a person enters a node he must also leave the node, unless this node is his origin or destination.

One should notice that with the help of the incidence matrix eqns (2.2) may be written as

$$Bx = e \tag{2.4}$$

where B denotes the incidence matrix, x the flow vector and e the right-hand side vector of eqn (2.2). It is the special structure of the incidence matrix B that guarantees the integrality of the optimal flow vector x.

If not only the shortest path between nodes s and t is wanted, but also those between s and all other nodes in X, then eqn (2.4) should be amended to

$$Bx = \begin{cases} (n-1) & \text{for } s \in X \\ -1 & \text{for all } k \in X \text{ and } k \neq s \end{cases} \tag{2.5}$$

where n is the number of nodes in the set X.

Obviously, both problems (2.1)–(2.3) and (2.1), (2.5) and (2.3) may be solved with the simplex algorithm; however, due to the special structure of the incidence matrix faster algorithms are available. The one presented here was developed by Dijkstra. For this algorithm it must be assumed that the costs $c_j \geqslant 0$ for all arcs $j \in A$. However, for problems we are considering this is not a restrictive assumption, because negative costs do not have a practical meaning. The algorithm is divided into two parts: firstly the shortest distances are found, and, secondly, the associated shortest paths.

Algorithm D (Dijkstra's algorithm for shortest distances)

To each vertex $x \in X$ a value $v(x)$ is assigned, which at the end will denote the shortest distance from some node $s \in X$. This value $v(x)$ may be temporary or permanent, where temporary means that $v(x)$ could still be reduced, and permanent indicates that this value already denotes the shortest distance.

D1 [Initialization]. Set $v(s) \leftarrow 0$ and mark this value as permanent. Set $v(x) \leftarrow \infty$ for all $x \in X$ and $x \neq s$ and mark these values as temporary. Set $p \leftarrow s$.

D2 [Updating the values]. For all nodes x which have temporary values $v(x)$ and which are connected by an arc from p, set $v(x) \leftarrow \min[v(x), v(p) + c_j]$, where c_j is the cost of the arc j from node p to node x.

D3 [Fixing a value as permanent]. Of all nodes x with associated temporary values find node y for which $v(y) = \min v(x)$. Mark the value $v(y)$ as permanent and set $p \leftarrow y$.