Large-Scale Networks: Theory and Design

Edited by
Francis T. Boesch

IEEE PRESS

Large-Scale Networks: Theory and Design

Edited by
Francis T. Boesch

Member, Technical Staff
Bell Laboratories

A volume in the IEEE PRESS Selected Reprint Series,
prepared under the sponsorship of the IEEE
Circuits and Systems Society.

IEEE
PRESS

The Institute of Electrical and Electronics Engineers, Inc. New York

International Standard Book Numbers: Clothbound: 0-87942-063-4
Paperbound: 0-87942-064-2

Library of Congress Catalog Card Number 75-25324

PRINTED IN THE UNITED STATES OF AMERICA

Large-Scale Networks: Theory and Design

OTHER IEEE PRESS BOOKS

Optical Fiber Technology
Edited by D. Gloge
Selected Papers in Digital Signal Processing, II
Edited by the Digital Signal Processing Committee
A Guide for Better Technical Presentations
Edited by R. M. Woelfle
Career Management: A Guide to Combating Obsolescence
Edited by H. G. Kaufman
Energy and Man: Technical and Social Aspects of Energy
Edited by G. Morgan
Magnetic Bubble Technology: Integrated-Circuit Magnetics for Digital Storage and Processing
Edited by H. Chang
Frequency Synthesis: Techniques and Applications
Edited by J. Gorski-Popiel
Literature in Digital Signal Processing: Author and Permuted Title Index (Revised and Expanded Edition)
Edited by H. D. Helms, J. F. Kaiser, and L. R. Rabiner
Data Communications via Fading Channels
Edited by K. Brayer
Nonlinear Networks: Theory and Analysis
Edited by A. N. Willson, Jr.
Computer Communications
Edited by P. E. Green, Jr. and R. W. Lucky
Stability of Large Electric Power Systems
Edited by R. T. Byerly and E. W. Kimbark
Automatic Test Equipment: Hardware, Software, and Management
Edited by F. Liguori
Key Papers in the Development of Coding Theory
Edited by E. R. Berlekamp
Technology and Social Institutions
Edited by K. Chen
Key papers in the Development of Information Theory
Edited by D. Slepian
Computer-Aided Filter Design
Edited by G. Szentirmai
Laser Devices and Applications
Edited by I. P. Kaminow and A. E. Siegman
Integrated Optics
Edited by D. Marcuse
Laser Theory
Edited by F. S. Barnes
Digital Signal Processing
Edited by L. R. Rabiner and C. M. Rader
Minicomputers: Hardware, Software, and Applications
Edited by J. D. Schoeffler and R. H. Temple
Semiconductor Memories
Edited by D. A. Hodges
Power Semiconductor Applications, Volume I: General Considerations
Edited by J. D. Harnden, Jr., and F. B. Golden
Power Semiconductor Applications, Volume II: Equipment and Systems
Edited by J. D. Harnden, Jr., and F. B. Golden
A Practical Guide to Minicomputer Applications
Edited by F. F. Coury
Active Inductorless Filters
Edited by S. K. Mitra
Clearing the Air: The Impact of the Clean Air Act on Technology
Edited by J. C. Redmond, J. C. Cook, and A. A. J. Hoffman

Preface

Many complex communication and distribution systems are designed by using large-scale network models. Such systems include telephone networks, gas pipeline networks, and urban transportation networks. The methods used in these designs form a body of knowledge that applies to all these diverse systems.

The philosophy of treating many large-scale network problems by a unified theory has gained a great momentum in recent years. A primary reference for this approach is probably the text *Communication, Transmission, and Transportation Network*, by H. Frank and I. T. Frisch, published by Addison Wesley in 1971. A number of important results have appeared in the literature since the publication of that text, and the intended purpose of the present book is to provide the reader with a single convenient source of some of the more important recent developments in both the theory and application of large-scale network methods. No attempt is being made to produce an exhaustive compilation. As a result of this objective, this book does not explicitly include the pioneering work of Dantzig; Elias, Feinstein, and Shannon; Ford and Fulkerson; and Mayeda. In an attempt to make this work self-contained, however, two specially written tutorial papers have been included to serve as an introduction. The first paper presents the basic concepts of networks and network flows. The second paper, prepared explicitly for inclusion herein by R. E. Thomas of Bell Laboratories, is a tutorial on network design problems using optimal path methods.

Finally, I would like to thank the many people who contributed in some way to this work. I thank the authors of the papers reproduced herein for their original research contributions. I would also like to give special thanks to the following people for their advice, support, and help in the preparation of this volume: M. R. Aaron, W. R. Crone, I. T. Frisch, S. K. Mitra, S. R. Parker, and R. E. Thomas.

F. T. BOESCH
Editor

Contents

Preface . . . vii

Introductory Papers . . . 1

An Introduction to the Theory of Large-Scale Networks, *F. T. Boesch* (October 1975) . . . 3
Optimum Path Problems, *R. E. Thomas* (October 1975) . . . 11

Part I: Basic Theory . . . 17

Network Analysis, *H. Frank and I. T. Frisch* (*Scientific American*, July 1970) . . . 19
An Appraisal of Some Shortest-Path Algorithms, *S. E. Dreyfus* (*Operations Research*, May–June 1969) . . . 28
Depth-First Search and Linear Graph Algorithms, *R. Tarjan* (*SIAM Journal on Computing*, June 1972) . . . 46
A New Algorithm for Finding All Shortest Paths in a Graph of Positive Arcs in Average Time $0(n^2 \log^2 n)$, *P. M. Spira* (*SIAM Journal on Computing*, March 1973) . . . 61
An Effective Heuristic Algorithm for the Traveling-Salesman Problem, *S. Lin and B. W. Kernighan* (*Operations Research*, March–April 1973) . . . 66
An Efficient Heuristic Procedure for Partitioning Graphs, *B. W. Kernighan and S. Lin* (*Bell System Technical Journal*, February 1970) . . . 85
Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems, *J. Edmonds and R. M. Karp* (*Journal of the Association for Computing Machinery*, April 1972) . . . 102
Exact Calculation of Computer Network Reliability, *E. Hänsler, G. K. McAuliffe, and R. S. Wilkov* (*Networks*, 1974) . . . 119
Disjoint Paths in a Network, *J. W. Suurballe* (*Networks*, 1974) . . . 137
Analysis and Design of Reliable Computer Networks, *R. S. Wilkov* (*IEEE Transactions on Communications*, June 1972) . . . 158
Methods for Investigating Connectivity for Large Graphs, *D. J. Kleitman* (*IEEE Transactions on Circuit Theory*, May 1969) . . . 177
Minimum Cost Routing for Static Network Models, *B. Yaged, Jr.* (*Networks*, 1971) . . . 179
Network Reliability Analysis, Part I, *R. Van Slyke and H. Frank* (*Networks*, 1972) . . . 213
On the Existence of n-Connected Graphs with Prescribed Degrees ($n \geq 2$), *D. L. Wang and D. J. Kleitman* (*Networks*, 1973) . . . 225
On the Design of Reliable Networks, *S. L. Hakimi and A. T. Amin* (*Networks*, 1973) . . . 240
Vulnerability of Directed Communication Nets, *I. T. Frisch and M. Malek-Zavarei* (*IEEE Transactions on Circuit Theory*, May 1972) . . . 260
A General Class of Invulnerable Graphs, *F. T. Boesch and A. P. Felzer* (*Networks*, 1972) . . . 262
Testing Graph Connectivity, *R. E. Tarjan* (*Proceedings of the 6th Annual ACM Symposium on Theory of Computing*, April 30–May 2, 1974) . . . 285
A$^n n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs, *J. E. Hopcroft and R. M. Karp* (*SIAM Journal on Computing*, December 1973) . . . 294
A Note on n-Edge Connectivity, *D. L. Wang and D. J. Kleitman* (*SIAM Journal on Applied Mathematics*, March 1974) . . . 301

Part II: Applications . . . 303

The One-Terminal Telpak Problem, *B. Rothfarb and M. Goldstein* (*Operations Research*, January–February 1971) . . . 305
Optimal Design of Offshore Natural-Gas Pipeline System, *B. Rothfarb, H. Frank, D. M. Rosenbaum, K. Steiglitz, and D. J. Kleitman* (*Operations Research*, November–December 1970) . . . 319
A Large Scale Multicommodity Flow Problem: Telpak, *D. J. Kleitman and A. Claus* (*Proceedings of the Symposium on Computer-Communications Networks and Teletraffic*, April 1972) . . . 348

An Experimental Interconnection of Computers Through a Loop Transmission System, *C. H. Coker* (*Bell System Technical Journal*, July–August 1972) 352
Topological Optimization of Computer Networks, *H. Frank and W. Chou* (*Proceedings of the IEEE*, November 1972) 361
Computer Communication Network Design: Experience with Theory and Practice, *H. Frank, R. E. Kahn, and L. Kleinrock* (*Networks*, 1974) 374
Networks and Vehicle Routing for Municipal Waste Collection, *E. J. Beltrami and L. D. Bodin* (*Networks*, 1974) 406
Performance Measurements on the ARPA Computer Network, *G. D. Cole* (*IEEE Transactions on Communications*, June 1972) 436
Terminal-Oriented Computer-Communication Networks, *M. Schwartz, R. R. Boorstyn, and R. L. Pickholtz* (*Proceedings of the IEEE*, November 1972) 443
Design of Long-Distance Telecommunication Networks—The Telpak Problem, *M. C. Goldstein* (*IEEE Transactions on Circuit Theory*, May 1973) 459
Modeling and Optimal Control of Congested Transportation Systems, *D. C. Gazis* (*Networks*, 1974) 466

Author Index 479

Subject Index 481

Editor's Biography 483

Introductory Papers

An Introduction to the Theory of Large-Scale Networks

F. T. BOESCH

Background

The basic mathematical concept underlying large-scale networks is some set of *nodes* which are interconnected by lines or curves called *edges*. We call such an object a *graph* or a *digraph*, respectively, as the edges are undirected or directed with arrows. If each edge is assigned a weight, then we call it a network, or directed network in the digraph case. The weight may represent length of a cable, probability of failure of a power line, capacity of a microwave link, etc. A great many problems regarding the design or analysis of networks can be formulated via this mathematical model. An extremely interesting and informative article by Frank and Frisch [6] serves as an excellent nontechnical introduction. An extensive account of network problems is given in the book by Frank and Frisch [7]. In fact, [7] serves as background for this work in that papers containing results that are included in [7] are not reproduced here. Thus, for example, this book does not include the pioneering papers of Ford and Fulkerson. However, as a service to the reader, this introductory paper will present a tutorial introduction to the theory. The second introductory paper, entitled "Optimum Paths in Networks," is a tutorial introduction prepared by R. E. Thomas.

The basic concepts of graph theory and flows in networks are the essential ingredients of an analytical approach to the design and analysis of large-scale networks. Graphs, digraphs, networks, tours, connectedness, and paths in graphs are all introduced and illustrated via the well-known Königsberg bridge problem. The essential ideas behind the Ford and Fulkerson theory of network flows are presented next. This will include directed and undirected network flow models, the max-flow, min-cut theorem, the equivalence of node flow and path flow, and the labeling algorithm. The application of these results to problems of network vulnerability will also be discussed in this introductory paper.

Probably the first problem to be formulated and solved by a graph model is the famous Königsberg bridge problem. It is used here to illustrate basic concepts. For a complete account of graph theory, consult Harary [11]. The city of Königsberg in East Prussia is divided by the Pregel River, which contains two islands that are part of that city, cf., Fig. 1. Königsberg (now Kaliningrad) is perhaps most famous for being the birthplace of "The Fox of Königsberg," Immanuel Kant (1724–1804). However, it is also the birthplace of the following graph theory puzzle: "Is it possible to cross each bridge once and only once and return to where you started?". Euler in 1736 answered this question by observing that the shape of the city had nothing to do with the problem; all that matters is the interconnection of the bridges, which may be represented by the graph of Fig. 2. In doing this, Euler created what we now call graph theory. Fig. 3 shows the graph model for New York City. Euler then proved that any such graph could be traversed (crossing each edge exactly once) if and only if it is in one piece (connected) and each node has an even number of edges touching it. The fact that even numbers are required is easily demonstrated: starting and finishing each utilizes an edge at the origin, and each time a node is entered it must be departed. The notion that this is all that is required to accomplish such a traversal needs proof. Euler's result should not be regarded as trivial. In fact, an arbitrary trip in a connected graph having the even property may not satisfy the requirements, e.g., the trip indicated by numbering the nodes as shown in Fig. 4.

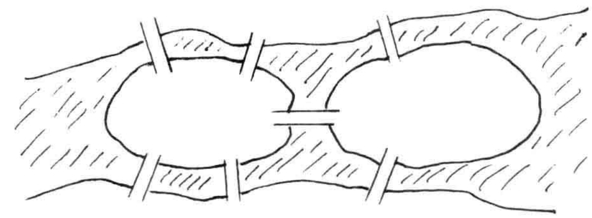

Fig. 1. The seven bridges of Königsberg.

In order to present further details, it is necessary to introduce some terminology.

Walk: A walk in a graph is an alternating sequence composed of nodes and edges from the graph such that each edge in the sequence connects the immediately preceding and succeeding nodes. If an edge is connected between two nodes, we say that the edge is *incident* at these nodes. Note that a walk may repeat edges or nodes. The walk is *open* if the first and last nodes in the sequence are distinct; it is *closed* otherwise.

Trail: A walk with no repeated edges (repeated nodes are allowed).

Path: (open path) A walk with all nodes distinct.

Cycle: A closed trail with all nodes distinct except the first and the last, which are equal.

Connected: There is a path between every distinct pair of nodes.[1]

The author is with Bell Laboratories, Holmdel, NJ 07733.

[1] We purposely ignore the problem of deciding whether or not the graph consisting of a single isolated node is connected.

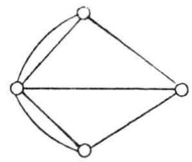

Fig. 2. The Königsberg graph.

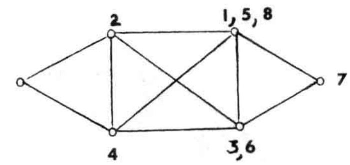

Fig. 4. A trip which does not cover all edges.

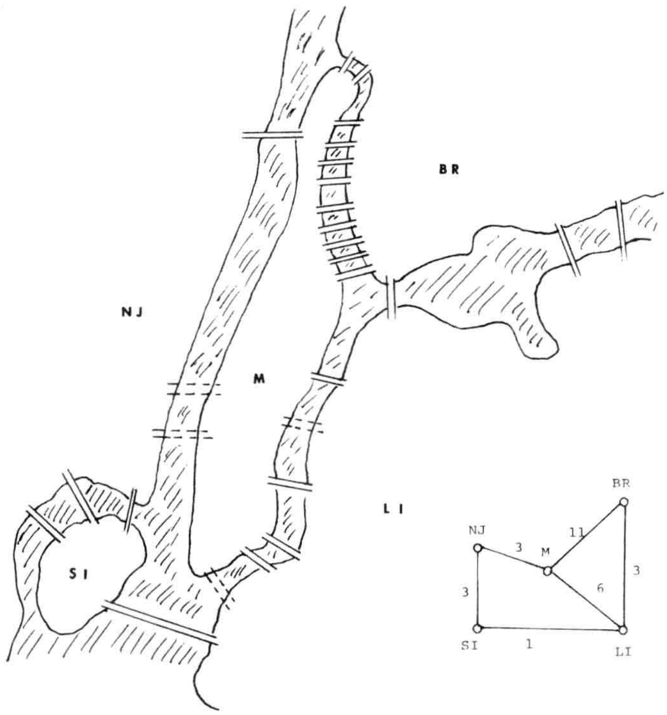

Fig. 3. The New York City bridge or tunnel problem and its graph; multiple edges are indicated by numbering.

Degree of a Node: The number of edges incident at this node. For node i, the degree is denoted by d_i.

To summarize, Euler's Theorem states that a graph has a closed trail which includes all the edges of the graph if and only if the graph is connected[2] and each node has even degree.

As a slight digresssion, we might consider the same problem, except that we are required to finish at a different node from the one we started at, i.e., cover all the edges with an open trail. In this case, it can be shown that the solution exists if and only if there are exactly two nodes with odd degree and the graph is connected,[2] e.g., Kaliningrad today (Fig. 5). If the graph has either an open trail or a closed trail covering all its edges, we call it a *unicursal* figure.

As we have seen, all of these questions depend on whether or not a graph is connected. Although connectedness is a simple idea, it may not be trivial to determine if a large graph is connected, cf., Figs. 6 and 7.

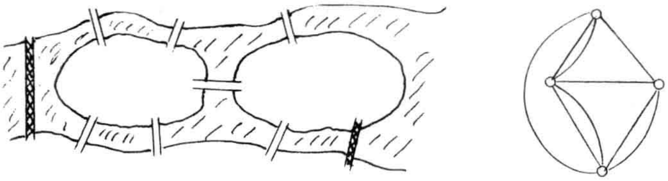

Fig. 5. Kaliningrad today with new bridges and its graph.

An Algorithm for Connectedness[3]

1) Start at any node and label it with a 0.

2) Label all nodes with a 1 that are connected to node 0 by an edge.

3) Label all unlabeled nodes with a $k + 1$ that are connected by an edge to at least one node labeled k (for $k \geqslant 1$).

4) If all nodes become labeled, the graph is connected.

5) If we arrive at a step where it is impossible to continue, yet there are unlabeled nodes, then the graph is disconnected.

For an example, see Fig. 8.

It should be clear that the numerical labels are not really necessary for the purpose of determining connectedness; however, they are useful in determining other properties of the graph which will be discussed subsequently. For the moment, consider the question of finding a path from n_0 to some node, say n_k in level k. Clearly, n_k is connected to some node in level $k - 1$, etc. Hence, such a path can be found.[4]

Let us now return to the problem of finding a closed trail which uses all the edges. The connectedness algorithm together with the obviously simple task of determining whether or not each node has even degree allows us to test for existence. However, this does not tell us what the trail is. As we have already seen (Fig. 5), an arbitrary trip need not generate the desired trail. The standard proof of Euler's Theorem involves a constructive algorithm for producing the desired trail. We shall not elaborate on this point here; the unicursal figure problem is merely being noted as an example. The interested reader may consult Edmonds and Johnson [4] for a complete discussion of the algorithms for Euler-type problems.

Before we conclude this section, we make an important observation regarding the relation of digraphs and graphs. The examples given above were all for graphs since they are

[2] We assume that the graph has no isolated nodes for the purpose of this statement.

[3] This simple algorithm is given for the sake of illustration. For a discussion of a more efficient algorithm, see Tarjan [13].

[4] A complete discussion and an introduction to path problems is given in the following paper written by R. E. Thomas. A comparison of path algorithms is given in Dreyfus [3].

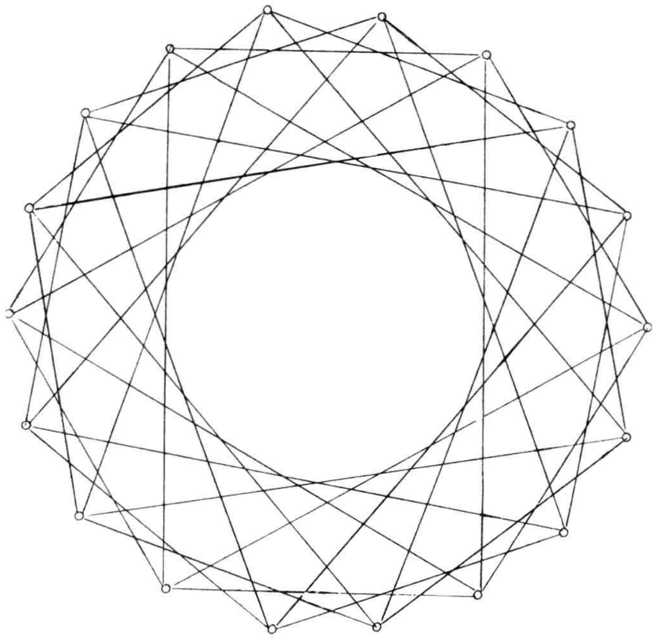

Fig. 6. A large graph that appears to be connected.

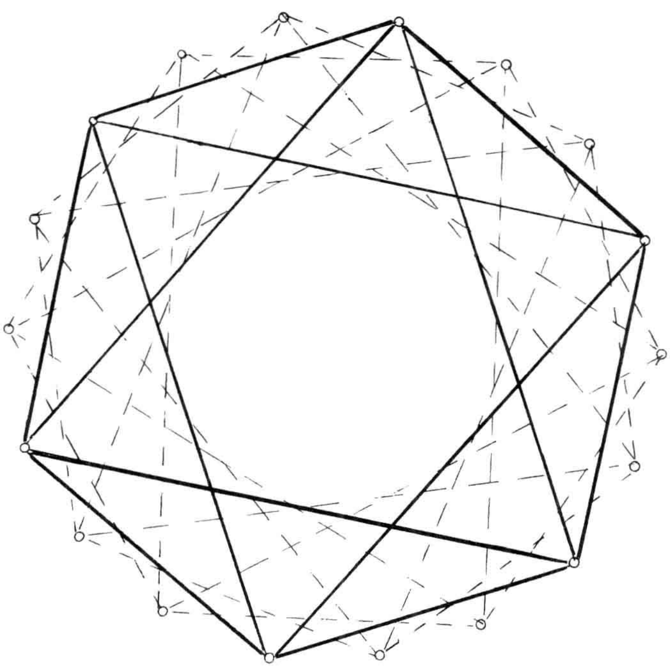

Fig. 7. A demonstration that the graph of Fig. 6 is disconnected.

slightly easier to describe mathematically. Most of these ideas carry over in a rather simple fashion to digraphs with appropriate modifications. However, it should be noted that the directed case is the more general for path problems, since an undirected graph may be modeled as what is known as a symmetric directed graph by allowing each edge to be replaced by two oppositely directed edges (see Fig. 9). In the next section, we will consider the networks as always being directed.

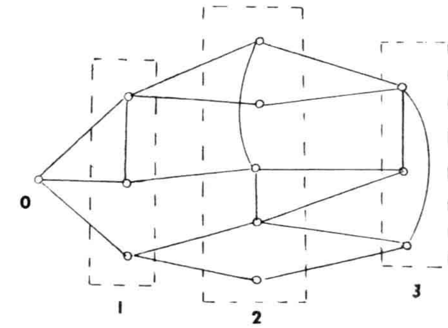

Fig. 8. The connectedness algorithm.

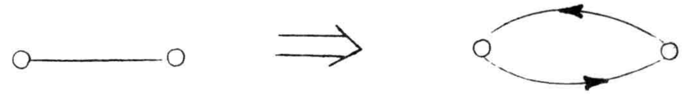

Fig. 9. Representing an undirected graph as a digraph.

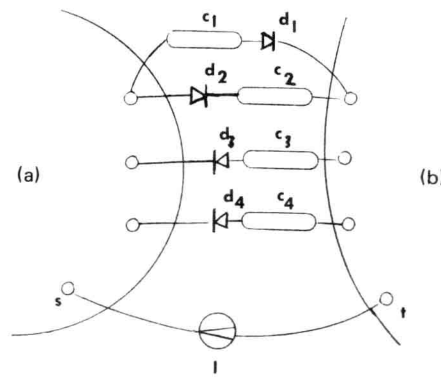

Fig. 10. An electric network flow problem.

Flows in Networks

We consider now the important case where a network has two distinguished nodes called a source n_s and a terminal n_t. We wish to transmit "as much as possible" from n_s to n_t; however, each edge has a weight which represents its maximum capacity. The questions of interest here are what is the limit on "how much we can transmit" and what routing scheme enables us to do it. To be more precise, we must specify the problem in more detail. We imagine that the network is directed (a diode in each edge), that each edge has capacity which cannot be exceeded (a fuse in each edge), and we wish to connect the largest possible dc current source between n_s and n_t without blowing a fuse. Of course, it is presumed that Kirchhoff's Nodal Current Laws are satisfied at all nodes. An extremely useful observation in this problem is that by applying Kirchhoff's Surface Current Law, we can determine an upper bound. Consider the situation pictured in Fig. 10.

Clearly, I can be no larger than the sum of the currents in d_1 and d_2, which are limited in turn by fuses c_1 and c_2. Furthermore, I can only be this large if the currents in d_3 and d_4 are zero. These observations can be summarized as follows.

1) Consider a directed network where each edge e_i has a

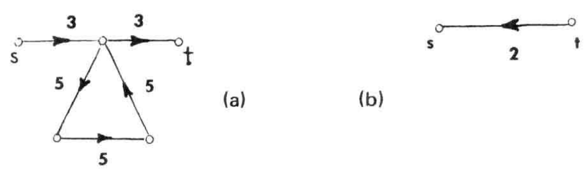

Fig. 11. Algebraic flow distributions that cannot be obtained by paths.

capacity c_i (the current source is regarded here as external to the network).

2) Each edge e_i can sustain a flow f_i with $0 \leqslant f_i \leqslant c_i$.

3) Any set of flows is allowed if it satisfies Kirchhoff's Current Law at all but two distinguished nodes n_s and n_t at which the total flow is v and $-v$, respectively.

4) Find the largest possible value of v.

Consider any partition of the nodes N of the networks into two subsets X and $\bar{X}$. The set of all edges which are directed out of X into $\bar{X}$ is called an n_s, n_t cut if $n_s \in X$ and $n_t \in \bar{X}$. Such a cut is denoted by $(X,\bar{X})$; the sum of the capacities of all the edges in a cut is denoted by $c(X,\bar{X})$, and this is called the cut capacity. A major result is known as the max-flow, min-cut theorem. It was proven by Ford and Fulkerson and independently by Elias, Feinstein, and Shannon.

The Max-Flow, Min-Cut Theorem

For any network, the maximum flow value from n_s to n_t is equal to the minimum cut capacity of all cuts which separate n_s from n_t.

Perhaps an even more important feature of this theorem is that Ford and Fulkerson have given an algorithm for finding this max flow. Before presenting this algorithm, however, we turn to an investigation of a related problem which further demonstrates the importance of these results. The mathematical model we have given for network flows clearly applies to the situation of currents in an electrical network. It is clear that it also applies to the case of fluid flow in pipes where the result may be stated loosely as the maximum flow through a system of pipes is limited only by the smallest bottleneck. Let us call any flow which satisfies the Kirchhoff flow requirements an algebraic flow. It is not at all obvious that this algebraic flow also applies to information transmission or transportation systems since messages or vehicles do not really split up and recombine at nodes in the nature of an electric current. A more appropriate model for an information and transportation system is as follows.

1) Let a directed network have a capacity assigned to each edge and have two distinguished nodes n_s and n_t.

2) Information can flow over any directed path from n_s to n_t; however, the total amount which can flow on any path cannot exceed the smallest capacity of any edge in that path.

3) Find the maximum of the total flow w, i.e., find the sum of the path flows which can be sustained simultaneously over different paths. This assumes that the amount of flow in each edge, limited by its capacity, is the sum of the flows on those paths which include this edge.

Let us call this the path-flow problem. Although the two flow problems are related, it is not obvious that they are equivalent. We first note by examining Fig. 11 that the two flow problems are not identical. No set of path flows can produce the individual edge flows shown in Fig. 11(a), but they satisfy the algebraic flow constraints. However, the total flow from s to t can always be produced by path flow. Also, the total flow in Fig. 11(b) cannot be produced by paths since it is negative, but it is a permitted algebraic flow. The next result due to Ford and Fulkerson shows how the two flow problems are related.

The Path-Flow and Algebraic-Flow Equivalence Theorem

Corresponding to an allowable path flow of total flow value w, there exists an allowable algebraic flow of total flow value v with $v = w$. Conversely, to any allowable algebraic flow of total flow value v, there corresponds an allowable path flow of total flow value $w = v$ (if $v \geqslant 0$).

The validity of this result can be demonstrated by the following observations.

1) Any path flow satisfies Kirchhoff's Current Law.

2) The superposition of path flows yields an algebraic flow which satisfies Kirchhoff's Current Law.

3) If an algebraic flow has total flow value $v > 0$, then there exists a directed $s - t$ path with positive flow on each of its edges.

Probably the most interesting claim is 3), whose proof really gives the basic ideas of the proof of the max-flow, min-cut theorem also. Consider, therefore, the following constructive proof.

Let a set of nodes X be defined by

a) $n_s \in X$.

b) If $n_i \in X$ and there is a flow $f(i,j) > 0$ on an edge directed from n_i to n_j, then place n_j in X.

c) If $n_t \in X$, then the subnetwork consisting of edges with nonzero flows is connected in that there is a directed n_s, n_t path.

d) If $n_t \notin X$, let $\bar{X}$ be all the nodes of the network that are not in X. Thus. $(X,\bar{X})$ is a cut and by Kirchhoff's Surface Current Law, the fact that $v \neq 0$ implies that at least one edge in $(X,\bar{X})$ has nonzero flow—which contradicts the definition of X.

A similar construction establishes the validity of the max-flow, min-cut theorem. The basic ideas are as follows.

a′) By Kirchhoff's Surface Current Law

$$\max v \leqslant \min c\,(X,\bar{X}).$$

b′) Suppose there is some total flow v which is maximum.[5] Then define a set X recursively as follows:

i) $n_s \in X$.

ii) If $n_i \in X$ and there is an edge from n_i to n_j with $f(i,j) < c(i,j)$, then $n_j \in X$.

iii) If $n_i \in X$ and there is an edge from n_j to n_i with $f(j,i) > 0$, then $n_j \in X$.

iv) Claim $n_t \notin X$. Suppose $n_t \in X$; then there would be an undirected path from n_s to n_t with all positive edges under capacity and all negative edges with nonzero flow. This implies that the total flow could be increased by increasing the flow on this path.

[5]The existence of such a maximum is being presumed here.

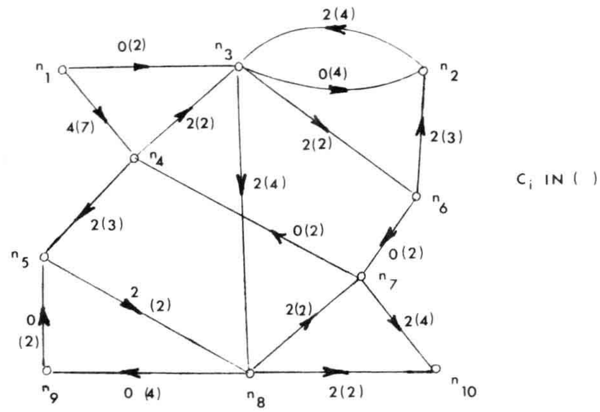

Fig. 12. A network with a feasible flow.

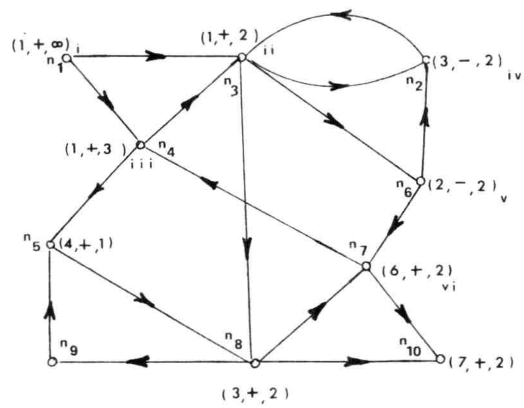

Fig. 13. Result of labeling algorithm for Fig. 12. Subscripts on triples indicate scanning sequence. Flow augmented by two along n_1, n_3, n_2, n_6, n_7, n_{10}.

v) As $n_t \notin X$, consider cut $(X,\bar{X})$. By definition of X and Kirchhoff's Surface Current Law, $v = c(X,\bar{X}) \geqslant \min c(X,\bar{X})$.

A careful examination of this proof reveals that it verifies the max-flow, min-cut theorem for any capacities which are real numbers, viz., it does not assume that the capacities are rational numbers. There remains an important problem which we have not as yet discussed. Implicit in the above contradiction proof is a method to obtain a max-flow by starting with any flow (zero is a possibility) and determining the set X described above. If $n_t \in X$, then there is a ***flow augmenting path***, and the total flow can be increased. If $n_t \notin X$, then this total flow must be maximum as it saturates some cut, and hence must saturate the min-cut. The only computational difficulty here is that we must find the flow-augmenting path. Ford and Fulkerson have also given a labeling algorithm which shows how to construct both the set X and the flow-augmenting path simultaneously.

The Labeling Algorithm

The labeling algorithm recursively assigns labels which are triples to the nodes. The three labels represent, respectively, the node number, a mark to indicate whether or not this node is either the source of an unsaturated edge or the sink of a nonempty edge, and the residual capacity of the portion of the flow augmenting path that has been determined at this stage. In order to accomplish this, we must perform two types of operations. The first involves the determination of what can

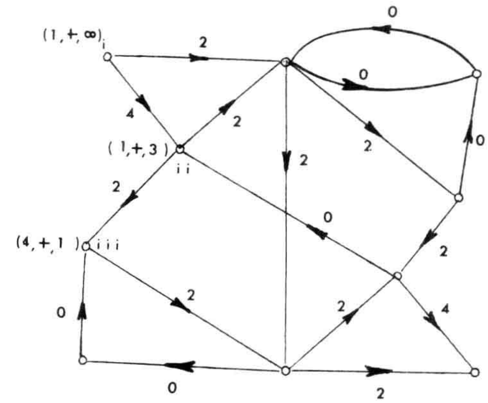

Fig. 14. Result of flow augmentation for Fig. 12 and new labeling indicating max flow.

be accomplished in terms of flow augmentation up to some node; this is called *labeling* a node. The second part involves what can be reached by a single edge from the labeled node; after this "fan-out," the node is called *scanned*. The algorithm can be described in detail as follows. Suppose the nodes of the network are $n_1, n_2, \cdots, n_n$ where n_1 is the source s and n_n is the sink t. Perform the following steps recursively.

Step 1: Label n_1 by $(1, +, \epsilon\ (1) = \infty)$ and call n_1 labeled and unscanned.

Step 2: Select any labeled and unscanned node n_i with label $(k, \pm, \epsilon_i)$.

a) If there is an edge from n_i to n_j, where n_j is unlabeled and

$$f_{ij} < c_{ij},$$

then assign n_j the label $(i, +, \epsilon_j)$ with

$$\epsilon_j = \min\ [\epsilon_i; c_{ij} - f_{ij}]\,.$$

Node n_j is now labeled and not scanned.

b) If there is an edge from n_j to n_i, where n_j is unlabeled and

$$f_{j,i} > 0,$$

then assign n_j the label $(i, -, \epsilon_j)$ with

$$\epsilon_j = \min\ [\epsilon_i; f_{ji}]\,.$$

Node n_j is now labeled and not scanned.

Step 3: After Step 2 is performed as many times as possible the node n_i becomes both labeled and scanned. (We might keep track of this fact by circling the $\pm$ sign.)

Step 4: If n_n never becomes labeled, then the flow is already maximum. When n_n becomes labeled, then both a flow-augmenting path and the value of this flow augmentation have been determined.

The example shown in Fig. 12 serves to illustrate the procedure. It is assumed that the indicated flow has already been obtained, and the capacities are shown by the bracketed numbers. One run of the labeling algorithm produces the assignment shown in Fig. 13. After augmentation, a rerun is shown in Fig. 14; the conclusion is that the flow is now maximum for this example. It should be noted that this algorithm merely shows how to increase any flow that is not maximum. It is possible to use the algorithm to augment some flows ad infinitum without ever reaching the maximum. An example where this happens is given by Ford and Fulkerson [8].[6] A variation of this labeling algorithm has been given recently by Karp and Edmonds [5], and they show that the new algorithm always converges.

To conclude this discussion of the labeling algorithm, we note an important corollary due to Ford and Fulkerson. First note that Ford and Fulkerson [8] did show that their algorithm converges if the capacities are rational numbers.[7] Furthermore, they show that integer capacities enable integer augmentations. Hence we have the following.

The Integrity Theorem of Ford and Fulkerson

If the capacities are all integers, then there is an integer max-flow value which corresponds to individual edge flows that are all integers.

Network Vulnerability

In this section, we assume that a network will be represented by a connected, undirected graph G, and we are interested in examining the possibilities for disconnecting the graph. A set of edges whose removal disconnects a connected graph will be called an *edge disconnecting* set; the size of the smallest edge disconnecting set is called the *cohesion*-$\lambda(G)$. A complete summary of the results related to vulnerability is given in the survey paper by Wilkov [15]. Now the failure of a node can be thought of as removal of that node together with all of its incident edges. A *node disconnecting* set is a set of nodes whose removal disconnects a connected graph; the size of the smallest node disconnecting set is called the *connectivity*-$\omega(G)$. In order to make $\omega(G)$ well defined for all graphs, we must discuss the case of a *complete* graph, i.e., a graph which has an edge between each pair of nodes. Clearly, we cannot disconnect such a graph by removing less than $n - 1$ nodes; the removal of $n - 1$ nodes creates an isolated node, but the definition of connectedness given earlier does not apply. In this case of the complete graph, we define the connectivity to be $n - 1$. This definition will provide consistent results in the basic properties of ω and λ. We shall now discuss two basis problems, namely, analysis and synthesis.

Synthesis of Invulnerable Graphs

Perhaps the most basic result on cohesion and connectivity is that it never requires more nodes than edges to disconnect a graph,[8] and we can always disconnect a graph by removing all the edges incident at any node. Hence,

$$n - 1 \geqslant \omega \leqslant \lambda \leqslant \text{degree of any node.}$$

[6]Some further pathological examples have been given by Zadeh [16].

[7]At first it might seem that only rational capacities are of any practical significance. An extremely important practical implication of the nonconvergence for irrational capacities is given in the Karp and Edmonds paper [5].

[8]Although intuitively clear, this is not trivial. A proof is given in Frank and Frisch [7].

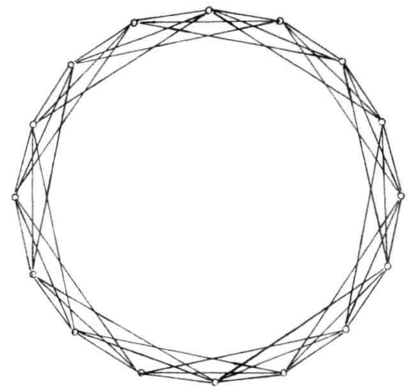

Fig. 15. A simple invulnerable graph.

We can obtain another inequality by observing that if we add all the degrees of the nodes, then we are counting each edge exactly twice. Hence,

$$2e = \sum_{i=1}^{n} d_i \geqslant nd_{\min} \cdot$$

This yields

$$\lambda \leqslant d_{\min} \leqslant 2e/n \cdot$$

Thus, for specified numbers of nodes and edges, the most invulnerable graph which could possibly exist would satisfy

$$\omega = \lambda = [2e/n]$$

where $[x]$ is a symbol that denotes the integer part of a real number, i.e., $[x]$ is the largest integer which is not greater than x. It has been shown that such graphs always do exist; however, a complete documentation of all such graphs has not yet been achieved [1], [2], [10]. A rather general class of graphs have been shown to be maximally invulnerable; they may be described as follows. Lay out n nodes equally spaced on the circumference of a circle, and label them clockwise with the integers $0, 1, 2, \cdots, (n-1)$. Let the arc length between any two consecutively numbered nodes be unity. Then to construct a graph of connectivity $2k$ with kn edges, connect every pair of nodes that have a corresponding arc length less than or equal to k; this construction is illustrated in Fig. 15 for $n = 15$ and $k = 3$. The case of odd connectivities can be handled by adding diameters to the circle. The general case may be described by connecting nodes whenever there is an arc between them of any of the following lengths: $1 = b_1 \leqslant b_2 \leqslant b_3, \cdots, \leqslant b_k$ where $b_k < n/2$. It has been shown that the resulting graph is maximally invulnerable if the sequence of successive differences of the b_i is nondecreasing.[9] For example, (1, 2, 4, 6, 9) defines an invulnerable graph, whereas the graph corresponding to the sequence (1, 2, 4, 5, 9) might not be an invulnerable graph. A proof of the invulnerability of graphs for the simple case, i.e., the sequence $(1, 2, 3, 4, \cdots, k)$, is given in the Appendix. Extensions to the directed case are given by Frisch and Malek-Zaverei [9].

[9]Such sequences are sometimes called convex.

Vulnerability Analysis

Clearly, an important problem is to determine the values of ω and λ for a given graph. One method for doing this is to introduce the concept of local cohesion and connectivity and then apply the network flow ideas. This may be accomplished as follows. Let ω_{ij} be the smallest number of nodes which must be removed in order to disconnect nodes i and j; likewise, λ_{ij} is the smallest number of edges which must be removed to disconnect nodes i and j. Of course, ω_{ij} is only defined when nodes i and j are not connected by a single edge. It is not difficult to verify that

$$\omega_{ij} \leqslant \lambda_{ij} ,$$

$$\omega = \min \omega_{ij} ,$$

and

$$\lambda = \min \lambda_{ij} \cdot$$

The fact that λ_{ij} can be calculated by the network flow methods should be obvious. All that is required is to define all capacities as unity, and then observe that a minimum $i-j$ disconnecting set is a min-cut and that any min-cut is a minimum $i-j$ disconnecting set.[10] It should be noted that the integrity theorem verifies that the flows are either zero or one and can be broken down into simple paths of unit flow. Furthermore, the labeling algorithm can be used to find the values. It can also be shown that it is not necessary to perform the labeling algorithm for each possible pair of nodes in order to obtain λ [7], [8].

The case of ω_{ij} can be handled via a trick which converts the node problem into an edge problem. The idea is to split each node into two copies of itself and connect these by an edge of unit capacity. All other edges are allowed to have infinite capacity. The problem for ω_{ij} is thereby converted to a λ_{ij} problem. Again it can be shown that a considerable amount of effort can be saved by refining this basic concept. The details can be found in Frank and Frisch [7]. Theoretical results which emerge from this conversion to a flow problem are the following.

1) λ_{ij} equals the maximum number of edge disjoint paths which connect i and j.

2) ω_{ij} equals the maximum number of node disjoint paths which connect i and j.

Connectedness

Another possibility for measuring the vulnerability of a network is to ascertain that the connectivity is no worse than some desired value. Namely, a graph is said to be k-connected if at least k nodes must be removed in order to disconnect the graph, i.e., $\omega \geqslant k$. An algorithm to show that $\omega \geqslant k$ for some given value of k that is simpler than the determination of ω was given by Kleitman [12]. The algorithm is described below for the case $\omega \geqslant 4$.

[10] This equivalence is due to the minimization. Disconnecting sets and cuts are not generally identical.

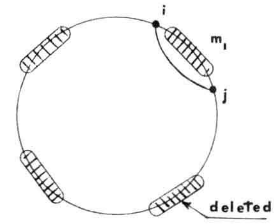

Fig. 16. A disconnecting set.

1) Choose any node n_1 and determine if there are four node disjoint paths from n_1 to all other nodes.

2) If not, then $\omega < 4$. If yes, remove n_1 from G to form G_1.

3) Choose n_2 in G_1 and see if there are three node disjoint paths from n_2 to all other nodes of G_1.

4) If not, then $\omega < 4$. If yes, remove n_2 to form G_2.

5) Choose any n_3 in G_2 and see if there are two node disjoint paths from n_3 to all the other nodes of G_2. If not, $\omega < 4$. If yes, remove n_3 to form G_3.

6) If final graph G_3 is connected, then $\omega \geqslant 4$.

A more recent result on algorithms for connectivity was given by Tarjan [14].

Appendix

Proof of Optimality of the Basic Invulnerable Graphs

Clearly, there is always a cycle through all the nodes. Thus, a disconnecting set can be pictured as shown in Fig. 16, where the disconnecting set is portrayed as being divided into consecutive sequences of removed nodes. Now if the consecutive sequence of nodes m_1 has less than k nodes, then the node i which precedes m_1 is adjacent to the node j which follows m_1. Hence, in order to disconnect the graph, at least two m_i must contain k or more nodes. Thus,

$$\omega \geqslant 2k \cdot$$

However, as all nodes have degree $2k$,

$$\omega \leqslant 2k \cdot$$

References

*[1] F. T. Boesch and A. P. Felzer, "A general class of invulnerable graphs," *Networks*, vol. 2, pp. 261–283, 1972.

[2] F. T. Boesch and R. E. Thomas, "On graphs of invulnerable communication nets," *IEEE Trans. Circuit Theory*, vol. CT-17, pp. 183–192, May 1970.

[3] S. E. Dreyfus, "An appraisal of some shortest-path algorithms," *Oper. Res. Soc. Amer.*, vol. 17, pp. 395–412, May–June 1969.

[4] J. Edmonds and E. L. Johnson, "Matching, Euler tours, and the Chinese postman problem," *Math. Programming*, vol. 5, no. 1, pp. 84–124, 1973.

*[5] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *J. Ass. Comput. Mach.*, vol. 19, no. 2, pp. 248–264, 1972.

*[6] H. Frank and I. T. Frisch, "Network analysis," *Sci. Amer.*, vol. 223, pp. 94–103, July 1970.

[7] ——, *Communication, Transmission and Transportation Networks*. Reading, MA: Addison-Wesley, 1971.

*Indicates papers included in this volume.