Network Design and Transportation Planning: Models and Algorithms*

T. L. MAGNANTI†

Massachusetts Institute of Technology, Cambridge, Massachusetts

R. T. WONG

Purdue University, West Lafayette, Indiana

Numerous transportation applications as diverse as capital investment decision-making, vehicle fleet planning, and traffic light signal setting all involve some form of (discrete choice) network design. In this paper, we review some of the uses and limitations of integer programming-based approaches to network design, and describe several discrete and continuous choice models and algorithms. Our objectives are threefold—to provide a unifying view for synthesizing many network design models, to propose a unifying framework for deriving many network design algorithms, and to summarize computational experience in solving design problems. We also show that many of the most celebrated combinatorial problems that arise in transportation planning are specializations and variations of a generic design model. Consequently, the network design concepts described in this paper have great potential application in a wide range of problem settings.

1. INTRODUCTION

As broadly construed, network design is a topic that captures many of the most salient features of transportation planning. Indeed, network design issues pervade the full hierarchy of strategic, tactical, and operational decision-making situations that arise in transportation. At the

* Presented at the Frontiers in Transportation Equilibrium and Supply Models Conference, Montreal, November 1981, and in revised form at NETFLOW '83, Pisa, Italy, March 1983.

† Supported in part by the National Science Foundation under Grant 79-26625-ECS and in part by the FHWA under Contract DTFH61-81-C-00084.

Transportation Science
Vol. 18, No. 1, February 1984

0041-1655/84/1801-0001 $01.25
© 1984 Operations Research Society of America

highest, most aggregate level of decision-making, network design choices do much to determine the effectiveness of strategic, long-term transportation planning. One hardly needs to reflect on the importance of strategic capital investment decisions involving highways, airports, fleet acquisitions, and mass transit, among others, and the role of these decisions in defining the infrastructure of transportation system networks and, hence, the resources that will be available for transportation planning in general. Tactical, intermediate term network design decisions are essential as well, though for planning for the effective use, rather than the acquisition, of these resources. Setting one-way street assignments in urban street networks or determining the best location for a company's warehouses would be examples of such tactical decisions. And, finally, network design decisions arise in certain operational, short-term planning problems, such as the setting of street light signals or the scheduling of repairs on urban streets.

Because many of these design decisions involve choices from a discrete set of alternatives, the design problems can often be cast in the form of integer or mixed integer programming models. The essence of these models can be stated quite simply. We are to choose those arcs (e.g., roadways or railbeds) to include in, or add to, a transportation network accounting for the effects that the design decision will have on the operating characteristics of the transportation system. This type of model is capable, at least in principle, of considering the system-wide interactions between design decisions and of modeling how design decisions affect the operations of a transportation network. In particular, the models can consider trade-offs between various design alternatives, striking a balance between increased design expenditures and resulting improvements in the system's operation (e.g., decreases in operating costs).

In this paper, we will consider both modeling and algorithmic development for these integer programming-based network design models. Our objectives are threefold—to present a unified view of modeling of network design problems, to propose a unifying framework for describing many network design algorithms, and to summarize computational experience in solving design problems.

First, we describe a general discrete choice model of network design, reviewing some of the features captured by the model and some of its specializations. As part of this discussion, we show how this general model is intimately related to many of the most renowned discrete choice problems encountered in transportation planning, including the minimal spanning tree problem, shortest path problem, traveling salesman and vehicle routing problems, and the traffic equilibrium problem. Next, we describe some of the integer programming algorithms and heuristic methods that have been developed for solving various versions of the design problem. In particular, we show that many of the approximation, or bounding, schemes that have been devised for solving this type of discrete optimization problem can be viewed in a unified way. We also describe several criteria for evaluating the effectiveness of heuristic algorithms as well as the application of these performance criteria to network design heuristics. Third, we summarize algorithmic experience in applying these optimization-based and heuristic algorithms to the network design problem. In the last section we also discuss several models and algorithmic approaches for related convex and concave cost problems in which design decisions are not limited to a discrete set of choices.

Before undertaking these objectives, we might pause briefly to comment on the potential uses and the limitations of these discrete choice models, and to contrast how discrete choice design problems in transportation compare with similar types of models that arise in other contexts such as communication planning.

Scope and Limitations

Although discrete choice integer programming models are generally ideal for addressing the type of combinatorial complexities and interaction effects that arise in network planning, they are generally not well-suited for dealing with the risk and uncertainties that are inherent in many strategic decision-making situations. In these instances, other planning tools, such as simulation, decision analysis, or multiattribute utility theory, are attractive alternatives to integer programming analysis. This is particularly true when evaluating the merits of any single network investment project such as the introduction of a new subway line, or the choice of size and location of a new airport. In these instances, social, political, and regional development effects are likely to predominate. Often, the effects of these new capital investments on the transportation system as a whole can be viewed adequately from such an aggregate point of view that there is no need for detailed network analysis. These factors help to explain the limited use of integer programming type models for transportation investment decision-making and suggest that other types of analyses might be more appropriate. For example, KEENEY AND RAIFFA[63] have described the use of multiattribute utility theory in analyzing a potential new location for the Mexico City airport. BEESLEY[6] has described the use of cost-benefit and social benefit analysis to study the possibility of constructing an underground railroad in London. WILSON AND HUDSON[118] have described the use of simulation analysis for planning in the Canadian National Railroad. These examples are meant merely to be illustrative. MANHEIM[81] gives a much more thorough guide to the literature, particularly concerning economic and social impact analysis.

On the other hand, integer programming-type models are likely to be much more attractive for situations in which a variety of different investment decisions are being considered simultaneously, so that project selection becomes a major issue. For example, when introducing a new transportation infrastructure in a developing country, the range of possible alternatives for designing a highway system might be so enormous that a case-by-case analysis of each alternative becomes prohibitive. In these instances, using aggregate point estimates for demand and for design and routing costs might suffice for decision-making and an integer programming model becomes attractive, especially if augmented by extensive sensitivity analyses and probationary what-if type of analyses aimed at addressing underlying risks and uncertainties. Or, integer programming could be used to generate potential investment strategies that could then be tested by a simulation analysis. Similarly, simulation could be used to screen several potential projects and integer programming would subsequently be used to evaluate both the trade-offs between the most attractive of these projects and the projects' effects upon network performance. This use of integer programming models as a screening and evaluation tool seems, as yet, to be far from fully exploited in the context of transportation planning.

Moreover, network design decisions often arise in ways that do not involve capital investment decision-making at all. In fact, some of the most attractive uses of network design models at a tactical and strategic planning level are in supporting decisions concerning the type of transportation services to be provided, rather than the physical construction of the transportation facilities. In this setting, we view the underlying transportation network not as a physical system, but rather as a schedule map. Providing airline service between New York and London, for example, or providing service on a particular bus route would be viewed as adding an arc (e.g., the New York-London arc) to the service network. When introduced, this service arc then becomes available for passenger flow. In this sense, we are "constructing" a network. Decisions concerning the location and sizing of warehouses, which often are rental decisions, give rise to node choice design models with similar characteristics. Because these service decisions are more readily reversible, the risks and uncertainties in this context are often considerably less than those in capital investment decision-making. Also, service decisions typically have network-wide impact. As a consequence, transportation planners are much more apt to adopt discrete choice network design models when designing service networks. Therefore, we encourage the reader to broadly interpret any reference to network design or construction decisions in our discussion as including service choice decisions as well.

Transportation Planning vs. Other Application Areas

Most of the transportation models and algorithms that we consider in this paper apply equally well to communication system planning, water resource planning, distribution planning, and other application domains. Even though the problem settings might be very different, the underlying mathematical models are often much the same. Nevertheless, some of the differences do affect the use of the models.

1. *Time Scale.* In general, transportation and water resource design decisions have long-term effects. In contrast, communication system designs and distribution system designs frequently are more readily altered. For this reason, uncertainties, rather than combinatorial effects, are often more predominate in transportation and water resource applications and, consequently, integer programming models are often more attractive in other settings. Again, designing transportation service networks would be a notable exception.

2. *Supply-Demand Effects.* Typically, constructing new highways or a new subway line has a pronounced effect upon land usage and, as a result, on demand patterns for transportation services. In addition, transportation design decisions are often instrumental in regional planning efforts. Boyce and other authors in the volume that he has edited (BOYCE[10]), MACKINNON,[74] and STEENBRINK[110, 111] discuss these issues and their impact on transportation planning. In general, these effects imply that social and political considerations are likely to have a pronounced impact on many transportation network design issues. In contrast, these supply and demand effects are likely to be less pronounced in communication and distribution systems. Consequently, integer programming models are likely to capture more of the essence of the underlying issues in these other problem settings.

3. *Queuing and Congestion Effects.* Buffers, queues, and queuing delays are prevalent at the nodes of a communication system. The users of transportation networks, though, spend relatively more time on the links of the system (rail freight is a notable exception). Consequently, communication systems are more likely to require nonlinear models to represent congestion effects adequately. The same type of congestion effects applies in certain transportation models as well, particularly link delays in public sector urban planning models.

4. *User vs. System Optimization.* In most multiuser communication systems, the system retains control over route choice, node operating protocol, and other operating decisions. Some transportation systems,

particularly freight or distribution systems, might be similar. On the other hand, in most passenger service systems (e.g., urban street networks, airline systems), the users exercise considerable control over their own routing. In these instances, design models must include user behavioral models, the well-known traffic equilibrium model being a noted prototype.

5. *Operating Characteristics.* Many transportation systems are subject to substantial variability in demand, particularly due to morning and evening peak periods. Communication, distribution, and water resource systems face the same phenomenon, though typically they either experience less variations in demand during large periods of the day, or have greater flexibility for controlling and smoothing demand (e.g., off-peak pricing, system optimal routing, back ordering, and so forth). For this reason, any design of a transportation system will usually provide substantial excess capacity during most of the system's operations.

These comparisons highlight the fact that problem domains might emphasize different aspects of a network's design and operating characteristics and that various network design models might require different types of data. Even so, the basic models and algorithms that we discuss in subsequent sections apply to all of these application contexts.

2. NETWORK DESIGN MODELS

IN THIS SECTION, we describe a general version and several specializations of a generic discrete choice network design model. The general formulation not only models a variety of network design issues, but also serves as an umbrella for integrating a number of related transportation models. Indeed, many of the most well-known and intensively studied problems encountered in transportation planning can be viewed as specializations and variations of this generic design model. This observation not only suggests that algorithmic strategies for solving special cases of the model, or even the general model itself, might be derived from available methods for other special cases, but also highlights prospects for synthesizing and unifying methods for solving a variety of related problems. Later in this paper we comment further on these possibilities.

A General Model

The basic ingredients of the model are a set N of nodes and a set A of arcs that are available for designing a network (unless otherwise noted we assume that arcs are directed). The selection of those arcs to be included in the network depends upon the interplay between the design and operating decisions, particularly trade-offs inherent in situations involving both fixed design costs and variable operating costs.

The model permits multiple commodities. These might represent distinct physical goods, or the same physical good, but with different points of origin and destination (for example, a passenger traveling from Boston to New York is a commodity distinct from a passenger traveling from Atlanta to Chicago). We let κ denote the set of commodities and for each $k \in \kappa$, let R_k denote the required amount of flow of commodity k to be shipped from its point of origin, denoted $O(k)$, to its point of destination, denoted $D(k)$.

The model contains two types of variables, one modeling discrete choice design decisions and the other modeling continuous flow decisions. Let y_{ij} be a binary variable that indicates whether ($y_{ij} = 1$) or not ($y_{ij} = 0$) arc (i, j) is chosen as part of the network's design. Let f_{ij}^k denote the flow of commodity k on arc (i, j). Then if $y \equiv (y_{ij})$ and $f \equiv (f_{ij}^k)$ are vectors of design and flow variables, the model becomes

$$\text{minimize } \phi(f, y) \tag{2.1}$$

subject to:

$$\sum_{j \in N} f_{ij}^k - \sum_{l \in N} f_{li}^k = \begin{cases} R_k & \text{if } \ i = O(k) \\ -R_k & \text{if } \ i = D(k) \quad \text{all} \quad k \in \kappa \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

$$f_{ij} \equiv \sum_{k \in \kappa} f_{ij}^k \leq K_{ij} y_{ij} \qquad \text{all} \quad (i, j) \in A \tag{2.3}$$

$$(f, y) \in S \tag{2.4}$$

$$f_{ij}^k \geq 0, \quad y_{ij} = 0 \quad \text{or} \quad 1 \qquad \text{all} \quad (i, j) \in A, \quad k \in \kappa. \tag{2.5}$$

When the arcs in A are undirected, the problem is modeled in much the same way; simply restrict the indices to $i < j$, remove the non-negativity condition imposed upon the flow variable, and impose the additional contraints $-f_{ij} \leq K_{ij} y_{ij}$.

When the objective function $\phi(f, y)$ in this formulation is linear,

$$\phi(f, y) = \sum_{k \in \kappa} \sum_{(i,j) \in A} c_{ij}^k f_{ij}^k + \sum_{(i,j) \in A} F_{ij} y_{ij}, \tag{2.6}$$

the model becomes a linear mixed integer program. In this case, the per unit arc routing costs c_{ij}^k for commodity k and the fixed arc design costs F_{ij} must be defined commensurably. Most frequently, assuming that demand and flows would remain (approximately) unchanged over the lifetime of the network's design, we would choose the flow costs as net present values of the per unit routing costs evaluated over the network's lifetime.

Nonlinear versions of the objective function could be used to model congestion effects or to model diseconomies other than fixed costs. For example, the well-known Bureau of Public Roads formula $t_{ij}f_{ij}[1 + \alpha_{ij}(f_{ij}/K_{ij})^{\beta_{ij}}]$ or the well-known queuing delay formula $t_{ij}f_{ij}/(K_{ij} - f_{ij})$, with parameters t_{ij}, α_{ij} and β_{ij}, would model flow delays (or costs) on each arc (i, j) due to congestion. Or, any concave function $\theta(f)$ would model diseconomies of scale in the network's flow.

Constraints (2.2) imposed upon each commodity k in the formulation (2.1)–(2.5) are the usual network flow conservation equations. The "forcing" or "bundle" constraints (2.3) state that the total flow f_{ij} of all commodities on arc (i, j) cannot exceed the capacity K_{ij} of the arc if it is chosen as part of the network design, i.e., $y_{ij} = 1$, and must be zero if arc (i, j) is not chosen as part of the design, i.e., $y_{ij} = 0$. The set S includes any side constraints imposed either singly or jointly on the flow and design variables. These constraints might, for example, model topological restrictions imposed upon the configuration of the network, including precedence relations (e.g., choose arc (i, j) only if arc (r, s) is chosen, i.e., $y_{rs} \le y_{ij}$) or multiple choice relations (choose at most [at least, exactly] two of some set $\{(i, j), (p, q), (r, s)\}$ of arcs, i.e., $y_{ij} + y_{pq} + y_{rs} \le 2$) or the side constraints might model limitations imposed upon resources shared by several arcs, an especially important version being a budget constraint:

$$\textstyle\sum_{(i,j)\in A} e_{ij}y_{ij} \le B. \tag{2.7}$$

The coefficient e_{ij} in this expression is the cost incurred if arc (i, j) is chosen (constructed) in the network design. The constraint itself, which is often used in lieu of imposing design costs in the objective function, states that the total expenditure for designing the network is limited by a budget B.

Additional side constraints of the form $y_{ij} = 1$ would specify arcs (i, j) that are necessarily included in the network's design. When prior specifications of this type have been made, the formulation (2.1)–(2.5) is often referred to as a network improvement model.

Modeling Variations

Like most integer programming problems, the network design problem can be modeled in a variety of ways. As we note in the next section, some of these alternatives might be preferred for algorithmic purposes. In particular, for unconstrained network design problems in which the capacity K_{ij} for any arc exceeds any possible flow on this arc, the forcing constraints (2.3) are equivalent to the more disaggregated constraints

$$f_{ij}^k \le K_{ij}y_{ij} \quad \text{for all} \quad (i, j) \in A, \quad k \in \kappa.$$

Both versions of these constraints force each f_{ij}^k for $k \in \kappa$ to be zero if

$y_{ij} = 0$ and become redundant if $y_{ij} = 1$. Surprisingly, the seemingly less efficient disaggregate formulation, which contains far more constraints, leads to more efficient algorithms. The reasons for this are twofold. First, many techniques for solving integer programs like the network design problem first solve the linear programming relaxation of the model (i.e., replace $y_{ij} = 0$ or 1 with $0 \leq y_{ij} \leq 1$). Because the linear programming version of the disaggregate formulation is more tightly constrained than is the linear programming version of the aggregate formulation, the disaggregate linear program provides a sharper lower bound on the value of the integer programming formulation. That is, it better approximates the integer program. Second, because the disaggregate linear programming formulation has more constraints, it has a richer collection of linear programming dual variables. The enhanced set of dual variables provides more flexibility in algorithmic development. Indeed, these two advantages of the disaggregate formulation are intimately related. MAGNANTI AND WONG[79] discuss this issue in some detail.

We might note that our assumption of a single design level K_{ij} for each arc is not restrictive. Introducing parallel arcs permits us to model multiple capacity levels.

For other alternative formulations of the network design problem, see MALEK-ZAVAREE AND AGGARWAL,[80] PETERSON[101] and RARDIN AND CHOI.[102]

Special Cases

The importance of the network design problem (2.1)–(2.5) stems not only from its practical significance as a design tool, but also from the impressive range of problems that it models. Indeed, as Table I demonstrates, many of the most noted problems that arise in network optimization are either special versions or close relatives of the network design problem.

All but the last of the examples in this table assume that the objective function is linear as in (2.6) and that all flow costs c_{ij}^k and design costs F_{ij} are non-negative. Several of the examples assume that the design problem is uncapacitated; that is, $K_{ij} > \sum_{k \in K} R_k$. This assumption applies frequently in practice when designing systems that will operate far enough below capacity so that congestion effects become negligible. In this instance, the ijth bundle constraint (2.3) becomes redundant whenever $y_{ij} = 1$. Without side constraints, this unconstrained, linear cost version of the problem is often referred to as the *fixed charge design problem.* With the budget side constraint (2.7) and no fixed costs in the objective function, this basic uncapacitated problem is often called the *budget design problem.*

TABLE I

Specializations and Variations of the Network Design Model

Demand Structure $O(k)$, $D(K)$, R_k	Objective Function $\phi(f, y)$	Capacities K_{ij}	Side Constraints S	Problem Type
Complete (undirected network)	Linear in design variables, no flow costs	Uncapacitated	None	Minimal spanning tree
Arbitrary	Linear in flows, no design costs	Uncapacitated	None	Shortest path
Complete on a subset of nodes (undirected network)	Linear in design variables, no flow costs	Uncapacitated	None	Steiner tree problem
Arbitrary	(Non)Linear flow costs, no design costs	Arbitrary	None	(Nonlinear cost) Multicommodity flow problem
Single source	Linear in design variables, no flow costs	Uncapacitated	None	Minimal directed spanning tree
Complete	Linear in design and flow variables, large constant fixed costs	Uncapacitated	None	Traveling salesman problem
Single source	Linear design variables, no flow costs	Fixed capacity on all arcs	Assignment constraints on design variables	Vehicle routing problem
Arbitrary	Linear in flow variables, fixed costs on split nodes	Capacities on split nodes	None	Facility location problem
Arbitrary	Linear in flows and design variables	Uncapacitated	None	Fixed charge network design problem
Arbitrary	Linear in flows, no design costs	Uncapacitated	Budget constraint on design costs	Budget design problem
Arbitrary	Arbitrary	Uncapacitated	Minimum cost route choice for each commodity	Network design traffic equilibrium problem

When the network is capacitated, the flow problem in the variable f becomes a multicommodity flow problem for any choice of the design variables. In particular, if the problem contains no side constraints and no fixed costs, so that choosing $y_{ij} = 1$ for all $(i, j) \in A$ is optimal, the design problem itself reduces the often elusive multicommodity flow problem.

The remaining examples show the intimate connection between the network design problem and several other mainstays of transportation network analysis—the shortest path problem, vehicle routing (and the traveling salesman problem), facility location models, and traffic equilibrium—as well as the connection with several basic combinatorial optimization problems that arise in communication, transportation, and other applications.

Minimal Spanning Trees and Shortest Paths

Consider the fixed charge design problem (i.e., uncapacitated arcs, linear costs, and no side constraints) defined on an undirected network. Then if all routing costs c^k_{ij} are zero and the demand pattern is complete (i.e., there is a demand for travel between every pair of nodes), the minimum cost solution must be a spanning tree (otherwise delete an arc from any cycle, saving some fixed cost); therefore, the problem reduces to a minimal spanning tree problem. On the other hand, if all fixed costs F_{ij} are zero, it is always best to set each $y_{ij} = 1$ and the resulting fixed charge design problem reduces to a shortest path problem for each commodity (the network could be directed, as well, for this case). Consequently, whenever either routing costs or design costs are dominant and the network is undirected and uncapacitated, the problem becomes relatively easy to solve by extremely efficient algorithms. It is only when both fixed and variable costs contribute in a substantive way that the undirected problem becomes difficult to solve.

We might note that minor, apparently innocuous, modifications to the design could make it far more formidable to solve. For example, the completeness assumption on demands is essential for reducing the uncapacitated design problem to a minimal spanning tree problem. If the demand pattern is complete on a subset of nodes and there are no routing costs, then the optimal solution will be the minimum cost spanning subtree containing this subset of nodes together with other nodes that might, but need not, be chosen. That is, the problem becomes the computationally elusive Steiner tree problem (BEASLEY,[5] CLAUS AND MACULAN,[14] HAKIMI,[52] WONG[121]).

As another example, suppose that we consider a directed, rather than undirected, fixed charge design problem with no routing costs. If the demand is complete, the problem does not appear to reduce to any easily identifiable, or solvable, special case. And yet, if the problem contains a single source node, the optimal solution will be a minimum cost spanning tree directed out of the source node (also called an optimal branching problem). Although more difficult than a minimal spanning tree problem on an undirected network or a shortest path problem, this minimal directed tree problem can be solved efficiently as well (EDMONDS[26]).

The lessons from these observations are worth noting. Assumptions concerning directed vs. undirected arcs, complete demand vs. incomplete demand, and single vs. multiple sources can profoundly affect our ability to solve network design problems.

Traveling Salesman and Vehicle Routing Problems

Consider a fixed charge design problem with a complete demand pattern, with a large fixed cost $F_{ij} = F$ defined on every arc (i, j) of A, and with given values of the flow costs $c_{ij}^k = c_{ij}$, the same for all commodity types. Since the demand pattern is complete, any feasible design must contain at least one arc directed into each node, i.e., at least $|N|$ arcs. But for fixed routing costs if the fixed cost F is sufficiently large, the optimal solution will, if at all possible, use exactly $|N|$ arcs. That is, the optimal solution will be a Hamiltonian tour of the nodes. Moreover, if the demand between every pair of nodes is the same, each arc on the tour will contain the same flow, independent of the chosen tour. But then, since $|N| \cdot F$ is a constant, the cost minimizing solution will be a minimum cost Hamiltonian tour with respect to the routine costs c_{ij}. *That is, the network design problem becomes equivalent to solving a traveling salesman problem on the network* (N, A) *with costs* c_{ij}.

Next, consider a problem with a single source, node s, and with a fixed capacity, $K_{ij} = K$, on all arcs. In addition, suppose that we impose the following assignment constraints on the design variables:

$$\textstyle\sum_{j\in N} y_{ij} = 1 \quad \text{for all} \quad i \in N\backslash\{s\}$$

$$\textstyle\sum_{i\in N} y_{ij} = 1 \quad \text{for all} \quad j \in N\backslash\{s\}$$

as well as the constraint

$$\textstyle\sum_{j\in N} y_{sj} \le NV.$$

Then the linear cost network design problem becomes a vehicle routing problem for a homogeneous fleet of NV vehicles, each domiciled at the source node s and each having capacity K. In this instance, we could interpret the design variable y_{ij} as specifying whether ($y_{ij} = 1$) or not ($y_{ij} = 0$) any vehicle travels from node i to node j. The added assignment constraints state that exactly one vehicle enters and leaves each node.

The forcing constraint (2.3) states that the flow on arc (i, j) cannot exceed any vehicle's capacity and, furthermore, that this flow must be zero if no vehicle travels on arc (i, j), i.e., $y_{ij} = 0$.

This formulation can be traced to GARVIN et al.[43] GAVISH AND GRAVES[44] have subsequently developed a more aggregate version of the formulation in terms of the flow variables f_{ij}. To obtain their formulation, we can add constraints (2.2) over $k \in \kappa$ and substitute $f_{ij} = \sum_{k \in \kappa} f_{ij}^k$ in the resulting constraints. Note that if $NV = 1$ and the problem is uncapacitated, the problem again reduces to the traveling salesman problem.

Facility Location Problems

In many facility location problems, the design decisions concern the location and sizing of facilities (e.g., warehouses, plants, hospitals) at nodes of a distribution or transportation network. To convert these problems into the form of a network design model, we can use a familiar node "splitting" device—replace any node j that is a candidate for a facility by two nodes, a "receiving" node j' and a "sending" node j''. Also, replace any arc (i, j) with an arc (i, j'), any arc (j, k) with an arc (j'', k), and add arc (j', j'') to the network. Associate the costs of (i, j) with (i, j') and the costs of (j, k) with (j'', k), and identify the fixed and variable throughput costs and the capacity of a potential facility at node j with arc (j', j''). Then the design variable $y_{j'j''}$ associated with arc (j', j'') indicates whether or not we decide to locate a facility at node j.

At times an even simpler transformation is possible. For example, whenever only one arc is directed into (out of) a node that is a potential location for a facility, we need not rely on node splitting. We merely associate the node design variable with the arc directed into (out of) that node. As an illustration, consider a warehouse location problem in which warehouses, which can be placed at several potential locations, are to supply several customers. The objective is to minimize the sum of the fixed charges for opening various warehouses and the routing costs for supplying customers from the warehouses that are opened. To convert this problem to a network design model like (2.1)–(2.5), we add a special node to the warehouse location network. This node will be the source of all flow required by the customers. We also add a set of special arcs from the special node to the potential warehouse sites (see Figure 1). The special arc directed into any warehouse site has a construction cost equal to the fixed charge associated with opening the warehouse and has no routing cost. The remaining arcs in the network have no construction cost, but have routing costs equal to the transportation costs from the warehouses to the customers. Solving the resulting network design problem solves the warehouse location problem.

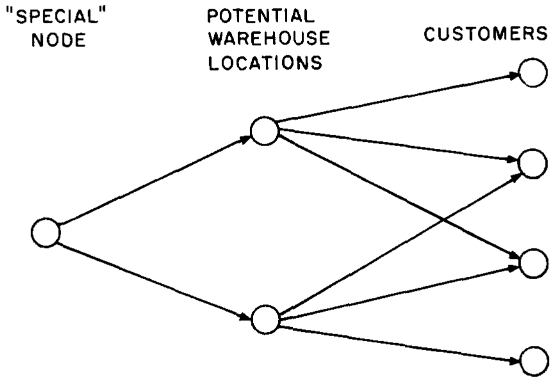

Fig. 1. Plant location as an arc design problem.

Several variants of this transformation are possible. For example, to model warehouse capacities, we add capacities to the special arcs. To model the so-called K-median problem in which we are to choose at most K of the warehouse locations, we add a budget constraint with $e_{ij} = 1$ for all the special arcs, $e_{ij} = 0$ otherwise, and select $B = K$ as the budget limitation.

These transformations invite comparison between the vast literature devoted to facility location problems and the network design literature. The recent survey of network location problems by TANSEL et al.[113] might serve as a starting point for making such comparisons.

Network Design and Traffic Equilibrium

In order to add traffic equilibrium conditions to the design model, we assume that the network is uncapacitated and that the set S is given implicitly as those flow vectors f that are solutions to the system

$$c_{ij}^k(f, y) + w_i^k - w_j^k \geq 0 \quad \text{for all} \quad i, j, k \tag{2.8}$$

$$[c_{i,j}^k(f, y) + w_i^k - w_j^k] f_{ij}^k = 0 \quad \text{for all} \quad i, j, k \tag{2.9}$$

$$w_{0(k)}^k \equiv 0 \quad \text{for all} \quad k$$

for some choice of the variables w_j^k. In these expressions, $c_{ij}^k(f, y)$ denotes the cost for commodity k, as a function of the full vectors f and y of the network flow and design variables, of traversing arc (i, j). To see the connection between these conditions and the more standard formulation of network equilibrium, suppose that P_k is *any* path connecting the origin $O(k)$ and destination $D(k)$ of commodity k. Adding the inequalities (2.8) for arcs (i, j) in P_k and noting that the terms involving the w_j^k telescope, gives

$$\textstyle\sum_{(i,j)\in P_k} c_{ij}^k(f, y) - w_{D(k)}^k \geq 0. \qquad (2.10)$$

Moreover, if $f_{ij}^k > 0$ for every arc (i, j) on P_k, then from (2.9) each of the inequalities in (2.8) for $(i, j) \in P_k$ becomes an equality as does the derived expression (2.10) as well. Consequently, we can interpret $w_{D(k)}^k$ as the length of the shortest path connecting $O(k)$ to $D(k)$ and the conditions (2.10) with

$$\textstyle\sum_{(i,j)\in P_k} c_{ij}^k(f, y) = w_{D(k)}^k \quad \text{if} \quad f_{ij}^k > 0 \quad \text{for all} \quad (i, j) \in P_k$$

becomes WARDROP'S[117] user equilibrium law. That is, the lengths $\sum_{(i,j)\in P_k} c_{ij}^k(f, y)$ of all paths carrying positive flow between any origin-destination pair are equal, and do not exceed the length of *any* path joining this origin-destination pair.

Consequently, adding the conditions (2.8) and (2.9) imbues the network design model with normative planning capabilities even when flow movements are governed by a user equilibrium behavioral principle.

Several modifications of this modeling approach are possible; for example, minor modifications to (2.8) and (2.9) will model situations involving capacitated design decisions.

Needless to say, modeling a normative equilibrium model in this way need not necessarily lead to efficient solution methods. Designing useful methods undoubtedly requires theory indicating how the solution to the equilibrium conditions responds to changes in the design decisions. Most iterative algorithms for solving the problem would rely upon such sensitivity analysis information to choose directions for improving the problem's objective function. Although HALL[53] describes initial efforts for addressing this sensitivity analysis issue, until recently the problem has remained essentially unsolved. Some new discoveries (DAFERMOS AND NAGURNEY[21]), however, are beginning to provide new insight about these problems.

3. OPTIMIZATION BASED METHODS

THE FOLLOWING two sections focus on a class of uncapacitated discrete choice network design problems. These models determine whether arcs (e.g., roadways, service arcs) should be added to the network or not. After briefly reviewing the formulation of these problems, this section discusses methods for obtaining optimal solutions to these models. Section 4 describes heuristic techniques for obtaining approximate solutions.

One of the major purposes of this section is to present a unified framework for viewing a number of optimization-based techniques. In particular, we show that many of these methods, even though not originally stated in this way, can be interpreted as relying on certain types of Benders cuts. We also describe another Benders cut that is apparently new.

Recall from Section 2 that R_k, for $k \in \kappa$, denotes the flow requirement between nodes $O(k)$ and $D(k)$ of a given network, that c_{ij}^k denotes the per unit routing cost on an arc (i, j) for "commodity k" goods, and that F_{ij} denotes the fixed cost of constructing arc (i, j). Therefore, our uncapacitated discrete choice network design model becomes:

$$\text{Minimize } \textstyle\sum_k \sum_{(i,j)\in A} c_{ij}^k f_{ij}^k + \sum_{(i,j)\in A} F_{ij} y_{ij}$$

subject to:

$$\textstyle\sum_j f_{ij}^k - \sum_r f_{ri}^k = \begin{cases} R_k & \text{if } \ i = O(k) \\ -R_k & \text{if } \ i = D(k) \text{ all } k \in \kappa \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

$$f_{ij}^k \le K_{ij} y_{ij} \quad \text{for all} \quad (i, j) \in A,\ k \in \kappa \tag{3.2}$$

$$\textstyle\sum_{(i,j)\in A} e_{ij} y_{ij} \le B \tag{3.3}$$

$$f_{ij}^k \ge 0 \quad \text{for all} \quad (i, j) \in A,\ k \in \kappa \tag{3.4}$$

$$y_{ij} = 0 \text{ or } 1 \quad \text{for all} \quad (i, j) \in A. \tag{3.5}$$

Since we are assuming that the problem is uncapacitated, $K_{ij} > \sum_{k\in\kappa} R_k$. Note that we have included both fixed costs in the objective function and a budget constraint (3.3) in this formulation. The term e_{ij} in the budget constraint is a cost function, which need not equal F_{ij}, related to the building of arc (i, j).

For notational ease, unless otherwise specified, we assume throughout the next two sections that the demand is complete and that the demand R_k between every pair of nodes equals 1. Most of the methodology discussed easily generalizes to situations involving arbitrary demand patterns.

Optimization Methods for Special Cases

Efficient algorithms are available for solving several special cases of the network design problem. Of course, there are very fast procedures for solving the shortest path and minimal spanning tree versions of the problem. HU[57] has proposed a $O(n^5)$ algorithm based on maximum flow computations for solving the uncapacitated budget design problem (i.e., every $F_{ij} = 0$) when every feasible design is a spanning tree and all routing costs are unity (i.e., $B = |N| - 1$ and $e_{ij} = c_{ij}^k = 1$ for all i, j and k). KARIV AND HAKIMI[60] have devised an efficient dynamic programming approach for the K-median location problem restricted to tree networks. As noted in Section 2, the K-median problem is a special case of the budget design model.

More general and complex network design problems seem to require mathematical programming algorithms for obtaining exact solutions. Benders decomposition and branch and bound have been the two most effective of the optimization-based techniques. The algorithmic strategy used by both of these methods is similar—develop and exploit lower bounds on the objective function of the problem. For this reason we devote most of the remainder of this section to various techniques used to generate lower bounds for Benders decomposition and branch and bound.

Benders Decomposition

BENDERS[7] decomposition is an algorithm for mixed integer programming that has been applied successfully to a variety of network design applications. FLORIAN et al.[34] have used the algorithm to schedule the movement of railway engines; RICHARDSON[103] has applied the algorithm to the design of airline routes; MAGNANTI et al.[78] have used the algorithm for fixed charge network design; and GEOFFRION AND GRAVES[47] have had great success applying the procedure to industrial distribution system network design. HOANG[56] has used an extended version of the algorithm, known as generalized Benders decomposition, to solve a class of nonlinear discrete network design models.

When applied to network design problems, Benders decomposition proceeds iteratively by choosing a tentative network configuration (i.e., setting values for the integer variables y_{ij}), solving for the optimal routing on this network, and using the solution to the routing problem to redefine the network configuration. Figure 2 illustrates this last step for an uncapacitated fixed cost design problem in which one unit of a good is to be sent from node 1 to node 6 (we let c_{ij} denote the routing cost for this single good). In this instance, the routing problem reduces to a shortest path computation between nodes 1 and 6.

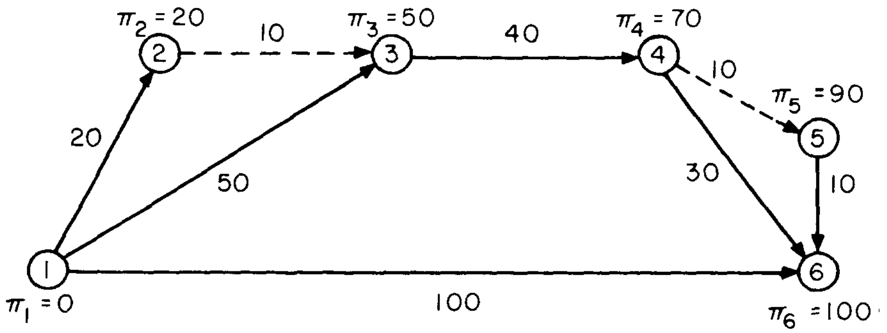

Fig. 2. A step of Benders decomposition.

The solid arcs in the figure are members of the current network configuration; the dashed arcs are candidates for inclusion in the optimal design. With respect to the routing costs shown next to each arc, the node numbers π_j are optimal dual variables for the linear programming routing problem. The dual variables indicate that introducing arc (2, 3) into the current design reduces the routing cost from node 1 to node 3 from $\pi_3 = 50$ to $\pi_2 + c_{23} = 20 + 10$ for a savings of $50 - 30 = 20$ units. Similarly, introducing arc (4, 5) reduces the routing cost from node 4 to node 6 by $\pi_5 - (\pi_4 + c_{45}) = 90 - (70 + 10) = 10$ units. Since either of these arcs might, but need not, become part of the shortest route path from node 1 to node 6 in the optimal network design, the optimal routing cost R is constrained by

$$R \geq 100 - 20y_{23} - 10y_{45}. \tag{3.6}$$

That is, at best, we could reduce the routing cost which is 100 units by the full savings of introducing arc (2, 3), i.e., setting $y_{23} = 1$ and the full savings of introducing arc (4, 5), i.e., setting $y_{45} = 1$. Another reason that expressions like this merely provide inequalities in the routing costs is that we might not be able to accrue the total savings for introducing each arc separately because of interactions and overcounting (since two arcs might lie on the same shortest path). For any network configuration, the total fixed charge design cost equals $\sum_{(i,j)\in A} F_{ij}y_{ij}$. Therefore, for the example in Figure 2, the following expression gives a lower bound on the minimum cost υ of any design for this problem:

$$\upsilon \geq 100 - 20y_{23} - 10y_{45} + \textstyle\sum_{(i,j)\in A} F_{ij}y_{ij}. \tag{3.7}$$

Constraints like (3.7), which are known as Benders cuts, are by-products of the optimal routing calculation for any tentative network configuration. Benders algorithm computes the new configuration at each step by minimizing the total network cost υ subject to the Benders cuts (3.7) generated by every previous configuration. This minimization, called the Benders master problem, is a mixed integer program in the integer variables y_{ij} and the single continuous variable υ. Since every conceivable constraint like (3.7) gives a valid lower bound expression for the optimal solution value υ, the optimal value υ^* of the master problem is also a lower bound, i.e., $\upsilon \geq \upsilon^*$. Also, every solution $y = \hat{y}$ to the master problem determines a network and the combined fixed and optimal routing cost on this network is an upper bound on the optimal value υ to the original problem. These two bounds permit early termination of the algorithm with an assessment of the degree of suboptimality.

When applied to more complicated design problems such as budget design problems, facility location problems, and even to general mixed integer programs, Benders decomposition operates in much the same way and has similar interpretations.

For a given model representation, it is possible to accelerate Benders decomposition by generating improved or "pareto-optimal" cuts at each iteration. Referring to Figure 2 will help illustrate this point. Note that the shortest path distance from node 3 to node 6 using *all* arcs that are candidates for inclusion in the optimal network design is 60 units. Since the distance to node 2 in the current design, as specified by the solid arcs, is 20 units and the current shortest path cost is 100 units, introducing arc (2, 3) whose routing cost is 10 units can save no more than 100 − (60 + 20 + 10) = 10 units, and not the 20 units computed earlier. Consequently, a valid Benders cut is

$$v \geq 100 - 10y_{23} - 10y_{45} + \textstyle\sum_{(i,j)\in A} F_{ij}y_{ij}. \tag{3.8}$$

Note that this cut is stronger than (3.7) in the sense that it provides a tighter lower bound on v; the right-hand side of (3.8) is as large as that of (3.7) for all 0-1 values of the decision variables y_{ij}, and exceeds the right-hand side of (3.7) whenever $y_{23} = 1$. This cut is also pareto-optimal, i.e., there is no other valid lower bound expression that is as good for all 0-1 values and better for at least one set of 0-1 values for the decision variables y_{ij}.

The opportunity to generate pareto-optimal cuts, like (3.8), is made possible because of degeneracy in the shortest path linear program, or equivalently because of multiple optimal solutions to its dual. In this example, $\pi_1 = 0$, $\pi_2 = 20$, $\pi_3 = 40$, $\pi_4 = 70$, $\pi_5 = 90$, and $\pi_6 = 100$ is an alternate optimal dual solution to that shown in Figure 2 (i.e., it satisfies $\bar{c}_{ij} \equiv c_{ij} + \pi_i - \pi_j \geq 0$ for all arcs with an equality for arc 1–6). Computing a Benders cut as before, but using these dual values, leads to the stronger cut (3.8). Because network problems are renowned for their degeneracy, considerations of this nature are attractive in a number of applications.

Magnanti and Wong[79] and Magnanti et al.[78] describe this pareto-optimal cut methodology in greater detail. They show how to generate pareto-optimal cuts for arbitrary mixed integer programs by solving a linear program to choose from among optimal dual solutions. They also discuss an implementation of Benders decomposition combined with a preprocessing procedure that can eliminate integer variables. Computational experience on a variety of the largest uncapacitated undirected fixed charge design problems ever solved was very promising. Benders decomposition with preprocessing and pareto-optimal, or other improved cuts, finds and verifies an optimal solution for 19 out of 24 test problems with up to 30 nodes, 130 arcs with 40 arcs fixed open, and 58 commodities (a corresponding mixed integer program contains 15,500 continuous variables and 90 integer variables). For all but one of these 24 test problems the algorithm finds feasible solutions that are guaranteed to be within at most 1.71% of optimality. The computations took from about 1 minute to about 1½ hours on a VAX 780 computer. Benders decomposition with preprocessing and the standard Benders cuts required from 1½ to 50 times more computation time.

Branch and Bound

Several researchers have proposed branch and bound algorithms for solving network design models (BOYCE et al.,[11] BOYCE AND SOBERANES,[12] DIONNE AND FLORIAN,[24] HOANG,[55] LEBLANC,[68] LOS AND LARDINOIS[72] ROTHENGATTER[104] and SCOTT[107]). To illustrate the nature of this work, we describe several of the contributions from this list.

Boyce et al. have suggested an enumerative algorithm for budget design problems with unit flow requirements between every pair of nodes. In this instance, the optimal routing, given any network configuration, is via shortest distance paths joining each origin-destination pair. Consequently, the objective function cost, denoted $F(y)$, is determined completely by the network configuration, i.e., by the choice of 0-1 values for the components y_{ij} of the vector y. At any node P in a branch and bound enumeration tree, certain arcs A_F are fixed as constructed ($\hat{y}_{ij} = 1$) or not ($\hat{y}_{ij} = 0$). Let $\bar{A}_F$ be the remaining arcs whose construction status has not yet been decided.

As a bounding mechanism for this algorithm, Boyce et al. have noted that any solution y with the arcs in A_F fixed at these same values satisfies the inequality

$$F(y) \geq F(y^P). \tag{3.9}$$

In this expression, y^P denotes a solution with $y_{ij} = 1$ for every arc $(i, j) \in \bar{A}_F$. Thus $F(y^P)$ is the best possible shortest route solution with the given fixed values of arcs in A_F.

Hoang has proposed an improvement on the basic bounding procedure of Boyce et al. At any node P in the branch and bound tree, he suggests the following lower bound function:

$$F(y) \geq F(y^P) + \textstyle\sum_{(i,j)\in\bar{A}_F} \bar{y}_{ij} I_{ij}(y^P) \tag{3.10}$$

where $\bar{y}_{ij} = 1 - y_{ij}$. The quantity $I_{ij}(y^P)$ denotes the increment to the shortest route cost from node i to node j when we delete (i, j) from the network defined by y^P.

Expression (3.10) has the following interpretation. If arc (i, j) is deleted (set $y_{ij} = 0$ and $\bar{y}_{ij} = 1$) from the network defined by y^P, then the cost of shipping the unit of demand between these nodes must increase by at least $I_{ij}(y^P)$ in the solution y. The cost of sending the unit of demand from i to j might increase by more because other arcs are being deleted as well. Hence, the right-hand side of (3.10) is a lower bound on the routing cost $F(y)$ of solution y.

Note that for all feasible values of y at the search node P, the lower bound function (3.10) dominates the one given by (3.9) in the sense that the bound given by (3.10) will be at least as large, and sometimes larger, than the bound of (3.9).

To compute lower bounds quickly, Hoang suggests relaxing the integer requirement on y_{ij} and minimizing the right-hand side of (3.10) subject to the budgetary constraint $\sum_{ij} e_{ij} y_{ij} \le B$ and $0 \le y_{ij} \le 1$, $y_{ij} = \hat{y}_{ij}$ for arcs $(i, j) \in A_F$. A solution y^* to this continuous knapsack problem has at most one fractional component and, in light of (3.10), gives a lower bound $F(y) \ge F(y^P) + \sum_{(i,j)\in \bar{A}_F} \bar{y}^*_{ij} I_{ij}(y^P)$ that applies to every node below node P in the branch and bound enumeration tree. Using this lower bound as a fathoming mechanism and branching from node P on (i.e., next fixing) the fractionally valued variable in the continuous knapsack solution, Hoang implements his algorithm in the framework of a straightforward branch and bound algorithm.

Dionne and Florian have noted several ways to improve this algorithm. First, in computing $I_{ij}(y^P)$ it is not necessary to resolve from scratch for shortest paths between all pairs of nodes. Specialized algorithms are available for recomputing shortest path distances when one arc has been deleted from a network. (Boyce et al. also advocated the use of these specialized algorithms in the context of computing the shortest path routing costs for adjacent nodes of the branch and bound tree.)

Second, they suggested branching on the variable y_{ij}, $(i, j) \notin A_F$ with highest incremental improvement per unit of budget, i.e., that arc (i, j) maximizing $I_{ij}(y^P)/e_{ij}$. These modifications lead to marked improvements in the algorithm. A typical example with a 65% budget level, i.e., $B = 0.65 \sum_{(i,j)\in A} e_{ij}$ with 20 nodes, and with 30 arcs, requires 20 seconds to solve on a CDC Cyber 75 computer with their algorithm, while Hoang's algorithm, after 500 seconds, is not able to determine that the current best solution is optimal. The authors note, however, that this branch and bound approach is probably limited to medium-sized networks like this, and that computation time seems to grow exponentially with a decrease in budget level. For example, the algorithm requires 288 seconds to solve the same problem with a 50% budget level.

BOFFEY AND HINXMAN[9] have used a similar procedure for the budget design problem.

Los and Lardinois[72] have adapted the Dionne and Florian lower bound for the uncapacitated fixed charge design problem. They utilize the inequality

$$F(y) \geq F(y^P) + \textstyle\sum_{(i,j)\in \bar{A}_F} \bar{y}_{ij} I_{ij}(y^P) + \sum_{(i,j)\in A} F_{ij} y_{ij}$$

where the added term $\sum_{(i,j)\in A} F_{ij} y_{ij}$ represents the fixed charge costs. LOS[77] has used another lower bound $F(y) \geq F(y^P) + \max(F_{MST}, \sum_{(i,j)\in A_F} F_{ij}\hat{y}_{ij})$ where F_{MST} is the cost of the minimum fixed cost spanning tree and y_{ij} is the 0-1 incident vector defining the current design. The last term in the inequality indicates that the total fixed charge cost of all nodes in the enumeration tree below node P must be at least the cost (i) of all arcs at P already fixed to value one (i.e., those arcs (i, j) in A_F with $\hat{y}_{ij} = 1$) *or* (ii) the fixed charge cost of a minimum spanning tree (since we are assuming all pairs of nodes have unit demands).

Los and Lardinois used a combination of the two lower bounds given above in a branch and bound procedure. They found that the method was effective only for small to medium-sized problems, with computation times growing exponentially in the problem size. For example, an 8-node and 23-arc problem required 5.4 seconds on a CDC Cyber 74 computer while a 12-node and 40-arc problem required 207 seconds.

As noted by LOS,[73] the above inequality (3.9) could be interpreted as a Benders cut (see (3.8)). Recall that at a node P in the branch and bound tree, the algorithm definitely excludes the arcs in A_F with $\hat{y}_{ij} = 1$. The only remaining decision concerns whether the arcs in $\bar{A}_F$ are to be included in the network design ($y_{ij} = 0$ or 1). If at node P we apply Benders decomposition to the network configuration with $y_{ij} = 1$ for all arcs $(i, j) \in \bar{A}_F$, then the usual (not pareto-optimal) Benders cut would be

$$v \geq F(y^P).$$

So the procedure of Boyce et al. can be interpreted as using Benders inequalities with an enumerative algorithm. This solution technique is similar to one suggested by LEMKE AND SPIELBERG[70] for more general classes of integer programming problems.

We can also derive (3.10) as a Benders cut in the same way except that we select an alternate dual solution to the shortest path linear program. In fact, there is a third cut that dominates these two cuts (3.9) and (3.10). Figure 3 illustrates an example of these three cuts.

The example involves an uncapacitated budget design problem in which one unit of good is to be sent from node 1 to node 2 and another unit from node 1 to node 3. The routing problem reduces to a pair of shortest path problems, one between nodes 1 and 2 and the other between nodes 1 and 3. Assume that we are at node P of the search tree where arcs (1, 2) and (2, 3) belong to $\bar{A}_F$ (i.e., are free arcs) and that y_{12} and y_{23} are set to one in the application of Benders decomposition. The remaining arcs belong to A_F with $\hat{y}_{ij} = 1$ so their values are permanently fixed at

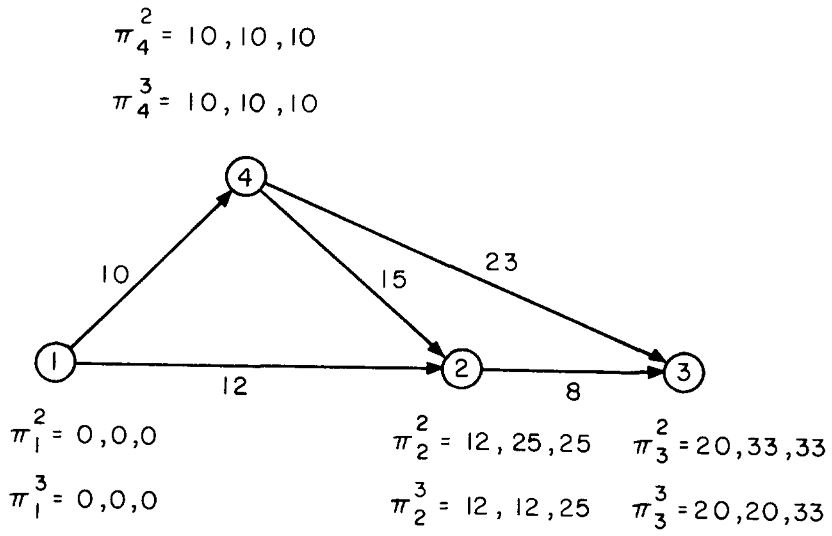

Fig. 3. A step of Benders decomposition with alternate dual solutions.

search node P. The optimal dual variables π_i^k, with $k = 2$ or 3 corresponding to the destination nodes of the two commodities, have three different values corresponding to the three different Benders cuts. The first set of dual variables yields the basic inequality, corresponding to (3.9), namely

$$v \geq 32.$$

The second set of dual variables indicates that deleting arc (1, 2) from the network increases the routing cost between nodes 1 and 2 from $\pi_1^2 + 12 = 12$ to $\pi_2^2 = 25$, for a net increase of 13 units. Thus the second type of cut, corresponding to (3.10), is

$$v \geq 32 + 13(1 - y_{12}).$$

The third set of dual variables dictates the same penalty of 13 units for the node 1 to node 2 problem. In addition, deleting arc (1, 2) increases the routing cost beween nodes 1 and 3 from $\pi_1^3 + 12 + 8 = 20$ to $\pi_3^3 = 33$ for a network cost of 13 units. Combining these results gives the new cut

$$v \geq 32 + (13 + 13)(1 - y_{12}).$$

The third cut improves upon (3.10) in the following way. In (3.10) when we deleted arc (i, j) from $\bar{A}_F$ we considered its effect on increasing the routing cost of the demand for the origin-destination pair (i, j). Thus, in the cut as a whole, we consider incremental increases in routing cost, but only for origin-destination pairs (i, j) corresponding to arcs in $\bar{A}_F$. The third cut sharpens the lower bound by considering incremental increases in routing cost for *all* origin-destination pairs in κ. In order to implement this modification, we consider any partition of the origin-destination pairs κ into sets S_{ij}. Then for any $(p, q) \in S_{ij}$, we define $I_{ij}^{pq}(y^P)$ as the increment to the shortest path between nodes p and q when arc (i, j) is deleted from the network defined by y^P. With this notation, we can define the third lower bound by the inequality

$$v \geq F(y^P) + \textstyle\sum_{(i,j)\in\bar{A}_F} \bar{y}_{ij} \sum_{(p,q)\in S_{ij}} I_{ij}^{pq}(y^P). \tag{3.11}$$

Note that if we require that $(i, j) \in S_{ij}$, the lower bound is never worse than, and usually dominates, the bounds given in (3.9) and (3.10). As an illustration, the third cut for the last numerical example corresponds to the choice of $S_{12} = \{(1, 2), (1, 3)\}$ and $S_{23} = \phi$.

We might note that if the network configuration contains a relatively small number of arcs (say, twice as many arcs as nodes as in a grid network) as is the case for most well-defined transportation networks, then several origin-destination shortest paths are likely to pass through most arcs. Consequently, the new cut in (3.11) is likely to improve significantly upon the cut (3.10).

A number of other Benders cuts can also be generated from the alternate optimal solutions to the shortest path linear programs.

GALLO[37] has independently derived the new cut (3.11) from a combinatorial argument. His computational results with one version of the new cut are very promising. For test problems with 14 nodes and 22 arcs the bounds provided by (3.11) are significantly stronger than the ones provided by (3.10) or another type of cut (3.12) that we define next.

Suppose that the following constraint is valid at some node P in a branch and bound tree:

$$\textstyle\sum_{(i,j)\in\bar{A}_F} y_{ij} \geq |\bar{A}_F| - Q.$$

Such a constraint could arise from connectivity requirements or by noting the maximum number of the arcs from $\bar{A}_F$ that could be deleted while still fully utilizing the given budget. Using the last inequality, GALLO[36] derived the lower bound relation

$$v \geq F(y^P) + (1/Q) \textstyle\sum_{(i,j)\in\bar{A}_F} \sum_{(p,q)\in\kappa} \bar{y}_{ij} I_{ij}^{pq}(y^P). \tag{3.12}$$

For small values of Q, (3.12) is a very effective lower bound. For example, if $Q = 1$ for the problem in Figure 3, (3.12) becomes

$$v \geq 32 + 26(1 - y_{12}) + 13(1 - y_{23})$$

which is a better lower bound (given the constraint $y_{12} + y_{23} \geq 1$) than the ones provided by the Benders cuts.

Note that for each (i, j) the second summation in (3.12) is taken over all origin-destination pairs, rather than over the set S_{ij} of a partition of the origin-destinations. Consequently, the summation might be double-counting the increment in costs for any origin-destination pair (p, q); that is, it double-counts the effect on (p, q) of deleting (i, j) and of deleting (i', j') when they both lie on the shortest path joining (p, q). The factor $1/Q$ compensates for this double counting.

Lagrangian and Linear Programming Methods

Other methods could also be useful in deriving lower bounds for network design problems. Lagrangian relaxation and dual ascent procedures have been very successful in providing bounds for the special cases of facility location problems (CORNUEJOLS et al.,[16] NARULA et al.,[92] ERLENKOTTER,[28] GALVAO[41] and MIRCHANDANI et al.[87]) and Steiner tree problems on a graph (BEASLEY[5] and WONG[121]). For more general design models, these methods are less easily applied due to the large number of formulation constraints. However, in preliminary work RARDIN AND CHOI[102] have found that a dual ascent procedure gives strong bounds for a fixed charge design model with a single source and multiple destinations. For a 157 node and 500 arc problem, they obtained a solution known to be within 2½% of optimality in about 300 seconds of CDC Cyber 76 computer time. MAGNANTI AND WONG[75] have very promising computational results with a dual ascent method for fixed charge design models with complete demand patterns. The dual ascent information provides good lower bounds and helps derive near-optimal design solutions. For problems with 10 nodes and 45 arcs they have found solutions known to be within about 1% of optimality in about 0.5 seconds of IBM 3033 computer time.

Note that Lagrangian relaxation and dual ascent bounds are closely related to Benders cuts since they can all be derived from the common framework of mathematical programming duality theory (MAGNANTI[77]). Thus it is possible to extend our Benders cut interpretation to derive the bounds given by Lagrangian relaxation and other dual based procedures (see also GUIGNARD[50]).

Special implementations of the simplex method have been successful in giving strong *LP* relaxation bounds for location problems (SCHRAGE[105, 106]). Future improvements in linear programming technology (e.g., TODD[114]) could make the method feasible for more general design models.

The effectiveness of both Lagrangian and linear programming-based methods depends crucially on the size of the gap between the *LP* relaxation value and the integer programming model value. The empirical results given in the references cited earlier indicate that the gap is usually quite small. Some recent theoretical results support this conclusion. For example, in a variety of specially structured location problems (KOLEN[66] and OUDJIT et al.[95]) there is no gap between the standard *LP* relaxation and the integer formulation values. For another class of location models, Cornuejols et al.[15] show that the worst-case value of the *LP* relaxation gap of certain formulations is bounded.

Finally, the addition of facet generating constraints can strengthen the *LP* relaxation of standard network design formulations. A notable example of this technique is the work of GROTSCHEL AND PADBERG[48, 49] for the traveling salesman problem (see also PADBERG AND HONG[96] and CROWDER AND PADBERG[18]). JOHNSON et al.[59] show how to derive certain facets of the fixed charge network design problem. PADBERG et al.[97] discuss facets for some models of capacitated fixed charge network flow problems. Conceivably, further studies of such facets might prove to be as useful for network design problems as they have been for the traveling salesman problem.

Summary

This section has reviewed a number of bounding procedures for obtaining optimal solutions to network design problems. Table II summarizes the computational results for the various proposed algorithms. We have also seen that the interpretation of lower bounds as Benders cuts unifies much of the previous bounding work and has also yielded a new lower bound. Gallo's work has indicated how stronger bounds can be synthesized with information not available to the Benders cuts. We are confident that future research will provide improved bounds and thus enlarge the scope of network design problems that can be solved optimally.

4. COMPUTATIONAL COMPLEXITY AND HEURISTICS FOR NETWORK DESIGN PROBLEMS

AS INDICATED in the previous section, past research gives extensive empirical evidence suggesting that large scale (50–100 nodes) network design problems are extremely difficult to solve. There is also theoretical evidence supporting this observation, since the general network design problem is *NP*-hard. This result suggests that there is little likelihood of devising an efficient (polynomial time) algorithm for solving the general network design problem (see KARP[61]).

Several authors have studied the complexity of specific design problems. JOHNSON et al.[58] have shown that the uncapacitated budget design problem is *NP*-hard. WONG[120] has shown that, when the budget design

TABLE II

Optimization Methods for Network Design

Authors	Problem Type (All Problems Are Uncapacitated)	Solution Algorithm	Computational Experience
Boffey and Hinxman[9]	Budget	Branch and bound	10 nodes, 45 arcs, ?
Boyce et al.[11]	Budget	Branch and bound	10 nodes, 45 arcs, 200 seconds, IBM 360/75
Dionne and Florian[24]	Budget	Branch and bound	20 nodes, 40 arcs, 20 seconds, CDC Cyber 74
Gallo[37]	Budget	Branch and bound	30 nodes, 60 arcs, 10 seconds, IBM 370/168
Hoang[56]	Budget	Branch and bound	20 nodes, 40 arcs, 500 seconds—problem not completed, Cyber 74
LeBlanc[68]	Budget—convex routing costs	Branch and bound	24 nodes, 76 arcs, 135 seconds, CDC 6400
Los and Lardinois[72]	Fixed charge	Branch and bound	12 nodes, 40 arcs, 207 seconds, CDC Cyber 74
Magnanti et al.[78]	Fixed charge	Benders decomposition	30 nodes, 90 arcs, 82 seconds, VAX 780

problem is restricted to unit demands between pairs of nodes, the problem of finding a solution whose value is less than $n^{1-\epsilon}$ times the optimal solution value (where n is the number of nodes and ϵ is any positive constant) is also NP-hard. This result indicates that even finding a near optimal solution for the budget design problem efficiently is also very difficult.

PLESNIK[100] gives results showing that finding near optimal solutions for a budget design problem with a minimax objective is also NP-hard.

As shown in Section 2, the uncapacitated fixed charge design problem contains a number of NP-hard problems (e.g., the plant location problem and the Steiner tree problem on a graph (GAREY AND JOHNSON[42]) as special cases. Thus the fixed charge problem is also NP-hard.

Since both theoretical and empirical results confirm the computational difficulty of the general network design problem, researchers and practitioners have often used heuristic algorithms as alternative solution methods.

An important issue in using heuristic approximation techniques is solution accuracy. The usual heuristic performance measure computes the relative percentage of error where

$$\text{Relative error} = (v_h(P) - v_*(P))/v_*(P).$$

The term $v_h(P)$ is the best heuristic solution value produced for a particular problem P; $v_*(P)$ is the optimal solution value for problem P.

Analysts have proposed other error measures (e.g., see ZEMEL[128]) for assessing heuristic procedures; however, the above formula is easy to use and understand and is a fairly accurate error measure for network design problems, as contrasted with facility location problems, where a simple transformation of the problem data, which preserves the choice of optimal variables, can change the relative error value to any positive number (see CORNUEJOLS et al.[16]).

Normally, we are interested in assessing the relative error of a heuristic when it is applied to a specific class $\mathscr{S}$ of design problems. For example, $\mathscr{S}$ could be the set of all uncapacitated budget design problems with unit demands between all pairs of nodes.

Recent research has focused on four different techniques for evaluating heuristic performance over a class of problems $\mathscr{S}$:

1. Worst case analysis:

$$\text{Worst case error} = \max_{P\in \mathscr{S}}[(v_h(P) - v_*(P))/v_*(P)].$$

2. Average case analysis:

$$\text{Average case performance} = \text{Prob}[((v_h - v_*)/v_*) \leq 1 + \epsilon]$$

where ϵ is some margin of error and the probability is computed by assigning a probability distribution to the set of problems $\mathscr{L}$.

3. Empirical analysis: This technique empirically evaluates the heuristic by assessing its relative error for a small set of test problems $T \subseteq \mathscr{L}$.
4. Statistical analysis: After repeatedly running the heuristic (with randomly varying starting solutions) on a particular problem P, we use the data to produce estimates $\hat{z}_*(P)$ for the optimal solution value and the relative error percentage $(\hat{z}_h(P) - \hat{z}_*(P))/\hat{z}_*(P)$.

Worst case performance analysis is rather conservative as it reflects only the worst possible situation for the heuristic. Such a situation could be atypical and thus misleading. However, when the worst-case error is small, we are guaranteed the heuristic will always produce near optimal solutions.

An alternative technique measures the average cost or "typical" relative error. When evaluated by this performance criteria, the heuristic receives a poor evaluation only when the problems on which it performs badly can occur relatively frequently. Unfortunately, average case analysis is difficult to perform since assigning realistic probability distributions to a class $\mathscr{L}$ is usually difficult, if not impossible. Also because the method frequently requires complex technical arguments, researchers have analyzed only very simple heuristics.

Empirical analysis is by far the most popular and most easily performed heuristic evaluation method. It avoids the complicated technical arguments that are frequently required for worst case and average case analysis. However, for large network design problems it is difficult to compute the optimal solution value or, equivalently, the relative error for the heuristic. So the heuristic's utility for large problems must be measured by its performance on much smaller test problems. In addition, the statistical inferences made from the empirical data are not very rigorous and could be misleading.

The statistical analysis method assumes that the heuristic solutions generated can be viewed as random samples from an extreme value Wiebull distribution. This technique is attractive because it estimates the relative error with only simple computational tests. However, as of yet, although there are some plausible intuitive arguments to support the Wiebull distribution assumption, there are no mathematical proofs to justify its validity.

Experience with Heuristics

A number of studies have applied these evaluation methods to a variety of proposed network design heuristics. The three most commonly used heuristics are the add, delete and interchange procedures. The add heuristics start with some feasible design and add arcs, one at a time, choosing at each stage the arc that gives the greatest decrease in cost, or some surrogate measure of cost. The delete heuristics are similar, but start with an initial design containing all candidate arcs, and delete arcs one at a time. Starting with some initial design, the interchange heuristics add and/or delete an arc at each step until no further improvement in cost is possible.

Scott,[107] Dionne and Florian,[24] Boffey and Hinxman,[9] BILLHEIMER AND GRAY,[8], Los and Lardinois[72] have used combinations of the above heuristics for solving uncapacitated (budget or fixed charge) design problems. Empirical analysis indicated that the heuristics were very accurate (relative error $\leq 0.5\%$) for medium sized (10–29 nodes and 40–60 arcs) problems. For larger problems of up to 15 nodes and 105 arcs, the Wiebull statistical analysis estimates generated by the Los and Lardinois procedure had relative errors of less than 0.2%. The heuristics were fairly efficient, requiring from 7 to 10 seconds on a Cyber 74 computer.

Recently, Dionne and Florian[24] have reported impressive computational experience with a new type of heuristic for budget design problems which we assume, for convenience, has one unit of demand between every two nodes in the network. They use their branch and bound algorithm described in the last section with the following modification. In place of the term $I_{ij}(y^P)$ in the lower bound expression (3.10), they use a term $I(y^P)$ which is the increment to shortest route costs between *every* pair of nodes, not just i and j, when arc (i, j) is deleted from the network. This algorithm is a heuristic because, as examples show, the new lower bound need not be valid.

The authors have tested this algorithm on problems ranging from 7 nodes and 16 candidate arcs to 29 nodes and 54 candidate arcs. The empirical analysis indicated the procedure had a very small ($\leq 0.03\%$) relative error and required from 0.05 to 8.47 seconds on a CDC Cyber 74 computer.

Dionne and Florian have suggested another alternative procedure, a modified add heuristic, for the budget design problem. Starting from a minimal spanning tree with respect to the arc construction costs e_{ij}, the procedure iteratively adds the arc with the smallest construction cost until it exhausts the construction budget B. Empirical analysis shows that this heuristic is extremely fast, but less accurate than the heuristics described previously. This result is to be expected since the procedure constructs a network according to a naive criteria that does not even evaluate the objective function.

It is interesting to note that network design researchers have used only empirical and statistical analysis to evaluate the class of add, delete and interchange heuristics. In contrast, for the special case of facility location problems, researchers have performed extensive worst case and probabilistic analyses for these heuristics (see CORNUEJOLS et al.[15,16] and FISHER[30]).

As described above, the add, delete and interchange heuristics are fairly effective for solving small to medium-sized discrete network design problems. However, for large-scale problems (e.g., 100 nodes and 600 arcs) these heuristics could require excessive amounts of computer time. These procedures rely on iteratively evaluating the design cost which is equivalent to resolving shortest path problems. Even with the use of specialized algorithms (e.g., MURCHLAND[91]), the computation times for large networks could be unreasonable. Thus different types of approximate procedures might be necessary to solve very large network design problems.

WONG[119] has analyzed a close variant of the modified add heuristic when the problem class is restricted to Euclidean problems. The nodes are points in the plane and arc routing and construction (e_{ij}) costs are multiples of the arc length. Assuming the budget is sufficiently large, $B/\sum_{ij} e_{ij} \geq (\log n/n^{0.6})$, Wong has shown that, for any $\epsilon > 0$, average case performance $=$ Prob$[(v_h - v_*)/v_* \leq (1 + \epsilon)]$ approaches one as n approaches infinity. Thus, in an asymptotic sense the modified add heuristic is very accurate. Empirical analysis confirms these analytic results with problems up to 100 nodes and 4950 arcs having a relative error of less than 3%. The procedure requires about 30 seconds on an IBM 3033 computer to solve a 100-node problem. This result indicates that the build heuristic can effectively exploit the geometric structure of large budget design problems.

Wong has also given related empirical and analytic results on a more restricted class of problems for a geometrically defined heuristic.

Several authors have also used asymptotic probabilistic analysis to evaluate heuristics for facility location problems. Consider a Euclidean version of the K-median problem (locate K facilities on a network to minimize the total cost of servicing all N nodes or customers, each of which is served from its closest facility). Let us assume that all nodes are randomly distributed points over a unit square and that the cost of travel between any two points is their Euclidean distance. Suppose that as N increases the number of medians increases as a function $K(N)$ of N.

FISHER AND HOCHBAUM[31] introduce an aggregation heuristic where the unit square is decomposed into $t^2 < N$ subsquares. After combining the nodes in each subsquare, they solve the aggregated K-median problem and convert its solution into one for the original problem. They then prove that if $(K(N)/N) < (\log N/N)$ and $\epsilon > 0$, the average case performance given by $\text{Prob}[((v_n - v_*)/v_*) \leq (1 + \epsilon)]$ of this heuristic approaches one as N approaches infinity.

When $(\log N/N) < (K(N)/N) < 1/\log N$ and $K(N)$ approaches infinity, PAPADIMITRIOU[98] derives a similar result for a "honeycomb" heuristic. This procedure covers the unit square with $K(N)$ regular hexagons of area $1/K(N)$. The heuristic solution consists of the best single median facility for each hexagon of the honeycomb.

Assuming that the points are generated by a Poisson distribution with mean N, Papadimitriou shows that when $(K(N)/N) > (1/(\log N)^3)$ and $\epsilon > 0$, a fast dynamic programming based heuristic has an asymptotic average case performance of one.

HAIMOVICH[51] shows that the hexagonal partitioning heuristic is optimal whenever $K(N)/N$ approaches zero as N approaches infinity. He does this by showing that the solution of the problem converges to the solution of the continuous analog of the problem where the demand is continuous and uniform over the square. Moreover, he extends the heuristic to solve situations with nonuniform distributions of customers. In this case, the partitioning into service regions is locally hexagonal, but the size of the hexagons varies with the density of the demand distribution in such a way that the density (i.e., number/area) of medians is asymptotically proportional to the ($2/3$)th power of the density of customers. When viewed in terms of the continuous analog, the Fisher and Hochbaum heuristic can be interpreted as a finite element type method for solving the underlying continuous problem.

When $K(N)/N$ approaches a positive constant less than one, and the customer distribution is uniform, the continuous approximation is no longer valid. However, Haimovich[51] and STEELE[109] show that the optimal objective value is proportional to $\sqrt{N}$. This result is based upon a subadditivity property of the optimal objective value and leads to the asymptotic optimality of partitioning heuristics of the type devised by KARP[62] for the traveling salesman problem.

Summary

This section has emphasized the design and analysis of efficient network design heuristics (see Table III for a summary). Current procedures appear quite adequate for solving medium-sized (50 nodes, 150 arcs) problems, but additional heuristic design is needed to handle larger scale (100 nodes, 400 arcs) problems. Although researchers and practitioners evaluate most approximate procedures by empirical analysis, the mathematical analysis of performance accuracy has recently become more widespread. The excellent average case results for network design and facility location problems indicate that near-optimal solutions for these models may be fairly easy to obtain. Computational results by

TABLE III

Heuristics for Network Design

Authors	Problem Type	Heuristic Type	Empirical Performance Analysis[a]	Additional Performance Analysis
Billheimer and Gray[8]	Fixed charge	Add and delete	68 nodes, 476 arcs, 180 seconds; IBM 360/67; RE = ?	—
Dionne and Florian[24]	Budget design	Delete and interchange	29 nodes, 54 arcs, 12 seconds; CDC Cyber 74; RE $\leq$ 0.05%	—
Dionne and Florian[24]	Budget design	Modified tree search	29 nodes, 54 arcs, 3 seconds; CDC Cyber 74; RE = 0	—
Fisher and Hochbaum[31]	K-median	Aggregation	—	Average case for some Euclidean problems
Haimovich[51]	K-median	Modified honeycomb	—	Average case for some Euclidean problems
Los and Lardinois[72]	Fixed charge	Add and delete	12 nodes, 40 arcs, 0.61 seconds; CDC Cyber 74; RE = 0	Statistical analysis; 15 nodes, 105 arcs; RE $\leq$ 2%
Papadimitriou[98]	K-median	Honeycomb, dynamic programming based	—	Average case for some Euclidean problems
Wong[120]	Budget design	Add	100 nodes, 4950 arcs, 30 seconds; IBM 3033 RE $\leq$ 3%	Average case for some Euclidean problems

[a] RE = relative error.

PEARMAN,[99] which support this hypothesis, suggest that network design problems are rich in good suboptimal solutions. Another advantage of the analytic approach is that the proof techniques frequently contain qualitative insight concerning the structure of optimal or near-optimal solutions. NEWELL[93] and MacKinnon[74] emphasize the utility of this qualitative information (e.g., the value of honeycomb, square grid, and other special network layouts; whether hiearchical or nonhierarchical designs are superior) in the context of the transportation planning process. MacKinnon argues that, due to the complexity of transportation systems, structural information about network design could be even more useful than the design of an efficient algorithm.

5. NETWORK DESIGN PROBLEMS WITH NONLINEAR ROUTING COSTS

THIS SECTION discusses a more complex type of network design problem where the arc routing costs can be nonlinear functions of arc flows. Such models occur in urban and highway transportation systems where convex arc routing costs reflect increasing user cost due to road or link congestion. Concave routing costs represent economies of scale present in freight transportation or multiplexed communication network applications.

Using the same notational conventions described in Sections 2 and 3, we have the following general formulation:

$$\text{minimize} \quad \textstyle\sum_{(i,j)\in A} c_{ij}(f_{ij}, y_{ij}) + \sum_{(i,j)\in A} F_{ij}(y_{ij})$$

subject to:

$$\textstyle\sum_j f_{ij}^k - \sum_r f_{ri}^k = \begin{cases} R_k & i = O(k) \\ -R_k & i = D(k) \quad \text{all} \quad k \in \kappa \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

$$f_{ij} = \textstyle\sum_k f_{ij}^k \tag{5.2}$$

$$f_{ij} \leq K_{ij} y_{ij} \tag{5.3}$$

$$\textstyle\sum_{(i,j)\in A} e_{ij} y_{ij} \leq B \qquad \text{all} \quad (i, j) \in A,\ k \in \kappa \tag{5.4}$$

$$f_{ij}^k \geq 0 \tag{5.5}$$

$$y_{ij} \in Y. \tag{5.6}$$

Notice that unlike the previous two sections, we permit the variables y_{ij} to be continuous in this model depending on our choice of the feasible set Y.

This section discusses several specializations of this model. We shall use the following terminology to distinguish between these problem variants:

Uncapacitated Design. If all the variables y_{ij} are 0-1 and every $K_{ij} \geq \sum_k R_k$, the constraints (4.3) impose no (capacity) restrictions on flow when $y_{ij} = 1$.

Convex Routing Cost. When all the variables are fixed with $y_{ij} = \hat{y}_{ij}$, the functions $c_{ij}(f_{ij}, \hat{y}_{ij})$ are convex functions of the f_{ij}. As we noted earlier, the convex routing cost function $t_{ij} f_{ij}[1 + \alpha_{ij}(f_{ij}/\hat{y}_{ij})^{\beta_{ij}}]$, where α_{ij}, β_{ij} and t_{ij} are constants, is often used to model highway congestion costs.

Concave Routing Cost. Same as the above discussion except that the routing costs are now concave.

Budget Design. The construction cost term $\sum_{(i,j)\in A} F_{ij}(y_{ij})$ is eliminated.

Convex Routing Cost Problems

Hoang[56] and CÔTÉS AND LAUGHTON[17] have applied an extension of Benders decomposition, known as generalized Benders decomposition (GEOFFRION[46]), to the uncapacitated discrete budget design problem with convex routing costs. The algorithmic strategy of this procedure is very similar to applying Benders procedure to the uncapacitated budget problem of Section 3. The algorithm starts with a tentative network configuration, but instead of solving shortest path subproblems as in Section 3, solves a convex (cost) multicommodity flow subproblem. There are a number of efficient procedures for solving these multicommodity flow problems (NGUYEN,[94] CANTOR AND GERLA,[13] LEBLANC et al.,[69] ASSAD,[3] DEMBO,[23] FLORIAN,[32] KENNINGTON,[64] MAGNANTI AND GOLDEN[76] among others). Then using the optimal dual variables of the nonlinear programming subproblem, the algorithm synthesizes a linear lower bound inequality of the form $v \geq a_0 + \sum_{ij} a_{ij} y_{ij}$. These dual variable values can be interpreted as the marginal cost of increasing the flow requirement between a pair of nodes when the current flow pattern is the optimal solution to the multicommodity subproblem.

When applied to this problem, generalized Benders decomposition would normally solve a mixed integer master problem consisting of the budget constraints as well as all the generated cuts. Hoang dualizes the problem and places all the Benders cuts, weighted by dual variables, in the objective function. He then heuristically solves the resulting knapsack problem to get a new tentative configuration and solves a relaxed version of the knapsack problem to obtain a new lower bound for the network design model. The procedure finds a solution known to be within 9% of optimality for a problem with 155 nodes, 376 arcs and 25 projects (set of arcs to be modified) in about 60 seconds of IBM 360/50 time. Côtés and Laughton derive the same knapsack problem, but solve it only heuristically; therefore, they use the generalized Benders algorithm as a heuristic, without using the algorithm's lower bounding provision. Their computational results are also quite promising.

Note that the Benders decomposition acceleration techniques of Magnanti and Wong[79] could be easily generalized and applied to Hoang and Côtés and Laughton's procedure.

LEBLANC[68] has solved a nonlinear-user optimal criteria budget design problem using a similar type of bounding procedure. At node P in the search tree, he bounds the optimal solution value by $G(y^P)$, the optimal value of the corresponding *system optimal* routing problem for the network configuration defined by y^P. This routing calculation requires solving a convex multicommodity flow problem. LeBlanc makes the assumption that the routing cost in his problem does not *increase* when an arc is added to the network configuration (the situation when routing cost does increase after an arc is added is known as Braess' paradox (see MURCHLAND[90] and KNÖDEL[65]. For recent discussions of this phenomenon, see FRANK[35] or STEINBERG AND ZANGWILL[112]). When the routing problem is a set of shortest path problems as for the budget design model, this assumption is always valid. LeBlanc's algorithm solved design problems containing 24 nodes, 76 arcs and 5 potential arcs to be added to the network in 135 seconds on a CDC 6400 computer.

MORLOK AND LEBLANC[87] have used an add-interchange heuristic to solve LeBlanc's problem. The heuristic was able to compute an essentially optimal solution to the same problem described above in 17.8 seconds on a Cyber 70 computer.

It is possible to derive the system optimal routing problem bound in LeBlanc's model by applying generalized Benders decomposition with y^P as the tentative configuration. This interpretation reveals that LeBlanc's bound is analogous to the bound used by Boyce et al. to solve the uncapacitated budget design problem as described in Section 3. Also, the improved bounds, given in Section 3 by Hoang, Dionne and Florian, and Gallo, and the pareto-optimal cut theory would apply in a generalized form to LeBlanc's model. Thus, improved bounds may be readily available for this problem.

BARBIER,[4] STAIRS[108] and HAUBRICH[54] discuss a problem similar to LeBlanc's except they replace the budget constraint by the construction cost term $\sum_{(i,j)\in A} F_{ij}(y_{ij})$ in the objective function. Barbier and Haubrich utilized a drop type heuristic to solve models of the Paris rail network and the Dutch rail network. Haubrich processed a network design problem with about 1250 nodes and about 8000 arcs in less than 2400 seconds of IBM 360/75 computer time.

DANTZIG et al.,[22] LEBLANC AND ABDULAAL[67] and LOS[71] have all studied a budget problem with convex routing costs and continuous capacity variables (see also DAFERMOS[20]). Their solution procedures are all based upon a decomposition scheme suggested by Steenbrink,[110,111] which first associates a dual variable λ with the budget constraint (5.4) and by dualizing obtains:

$$\begin{aligned}&\text{minimize} \quad \textstyle\sum_{(i,j)\in A} c_{ij}(f_{ij}, y_{ij}) + \lambda \sum_{(i,j)\in A} e_{ij}y_{ij} \\ &\text{subject to (5.1), (5.2), (5.3), (5.5) and (5.6).}\end{aligned} \tag{5.7}$$

It is usually necessary to solve this relaxed problem with several different values of λ in order to obtain a solution to the original budget design model. The following discussion concentrates on solving the relaxed problem with a fixed value of λ.

We can rewrite the relaxed problem as

$$\text{minimize}_{f\in G} \textstyle\sum_{(i,j)\in A} [\text{minimize}_{y_{ij}\in Y(f_{ij})} c_{ij}(f_{ij}, y_{ij}) + \lambda e_{ij}y_{ij}] \tag{5.8}$$

where $f = (f_{ij})$ is a vector of arc flows and the set G contains all values of f that satisfy the constraints (5.1), (5.2), and (5.5). The set $Y(f_{ij})$ contains all values of the y_{ij} that satisfy (5.3) and (5.6) when f_{ij} is held fixed.

Now define

$$H_{ij}(f_{ij}) = \min_{y_{ij}\in Y(f_{ij})} [c_{ij}(f_{ij}, y_{ij}) + \lambda e_{ij}y_{ij}]. \tag{5.9}$$

The solution y_{ij}^* to this minimization problem can be interpreted as the optimal capacity level for arc (i, j) given that the total arc flow is f_{ij}.

Rewriting (5.8) in terms of (5.9) yields:

$$\text{minimize}_{f\in G} \textstyle\sum_{(i,j)\in A} H_{ij}(f_{ij}). \tag{5.10}$$

Note that (5.10) is a multicommodity flow problem in which the constraints (5.1), (5.2), and (5.5) that describe G represent the conservation of flow constraints and $H_{ij}(f_{ij})$ is the flow cost function for arc (i, j).

This decomposition scheme has separated (5.7) into a master problem (5.10) and a series of subproblems (5.9) with one subproblem for each arc (i, j). The main drawback to its implementation resides in the definition of the functions H_{ij} only implicitly as the optimal objective value to the optimization problems (5.9). Also, the method becomes particularly attractive only when the functions H_{ij} are convex so that we can apply any of the efficient convex multicommodity flow algorithms cited earlier. Dantzig et al., LeBlanc and Abdulaal, and Los have found a wide range of useful models where the H_{ij} functions have a closed form representation and are convex. They use the Kuhn-Tucker conditions for (5.9) to obtain a relationship between y_{ij}^*, the optimal solution for (5.9), and f_{ij}. Then substituting for y_{ij}^* in (5.8) gives a closed form expression for H_{ij}.

For example, Los considered the case where the constraints (5.3) are omitted (i.e., the model contains no explicit capacity constraints), the constraints (5.6) only restrict the capacity variables y_{ij} to be non-negative, and the routing cost function is given by

$$c_{ij}(f_{ij}, y_{ij}) = \begin{cases} 0 & \text{if } y_{ij} = 0 \\ g_1[1 + g_2(f_{ij}/y_{ij})^{g_3}]f_{ij} & \text{otherwise.} \end{cases} \tag{5.11}$$

As mentioned previously, this cost function is often used to model highway congestion costs. For this problem we determine the solution y_{ij}^* to (5.9) by setting $y_{ij}^* = 0$ if $f_{ij} = 0$ and by setting the derivative of the objective function in (5.9) to zero if $f_{ij} > 0$. Setting this derivative to zero and solving for y_{ij}^* gives

$$y_{ij}^* = f_{ij}(g_1 g_2 g_3/\lambda e_{ij})^{1/(g_3+1)}. \tag{5.12}$$

Substituting into (5.9) yields

$$H_{ij}(f_{ij}) = (g_1[1 + g_2(g_1 g_2 g_3/\lambda e_{ij})^{-g_3/(g_3+1)}] + \lambda e_{ij}(g_1 g_2 g_2/\lambda e_{ij})^{1/(g_3+1)})f_{ij}.$$

Since this expression is linear in the flow variables f_{ij}, the multicommodity flow master problem (5.10) reduces to a set of shortest path problems. After obtaining the optimal values for the f_{ij} from (5.10), we can use (5.12) to derive the optimal values for the y_{ij}. Thus, under the model restrictions assumed by Los, Steenbrink's decomposition scheme can efficiently generate a complete solution to the relaxed problem (5.7).

Dantzig et al. and LeBlanc and Abdulaal describe other variants of the budget design model where the functions H_{ij} are easily characterized and are convex. They focus on applications with a nonlinear version $\sum_{(i,j)\in A} G_{ij}(y_{ij}) \le B$ of the budget constraint (5.5).

Dantzig et al. show that when each budget cost function G_{ij} is convex and the routing cost function is specified as stated in (5.11), the functions H_{ij} are easily described and are also convex. The resulting master problems (5.10) are also convex flow problems. These authors derive similar results when the routing cost functions are piecewise linear and convex for every value of the capacity variable y_{ij}. Computational results for this model required about 300 seconds on an IBM 370/168 computer for a network with 394 nodes, 1042 arcs, and 84 origin nodes. In contrast, a linear programming approach (MORLOK et al.[89]) required over 2400 seconds on an IBM 370/168 computer to solve a design model with 24 nodes and 76 arcs.

Steenbrink[110,111] solved a similar type of problem, but with nonlinear construction costs $\sum_{(i,j)\in A} F_{ij}(y_{ij})$ present in the objective function instead of the budget constraint. The functions H_{ij} are easily derived but are, in general, nonconvex so the master problems (5.10) must be analyzed with a heuristic technique such as an incremental loading traffic assignment procedure (MARTIN AND MANHEIM[85]). Steenbrink applied this method to a Dutch roadway design problem containing 2000 nodes and 6000 arcs. The heuristic procedure required about 3000 seconds of IBM 360/65 computer time. Due to the size of the problem, it is not possible to evaluate the quality of Steenbrink's solution.

LeBlanc and Abdulaal discuss problems similar to the ones considered by Dantzig et al. and specify instances where the budget cost functions G_{ij} are nonconvex, but the functions H_{ij} are easily described and are convex.

McCallum[86] considers a convex routing cost design model concerning the location of circuits in a communication (telephone) network. Each arc routing cost function in his model is piecewise linear and each construction cost function is linear. The model permits only a few paths as flow routes between every pair of nodes and has no budget constraint. McCallum formulates this model as a linear program and uses a specialized implementation of the generalized upper bounding procedure to solve it. His code has solved a problem with 563 arcs and 1857 required flows in 173 seconds on an IBM 370/165 computer.

Note that McCallum's model is a special case of the problem considered by Dantzig et al. and LeBlanc and Abdulaal when their budget constraint is dualized. Recall that the implementation of Steenbrink's decomposition algorithm by Dantzig et al. significantly outperformed a linear programming approach that Morlok et al. used for a problem similar to McCallum's. Therefore, Steenbrink's decomposition is applicable to McCallum's model and should be quite effective in solving it.

All of the continuous capacity variable design problems discussed to this point have assumed that flow patterns are governed by system optimal routing. It is also possible to formulate and study analogous user-optimized design problems.

Abdulaal and LeBlanc[2] reformulate the model as an unconstrained optimization problem where the cost functions are implicitly defined as the total cost of user optimized traffic assignment problems corresponding to fixed values of the capacity variables. Marcotte[83] incorporates the user-optimal routing conditions as constraints by modeling them as variational inequalities. Marcotte[84] analyzes heuristics for this problem by giving convergence results and worst-case error bounds.

Concave Cost Design Problems

A variety of important communication and freight distribution problems can be viewed as concave routing cost problems. For example, Yaged[122] describes a communication network design model

$$\begin{aligned} &\text{minimize} \quad \textstyle\sum_{(i,j)\in A} F_{ij}(y_{ij}) \\ &\text{subject to (5.1), (5.2), (5.5) and } f_{ij} \le y_{ij} \end{aligned} \qquad (5.13)$$

whose component cost functions F_{ij} are concave, reflecting the economies of scale in telephone circuit costs. Since the cost functions F_{ij} are non-negative as well, $f_{ij} = y_{ij}$ in any optimal solution to this problem. Therefore, we can eliminate the y_{ij} variables and reduce the problem to the form

$$\begin{aligned} &\text{minimize} \quad \textstyle\sum_{(i,j)\in A} F_{ij}(f_{ij}) \\ &\text{subject to (5.1), (5.2) and (5.5)} \end{aligned} \tag{5.14}$$

(This problem transformation could also be derived by applying Steenbrink's decomposition procedure to (5.13).)

The application of Steenbrink's decomposition scheme demonstrates that a number of other network design models are equivalent to the concave cost multicommodity flow problem (5.14). For example, in the fixed charge network design problem introduced in Section 2, if $c_{ij} = c_{ij}^k$ for all k, and the problem contains no side constraints on the design variables y_{ij}, then we can substitute $f_{ij} = \sum_k f_{ij}^k$ for y_{ij} in the objective function to give the concave nondifferentiable function

$$F_{ij}(f_{ij}) = \begin{cases} c_{ij}f_{ij} + F_{ij} & \text{if} \quad f_{ij} > 0 \\ 0 & \text{if} \quad f_{ij} = 0. \end{cases}$$

Consequently, the Steenbrink decomposition master problem for this class of models reduces to (5.14) and the fixed charge network design problem and its various special cases, including the traveling salesman problem and the Steiner tree problem on a graph, can all be viewed as concave cost flow problems in the variables f_{ij}. These examples demonstrate that the concave cost flow problem (5.14) is an important and useful model for formulating network design problems. Since (5.14) contains many *NP*-hard problems as special cases, the concave cost problem is also *NP*-hard and, therefore, it is unlikely that there are efficient general solution techniques for the problem. The remainder of this section focuses on various proposed solution techniques for (5.14).

Incremental Improvement Algorithms via Linearization

Since the constraints of (5.14) are linear and its objective function is concave and non-negative, it always has an optimal solution at a vertex of the feasible region. One class of solution techniques exploits this property by either (i) moving from one extreme point to another, or (ii) constructing an extreme point solution. Unfortunately, these adjacent vertex following routines can be suboptimal since (5.14) generally has many local optima. Another class of solution strategies linearizes the objective function and then solves the resulting linear program as a series of shortest-path problems. The process is repeated a number of times with the linearization modified according to the previous iteration's shortest-path solutions. Under certain conditions, it is possible to prove that these linearization schemes converge, but again, only to local optima.

The Frank-Wolfe algorithm and its variants is one type of linearization procedure suitable for problems with differentiable cost functions. Starting with a feasible solution vector $f^1 \equiv (f_{ij}^k)$, the algorithm forms a Taylor first-order linearized approximation to the problem. That is, the objective function coefficient of f_{ij} becomes $\partial F_{ij}/\partial f_{ij}$ evaluated at f^1. If the objective value for the solution f^2 to this linearized problem is negative, then f^2 is a direction of improvement for the objective function. The Frank-Wolfe algorithm would then perform a one-dimensional line search to minimize the objective function in the line segment joining f^1 and f^2. In this case, however, since all the component cost functions $F_{ij}(f_{ij})$ are concave and f^2 is a direction of improvement, the line search always solves at f^2 (see Yaged[122]). The method next forms a new linearized problem using f^2 and then repeats the entire process. The procedure terminates when the point f^1, about which we are linearizing, solves the linearized problem. Since this procedure is the Frank-Wolfe algorithm applied to (5.14), a standard convergence proof shows that it will always converge to a local optimal solution.

Another type of linearization procedure uses an estimate of the average arc flow cost instead of the marginal flow cost $\partial F_{ij}/\partial f_{ij}$. If u_{ij} is an estimate of the total flow on arc (i, j),

$$t_{ij} \cdot f_{ij} \quad \text{where} \quad t_{ij} = F_{ij}(u_{ij})/u_{ij}$$

would approximate the cost function F_{ij}. This linearization can give more accurate cost approximations than the marginal cost procedure when the cost functions F_{ij} contain discontinuities (fixed charges) as in the case of the fixed charge network design problem. Also, this scheme does not require the functions F_{ij} to be differentiable.

The above approximations lead to the linear program:

$$\begin{aligned} &\text{minimize} \quad \textstyle\sum_{(i,j)\in A} t_{ij} f_{ij} \\ &\text{subject to (5.1), (5.2) and (5.5)} \end{aligned}$$

which can be solved efficiently as a series of shortest-path problems. If the optimal value for f_{ij} is u_{ij}, then the linear approximation gives the same flow cost as the concave cost function. Just as in the Frank-Wolfe procedure, the solution to the linear program approximation supplies a feasible solution to the concave cost flow problem (5.14) and defines a new average cost linearization for repeating the procedure. The algorithm terminates when two successive linear program solutions are identical.

It is interesting to note that the average cost linearization scheme is equivalent to the capacity restraint procedure (Martin and Manheim[85]) that was widely used many years ago to solve convex cost multicommodity flow traffic assignment problems. One disadvantage of this approach for both convex and concave cost problems is that its convergence properties are not known. However, empirical evidence (Martin and Manheim,[85] and Yaged[122, 123]) seems to indicate the procedure usually converges to a suboptimal solution.

Yaged[122] performed extensive computational tests with both the marginal cost and average cost linearization schemes. He also tested a third approximation technique that used a linear combination of these two linearizations. His tests, with problems, containing 100 nodes and 210 arcs and concave differentiable cost functions, indicated that all these iterative schemes converged though he gave no computer times or iteration counts. He found that, of the three approximation procedures, it was the most effective to apply the average cost scheme until it terminated and then improve its solution with the marginal cost scheme.

Adjacent Extreme Point Search Methods

One drawback to the linearization approaches is that they fail to fully utilize the property that an optimal solution lies on an extreme point of the feasible region. For example, the marginal cost scheme finds an extreme point solution that is locally optimal relative to all points within a neighborhood. However, neighboring *extreme points* could provide improved solutions. Various authors including ZADEH,[124, 125] DAENINCK AND SMEERS,[19] and GALLO AND SODINI[38, 39] have proposed methods for improving solutions by considering neighboring extreme points.

Zadeh[124, 125] introduced a series of heuristic techniques for improving concave cost flow solutions by rerouting flow about cycles in the network. He says that an arc (i, j) is in the solution f if the flow f_{ij} on it is strictly positive. His methods involve adding and/or deleting arcs from the solution to form and/or destroy cycles in the solution. He also considers rerouting flows about cycles without any arc addition or deletion. For example, consider Zadeh's[124] 5-node cyclic network specified in Figure 4. Each arc (i, j) has a cost function $F_{ij}(f_{ij}) = f_{ij}^{0.3}$. The demands r_{ij} between pairs of nodes i and j are given by $r_{12} = r_{23} = r_{34} = r_{45} = r_{15} = 1$.

Figure 4a gives a feasible problem solution (with cost 5) where the arc labels indicate the arc flows. The marginal cost scheme indicates that no further improvement is possible. However, by deleting arc (1, 2) from the cycle in Figure 4a and rerouting the flow requirement r_{12}, we get the solution in Figure 4b with cost of about 4.92.

Zadeh's computational results indicate that his techniques are usually able to improve upon marginal cost method solutions. The disadvantage with his methods is that the number of possible network cycles grows extremely rapidly with problem size. Therefore, these techniques could require unacceptable amounts of computation time for large networks.

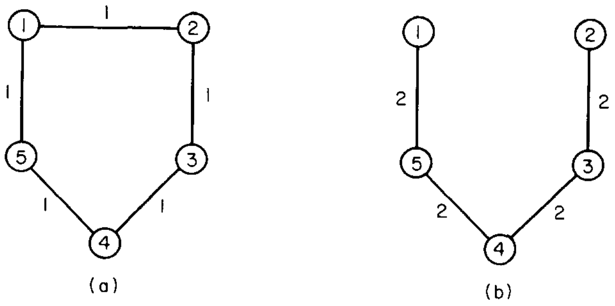

Fig. 4. Concave cost network example.

Other researchers have used more formal techniques for exploiting the extreme point property of concave cost flow problems. Daeninck and Smeers[19] and Gallo and Sodini[38,39] propose procedures that start from an initial extreme point solution and repeatedly move to the best adjacent vertex solution until no further objective function improvements are possible.

We will say that a solution satisfies the *tree property* if for any node v the set of arcs used by flows orginating at that node forms a tree (is cycle free). A solution corresponds to an extreme point if and only if it satisfies the tree property. Daeninck and Smeers and Gallo and Sodini show that moving to an adjacent vertex corresponds to rerouting the flow for a single-origin destination pair while maintaining the tree property. The cost of the best reroute can be computed via a specially derived shortest-path problem. So in order to evaluate all adjacent vertices and to find the best one, the procedure must solve a shortest-path problem for each origin-destination pair.

For the example given by Figure 4a, the flows originating at node 1 (r_{12} and r_{15}) use the arcs (1, 2) and (1, 5). Rerouting the flow between nodes 1 and 2 gives the solution in Figure 4b and the arcs used by r_{12} and r_{15} change to (1, 5), (4, 5), (3, 4), and (2, 3). This set of arcs satisfies the tree property and closer inspection of the other origin nodes shows that the solutions given in Figures 4a and 4b are adjacent extreme points.

These procedures appear most effective when flow originates at a single node since this assumption limits the number of shortest-path problems that the procedure must solve. Computational results with these algorithms have been very promising. For example, Gallo and Sodini, in solving a class of problems with 48 nodes, 174 arcs, 1 origin, and 20 destinations were able to improve the marginal cost algorithm's solutions by an average of 20% in 30 to 60 seconds of IBM 370/168 computer time.

Branch and Bound Procedures

A variety of authors have used discrete optimization techniques such as branch and bound and dynamic programming to exploit the tree property of single commodity concave cost flow problems. These methods are able to compute optimal solutions, but only with the usual cost of increased computational effort.

FLORIAN AND ROBILLARD[33] and Gallo et al.[40] use similar branch and bound approaches to determine which arcs will carry flow in a single commodity concave cost flow problem. Each vertex of the branch and bound tree has two successors where a particular arc is either included in or excluded from the solution. The methods calculate lower bounds from average cost linearizations that are solved as minimum cost flow problems. Gallo et al. are able to optimally solve a problem with 34 nodes, 122 arcs, 1 origin and 10 destinations in 184 seconds on an IBM 370/168 computer.

ACHIM et al.[1] use a very different branch and bound approach for the same problem by adopting FALK AND SOLAND's[29] procedure for nonconvex optimization problems. At each vertex of the branch and bound tree, they solve an average cost linearization (a minimum cost flow problem) to derive a lower bound on the optimal objective value. Suppose that, at the vertex, the feasible set of flow values for an arc (i, j) is the interval $[l_{ij}, u_{ij}]$. Bisecting this interval creates two successors to the vertex that restrict arc (i, j)'s flow to the intervals

$$[l_{ij}, (l_{ij} + u_{ij})/2] \quad \text{and} \quad [(l_{ij} + u_{ij})/2, u_{ij}].$$

Each successor vertex gives rise to a more refined average cost approximation and a more accurate lower bond. The algorithm converts each linear approximation solution into a feasible solution and, consequently, specifies an upper bound for the search process. The bisecting of arc intervals (expansion of the search tree) continues until the difference between the upper and lower bounds is sufficiently small. The procedure has optimally solved a network with 24 nodes, 57 arcs, 4 source nodes, 8 destination nodes and piecewise linear arc cost functions in about 25 seconds on a CDC 6400 computer.

Dynamic Programming

Recently, ERICKSON et al.[27] proposed a dynamic programming approach to single-commodity-concave cost flow problems that generalizes many previous contributions from the literature. For ease of exposition, we restrict our discussion to the case of a single source node supplying a set D of demand nodes. Their method easily generalizes to situations in which a set of sources supply a set of demand nodes.

The tree property given earlier indicates that for an extreme point solution, the set of arcs used to supply the demand set D forms a tree, which we will call a supply tree. Let C_{iI} be the optimal cost of supplying a set of nodes I from supply at node i. Then C_{sD} gives the optimal cost of the entire problem. The C_{iI} values can be computed recursively with the relationships

$$C_{iI} = \min(\min_j[C_{jI} + c_{ij}(r_I)], B_{iI})$$

and

$$B_{iI} = \min_{\phi \subset J \subset I}[C_{iJ} + C_{i,I-J}].$$

The initial conditions are given by

$$C_{iI} = \begin{cases} 0 & \text{if } I = \{i\} \\ \infty & \text{otherwise.} \end{cases}$$

The first equation indicates that the best supply tree rooted at node i satisfies one of two possible conditions. First, node i in the supply tree has a single son j with a total cost given by C_{jI} (the cost of supplying the demand set I from node j) plus $c_{ij}(r_I)$ (the cost of sending the total demand r_I in the set I from node i to node j). The inner minimization selects the best possible single son j.

The second condition corresponds to a supply tree in which root node i has two or more sons. The B_{iI} term represents the optimal splitting of the supply tree rooted at node i into two trees rooted at i supplying J and $I - J$, respectively. The initial conditions indicate the cost of supplying demand node i from node i is zero.

The above dynamic programming recursion essentially requires the computation of C_{iI} for all values of i and all possible subsets I of the demand set D. For the general case, if there are d nodes in D and n nodes in the entire network, the number of C_{iI} computations become proportional to $n2^d$ which grows exponentially with network size in the worst case.

Erickson, Monma and Veinott describe a special class of networks where their procedure has a polynomial time bound. A network is called D-planar if it can be embedded in the plane so that all demand nodes are on the boundary of the outer face. Now assume the demand nodes of a D-planar network are always labeled cyclically $1, 2, \cdots, d$ around the outer face. We define an *interval set* as having either the form $\{q, q + 1, \cdots, q + k\}$ or $\{q, q + 1, \cdots d, 1, 2, \cdots, t\}$ for some q, k, and t. Figure 5 gives a D-planar network where $D = \{1, 2, 3, 4, 5\}$. The sets $\{2, 3, 4\}$ and $\{4, 5, 1, 2\}$ are interval sets, but $\{4, 5, 1, 3\}$ and $\{1, 2, 4\}$ are not interval sets.

Using the extreme point (tree) property of concave cost flow problems,

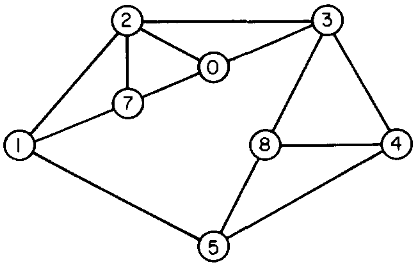

Fig. 5. D-Planar network.

the authors show that, in order to find the optimal solution, the only C_{iI} that must be evaluated are the ones where I is an interval set. Since in the worst case, there are $O(n^2)$ interval sets, the algorithm requires at most $O(n^5)$ computations to find the optimal supply tree.

The above solution approach is applicable to a variety of special network design problems and is as efficient as specialized algorithms for particular problems. A number of distribution system problems can be modeled as single commodity concave cost flow problems on D-planar networks (WAGNER AND WHITIN,[116] ZANGWILL,[126,127] MANNE AND VEINOTT[82] and VEINOTT[115]). The Erickson, Monma and Veinott algorithm presents a unified approach to all of these problems which is as efficient as the algorithms proposed for each special problem. Since the Steiner tree problem on a graph (introduced in Section 2) can be viewed as a single commodity concave cost flow problem on a general network, the above algorithm could be used with an exponential worst-case time bound. This time complexity matches the bound for a special purpose Steiner problem dynamic programming approach proposed by DREYFUS AND WAGNER.[25] In fact, the two dynamic programming approaches are very similar in structure.

Summary

This section has considered a wide variety of network design models and solution techniques for problems with nonlinear routing costs. As is the case with linear cost models, decomposition techniques are especially useful for solving these network design problems. Benders decomposition partitions a design model into a multicommodity flow subproblem and a master problem that selects candidate values for arc capacities. Steenbrink decomposition is another type of resource directive decomposition (GEOFFRION[45]). It utilizes a multicommodity flow master problem (which incorporates the routing and construction costs into a single objective function) and a simple set of subproblems that selects the arc

TABLE IV
Optimization Methods for Network Design with Nonlinear Routing Costs

Author	Problem Type	Solution Algorithm	Computational Experience
Côtés and Laughton[17]	Uncapacitated budget design with convex routing costs	Generalized Benders decomposition (heuristic)	155 nodes, 376 arcs, 25 improvable arcs, 75 seconds; IBM 370/168
Dantzig et al.[22]	Budget design with convex routing costs	Steenbrink decomposition	394 nodes, 1042 arcs, 103 improvable arcs, 300 seconds, IBM 370/168
Haubrick[54]	Uncapacitated design with convex routing costs	Delete heuristic	1250 nodes, 8000 arcs, 2400 seconds; IBM 360/65
Hoang[56]	Uncapacitated budget design with convex routing costs	Generalized benders decomposition	155 nodes, 376 arcs, 25 improvable arcs, 60 seconds; IBM 360/50
LeBlanc[68]	Uncapacitated budget design with convex routing costs	Branch and bound	24 nodes, 76 arcs, 5 improvable arcs, 135 seconds; CDC 6400
LeBlanc and Abdulaal[67]	Budget design with convex routing costs	Steenbrink decomposition	24 nodes, 76 arcs, 28 improvable arcs, 153 seconds; Cyber 70
Morlok and LeBlanc[88]	Uncapacitated budget design with convex routing costs	Add-delete heuristic	24 nodes, 76 arcs, 5 improvable arcs, 18 seconds; Cyber 70
McCallum[86]	Convex routing costs with limited no. of paths	Linear programming generalized upper bounding code	?, 563 arcs, 173 seconds; IBM 370/165
Steenbrink[110]	Convex routing cost	Steenbrink decomposition (heuristic)	2000 nodes, 6000 arcs, 2880 seconds; IBM 360/65
Yaged[122]	Concave capacity costs	Marginal and average cost linearization heuristics	100 nodes, 210 arcs

TABLE V

Optimization Methods for Single Commodity Concave Cost Flow Problems[a]

Authors	Solution Algorithm	Computational Experience
Achim et al.[1]	Branch and bound	24 nodes, 57 arcs, 4 sources, 8 destinations, 25 seconds; CDC 6400
Daeninck and Smeers[19]	Adjacent vertex search heuristic	83 nodes, 234 arcs, 1 origin, 35 destinations, 3.6 seconds; IBM 370/158
Gallo et al.[40]	Branch and bound	34 nodes, 122 arcs, 1 source, 10 destinations, 184 seconds; IBM 370/168
Gallo and Sodini[39]	Marginal cost linear approximation and adjacent vertex search heuristic	48 nodes, 174 arcs, 1 origin, 20 destinations, 40 seconds; IBM 370/168

[a] All problems are single commodity concave cost flow problems.

capacity as a function of arc flow. A crucial factor determining the effectiveness of these schemes is the difficulty of solving the resulting multicommodity flow problem. When the flow problems have convex costs and thus can be solved efficiently, the decomposition schemes have been reasonably successful. When the flow problems have concave or other nonlinear costs, they are much harder to solve and the decomposition scheme usually produces suboptimal results. These results underscore the close relationship between solving network design problems and solving network flow problems.

Since the concave cost flow problem arises in many applied network design problem contexts and is difficult to solve, researchers have been motivated to develop a variety of solution techniques. The linear (marginal cost and average cost) approximation methods are equivalent to methods used to solve convex cost flow problems. Other techniques such as the adjacent vertex search procedure attempt to exploit the property that an optimal solution for a concave cost flow problem is always located at an extreme point. For the special case of a single commodity concave cost flow model, several branch and bound and dynamic programming algorithms are available to solve medium-sized networks.

Tables IV and V summarize the computational results on solving network design models with nonlinear routing costs.

REFERENCES

1. C. ACHIM, M. FLORIAN AND P. ROBILLARD, "Experiments in Solving Concave Cost Network Flow Problems," Department d'informatique, Université de Montreal, 1975.

2. M. ABDULAAL AND L. J. LEBLANC, "Continuous Equilibrium Network Design Models," *Trans. Res.* **13B,** 65–80 (1979).
3. A. ASSAD, "Multicommodity Network Flows—A Survey," *Networks* **8,** 37–91 (1978).
4. M. BARBIER, "Le futur reseau de transports en region de Paris," *Cahiers de l'Institute d'Amenagement et d'Urbanisme de la Region Parisienne* **4,** 4–5 (1966).
5. J. E. BEASLEY, "An Algorithm for the Steiner Problem in Graphs," Department of Management Science, Imperial College, 1982; *Networks* (to appear).
6. M. E. BEESLEY, *Urban Transport: Studies in Economic Policy,* Butterworths, London, 1973.
7. J. F. BENDERS, "Partitioning Procedures for Solving Mixed Variable Programming Problems," *Num. Math.* **4,** 238–252 (1962).
8. J. BILLHEIMER AND P. GRAY, "Network Design with Fixed and Variable Cost Elements," *Trans. Sci.* **7,** 49–74 (1973).
9. T. B. BOFFEY AND A. I. HINXMAN, "Solving for Optimal Network Problem," *Eur. J. Opnl. Res.* **3,** 386–393 (1979).
10. D. E. BOYCE (ed.), *Trans. Res.* **13B,** No. 1, 1–3 (1979).
11. D. E. BOYCE, A. FARHI AND R. WEISCHEDEL, "Optimal Network Problem: A Branch-and-Bound Algorithm," *Environ. Plan.* **5,** 519–533 (1973).
12. D. E. BOYCE AND J. L. SOBERANES, "Solutions to the Optimal Network Design Problem with Shipment Related to Transportation Cost," *Trans. Res.* **13B,** 65–80 (1979).
13. D. G. CANTOR AND M. GERLA, "Optimal Routing in a Packet Switched Computer Network," *IEEE Trans. Comput* **C-23,** 1062–1068 (1974).
14. A. CLAUS AND N. MACULAN, "Une Nouvelle Formulation du Probleme de Steiner sur un Graph," Publication 280, Centre de recherché sur les tranports, Université de Montreal (January), 1983.
15. G. CORNUEJOLS, M. L. FISHER AND G. L. NEMHAUSER, "Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms," *Mgmt. Sci.* **23,** 789–810 (1977).
16. G. CORNUEJOLS, G. L. NEMHAUSER AND L. A. WOLSEY, "Worst-Case and Probabilistic Analysis of Algorithms for a Location Problem," *Opns. Res.* **28,** 847–858 (1980).
17. G. CÔTÉS AND M. LAUGHTON, "Large-Scale Mixed Integer Programming Benders-Type Heuristics," Institut de recherche de l'hydro-Quebec, Varennes, Quebec, Canada, 1981.
18. H. CROWDER AND M. W. PADBERG, "Solving Large Scale Symmetric Travelling Salesman Problems to Optimality," *Mgmt. Sci.* **26,** 495–509 (1980).
19. C. DAENINCK AND Y. SMEERS, "Using Shortest Paths in Some Transshipment Problems with Concave Costs," *Math. Program.* **12,** 18–25 (1977).
20. S. DAFERMOS, "Traffic Assignment and Resource Allocation in Transportation Networks," Ph.D. dissertation (Chapter 3), Johns Hopkins University, 1968.
21. S. DAFERMOS AND A. NAGURNEY, "Sensitivity Analysis for the General

Asymmetric Network Equilibria Problem," Lefschetz Center for Dynamical Systems Report 82-7 (May), 1982.

22. G. DANTZIG, R. HARVEY, Z. LANSDOWNE, D. ROBINSON AND S. MAIER, "Formulating and Solving the Network Design Problem by Decomposition," *Trans. Res.* **13B,** 5–17 (1979).
23. R. DEMBO, "The Design and Implementation of Software for Large-Scale Nonlinear Network Optimization," paper presented at the conference NETFLOW 83, Pisa (April), 1983.
24. R. DIONNE AND M. FLORIAN, "Exact and Approximate Algorithms for Optimal Network Design," *Networks* **9,** 37–59 (1979).
25. S. E. DREYFUS AND R. A. WAGNER, "The Steiner Problem in Graphs," *Networks* **1,** 195–207 (1972).
26. J. EDMONDS, "Optimal Branching," *J. Res. NBS* **71B,** 233–240 (1967).
27. R. E. ERICKSON, C. L. MONMA AND A. F. VEINOTT, "Minimum Concave Cost Network Flows," unpublished manuscript (August), 1981.
28. D. ERLENKOTTER, "A Dual Based Procedure for Uncapacitated Facility Location," *Opns. Res.* **26,** 992–1009 (1978).
29. J. E. FALK AND R. M. SOLAND, "An Algorithm for Separable Non-Convex Programming Problems," *Mgmt. Sci.* **15,** 550–569 (1969).
30. M. L. FISHER, "Worst-Case Analysis of Heuristic Algorithms," *Mgmt. Sci.* **26,** 1–17 (1980).
31. M. L. FISHER AND D. S. HOCHBAUM, "Probabilistic Analysis of the Planar *K*-Median Problem," *Math. Opns. Res.* **5,** 27–34 (1980).
32. M. FLORIAN, "Nonlinear Cost Network Models in Transportation Analysis," Publication 287, Centre de recherché sur les transport, Université de Montreal (March), 1983.
33. M. FLORIAN AND P. ROBILLARD, "An Implicit Enumeration Algorithm for the Concave Cost Network Flow Problem," *Mgmt. Sci.* **18,** 184–193 (1971).
34. M. FLORIAN, G. G. GUERIN AND G. BUSHEL, "The Engine Scheduling Problem on a Railway Network," *INFOR J.* **14,** 121–128 (1976).
35. M. FRANK, "The Braess Paradox," *Math. Prog.* **20,** 283–302 (1981).
36. G. GALLO, "A New Branch-and-Bound Algorithm for the Network Design Problem," Report L81-1, Instituto Di Elaborazione Dell' Informazione, Pisa, Italy, 1981.
37. G. GALLO, "Lower Planes for the Network Design Problem," *Networks,* **13,** 411–426 (1983).
38. G. GALLO AND C. SODINI, "Concave Cost Minimization on Networks," *Eur. J. Opnl. Res.* **3,** 239–249 (1979).
39. G. GALLO AND C. SODINI, "Adjacent Extreme Flows and Application to Min Concave Cost Flow Problems," *Networks* **9,** 95–221 (1979).
40. G. GALLO, C. SANDI AND C. SODINI, "An Algorithm for the Min Concave Cost Flow Problem," *Eur. J. Opnl. Res.* **4,** 248–255 (1980).
41. R. D. GALVAO, "A Dual-Bounded Algorithm for the *p*-Median Problem," *Opns. Res.* **28,** 1112–1121 (1980).
42. M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability: A Guide*

to the Theory of NP Completeness, W. H. Freeman, San Francisco, 1979.

43. W. W. GARVIN, H. W. CRANDALL, J. B. JOHN AND R. A. SPELLMAN, "Applications of Vehicle Routing in the Oil Industry," *Mgmt. Sci.* **3,** 407–430 (1957).
44. B. GAVISH AND S. GRAVES, "Travelling Salesman Problem and Related Problems," Working Paper OR 078-78, Operations Research Center, MIT (July), 1978.
45. A. M. GEOFFRION, "Primal Resource-Directive Approaches for Optimizing Nonlinear Decomposable Systems," *Opns. Res.* **18,** 375–403 (1970).
46. A. M. GEOFFRION, "Generalized Benders Decomposition," *J. Optim. Theory Appl.* **10,** 237–260 (1972).
47. A. M. GEOFFRION AND G. GRAVES, "Multicommodity Distribution System Design by Benders Decomposition," *Mgmt. Sci.* **5,** 822–844 (1974).
48. M. GROTSCHEL AND M. W. PADBERG, "On the Symmetric Traveling Salesman Problem; I. Inequalities," *Math. Program.* **16,** 265–280 (1979).
49. M. GROTSCHEL AND M. W. PADBERG, "On the Symmetric Traveling Salesman Problem; II. Lifting Theorems and Facets," *Math. Program.* **16,** 281–302 (1979).
50. M. GUIGNARD, "Preprocessing and Optimization in Network Flow Problems with Fixed Charges," Department of Statistics, Report 47, Wharton School, University of Pennsylvania, 1982.
51. M. HAIMOVICH, "Asymptotic Properties of Geometric Location Problems," Doctoral dissertation, Department of Aeronautics and Astronautics, M.I.T. (in preparation).
52. S. L. HAKIMI, "Steiner's Problem in a Graph and Its Implications," *Networks* **1,** 113–133 (1971).
53. M. HALL, "Properties of the Equilibrium State in Transportation Networks," unpublished report, Bell Laboratories, Piscataway, N.J., 1976.
54. G. TH. M. HAUBRICH, "De Optimalisering van het spoorwegnet in Nederland. Korte beschrijving van methode en resultaten," *Tijdschrift voor Vervoerswetenschap* **8** (1972).
55. H. H. HOANG, "A Computational Approach to the Selection of an Optimal Network," *Mgmt. Sci.* **19,** 488–498 (1973).
56. H. H. HOANG, "Toplogical Optimization of Networks: A Nonlinear Mixed Integer Model Employing Generalized Benders Decomposition," *IEEE Trans. Automatic Control* **AC-27,** 164–169 (1982).
57. T. C. HU, "Optimum Communication Spanning Trees," *SIAM J. Comput.* **3,** 188–195 (1974).
58. D. S. JOHNSON, J. K. LENSTRA AND A. H. G. RINNOOY KAN, "The Complexity of the Network Design Problem," *Networks* **8,** 279–285 (1978).
59. E. L. JOHNSON, M. PIERI AND P. PIERONI, "Facets for Polyhedra of Fixed Charge Shortest Path Problems," Report 89, Departimento di ricerca operativa e scienze statistiche, University di Pisa, 1982.
60. O. KARIV AND S. L. HAKIMI, "An Algorithmic Approach to Network Location Problem; II. The p-medians," *SIAM J. Appl. Math* **37,** 539–560

(1979).

61. R. M. KARP, "On the Computational Complexity of Combinatorial Problems," *Networks* **5,** 45–68 (1975).
62. R. M. KARP, "Probabilistic Analysis of Partitioning Algorithms for the Traveling Salesman Problem on the Plane," *Math. Opns. Res.* **2,** 209–224 (1977).
63. R. L. KEENEY AND H. RAIFFA, *Decision with Multiple Objectives Preferences and Value Tradeoffs*, Ch. 8, John Wiley & Sons, New York, 1976.
64. J. KENNINGTON, "Multicommodity Flows: A State-of-the-Art Survey of Linear Models and Solution Techniques," *Opns. Res.* **26,** 209–236 (1978).
65. W. KNÖDEL, *Graphentheoretische Methoden und ihre Anwendungen*, Springer-Verlag, Berlin, 1969.
66. A. KOLEN, "Location Problems on Trees and in the Rectilinear Plane," Mathematisch Centrum, 1981.
67. L. J. LEBLANC AND M. ABDULAAL, "An Efficient Dual Approach to the Urban Road Network Design Problem," *Comput. Math. Appl.* **5,** 11–19 (1979).
68. L. J. LEBLANC, "An Algorithm for the Discrete Network Design Problem," *Trans. Sci.* **9,** 283–287 (1975).
69. L. J. LEBLANC, E. K. MORLOK AND W. P. PIERSKALLA, "An Efficient Approach to Solving the Road Network Equilibrium Traffic Assignment Problem," *Trans. Res.* **9,** 309–318 (1975).
70. C. E. LEMKE AND K. SPIELBERG, "Direct Search Zero One and Mixed Integer Programming," *Opns. Res.* **15,** 892–914 (1967).
71. M. LOS, "A Discrete-Convex Programming Approach to the Simultaneous Optimization of Land Use and Transportation." *Trans. Res.* **13B,** 33–48 (1979).
72. M. LOS AND C. LARDINOIS, "Combinatorial Programming, Statistical Optimization and the Optimal Transportation Network Problem," *Trans. Res.* **16B,** 89–124, 1980.
73. M. LOS, private communication, 1979.
74. R. D. MACKINNON, "Optimization Models of Transportation Network Improvement: Review and Further Prospects," Working Paper, IIASA, Luxemburg, Austria (February), 1976.
75. T. L. MAGNANTI AND R. T. WONG, "A Dual Ascent Approach for Network Design Problems" (1983).
76. T. L. MAGNANTI AND B. L. GOLDEN, "Transportation Planning: Network Models and Their Implementation", in *Studies in Operations Management*, A. C. Hax (ed.), North-Holland, Amsterdam, 1978.
77. T. L. MAGNANTI, "Optimization for Sparse Systems," in *Sparse Matrix Computations*, J. R. Bunch and D. J. Rose (eds.), Academic Press, New York, 1976.
78. T. L. MAGNANTI, P. MIREAULT AND R. T. WONG, "Tailoring Benders Decomposition for Network Design," Working Paper OR 125-83, Operations Research Center, Massachusetts Institute of Technology (December), 1983.

79. T. L. MAGNANTI AND R. T. WONG, "Accelerating Benders Decomposition: Algorithmic Enhancement and Model Selection Criteria," *Opns. Res.* **29,** 464–484 (1981).

80. M. MALEK-ZAVAREE AND J. K. AGGARWAL, "Optimal Flow in Networks with Gains and Costs," *Networks* **1,** 355–365 (1972).

81. M. L. MANHEIM, *Fundamentals of Transportation Systems Analysis, Vol. I. Basic Concepts*, MIT Press, Cambridge, Mass., 1979.

82. A. A. MANNE AND A. F. VEINOTT, JR., "Optimal Plant Size with Arbitrary Increasing Time Paths of Demand" in *Investments for Capacity Expansion: Size, Location and Time Phasing*, A. S. Manne (ed.), MIT Press, Cambridge, Mass., 1967.

83. P. MARCOTTE, "An Analysis of Heuristics for the Network Design Problem," Publication 200, Centre de recherché sur les transports, Université de Montreal (May), 1982.

84. P. MARCOTTE, "Network Optimization with Continuous Control Parameters," *Trans. Sci.* **17,** 181–197, 1983.

85. B. V. MARTIN AND M. L. MANHEIM, "A Research Program for Comparison of Traffic Assignment Techniques," *Highway Res. Rec.* **88,** 69–84 (1965).

86. C. J. MCCALLUM, "A Generalized Upper Bounding Approach to a Communications Network Planning Problem," *Networks* **7,** 1–23 (1976).

87. P. B. MIRCHANDANI, A. OUDJIT AND R. T. WONG, "Location Decisions on Stochastic Multidimensional Networks," presented at the International Symposium on Locational Decisions, Skodsborg, Denmark, 1981.

88. E. K. MORLOK AND L. J. LEBLANC, "A Marginal Analysis Technique for Determining Improvements to an Urban Road Network," paper presented at ORSA/TIMS Meeting, Las Vegas, Nevada, Fall 1975.

89. E. K. MORLOK, J. L. SCHOFER, W. P. PIERSKALLA, R. E. MARSTEN, S. K. AGARWAL, J. L. EDWARDS, L. J. LEBLANC, D. T. SPACEK AND J. W. STONER, "Development and Application of a Highway Network Design Model" (Final Report prepared for Federal Highway Administration, Environmental Planning Branch), Dept. of Civil Engineering, Northwestern University, Evanston, Ill., 1973.

90. J. D. MURCHLAND, "Braess' Paradox of Traffic Flow," *Trans. Res.* **4,** 391–394 (1970).

91. J. D. MURCHLAND, "A Fixed Matrix Method for Finding All Shortest Distances in a Directed Graph and for the Inverse Problem," Ph.D. thesis, University of Karlsruhe, 1970.

92. S. C. NARULA, U. I. OGBU AND H. M. SAMUELSSON, "An Algorithm for the *p*-Median Problem," *Opns. Res.* **25,** 709–712 (1977).

93. G. F. NEWELL, "Optimal Network Geometry," in *Sixth International Symposium on Transportation and Traffic Theory*, Elsevier, New York, 1974.

94. S. NGUYEN, "An Algorithm for the Traffic Assignment Problem," *Trans. Sci.* **8,** 203–216 (1974).

95. A. OUDJIT, J. WARD AND R. T. WONG, "Linear Programming Relaxations of Location Problems on Graphs" (in preparation).

96. M. W. PADBERG AND S. HONG, "On the Symmetric Travelling Salesman

Problem: A Computational Study," *Math. Prog. Stud.* **12,** 78–107 (1980).

97. M. W. PADBERG, T. J. VAN ROY AND L. A. WOLSEY, "Valid Linear Inequalities for Fixed Charge Problems," CORE Discussion Paper #8232, Center for Operations Research and Econometrics, Université Catholique de Louvain, 1982.
98. C. H. PAPADIMITRIOU, "Worst-Case and Probabilistic Analysis of a Geometric Location Problem," *SIAM J. Comput.* **10,** 542–557 (1981).
99. A. D. PEARMAN, "The Structure of the Solution Set to Network Optimization Problems," *Trans. Res.* **13B,** 81–90 (1979).
100. J. PLESNIK, "The Complexity of Designing a Network with Minimum Diameter," *Networks* **10,** 33–45 (1980).
101. B. E. PETERSEN, "A Cut-Flow Procedure for Transportation Network Optimization," *Networks* **10,** 33–45 (1980).
102. R. L. RARDIN, AND U. CHOI, "Tighter Relaxations of Fixed Charge Network Flow Problems," Technical Report, No. J-79-18, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta (May), 1979.
103. R. RICHARDSON, "An Optimization Approach to Routing Aircraft," *Trans. Sci.* **10,** 52–71 (1976).
104. W. ROTHENGATTER, "Application of Optimal Subset Selection to Problems of Design and Scheduling on Urban Transportation Networks," *Trans. Res.* **13B,** 49–63 (1979).
105. L. SCHRAGE, "Implicit Representation of Variable Upper Bounds in Linear Programming," *Math. Prog. Stud.* **4,** 118–132 (1975).
106. L. SCHRAGE, "Implicit Representation of Generalized Variable Upper Bounds in Linear Programs," *Math. Program.* **14,** 11–20 (1978).
107. A. J. SCOTT, "The Optimal Network Problem: Some Computational Procedures," *Trans. Res.* **3,** 201–210 (1969).
108. S. STAIRS, "Selecting a Traffic Network," *J. Trans. Econ. Policy* **2,** 218–231 (1968).
109. J. M. STEELE, "Subadditive Euclidean Functions and Nonlinear Growth in Geometric Probability," *Ann. Prob.* **9,** 365–376 (1981).
110. P. A. STEENBRINK, "Transport Network Optimization in the Dutch Integral Transportation Study," *Trans. Res.* **8,** 11–27 (1974).
111. P. A. STEENBRINK, *Optimization of Transportation Networks,* John Wiley & Sons, London, 1974.
112. R. STEINBERG AND W. I. ZANGWILL, "The Prevalence of Braess' Paradox," *Trans. Sci.* **17,** 301–318, 1983.
113. B. C. TANSEL, R. L. FRANCIS AND T. L. LOWE, "Location on Networks: A Survey; Part I. The *p*-Center and *p*-Median Problems; Part II. Exploiting Tree Network Structure," *Mgmt. Sci.* **29,** 482–511 (1983).
114. M. J. TODD, "An Implementation of the Simplex Method for Linear Programming Problems with Variable Upper Bounds," *Math. Prog.* **23,** 34–49 (1982).
115. A. F. VEINOTT, "Minimum Concave-Cost Solution of Leontief Substitution Models of Multi-Facility Inventory Systems," *Opns. Res.* **17,** 262–291 (1969).

116. H. M. WAGNER AND T. M. WHITIN, "Dynamic Version of the Economic Lot Size Model," *Mgmt. Sci.* **5,** 89–96 (1958).
117. J. G. WARDROP, "Some Theoretical Aspects of Road Traffic Research," *Proc. Inst. Civ. Eng., 1, Part II,* 325–362, 1952.
118. P. B. WILSON AND C. J. HUDSON, "Development Validation and Application to the CN Network Model," *PROC,* 61–65 (1970).
119. R. T. WONG, "Probabilistic Analysis of an Optimal Network Program Heuristic," Dept. of Math Sciences, Rensselaer Polytechnic Institute, Troy, N.Y., 1981 (revised version in preparation).
120. R. T. WONG, "Worst-Case Analysis of Network Design Problem Heuristics," *SIAM, J. Alg. Disc. Meth.* **1,** 51–63 (1980).
121. R. T. WONG, "Dual Ascent Approach for Steiner Tree Problems on a Directed Graph," to appear in *Math. Prog.* (1983).
122. B. YAGED, JR., "Minimum Cost Routing for Static Network Models," *Networks,* **1,** 139–172 (1971).
123. B. YAGED, JR., "Minimum Cost Routing for Dynamic Network Models," *Networks,* **3,** 193–224 (1973).
124. N. ZADEH, "On Building Minimum Cost Communication Networks," *Networks,* **3,** 315–331 (1973).
125. N. ZADEH, "On Building Minimum Cost Communication Networks Over Time," *Networks* **4,** 19–34 (1974).
126. W. I. ZANGWILL, "Minimum Concave Cost Flows in Certain Networks," *Mgmt. Sci.* **14,** 429–450 (1968).
127. W. I. ZANGWILL, "A Backlogging Model and a Multi-Echelon Model of a Dynamic Economic Lot Size Production System—A Network Approach," *Mgmt. Sci.* **15,** 506–527 (1969).
128. E. ZEMEL, "Measuring the Quality of Approximate Solutions to Zero-One Programming Problems," *Math. Opns. Res.* **6,** 319–332 (1981).