Modeling and Heuristic Worst-case Performance Analysis of the Two-level Network Design Problem

Anantaram Balakrishnan,
Thomas L. Magnanti
and
Prakash Mirchandani

WP# 3498-92-MSA November, 1992

Modeling and Heuristic Worst-case Performance Analysis of the Two-level Network Design Problem

Anantaram Balakrishnan†
Thomas L. Magnanti

Sloan School of Management
M. I. T.
Cambridge, MA

Prakash Mirchandani*

Katz Graduate School of Business
University of Pittsburgh
Pittsburgh, PA

Revised: November 1992

---

† Supported in part by a grant from the AT&T Research Fund
* Supported in part by a Faculty Grant from the Katz Graduate School of Business, University of Pittsburgh

Abstract

This paper studies a multi-facility network synthesis problem, called the Two-level Network Design (TLND) problem, that arises in the topological design of hierarchical communication, transportation, and electric power distribution networks. The nodes of a multi-level network have varying levels of importance; more critical or higher level nodes require more expensive higher grade interconnections. Given an undirected network with L possible facility types for each edge, and a partition of the nodes into L levels, the multi-level network design problem seeks a connected design that minimizes total cost while spanning all the nodes, and connecting nodes at each level via facilities of the corresponding or higher type. The TLND problem is a special case of multi-level network design with L = 2. This problem generalizes the well-known Steiner network problem and the hierarchical network design problem. In this paper, we study the relationship between alternative model formulations for this problem, and analyze the worst-case performance for a composite TLND heuristic based upon Steiner and spanning tree computations. When the ratio of higher to lower grade facility costs is the same for all edges, the worst-case performance ratio of the TLND heuristic is 4/3 if we can solve an embedded Steiner network problem optimally. For other cases, we express the TLND heuristic worst-case ratio in terms of the performance ratio of the Steiner solution method. A companion paper develops and tests a dual ascent procedure that generates tight upper and lower bounds on the optimal value of the multi-level network design problem.

Keywords: Network design, integer programming, problem formulation, valid inequalities, worst-case analysis of heuristics

1. Introduction

1.1 Motivation

This paper studies a multi-facility, network synthesis problem which we call the **Multi-level Network Design (MLND)** problem. The problem, which generalizes several well-known optimization models, addresses design decisions for hierarchical telecommunications, transportation, and electric power distribution networks. The nodes in the network have different levels of importance, with more critical or higher level nodes requiring higher grade (e.g., higher capacity or more reliable), but more expensive, interconnections. Designing the topology for such hierarchical networks motivates the following MLND problem. The problem input consists of an undirected graph whose nodes are partitioned into *L levels*; each edge can contain one of L different *facility types*, with higher grade facilities requiring higher fixed costs. We must select a connected subgraph, and choose a facility type for each selected edge so that all nodes at any level *l* communicate via facilities of grade *l* or higher. The objective is to minimize the total cost of the chosen facilities. We refer to the special version of the MLND problem containing only two node levels—*primary* and *secondary* nodes—as the **Two-level Network Design (TLND)** problem.

Two-level network design problems have considerable economic significance. Consider, for instance, the telecommunications context. With increasing demand for higher bandwidth telecommunication services, regional telephone companies are rapidly modernizing their metropolitan networks by replacing copper cables with fiber optic systems. For instance, from 1987 to 1991, the regional telephone companies in the United States nearly quadrupled their deployment of fiber optic equipment to an installed base of over 8.2 million fiber-kilometers. Total U. S. sales of fiber optic cables and systems was approximately $ 1.7 billion in 1991, and the demand is expected to grow further as the deployment of fiber optics in local loops, metropolitan area networks, and cable television systems increases (U.S. Industrial Outlook 1992). Since network modernization entails enormous investments, planners require new models and methods to design cost-effective two-level networks combining fiber optic transmission systems (primary facilities) and copper cables (secondary facilities). The switching centers and certain critical customers (e.g., large businesses) are primary nodes, and households are secondary nodes. Because primary nodes have greater traffic volume and higher transmission frequency, the network must connect them using fiber optic systems (which have higher bandwidth, but also higher cost). Secondary nodes, on the other hand, might access the network via either

fiber or copper cables. When the fixed cable installation costs dominate (relative to throughput-dependent costs), this network configuration problem reduces to the TLND problem.

Similar applications arise in road network planning and electric power distribution planning (see, for instance, Patel [1979], Current, ReVelle, and Cohon [1986]). In the transportation context, all-weather highways and rough roads might represent primary and secondary facilities, with major cities and rural communities serving as primary and secondary nodes. For electric power distribution, the primary and secondary facilities correspond to high and low voltage transmission lines.

Admittedly, the TLND problem or its multi-level generalization might not completely capture all the complexities of the actual design task. For instance, the model incorporates only a coarse representation of capacity constraints via the discrete facility types. However, TLND solutions provide insights and principled starting points for an overall network design exercise.

The TLND model also has theoretical significance because it generalizes several classical discrete and network optimization models. For instance, the model generalizes the **Hierarchical Network Design (HND)** problem, defined by Current et al. [1986]. The HND problem designates exactly **two** nodes of the network as primary nodes. Its solution consists of a primary path connecting these two nodes; secondary edges connect the remaining nodes to this path. The TLND problem also generalizes the Steiner Network problem (Dreyfus and Wagner [1972]) which, in turn, generalizes the shortest path and minimum spanning tree problems. To model the Steiner network problem, we treat the terminal vertices as primary nodes in the TLND problem, designate the potential Steiner vertices as secondary nodes, and use zero secondary costs for all edges (the primary costs are the original edge lengths in the Steiner problem). Deleting all the secondary edges from the optimal TLND solution gives the minimum cost Steiner network.

1.2 Previous research

Iwainsky [1985] introduced the multi-level network design problem, and Duin and Volgenant [1989] showed how to transform this problem (which they call the *Multi-weighted Steiner tree problem*) into an equivalent directed Steiner tree model over an expanded network. For a MLND problem with n nodes, m undirected edges, and L levels, the expanded network contains nL nodes, and (2Lm + Ln) directed arcs. Researchers have

extensively studied two special cases of the TLND problem: the Steiner network problem and the HND problem. The vast literature on the *Steiner network problem* (see Winter [1987] for a recent survey) addresses issues of model formulation and polyhedral representations (e.g., Prodon, Liebling and Gröflin [1985], Chopra and Rao [1988a] [1988b]), worst-case analysis of heuristics (e.g., Takahashi and Matsuyama [1980], Kou, Markowsky, and Berman [1981], Goemans and Bertsimas [1990]), and computational testing of optimization-based solution methods (e.g., Wong [1984], Beasley [1984], [1989]). The literature on the *HND problem* is relatively recent. Current et al. [1986], Shier [1991], and Pirkul, Current, and Nagarajan [1991] describe heuristic solution methods, and Orlin [1991] analyzes the problem's computational complexity and heuristic worst-case performance. Duin and Volgenant [1989] describe methods to identify optimal edges and eliminate variables from the HND problem formulation, and present computational results to demonstrate the effectiveness of these reduction strategies. For the *TLND problem*, Duin and Volgenant [1991] propose two approximate methods that are analogous to previous Steiner tree heuristics, and discuss problem reduction tests.

The TLND problem is NP-hard since it generalizes the Steiner network problem (Garey and Johnson [1979]). Orlin [1991] showed that even the HND special case is NP-hard. Furthermore, the HND problem remains NP-hard even when all the edges have the same primary-to-secondary cost ratio, or if all the edges have unit primary costs and binary secondary costs (Orlin [1991]). This paper considers modeling issues for the TLND problem, and develops worst-case bounds for a combined heuristic based on Steiner and spanning tree solutions. A companion paper (Balakrishnan, Magnanti and Mirchandani [1992]) develops and tests an algorithm that combines problem preprocessing, dual ascent, and local improvement to approximately solve the MLND problem. Using this method, we have solved large-scale TLND problems containing up to 500 nodes and 5000 edges to within 0.9% of optimality; the mixed integer formulation for our largest test problem contains 20,000 integer variables and over 5 million constraints.

This paper is organized as follows. Section 2 introduces our notation and presents two related integer programming formulations for the undirected TLND problem—a Steiner–Spanning tree formulation and a multicommodity flow-based formulation. We also describe a class of inequalities, called the bidirectional commodity-pair inequalities, that produce a considerably stronger "enhanced" linear programming relaxation. In Section 3, we consider a more compact formulation corresponding to the directed version of the TLND problem, and show that this formulation has the same optimal linear programming

value as the enhanced undirected formulation. Section 4 describes several natural heuristic strategies based upon minimum spanning tree and Steiner tree solutions (Duin and Volgenant's [1991] heuristics are particular implementations of these strategies), and derives worst-case performance bounds for a combined TLND heuristic. For problem instances with proportional primary and secondary costs (i.e., the ratio of primary to secondary cost is the same for all edges), our method's worst-case bound is 4/3 if we solve an embedded Steiner tree problem exactly; if we use a Steiner heuristic with worst-case bound of ρ, the TLND worst-case bound is ρ if $\rho \geq 2$, or $4/(4-\rho)$ if $\rho < 2$. The TLND worst-case performance ratio increases to $(\rho+1)$ for problems with nonproportional costs. As part of our analysis, we provide worst-case examples to show that these bounds are tight. Section 5 offers some concluding comments.

2. Modeling the Undirected TLND Problem

The TLND problem is defined over an undirected network G=(N,E) with nodes partitioned into two subsets—*primary (level 1) nodes* and *secondary (level 2) nodes*. Let p denote the number of primary nodes. For convenience, we index the primary nodes from 1 to p, and the secondary nodes from (p+1) to n. Every candidate edge {i,j} in E has a *primary cost* a_{ij} and a *secondary cost* b_{ij}, with $a_{ij} \geq b_{ij} \geq 0$. We assume, without loss of generality, that each edge can contain either facility type. If the problem context prohibits edge {i,j} from containing a primary facility, we can set the primary cost a_{ij} to a very high value; similarly, setting $b_{ij} = a_{ij}$ permits us to model edges that can contain only primary facilities.

The TLND problem seeks a tree that spans all the nodes and contains a subtree of primary facilities connecting all the primary nodes. This primary subtree might (optionally) span some secondary nodes. Note that if all the nodes are primary nodes or if the primary cost equals the secondary cost for all edges, the TLND problem reduces to the *minimum spanning tree* problem. At the other extreme, the *shortest path* problem corresponds to the special case in which the network contains only two primary nodes, and all secondary costs are zero; with more than two primary nodes, this model becomes a *Steiner network* problem.

To formulate the TLND problem as an integer program, we first represent it as two linked subproblems—a Steiner tree subproblem and a spanning tree subproblem. We then expand (in Section 2.2) the Steiner and spanning tree constraints in terms of binary design

variables and continuous flow variables to obtain a basic flow-based formulation. Section 2.3 describes some valid inequalities to strengthen this formulation. Using a small example, Section 2.4 demonstrates how these additional inequalities significantly improve the optimal value of the linear programming relaxation. In Section 3, we transform the undirected problem into a directed problem, and prove that the linear programming relaxation of the directed formulation has the same optimal objective function value as the linear programming relaxation for the enhanced undirected formulation. This result enables us to apply a dual ascent method for the directed problem, which is easier to describe and implement (Balakrishnan et al. [1992]).

2.1 Steiner–Spanning Tree (S–ST) Formulation

This problem formulation exploits the following two observations concerning the optimal TLND solution:

(i) the optimal design is a spanning tree of the original graph G (since all costs are nonnegative), and

(ii) the edges containing primary facilities constitute a Steiner tree, with the primary nodes as terminals, that is embedded in the spanning tree (since $a_{ij} \geq b_{ij}$).

Correspondingly, we have two sets of binary decision variables:

u_{ij} = 1 if edge {i,j} contains a *primary* facility, and
0 otherwise.

w_{ij} = 1 if edge {i,j} belongs to the optimal design, and
0 otherwise.

We let $e_{ij} = a_{ij} - b_{ij} \geq 0$ denote the **incremental cost** of edge {i,j}.

Let U be the set of all Steiner trees with primary nodes as terminals (and secondary nodes as Steiner points). $u = \{u_{ij}\}$ is the characteristic vector of a Steiner tree in U, i.e., $u_{ij} = 1$ if edge {i,j} belongs to the Steiner tree, and $u_{ij} = 0$ otherwise. Similarly, let W be the set of all spanning trees of the graph G, and w denote the characteristic vector of a spanning tree. The TLND problem then has the following **Steiner–Spanning tree (S–ST) formulation:**

[S–ST] minimize $\sum_{\{i,j\} \in E} (e_{ij}\, u_{ij} + b_{ij}\, w_{ij})$ (2.1)

subject to

Steiner tree constraints:

$u \in U,$ (2.2)

Spanning tree constraints:

$w \in W,$ (2.3)

Linking constraints:

$u_{ij} \leq w_{ij}$ for all $\{i,j\} \in E$, and (2.4)

Integrality constraints:

$u_{ij}, w_{ij} = 0 \text{ or } 1$ for all $\{i,j\} \in E.$ (2.5)

The objective function (2.1) minimizes the secondary cost for the spanning tree w and the incremental cost of the Steiner subtree u. Constraints (2.2) and (2.3) specify that the primary subnetwork must be a Steiner tree while the overall network is a spanning tree. The linking constraints (2.4) ensure that the Steiner tree is embedded in the selected spanning tree.

The S–ST formulation extends easily to the general MLND problem with more than two levels. Consider L different sets U^l of Steiner trees, one for each level $l = 1, 2, \ldots,$ L of the network; the set U^l contains all Steiner trees using level l or higher level nodes as terminals (in our terminology a higher level has a lower index l). Correspondingly, for each edge $\{i,j\}$, we have L different design variables u^l_{ij}, for $l = 1, 2, \ldots,$ L. The linking constraints are:

$$u^l_{ij} \leq u^{l+1}_{ij} \quad \text{for all edges } \{i,j\} \in E, \text{ and } l = 1, 2, \ldots, L\text{–}1.$$

The objective function coefficient e^l_{ij} for variable u^l_{ij} equals the difference in cost between the level l facility and the level $(l+1)$ facility on edge $\{i,j\}$.

Note that for the HND special case (which contains only two primary nodes), the Steiner tree component of the S–ST formulation reduces to a shortest path restriction, i.e., U is the set of all simple paths in the network connecting the two primary nodes. Also, if we omit the linking constraints (2.4), the formulation decomposes into two independent subproblems: a Steiner tree subproblem (over the primary nodes) using the incremental edge costs e_{ij}, and a spanning tree subproblem using the secondary costs b_{ij}. The sum of the optimal values for these two subproblems, therefore, provides a lower bound on the

optimal value of the TLND problem. We exploit this observation in Section 4 when we derive heuristic worst-case bounds for the TLND problem.

2.2 Basic Undirected Flow-based Formulation

This section reformulates the TLND problem by expanding the set constraints (2.2) and (2.3) in the [S–ST] model using multi-commodity flow formulations of the Steiner tree and spanning tree subproblems. To formulate these subproblems in terms of network flows, we introduce (n–1) unit demand *commodities*, all originating at a common root node; for convenience, we designate the primary node 1 as the root node. We index the commodities from 2 to n, and impose flow constraints for commodity k = 2, 3,..., n requiring that we send one unit of flow from node 1 to node k. We refer to commodities 2 to p (i.e., commodities with primary nodes as destinations) as **primary commodities**, and commodities (p+1) to n as **secondary commodities**, and let P and S denote the set of primary and secondary commodities. To determine the routing of each commodity k, we introduce *directed* (continuous) flow variables f_{ij}^k and f_{ji}^k for each edge {i,j}. The variable f_{ij}^k (f_{ji}^k) denotes the fraction of commodity k's (unit) demand flowing from node i to node j (from node j to node i). We next expand the Steiner tree and spanning tree constraints (2.2) and(2.3) using these flow variables and the design variables u_{ij} and w_{ij}.

First, consider the Steiner tree constraints (2.2). By definition, the Steiner tree must span all the primary nodes, i.e., it must provide an origin-to-destination flow path for every *primary* commodity. This requirement translates into the following flow conservation and forcing constraints (Wong [1984] proposed this Steiner tree formulation):

Steiner tree flow conservation equations:

$$\sum_{i \in N} f_{ij}^k - \sum_{i \in N} f_{ji}^k = \begin{cases} -1 & \text{if } j = 1 \\ 1 & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases} \quad \text{for all } j \in N,\ k \in P, \qquad (2.2a)$$

Steiner tree forcing constraints:

$$f_{ij}^k \leq u_{ij}, \text{ and}$$
$$f_{ji}^k \leq u_{ij} \quad \text{for all } \{i,j\} \in E,\ k \in P, \qquad (2.2b)$$

Nonnegativity constraints:

$$f_{ij}^k, f_{ji}^k \geq 0 \quad \text{for all } \{i,j\} \in E,\ k \in P. \qquad (2.2c)$$

The Steiner tree forcing constraints (2.2b) ensure that primary commodities flow (in either direction) only on edges {i,j} containing primary facilities, i.e., only if $u_{ij} = 1$.

We can rewrite the spanning tree constraints (2.3) using an analogous flow formulation. The spanning tree must carry one unit of flow from the root node to every other node of the network; an edge {i,j} can carry flow only if we include it in the design (i.e., only if $w_{ij} = 1$). The following constraints express these conditions.

Spanning tree flow conservation equations:

$$\sum_{i \in N} f_{ij}^k - \sum_{i \in N} f_{ji}^k = \begin{cases} -1 & \text{if } j = 1 \\ 1 & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases} \quad \text{for all } j \in N,\ k \in P \cup S, \qquad (2.3a)$$

Spanning tree forcing constraints:

$$\begin{aligned} f_{ij}^k &\leq w_{ij}, \text{ and} \\ f_{ji}^k &\leq w_{ij} \end{aligned} \quad \text{for all } \{i,j\} \in E,\ k \in P \cup S, \qquad (2.3b)$$

Nonnegativity constraints:

$$f_{ij}^k,\ f_{ji}^k \geq 0 \quad \text{for all } \{i,j\} \in E,\ k \in P \cup S. \qquad (2.3c)$$

Replacing constraint (2.2) with (2.2a), (2.2b) and (2.2c), and constraint (2.3) with (2.3a), (2.3b), and (2.3c) in formulation [S–ST], and eliminating the redundant constraints (2.2a) and (2.2c) for $k \in P$, we obtain the expanded S–ST formulation containing the following constraints:

- flow conservation equations (2.3a) for each commodity at every node;
- Steiner tree forcing constraints (2.2b) for all edges;
- spanning tree forcing constraints (2.3b) for all edges;
- linking constraints (2.4) for all edges;
- nonnegativity constraints (2.3c) for all the flow variables; and,
- integrality restrictions (2.5) for all the design variables.

Consider the following change of variables in the expanded S–ST formulation: for every edge {i,j}, we introduce a new decision variable v_{ij}, and replace the spanning tree edge selection variable w_{ij} with $(u_{ij} + v_{ij})$. Since both w_{ij} and u_{ij} are binary variables and since $w_{ij} \geq u_{ij}$ (constraint (2.4)), v_{ij} is also binary, and has the following interpretation:

$$v_{ij} = \begin{cases} 1 & \text{if edge } \{i,j\} \text{ contains a } \textit{secondary} \text{ facility, and} \\ 0 & \text{otherwise.} \end{cases}$$

We refer to u_{ij} and v_{ij}, respectively, as *primary* and *secondary edge selection variables*. Substituting for w_{ij} in the linking constraint (2.4) gives

$$u_{ij} \leq u_{ij} + v_{ij},$$

which reduces to nonnegativity constraints for the v variables. Constraints (2.2b) remain unchanged, while constraint (2.3b) becomes

$$\begin{aligned} f_{ij}^k &\leq u_{ij} + v_{ij}, \text{ and} \\ f_{ji}^k &\leq u_{ij} + v_{ij} \qquad \text{for all } \{i,j\} \in E, k \in S. \end{aligned}$$

These constraints specify that a secondary commodity k can flow in either direction on edge {i,j} only if this edge contains a primary or secondary facility. Finally, with the change of variables, the primary cost a_{ij} replaces the incremental cost e_{ij} as the objective coefficient of u_{ij}, and variable v_{ij} has the secondary cost b_{ij} as its objective coefficient. We refer to this revised formulation as the **Basic Undirected Flow-based (BUF)** formulation .

[BUF] minimize $$\sum_{\{i,j\} \in E} a_{ij} u_{ij} + \sum_{\{i,j\} \in E} b_{ij} v_{ij} \qquad (2.6)$$

subject to

Commodity flow conservation:

$$\sum_{i \in N} f_{ij}^k - \sum_{i \in N} f_{ji}^k = \begin{cases} -1 & \text{if } j = 1 \\ 1 & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases} \qquad \text{for all } j \in N, k \in P \cup S, \qquad (2.7)$$

Primary forcing constraints:

$$\begin{aligned} f_{ij}^k &\leq u_{ij}, \text{ and} && (2.8a) \\ f_{ji}^k &\leq u_{ij} \qquad \text{for all } \{i,j\} \in E, k \in P, && (2.8b) \end{aligned}$$

Secondary forcing constraints:

$$\begin{aligned} f_{ij}^k &\leq u_{ij} + v_{ij}, \text{ and} && (2.9a) \\ f_{ji}^k &\leq u_{ij} + v_{ij} \qquad \text{for all } \{i,j\} \in E, k \in S, && (2.9b) \end{aligned}$$

Nonnegativity, integrality:

$$\begin{aligned} u_{ij}, v_{ij} &= 0 \text{ or } 1 \qquad \text{for all } \{i,j\} \in E, && (2.10a) \\ f_{ij}^k, f_{ji}^k &\geq 0 \qquad \text{for all } \{i,j\} \in E, k \in P \cup S. && (2.10b) \end{aligned}$$

The flow-based model easily accommodates variable (flow-dependent) costs (these costs appear as objective function coefficients for the flow variables f_{ij}^k), and also extends

to the more general multi-level network design problem. Like the spanning tree and Steiner network problems, the TLND problem also has several alternate formulations. For instance, we can reformulate the problem in cutset form using only the primary and secondary design variables (see Aneja [1980] and Chopra and Rao [1988a] for cutset formulations of the Steiner network problem). This formulation contains an exponential number of constraints (corresponding to all possible cutsets in the graph). A second variant would express the connectedness constraint by defining a different commodity for every pair of nodes in the network.

For a network with n nodes and m edges, formulation [BUF] has O(m) binary variables, O(mn) flow variables, $O(n^2)$ flow conservation constraints (2.7), and O(mn) forcing constraints (2.8) and (2.9). In the next section, we describe model enhancements that strengthen the formulation but increase the number of forcing constraints to $O(mn^2)$.

2.3 Strengthening the Undirected Flow-based Formulation

As we illustrate in Section 2.4, the basic undirected flow-based formulation [BUF] has a relatively weak linear programming relaxation, making it unsuitable for LP-based solution methods such as the dual ascent procedure we develop in Balakrishnan et al. [1992]. To strengthen this relaxation, we describe a class of additional valid inequalities which we call the **bidirectional commodity-pair forcing constraints.** As their name suggests, these constraints contain flow variables for pairs of commodities flowing in opposite directions on each edge; they replace the primary and secondary forcing constraints (2.8) and (2.9) of formulation [BUF].

Let us first consider the primary forcing constraints (2.8) in formulation [BUF]. To strengthen these constraints, we exploit the following property of the optimal TLND solution. Since the primary and secondary costs are nonnegative and since all commodities share the same origin, the TLND problem has an optimal tree solution that routes all commodities flowing on an edge in the same direction on that edge. In particular, if this solution routes a pair of primary commodities k and h on edge (i,j), then both commodities must flow either from node i to node j or from node j to node i. This observation motivates the following stronger forcing constraints, which we call the primary commodity-pair forcing constraints or **P-P forcing constraints**:

$$f_{ij}^{k} + f_{ji}^{h} \leq u_{ij} \qquad \text{for all } (i,j) \in E,\ k,h \in P. \qquad (2.11)$$

The same principle also applies to commodity pairs containing secondary commodities. However, if either commodity k and/or commodity h is a secondary commodity, we must add the secondary design variable v_{ij} to the right-hand side. Thus, for mixed (primary and secondary) commodity pairs we add the **P-S forcing constraints**

$$f_{ij}^k + f_{ji}^h \leq u_{ij} + v_{ij} \quad \text{for all } \{i,j\} \in E, k \in P \ \& \ h \in S \text{ or } k \in S \ \& \ h \in P. \quad (2.12)$$

For pairs of secondary commodities, we replace the secondary forcing constraints (2.9) in formulation [BUF] with the **S-S forcing constraints**

$$f_{ij}^k + f_{ji}^h \leq u_{ij} + v_{ij} \quad \text{for all } \{i,j\} \in E, k,h \in S. \quad (2.13)$$

Let **[EUF]** denote the **Enhanced Undirected Flow-based** formulation containing these three sets of constraints instead of the single-commodity forcing constraints (2.8) and (2.9).

The bidirectional commodity-pair forcing constraints (2.11), (2.12), and (2.13) strengthen the original forcing constraints since they contain additional flow variables on the left-hand side. However, since these constraints apply to every *pair* of commodities, the enhanced formulation for a network with n nodes and m edges contains $O(mn^2)$ forcing constraints rather than the O(mn) forcing constraints in the basic formulation. For the largest network size that we tested in our computational study (Balakrishnan et al. [1992]), formulation [EUF] contains more than 450 million constraints.

Researchers have previously proposed bidirectional commodity-pair forcing constraints for related problems. Unlike our P-P, P-S, and S-S forcing constraints (2.11)–(2.13) which incorporate the MLND problem's multiple commodity (and facility) types, the previous constraints consider only a single commodity type. Magnanti and Wong [1981] described commodity-pair forcing constraints for the *uncapacitated network design problem*; Balakrishnan, Magnanti and Wong [1989] incorporated these constraints in a dual ascent algorithm. Martin [1986] showed that adding the commodity-pair forcing constraints to the undirected multicommodity flow formulation of the *minimum spanning tree problem* gives an exact formulation, i.e., the LP relaxation of this formulation has integer extreme points (this property does not hold for the TLND problem).

2.4 Impact of Adding the Commodity-pair Forcing Constraints

Using the simple 6-node example shown in Figure 1, we illustrate the impact of successively adding the commodity-pair forcing constraints on the linear programming lower bound. This example has 3 primary nodes (nodes 1, 2, and 3) and 3 secondary nodes. The numbers on each arc denote the corresponding primary and secondary costs. Primary nodes are shaded circles, and secondary nodes are hollow circles.

Figures 2(a) through 2(e) depict the optimal linear programming solutions as we progressively strengthen the basic undirected formulation [BUF] by adding the P-P forcing constraints, the S-S forcing constraints, and the P-S forcing constraints. In these figures, dark and light lines represent, respectively, primary and secondary edges with positive LP solution values. Solving the LP relaxation of the basic formulation [BUF] gives the solution shown in Figure 2(a) with a cost of 78.5; this solution violates (for example) the P-P forcing constraints on edge {2,3}. Adding the P-P constraints (for all edges) increases the optimal LP value to 88.5, and gives the solution shown in Figure 2(b). Contrast this solution with the optimal values (Figure 2(c)) obtained by enforcing integrality for only the primary design variables u_{ij} (and keeping the secondary design variables continuous); this mixed integer program has an optimal value of 91.

The solution in Figure 2(b) (with the P-P forcing constraints added to formulation [BUF]) violates the S-S forcing constraints for edges {4,5} and {5,6}. Adding the S-S forcing constraints for all edges eliminates this solution. However, the new optimal LP solution, shown in Figure 2(d), still contains fractional values. In particular, this solution has a cost of 101 and violates the P-S forcing constraint for edge {3,6}. Finally, when we add all the P-S forcing constraints (i.e., we use the complete enhanced formulation [EUF]), the optimal LP solution is integral, and is therefore optimal for the original problem as well; Figure 2(e) shows this optimal TLND solution.

For this example, the lower bound progressively increases from the basic LP value of 78 to the optimal IP value of 106, fully eliminating the original integrality gap of 35% as we successively introduce the commodity-pair forcing constraints (2.11), (2.12) and (2.13). This example also shows that none of these three classes of commodity-pair forcing constraints is redundant. We next describe a compact directed problem formulation, and prove that this formulation has the same optimal linear programming value as the enhanced undirected formulation [EUF].

3. The Directed TLND Model

The **Directed TLND problem** seeks a minimum cost directed spanning arborescence rooted at a specified primary node; this arborescence must contain a rooted subtree of primary arcs that spans all the primary nodes (and optionally includes secondary nodes). Given an undirected TLND problem, we consider an equivalent directed TLND problem by

(i) choosing an arbitrary primary node, say node 1, as the root node, and,

(ii) replacing each undirected edge {i,j} in the given graph with two *directed arcs* (i,j) and (j,i), both having the same primary and secondary costs (a_{ij} and b_{ij}) as the original edge.

Let A denote the set of arcs in this directed network. Since the primary and secondary arc costs are both nonnegative, the directed TLND problem defined over the transformed network has an optimal solution that selects at most one of the arcs (i,j) and (j,i). Therefore, ignoring the arc directions in this solution gives the optimal undirected solution with the same total cost. Note that this transformation is valid only for problems with a single-source (or single-destination) commodity flow pattern. For problem contexts requiring multicommodity flows (with associated flow costs) between multiple origins and destinations, the directed model is not equivalent to the undirected model because it double counts the fixed cost for edges that carry flow in both directions.

3.1 The Directed Flow-based Formulation

To formulate the directed TLND problem as a mixed-integer program, as in the undirected problem, we use a commodity flow pattern with n–1 unit demand commodities, all originating at the common primary root node 1. As before, the formulation uses directed (continuous) commodity flow variables f_{ij}^k, for all arcs (i,j) $\in$ A, denoting the proportion of commodity k's demand flowing from node i to node j. The formulation contains directed (binary) arc selection variables x_{ij} and y_{ij} for each arc (i,j) $\in$ A. The *primary arc selection* variable x_{ij} has value 1 if we select arc (i,j) as a primary arc, and 0 otherwise. The *secondary arc selection* variable y_{ij} has value 1 if we install a secondary facility on arc (i,j), and 0 otherwise. We obtain the following **Directed Flow-based** formulation, denoted **[DF]**, by replacing the primary and secondary edge selection variables u_{ij} and v_{ij} in the basic undirected flow-based formulation [BUF] with the directed arc selection variables x_{ij} and y_{ij}.

[DF] minimize $\sum_{(i,j)\in A} a_{ij} x_{ij} + \sum_{(i,j)\in A} b_{ij} y_{ij}$ (3.1)

subject to

Commodity flow conservation:

$$\sum_{i\in N} f_{ij}^k - \sum_{i\in N} f_{ji}^k = \begin{cases} -1 & \text{if } j = 1 \\ 1 & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases} \quad \text{for all } j\in N,\ k\in P\cup S, \qquad (3.2)$$

Primary forcing constraints

$$f_{ij}^k \leq x_{ij} \quad \text{for all } (i,j)\in A,\ k\in P, \qquad (3.3)$$

Secondary forcing constraints

$$f_{ij}^k \leq x_{ij} + y_{ij} \quad \text{for all } (i,j)\in A,\ k\in S, \qquad (3.4)$$

Nonnegativity, integrality:

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

$$f_{ij}^k \geq 0 \quad \text{for all } (i,j)\in A,\ k\in P\cup S. \qquad (3.5b)$$

Observe that, unlike the enhanced undirected formulation [EUF], the directed formulation uses only the O(mn) (unidirectional) single-commodity forcing constraints (3.3) and (3.4). Yet, as we show next, this formulation has the same optimal linear programming value as formulation [EUF].

3.2 LP-Equivalence of Directed and Enhanced Undirected Models

This section shows that, when the primary and secondary costs are nonnegative (and all commodities originate at a single node), the linear programming relaxations of the directed flow-based formulation [DF] and the enhanced undirected formulation [EUF] have the same optimal values. Previously, Goemans and Myung [1991] have considered a similar result for the Steiner tree problem. They showed that the polyhedron determined by the linear programming relaxation of the enhanced undirected Steiner tree formulation (without the explicit upper bound constraints $u_{ij} \leq 1$) is a projection of the polyhedron determined by the linear programming relaxation of the directed formulation. This property is a polyhedral result that does not depend upon the sign of the objective function coefficients. Our result applies to the broader class of multi-level design problems, but considers only optimal solutions of the linear programming relaxation, and requires nonnegative costs. We might also note that, for the Steiner network problem, Chopra and Rao [1988a] have shown a related equivalence between a directed cutset formulation and an (enhanced)

undirected multi-cut formulation; both these formulations use only design variables, and do not contain the unit upper bounds on these variables.

Let [LDF] and [LEUF] denote the linear programming relaxations of [DF] and [EUF], obtained by replacing the integrality (0 or 1) restrictions (2.10a) and (3.5a) on the edge and arc selection variables with nonnegativity constraints and unit upper bounds (e.g., replace the constraint $u_{ij} \in \{0,1\}$ in [BUF] with the constraints $0 \le u_{ij} \le 1$).

Theorem 1: *For problems with nonnegative primary and secondary costs, the linear programming relaxations of formulations [EUF] and [DF] have equal optimal objective function values.*

Proof:

We prove the theorem by showing that, given an optimal solution to either formulation ([LDF] or [LEUF]), we can construct a feasible solution to the other formulation with the same objective function value. Given a solution to one formulation, we will use the same flow solution $\{f_{ij}^k\}$ for the other formulation to determine appropriate values of the design variables for the second formulation.

We first note that, for a given flow solution $\{f_{ij}^k\}$, it is easy to find the optimal values of the design variables for either formulation. For any arc $(i,j) \in A$, let F_{ij}^P and F_{ij}^{PS} denote the maximum primary flow and maximum combined (primary or secondary) flow on this arc in the direction i to j, i.e.,

$$F_{ij}^P = \max \{ f_{ij}^k : k \in P \}, \text{ and}$$
$$F_{ij}^{PS} = \max \{ f_{ij}^k : k \in P \cup S \}.$$

For given flows, we use the following equations to compute the values of the directed design variables x_{ij} and y_{ij} in formulation [LDF]:

$$x_{ij} = F_{ij}^P, \text{ and} \qquad (3.6a)$$

$$y_{ij} = F_{ij}^{PS} - x_{ij} \qquad \text{for all } (i,j) \in A. \qquad (3.6b)$$

Equations (3.6a) and (3.6b) ensure that the directed design variables x_{ij} and y_{ij} are nonnegative, and have the smallest possible values that satisfy the primary and secondary forcing constraints (3.3) and (3.4). Since the primary and secondary costs are nonnegative, this solution also has the smallest possible design cost that accomodates the given flows.

Similarly, we can express the undirected design solution to [LEUF] in terms of given flow values as:

$$u_{ij} = \max \{ f_{ij}^k + f_{ji}^h : k,h \in P \} = F_{ij}^P + F_{ji}^P, \text{ and} \quad (3.7a)$$

$$v_{ij} = \max \{ f_{ij}^k + f_{ji}^h : k,h \in P \cup S \} - u_{ij} = F_{ij}^{PS} + F_{ji}^{PS} - u_{ij} \geq 0. \quad (3.7b)$$

Again, the undirected design variables are nonnegative, and satisfy all the commodity-pair forcing constraints (2.11)–(2.13) of formulation [LEUF]. Equations (3.7a) and (3.7b) select the lowest possible values of the primary and secondary edge selection variables u_{ij} and v_{ij} that accomodate the given flows.

We refer to the values of the directed and undirected design variables satisfying (3.6) or (3.7) as *tight design values*. When defined by the same flows f_{ij}^k, these values satisfy the following relationships:

$$u_{ij} = x_{ij} + x_{ji}, \text{ and} \quad (3.8a)$$

$$v_{ij} = y_{ij} + y_{ji} \qquad \text{for all } \{i,j\} \in E. \quad (3.8b)$$

Next observe that given an optimal solution to either the directed or undirected model, we can (i) assume it has tight design values, and (ii) use the given flows to construct tight design values to the other model satisfying (3.8). Since the costs are symmetric, the solutions satisfying (3.8) have the same objective function value.

So far, we have shown that the transformations via equations (3.6) and (3.7) from an optimal solution of either problem give directed and undirected design solutions that are nonnegative, satisfy the forcing constraints of [LDF] and [LEUF], and have equal objective function values. To complete the proof of equivalence, we need to show that the computed design variables have values less than or equal to 1. The transformation from [LEUF] to [LDF] via equations (3.6) clearly satisfies this condition, since $u_{ij} \leq 1$ and $v_{ij} \leq 1$ in the given [LEUF] solution and the computed values of x_{ij} and y_{ij} are nonnegative and satisfy equations (3.8). The following claim, which we prove (in the Appendix) using the flow decomposition property (see, for example, Ahuja, Magnanti, and Orlin [1993]), establishes that the reverse transformation from [LDF] to [LEUF] gives an undirected design solution that also satisfies the unit upper bounds.

Claim: *The linear programming relaxation [LDF] of the directed formulation has an optimal solution satisfying the conditions:*

$$x_{ij} + x_{ji} \leq 1, \text{ and}$$
$$y_{ij} + y_{ji} \leq 1 \qquad \text{for all edges } \{i,j\} \in E.$$

Proof: See Appendix 1.

Recall that the values of the undirected design variables u_{ij} and v_{ij} that we derive from the optimal directed solution to [LDF] satisfy equations (3.8). Therefore, given a directed design solution satisfying the conditions of the claim, the derived undirected solution also satisfies the unit upper bounds. These arguments prove that the directed formulation and enhanced undirected formulation have the same optimal LP value. ◆

Since the directed formulation [DF] and the enhanced undirected formulation [EUF] are LP-equivalent, in Balakrishnan et al. [1992] we use the directed formulation to develop the dual ascent algorithm. This algorithm is much easier to describe and implement than its equivalent undirected version. Finally, we note that the LP-equivalence of the directed and enhanced undirected formulations also extends to the more general single-origin (or single-destination) two-level network design model *with flow costs*. In this model, routing commodity k on arc (i,j) incurs a nonnegative per unit cost of c_{ij} (we assume this per unit cost to be the same for all commodities) in addition to the fixed primary or secondary cost. We can apply a slight extension of the previous proof to problems with flow costs by additionally showing that the flow rerouting step (see Appendix 1) will both ensure a feasible design and not increase the total flow cost of the optimal LP solution.

4. Worst-case Analysis of TLND Heuristics

This section analyzes the worst-case performance of several heuristic methods for the undirected TLND problem. Duin and Volgenant [1991] describe two approximate algorithms—a modified weight heuristic and a branch chord heuristic—for the TLND problem. These methods apply a Steiner tree heuristic (Takahashi and Matsuyama [1980]) to select the primary edges; the methods differ in the edge costs they use to construct the Steiner tree. Duin and Volgenant [1991] raised the issue of whether these heuristics have the same worst-case performance ratio of 2 as the underlying (undirected) Steiner heuristics, but report that they had not found tight bounds.

We propose two broad classes of heuristic strategies for the TLND problem—"forward" heuristics and "reverse" heuristics. A forward heuristic first selects the configuration of primary edges, and then adds secondary edges to connect the remaining secondary nodes. In contrast, a reverse heuristic first installs secondary facilities to connect all secondary nodes, and then upgrades or installs primary facilities to connect the primary nodes. We can implement these two strategies in various ways by solving minimum spanning tree or Steiner network subproblems to determine the primary and/or secondary subtree. (We can interpret Duin and Volgenant's [1991] two heuristics as specific implementations of the forward and reverse strategy respectively.) We consider four distinct methods, two forward and two reverse, and analyze the worst-case performance of a *composite* heuristic that selects, for each problem instance, the best (lowest cost) among the four heuristic solutions.

Motivated by Orlin's [1991] heuristic worst-case analysis for the HND problem, we first consider a special class of TLND problems in which the ratio of primary to secondary costs is the same for all edges. For this class of problems, we show if ρ is the worst-case ratio of the heuristic used to solve the embedded Steiner network subproblem, then the composite TLND heuristic has a worst-case performance ratio of ρ or $4/(4-\rho)$ depending on whether $\rho \geq 2$ or $\rho < 2$. This result implies that, if we can solve the Steiner network subproblem optimally (as in the HND problem), the composite heuristic produces a solution to the proportional cost TLND problem that is guaranteed to cost no more than 4/3 the optimal cost (compared to Orlin's worst-case bound of 1.618). When the primary to secondary cost ratio varies by edge, the composite heuristic has a worst-case ratio of $(\rho+1)$. We also provide worst-case examples to prove that these bounds are tight. Our worst-case results resolve Duin and Volgenant's [1991] conjecture about the relationship between Steiner and TLND heuristic worst-case ratios. We note that transforming the TLND problem into an equivalent directed Steiner tree problem (Duin and Volgenant [1989]) is not an effective tactic for TLND worst-case analysis since the lowest worst-case ratio known to date for the directed Steiner tree problem with w terminal nodes ($w = p+n$ in the equivalent directed Steiner tree representation of the TLND problem) is $\log(w)$; this result relies on transforming and solving the directed Steiner problem as a set covering problem (Goemans [1992]).

4.1 TLND Heuristics

4.1.1 Forward Heuristics

Forward heuristics first select the edges interconnecting the primary nodes (these edges must contain primary facilities), and then complete the design by adding secondary edges to connect the remaining secondary nodes (those that do not already belong to the primary subtree). We describe two such methods that respectively use the minimum spanning tree and minimal Steiner tree to construct the primary subtree.

The basic version of the **Minimum Spanning Tree (MST) heuristic** for the TLND problem constructs a feasible solution by installing primary facilities on all the edges of the minimum tree T spanning *all* the nodes of the original graph G (using primary edge costs). The primary subtree, denoted as T_p, is the minimal subtree of T that spans all the primary nodes.

We can improve the MST heuristic as follows. Instead of installing primary facilities on all edges of T, we (i) install primary facilities on all edges of subtree T_p, and (ii) select secondary edges to span the remaining secondary nodes by applying the following **optimal secondary completion procedure:**

> Aggregate all the nodes spanned by the primary subtree into a single node. If this aggregation process creates parallel edges, discard all but the cheapest (in terms of secondary costs) parallel edge. Find the minimum spanning tree of this condensed graph using secondary costs. Install secondary facilities on the edges of this subtree.

We refer to process of first finding the minimum spanning tree and then applying optimal secondary completion to T_p as the **Enhanced Minimum Spanning Tree (EMST) heuristic.**

The **Forward Steiner Tree (FST) heuristic** for the TLND problem first finds an exact or approximate Steiner tree (using primary edge costs) spanning all the primary nodes (and optionally covering some secondary nodes). This Steiner tree serves as the primary subtree in the heuristic TLND solution; we determine the secondary edges by applying the optimal secondary completion procedure to the primary subtree. Since the Steiner network problem is itself NP-hard, we might consider using an approximate method to solve the Steiner subproblem; consequently, we will express our TLND worst-case results in terms of the worst-case ratio ρ of the Steiner tree solution method ($\rho = 1$ if we solve the Steiner tree subproblem exactly). For the HND special case, we can solve the Steiner subproblem exactly since this subproblem corresponds to finding the shortest path between the two

primary nodes (using primary edge costs). We refer to this specialization of the FST heuristic as the **Shortest Path (SP) heuristic.**

We can interpret Duin and Volgenant's [1991] modified weight heuristic for the TLND problem as one version of the FST method. Using an edge weight function that reflects savings in secondary costs, they construct a primary subtree using a method analagous to Takahashi and Matsuyama's [1980] greedy heuristic for solving Steiner tree problems. The modified weight heuristic then completes the TLND solution by applying the optimal secondary completion procedure.

4.1.2 Reverse Heuristics

The MST and FST heuristics first connect the primary nodes (possibly via intermediate secondary nodes), and then choose secondary edges using the optimal secondary completion procedure. We now consider analagous "reverse" methods that first connect the secondary nodes, and then use incremental edge costs to install primary facilities. This reverse strategy might be intuitively appealing for problem instances with secondary costs close to primary costs. We describe two alternative implementations of this strategy.

The **Incremental Steiner Tree (IST) heuristic** first finds the minimum tree spanning the node set $S \cup \{1\}$ using secondary costs (recall that primary node 1 is the root node). Using incremental (= primary–secondary) costs for the edges of this tree, and the original primary costs on the remaining edges of G, the method constructs a Steiner tree with primary nodes as terminals. Adding the edges (with primary facilities) of the Steiner tree to the original subtree (containing secondary facilities), and successively dropping secondary edges to eliminate any cycles gives a feasible TLND solution.

The **Overlay Steiner Tree (OST) heuristic** begins with the secondary-cost minimum spanning tree over all the nodes (the IST heuristic connects only nodes $S \cup \{1\}$). We then construct a Steiner tree (with primary nodes as terminals) using incremental costs for edges in the minimum spanning tree, and primary costs for the remaining edges. As before, we install primary facilities on the edges of the Steiner tree, and eliminate cycles by successively dropping secondary edges.

Like the FST heuristic, the IST and OST heuristics might employ either an exact or approximate method for solving the Steiner subproblem. For the HND problem, we can solve this subproblem exactly using a shortest path algorithm. Although all three Steiner

tree-based heuristics—FST, IST, and OST—solve Steiner subproblems with the primary nodes as terminals, they use different costs (primary costs for FST, and different incremental costs for IST and OST). Just as the EMST heuristic improves the MST solution, we might consider the following enhancement of the IST (or OST) heuristic solution: after solving the Steiner tree subproblem in the second step, apply optimal secondary completion to the primary subtree (chosen by the exact or approximate Steiner solution method) in order to decide the configuration of secondary facilities. We will refer to this improved method as the Enhanced IST (or OST) heuristic.

In addition to the Steiner tree-based reverse heuristics IST and OST, we might also consider a minimal spanning tree-based reverse heuristic analagous to the MST heuristic. This method first finds the secondary-cost minimal spanning tree, determines the primary subtree of this spanning tree, and upgrades the secondary facilities to primary facilities on all edges of this primary subtree. Note, however, that the OST heuristic dominates this method since the primary subtree (of the secondary minimal spanning tree) is one of many possible heuristic solutions to the Steiner subproblem in the second step of the OST method. Finally, we can also interpret Duin and Volgenant's [1991] branch chord heuristic as a version of the enhanced OST heuristic; starting with the secondary minimum spanning tree, the branch exchange method constructs the primary subtree by successively exchanging secondary edges for primary edges.

4.2 Worst-case Bounds for the Proportional Costs case

This section considers the special class of TLND problems having the same **primary-to-secondary cost ratio**, say r, for all edges, i.e.,

$$r = a_{ij} / b_{ij} \quad \text{for all edges } \{i,j\} \in E.$$

In this case, the incremental cost e_{ij} of edge $\{i,j\}$ equals $(r-1)\, b_{ij}$. In the following discussion, we let T(G) denote the minimum tree (using secondary costs) spanning all the nodes of the graph G. To simplify our notation, we assume without loss of generality, that we have scaled the costs so that the secondary cost of the minimum spanning tree T(G) equals 1, i.e.,

$$\sum_{\{i,j\} \in T(G)} b_{ij} = 1.$$

Let s denote the (unknown) **secondary cost of the optimal Steiner tree** spanning all the primary nodes. Note that, since the spanning tree T(G) is a feasible solution to the

Steiner network problem with primary nodes as terminals, its secondary cost must be an upper bound on the secondary cost of the optimal Steiner tree, i.e., $s \leq 1$.

To evaluate the worst-case performance of the spanning and Steiner tree-based heuristics, we develop some lower bounds (in terms of the Steiner tree cost s, the cost ratio r, and the normalized secondary cost of the minimum spanning tree T(G)) on the optimal value, say Z*, of the TLND problem. We then derive upper bounds for each heuristic separately, and for a composite heuristic that applies all four methods and selects the best heuristic solution.

4.2.1 Lower Bounds on Z*

Our first lower bound follows from the S–ST formulation described in Section 2. Suppose we relax this formulation by removing the linking constraints (2.4). The problem then decomposes into two subproblems: (i) a Steiner tree subproblem (involving the u variables) with primary nodes as terminals and with the incremental costs e_{ij} as arc lengths; and (ii) a minimum spanning tree subproblem (the w-subproblem) over the original graph, with the secondary costs as arc lengths. Since we have relaxed the original formulation, adding the optimal values for these two subproblems provides a valid lower bound $\underline{Z}_1 = (r-1)\, s + 1$ on the optimal value Z*. Note that deleting the linking constraints (2.4) corresponds to dualizing these constraints using multipliers $\mu_{ij} = 0$; thus, $\underline{Z}_1$ is the optimal value of the Lagrangian subproblem for this special set of multipliers. We, therefore, refer to $\underline{Z}_1$ as the **Lagrangian lower bound.**

The second lower bound follows from a different relaxation of the S–ST formulation. Suppose we delete the spanning tree constraints (2.3) from formulation [S–ST]. Since all the secondary edge costs are nonnegative, this relaxed problem must have an optimal solution with $w_{ij} = u_{ij}$. Thus, we can eliminate the w-variables by substituting u_{ij} for w_{ij} in the objective function, and removing constraints (2.4); observe that, after we make this substitution, u_{ij} has the primary cost $a_{ij} = e_{ij} + b_{ij}$ as its objective function coefficient. Consequently, the residual problem seeks the optimal Steiner tree (with primary nodes as terminals) using the primary costs. Since we have relaxed the original formulation, the primary cost (= r s) of the optimal Steiner tree is a valid lower bound for Z*. We denote this **Steiner tree lower bound** as $\underline{Z}_2$.

Note that $\underline{Z}_1 \geq \underline{Z}_2$ since $s \leq 1$. We can obtain a third lower bound by omitting the Steiner tree constraint (2.2) in formulation [S–ST]. The optimal solution to this relaxation

is the secondary minimum spanning tree T(G), with cost $\underline{Z}_3 = 1$. Again, $\underline{Z}_1$ dominates this lower bound. Therefore, we use only the lower bound $\underline{Z}_1$ in all our subsequent discussions.

4.2.2 Upper bounds on heuristic solutions

We now determine upper bounds on the cost of the heuristic solutions produced by the four methods—one spanning tree method (EMST) and three Steiner tree-based methods (FST, IST, and OST)—described in Section 4.1. Let Z_H denote the cost of the solution produced by the heuristic method H. We begin by analyzing the forward heuristics.

The basic *MST heuristic* installs a primary facility on every edge of the minimum spanning tree of the original graph. With the constant primary-to-secondary cost ratio r, and our cost scaling assumption (i.e., the secondary minimum spanning tree has unit cost), the cost of the MST heuristic solution is r. Since the EMST heuristic improves the MST solution,

$$Z_{EMST} \leq r.$$

To analyze the worst-case performance of the Steiner tree-based heuristic methods, we will assume that the embedded Steiner network solution method has a known worst-case performance ratio ρ. Consequently, in the *FST heuristic* solution, the primary subtree is no more than ρ times the primary cost (= rs) of the optimal Steiner tree solution. Furthermore, the optimal secondary completion of this primary subtree must cost no more than the secondary minimum spanning tree for the original graph G. Therefore,

$$Z_{FST} \leq \rho rs + 1.$$

Let us now consider the reverse heuristics. The *IST heuristic* incurs a secondary cost of at most 1 unit in the first step, and a maximum primary cost of ρrs in the second step (for the approximate Steiner tree connecting the primary nodes). Therefore,

$$Z_{IST} \leq \rho rs + 1.$$

The *OST heuristic* incurs a cost of 1 in the first step, and an incremental cost of at most ρrs in the second step (since the incremental Steiner tree must cost less than the primary Steiner tree cost s). Consequently,

$$Z_{OST} \leq \rho r s + 1.$$

Note that all three Steiner tree-based heuristics (FST, IST, and OST) have worst-case values of (ρrs + 1).

4.2.3 Worst-case performance ratio

Let ω_{Span} and $\omega_{Steiner}$ represent, respectively, the worst-case performance ratios (i.e., ratio of heuristic solution cost to optimal TLND value) of the spanning tree and Steiner tree-based heuristics. Based on our observations in Sections 4.2.1 and 4.2.2, these two ratios have the following upper bounds:

$$\begin{aligned} \omega_{Span} &\leq Z_{EMST}/Z_1 &&= r / \{(r-1)s + 1\}, \text{ and} \\ \omega_{Steiner} &\leq Z_{FST}/Z_1 &&= \{\rho rs + 1\} / \{(r-1)s + 1\}. \end{aligned}$$

For any given value of $r \geq 1$, the upper bound on the performance ratio ω_{Span} is decreasing in s; as $s \rightarrow 0$, this upper bound tends to r. On the other hand, for the Steiner tree-based heuristics, the upper bound on the worst-case ratio $\omega_{Steiner}$ increases with s. Since $s \leq 1$, $\omega_{Steiner}$ has an upper bound of $(\rho + 1/r)$.

Since the two upper bounds on the performance ratio respectively decrease and increase with s, we consider a *Composite heuristic* that selects the best among all the spanning and Steiner tree-based heuristic solutions for a given problem instance. Let $\hat{Z} = \min \{Z_{EMST}, Z_{FST}, Z_{IST}, Z_{OST}\}$ be the value of the composite heuristic solution, and let ω denote its worst-case performance ratio. Note that

$$\hat{Z} \leq \min \{r, \rho rs + 1\}.$$

Therefore, the composite heuristic's worst-case performance ratio ω satisfies

$$\omega \leq \min (r, \rho rs + 1) / \{(r-1) s + 1\}. \qquad (4.1)$$

For fixed r, the right-hand side of (4.1) achieves its maximum value when $s = (r-1)/\rho r$. Substituting this value of s in the right-hand side of inequality (4.1) gives the following result.

Theorem 2:

If the ratio of primary-to-secondary costs is constant for all edges, then

$$\omega \leq \rho / \{1 + (\rho-2)/r + 1/r^2\}. \qquad (4.2)$$

Note that this result gives a worst-case bound on the performance of the composite heuristic as a function of the primary-to-secondary cost ratio r. For example, if r = 1, then as we might expect, the worst-case ratio is 1. In order to obtain a worst-case bound that applies simultaneously to all values of r, we consider two cases: $\rho \geq 2$ and $\rho < 2$. If $\rho \geq 2$, the right-hand side of inequality (4.2) is less than equal to ρ since $r \geq 1$ (by definition).

Now consider the case with $\rho < 2$. For fixed ρ, the expression on the right-hand side of inequality (4.2) achieves its maximum value of $4/(4-\rho)$ at $r^* = 2/(2-\rho)$. Since $\rho \geq 1$ (by definition) and $\rho < 2$ (by assumption), $r^* \geq 1$ as required. These arguments establish the following corollary to Theorem 2.

Corollary:

For TLND problems with proportional costs,

$$\begin{aligned} \omega &\leq \rho && \text{if } \rho \geq 2 \\ &\leq 4/4-\rho && \text{if } \rho < 2. \end{aligned}$$

Observe that, if the FST heuristic uses an *exact* Steiner tree solution method (with $\rho = 1$), this corollary implies a worst-case bound of 4/3. In particular, for the HND problem (with only two primary nodes) the SP heuristic produces a solution that is at most $33\frac{1}{3}\%$ more expensive than the optimal solution. Since we use the Lagrangian lower bound for characterizing the composite heuristic's worst-case performance, the analysis leading to Theorem 2 also provides bounds on the linear programming relaxation of the model [S-ST]. In a subsequent paper, we will develop these results in a more general problem setting than the two level network design problem.

4.2.4 Worst-case Examples

We now present worst-case examples to show that the bounds of Theorem 2 are tight. We separately consider the HND problem and the general TLND problem. For the HND problem, $\rho = 1$ since the shortest path algorithm solves the Steiner tree subproblems exactly. For the general TLND problem, we will assume a particular approximate method to solve the Steiner tree subproblems with worst-case ratio $\rho = 2$. To prove the tightness of bounds in Theorem 2, we show that the upper bound (4.2) on ω is achievable for arbitrary values of r. In particular, for the HND problem, we show an example with $\omega = r^2/(r^2-r+1)$ that satisfies (4.1) as an equality, and for the general TLND problem our worst-case example achieves $\omega = 2r^2/(r^2+1)$. In all our examples, the enhancement to the MST, IST and OST heuristics, i.e., applying optimal secondary completion with respect to the primary subtree in the final solution does not improve the solution.

Worst-case example for the HND problem:

Figure 3(a) shows the HND worst-case example. This network has two primary nodes (shown as solid circles), and q secondary nodes (the hollow circles). The parameter ε has

a small, positive value, and q is sufficiently large. We select a sufficiently integer large value for the parameter d so that $d \geq q/(r-1)$. The number on each edge denotes its secondary cost; the primary cost is r times this value. For this example,

(i) the EMST heuristic (Figure 3(b)) selects the primary edges on the lower path (selecting a large value of d ensures this MST configuration), with a total cost of

$$Z_{EMST} = r\{q[r-1]/q + 1\} = r^2;$$

(ii) the SP heuristic (Figure 3(c)) installs a primary facility on the direct edge between the primary nodes; the optimal secondary completion installs secondary facilities on the lower path (excluding the last edge incident to the primary node on the right). This solution costs

$$Z_{SP} = r(r-1-\varepsilon) + 1 + (q-1)(r-1)/q = r^2 - (r-1)/q - r\varepsilon.$$

Note that $Z_{SP} \rightarrow (r^2 - r\varepsilon)$ as $q \rightarrow \infty$;

(iii) the IST heuristic first selects the lower path as the minimum tree spanning the secondary nodes (and node 1, the primary node on the left-hand side); the method then installs a primary facility on the direct edge connecting the two primary nodes. This solution is the same as the SP solution (Figure 3(c));

(iv) the OST heuristic first installs secondary facilities on the lower path connecting the two primary nodes (this is the minimum spanning tree), then installs a primary facility (with incremental cost r(r–1)) on the direct edge connecting the primary nodes, and finally deletes one of the (secondary) edges in the lower path during the drop phase. This procedure also gives the SP solution (Figure 3(c));

(v) the optimal solution (Figure 3(d)) consists of primary edges on the lower path, and the pendant secondary edge. This solution has cost

$$Z^* = rq(r-1)/q + 1 = (r^2-r+1).$$

As $\varepsilon \rightarrow 0$, $\omega = \min\{Z_{EMST}, Z_{SP}\}/Z^*$ approaches $r^2/\{r^2-r+1\}$ as desired. Note that if $r = 2/(2-\rho) = 2$, we achieve the bound of 4/3 implied by Theorem 2 for the composite heuristic.

Worst-case example for the TLND problem:

For the general TLND problem, we will assume that the FST, IST, and OST methods use the following **Terminal Tree heuristic** to solve the Steiner subproblems: select the minimum tree spanning only the terminal (primary) nodes. If the edge costs do not satisfy the triangle inequality, we set the length of edge (i,j), for every pair of nodes i and j, equal to the shortest path distance from i to j in the original graph. The cost of this solution is at most twice the cost of the optimal (undirected) Steiner tree (Takahashi and Matsuyama [1980], Lou, Markowsky and Berman [1981]), i.e., $\rho = 2$ for the Terminal Tree heuristic. As Sullivan [1982] has noted, adapting this heuristic to solve the *directed* Steiner tree problem increases its worst-case performance ratio to w, the number of terminal nodes; thus, although we can transform the TLND problem into a directed Steiner tree problem (Duin and Volgenant [1989]), this transformation does not provide effective heuristic worst-case bounds.

Figure 4(a) shows the TLND worst-case example. This example has q primary nodes (solid circles) on the circumference; the edges connecting each primary node to its neighbors have a secondary cost of (r–1)/q. Each primary node is connected to a central secondary node via a "direct" path containing t edges (and t–1 intermediate secondary nodes), each with a secondary cost of (r–1)/2qt. Every pair of adjacent nodes i and j on this path is also connected by a string of (d+1) edges (with d intermediate secondary nodes). The first edge on this string has a secondary cost of (r–1)/2qt; all remaining edges have secondary costs of (r+1)/2dqt. Thus, the cost of the string from i to j is r/qt. In all, the network contains qt such strings. If we use only these strings to connect each primary node to the central secondary node, we incur a total secondary cost of r.

Figure 4(b) shows the EMST solution for this example. Its total cost is

$$Z_{EMST} = r^2.$$

The FST, IST, and OST heuristics generate the solution shown in Figure 4(c). This solution has cost

$$Z_{FST} = r(q-1)(r-1)/q + q(t-1)(r+1)/2qt + (r-1)/2qt + (r+1)/2,$$

which approaches r^2 as q and $t \rightarrow \infty$. Finally, the cost of the optimal solution (Figure 4(d)) approaches

$$Z^* = rqt(r-1)/2qt + (r+1)/2 = (r^2+1)/2.$$

Therefore, for this example, the performance ratio for the composite heuristic is

$$\omega = 2r^2/(r^2+1).$$

Again, as $r \rightarrow \infty$, $\omega \rightarrow 2$ as indicated in Theorem 2. To ensure unique solutions we can perturb the costs by some small $\epsilon > 0$.

4.3 Worst-case analysis for the Nonproportional Costs case

We now analyze the composite heuristic's worst-case performance when the ratio of primary to secondary costs varies by edge.

Theorem 3:

For TLND problems with nonproportional costs, the composite heuristic has a worst-case performance ratio ω of at most $(\rho + 1)$, where ρ is the worst-case ratio of the embedded Steiner tree heuristic.

Proof:

Let Z^*, Z_{ST} and $Z_{T(G)}$ denote, respectively, the optimal values of the TLND, the Steiner network spanning the primary nodes, and the minimum spanning tree T(G) of the original graph G using secondary costs. Since both the Steiner tree and secondary minimum spanning tree problems are relaxations of the TLND problem (see Section 4.2.1),

$$Z_{ST} \leq Z^*, \text{ and}$$
$$Z_{T(G)} \leq Z^*.$$

But, the Forward Steiner Tree heuristic finds a primary subtree with a cost that is at most ρ times the optimal Steiner tree cost; and applying secondary completion to this subtree increases the total cost by at most the cost of the secondary minimum spanning tree. Thus,

$$Z_{composite} \leq Z_{FST} \leq \rho Z_{ST} + Z_{T(G)}.$$

From the previous inequalities,

$$Z_{composite} \leq \rho Z^* + Z^*.$$

Therefore,

$$\omega = Z_{composite}/Z^* \leq (\rho + 1).$$ ♦

Corollary:

For TLND problems with nonproportional costs, using an exact solution method for the Steiner network subproblem produces a heuristic solution that is no more than twice as expensive as the optimal solution.

4.3.1 Worst-case examples

This section presents examples with varying primary-to-secondary cost ratios that achieve the worst-case heuristic performance bound of $(\rho+1)$. Again, we separately consider the HND problem and the general TLND problem. The worst-case examples have the same network configuration as before (i.e., same as Figures 3 and 4 for the HND and TLND problems, respectively), but have different (nonproportional) costs. Again, enhancing the MST, IST, and OST heuristics by applying optimal secondary completion as a final step does not improve the solutions.

Figure 5 shows the worst-case example for the HND problem. The primary and secondary costs are equal for all edges except the edge connecting nodes A and B, and the edges on the path connecting nodes B and C; these edges have a primary cost of 1 and a secondary cost of 0. The direct edge from A to D has primary and secondary cost equal to $q - \varepsilon$. For this example, the EMST, SP (and IST, OST) and optimal solutions have the same configurations as before (Figures 3(b), 3(c), and 3(d)). The cost of the EMST solution (Figure 3(b)) is

$$Z_{EMST} = 2q.$$

The SP solution (Figure 3(c)), obtained by first finding the shortest path between the two primary nodes, has cost

$$Z_{SP} = 2q - 1 - \varepsilon.$$

The optimal solution, shown in Figure 3(d), has a cost of q. Thus, the performance ratio for the composite heuristic is arbitrarily close to $2 = \rho + 1$.

Now consider the general TLND problem with more than 2 primary nodes and nonproportional costs. Figure 6 shows the TLND worst-case example. The optimal and heuristic (EMST and FST, IST, OST) solutions to this problem instance have the same structure as our previous example (Figure 4). The cost of the EMST solution (Figure 4(b)) is

$$Z_{EMST} = 3q.$$

The cost of the FST (and IST, OST) solution using the Terminal Tree heuristic to construct the approximate Steiner tree is

$$Z_{FST} = 2(q-1) + q(t-1)/t + 1/t,$$

which is arbitrarily close to 3q for large values of q and t. The optimal solution, on the other hand, has cost

$$Z^* = qt/t = q.$$

Thus, the performance ratio ω is arbitrarily close to $3 = (\rho + 1)$.

4.5 Summary of Worst-case Results

This section has described several heuristic methods for the TLND problem, developed worst-case performance bounds for two cost structures, and proved that these bounds are tight. We note that the analysis for the proportional cost case might extend to problems whose the primary-to-secondary cost ratios for different edges belong to a prespecified range $[r_l, r_u]$ instead of a single value r. In this case, the worst-case bound would depend on the values of the upper and lower limits r_l and r_u. This bound is likely to be superior to the bound of $(\rho+1)$ for the general cost structure. The following table summarizes the worst-case performance bounds for the different heuristics under various problem scenarios.

Worst-case Performance Ratios for TLND heuristics

Heuristic Method	Proportional costs	Nonproportional costs
FORWARD heuristics:		
EMST heuristic	r	∞
FST heuristic	$\rho + 1/r$	$\rho + 1$
REVERSE heuristics:		
IST heuristic	$\rho + 1/r$	$\rho + 1$
OST heuristic	$\rho + 1/r$	$\rho + 1$
COMPOSITE heuristic:		
(i) $\rho < 2$	$4/(4-\rho)$	$\rho + 1$
(ii) $\rho \geq 2$	ρ	$\rho + 1$

The results in this table show the power of combining two classes of heuristics, one whose performance ratio decreases and one whose performance ratio increases as a function of some underlying problem parameter; in our analysis, this *balancing* parameter is s, the cost of the optimal Steiner tree. By balancing the effects of these two trends, the composite procedure is able to achieve a better performance ratio than each heuristic alone. The bound of 4/3 for the Shortest Path heuristic is one example. Our analysis has the added novelty that the balancing parameter s, which is the optimal objective value (using

secondary costs) of the Steiner tree problem with the primary nodes as terminals, is unknown.

5. Conclusion

This paper has examined modeling issues and analyzed the worst-case performance of heuristics for a new class of multi-level network design problems. The model has applications in telecommunication, transportation, and electric distribution network planning. We showed that the basic flow-based formulation for the undirected problem is relatively weak (in terms of its LP value), and the additional bidirectional commodity-pair forcing constraints considerably strengthen the linear programming relaxation. When all the commodities originate at a single node (or have a single destination) this enhanced formulation is LP-equivalent to a more compact directed formulation. To analyze heuristic worst-case performance, we considered a composite heuristic that selects the best among several heuristic solutions based upon minimal Steiner and spanning trees. For the HND special case with only two primary nodes, this heuristic gives a solution that is guaranteed to be no more than $33\frac{1}{3}\%$ more expensive than the optimal solution. For the general case (with more than two primary nodes, and arbitrary primary and secondary costs), the composite heuristic's worst-case performance ratio of $\rho+1$ depends on the worst-case performance ratio ρ of any Steiner network heuristic.

In a companion paper (Balakrishnan et al. [1992]), we develop and test an optimization-based heuristic methodology for solving the multi-level network design problem. This method first applies certain preprocessing tests to reduce the problem by eliminating or installing primary or secondary facilities before solving the problem. The core of the method consists of a dual ascent algorithm to generate good linear programming-based lower bounds and heuristic upper bounds. Computational experience on large-scale problems (containing up to 500 nodes and 5000 edges) shows that the method provides very good heuristic solutions that are within 0.9% of optimality.

Acknowledgments: We appreciate the helpful comments from the referee and Mr. S. Raghavan, and thank a referee for bringing the work of Iwainsky and Duin and Volgenant to our attention. We are indebted to Professor James Orlin for illuminating discussions about HND heuristics. Our results concerning the relationship between directed and undirected formulations are rooted in many discussions about network design with Dr. Richard Wong.

Appendix 1

Proof of Claim in Section 3.2

Claim: *The linear programming relaxation [LDF] of the directed formulation has an optimal solution satisfying the conditions:*

$$x_{ij} + x_{ji} \leq 1, \text{ and}$$
$$y_{ij} + y_{ji} \leq 1 \qquad \text{for all edges } \{i,j\} \in E.$$

Proof:

To establish this claim, we use the flow decomposition property (see, for example, Ahuja, Magnanti, and Orlin [1992]). Let {i,j} be any edge for which

$$x_{ij} + x_{ji} > 1$$

in the given optimal solution to the linear programming relaxation [LDF] of the directed formulation.

We will construct an alternate solution to [LDF] that has equal (or lesser) cost but less flow in the j-to-i direction, and hence a lower value of x_{ji}. First, if a flow pattern for any commodity contains a cycle, we can eliminate this cycle by reducing the flow on all of its arcs; since the costs are nonnegative, the new directed design solution derived from equations (3.6) has equal or lower cost. Therefore, we will assume that the given flow solution routes all commodities on simple paths. Let h and k be the indices of the "bottleneck" primary commodities in the i-to-j and j-to-i directions, respectively, i.e.,

$$f_{ij}^h = F_{ij}^P = x_{ij}, \text{ and}$$
$$f_{ji}^k = F_{ji}^P = x_{ji}.$$

Note that, since $f_{ij}^k + f_{ji}^k \leq 1$ (since commodity k has unit demand and its flow pattern does not contain cycles) and $f_{ij}^h + f_{ji}^k > 1$ (by assumption), commodity k cannot be a bottleneck flow in the i-to-j direction, i.e.,

$$f_{ij}^k < f_{ij}^h .$$

Let Π_i denote the set of feasible flow paths from the root node 1 to node i *not containing node j* defined by arcs with x-variables greater than zero. Similarly, let Π_j denote the set of paths from node 1 to node j not containing node i. We will maintain commodity k's current flow into node i by increasing its flow on paths Π_i by $\phi = (f_{ij}^h + f_{ji}^k - 1)$ units, and correspondingly decreasing its flow on paths Π_j and arc (j,i). Observe that $0 < \phi \leq f_{ji}^k$. We next argue that we can perform this rerouting without increasing the cost of the [LDF] solution.

Since commodity h's flow paths do not contain cycles, its i-to-j flow must enter node i solely on paths Π_i. Consequently, the values of the design variables in the given [LDF] solution must create a total capacity of at least f_{ij}^h units on the paths Π_i. We can, therefore, increase commodity k's flow on these paths by $\varepsilon = (f_{ij}^h - f_{ij}^k) > 0$. Since $f_{ij}^k \le 1 - f_{ji}^k$, we have $\varepsilon \ge (f_{ij}^h + f_{ji}^k - 1) = \phi$. Thus, we can increase commodity k's flow on paths Π_i by ϕ units and correspondingly decrease its flow on paths Π_j and arc (j,i) by ϕ units without increasing the values of the design variables x_{ij} and y_{ij} (and hence without increasing the cost of the [LDF] solution). Let $f_{ji}^k = f_{ji}^k - \phi \ge 0$ denote the new value of commodity k's flow on arc (j,i) after the rerouting step. We also update the values of the design variable x_{ji} using equation (3.6); let x'_{ji} denote the new design value. Note that commodity k's new flow value from node j to node i satisfies the condition:

$$\begin{aligned} f_{ij}^h + f'_{ji}{}^k &= f_{ij}^h + f_{ji}^k - \phi \\ &= 1. \end{aligned}$$

If commodity k continues to be the bottleneck commodity in the j-to-i direction, then the previous inequality implies $x_{ij} + x'_{ji} \le 1$, as required. Otherwise, we successively perform the rerouting step for each new bottleneck commodity in the j-to-i direction until the sum of the design variables values in the i-to-j and j-to-i directions is less than or equal to 1.

A similar constructive argument proves that [LDF] must have an optimal solution satisfying $y_{ij} + y_{ji} \le 1$. ♦

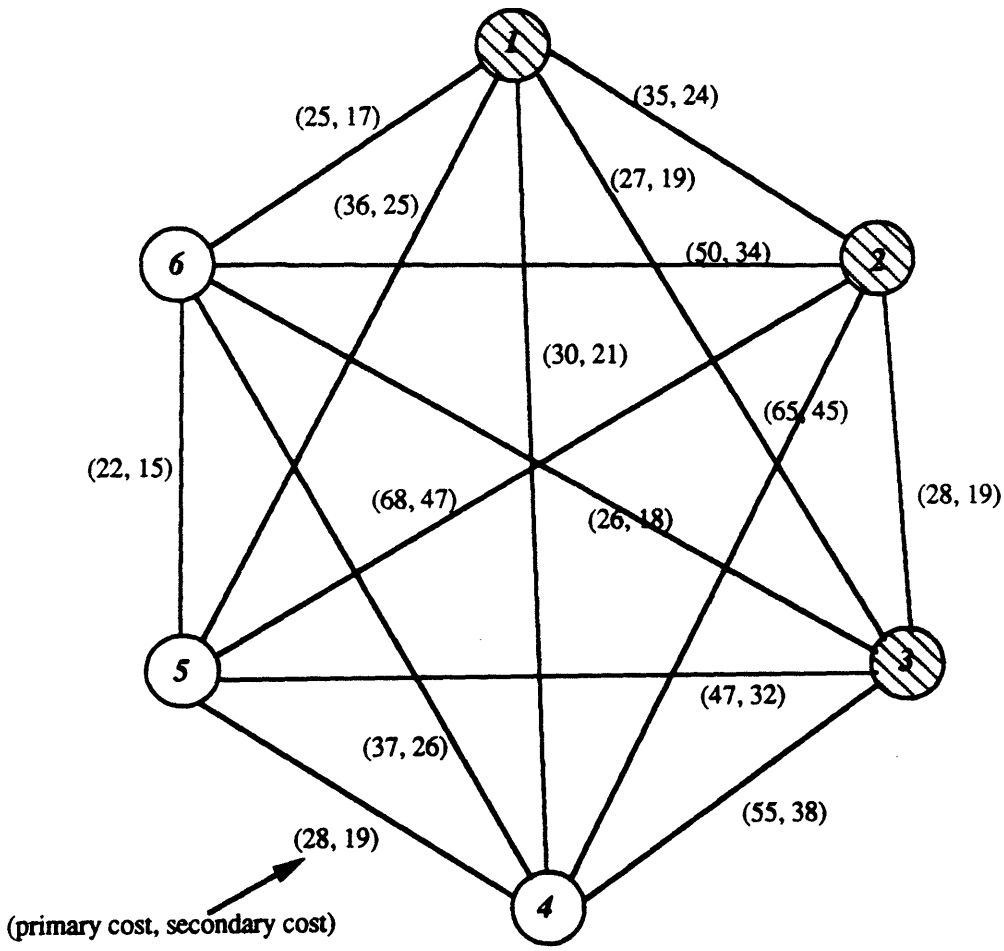


Figure 1: 6-node TLND example

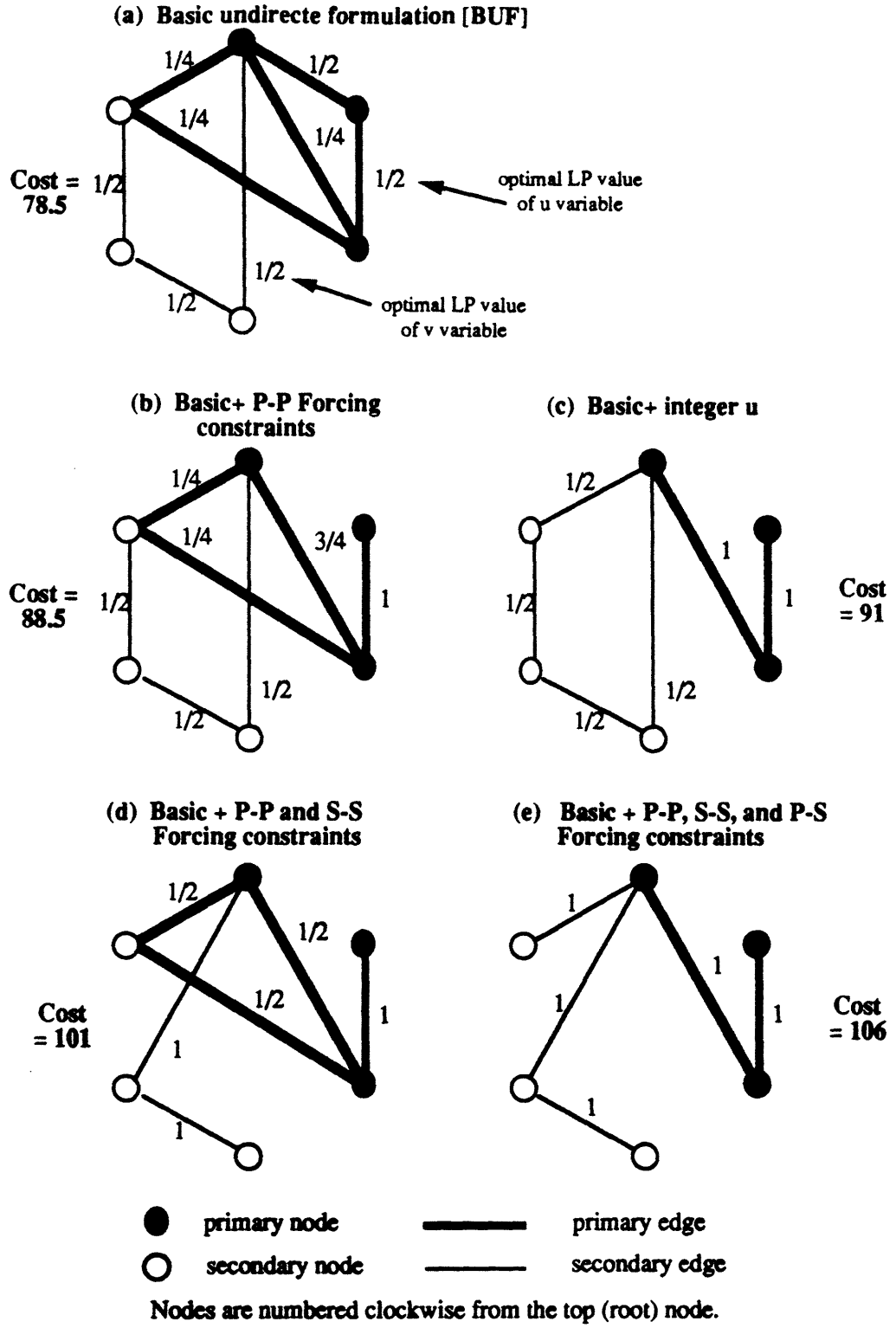


Figure 2: LP solutions for 6-node example

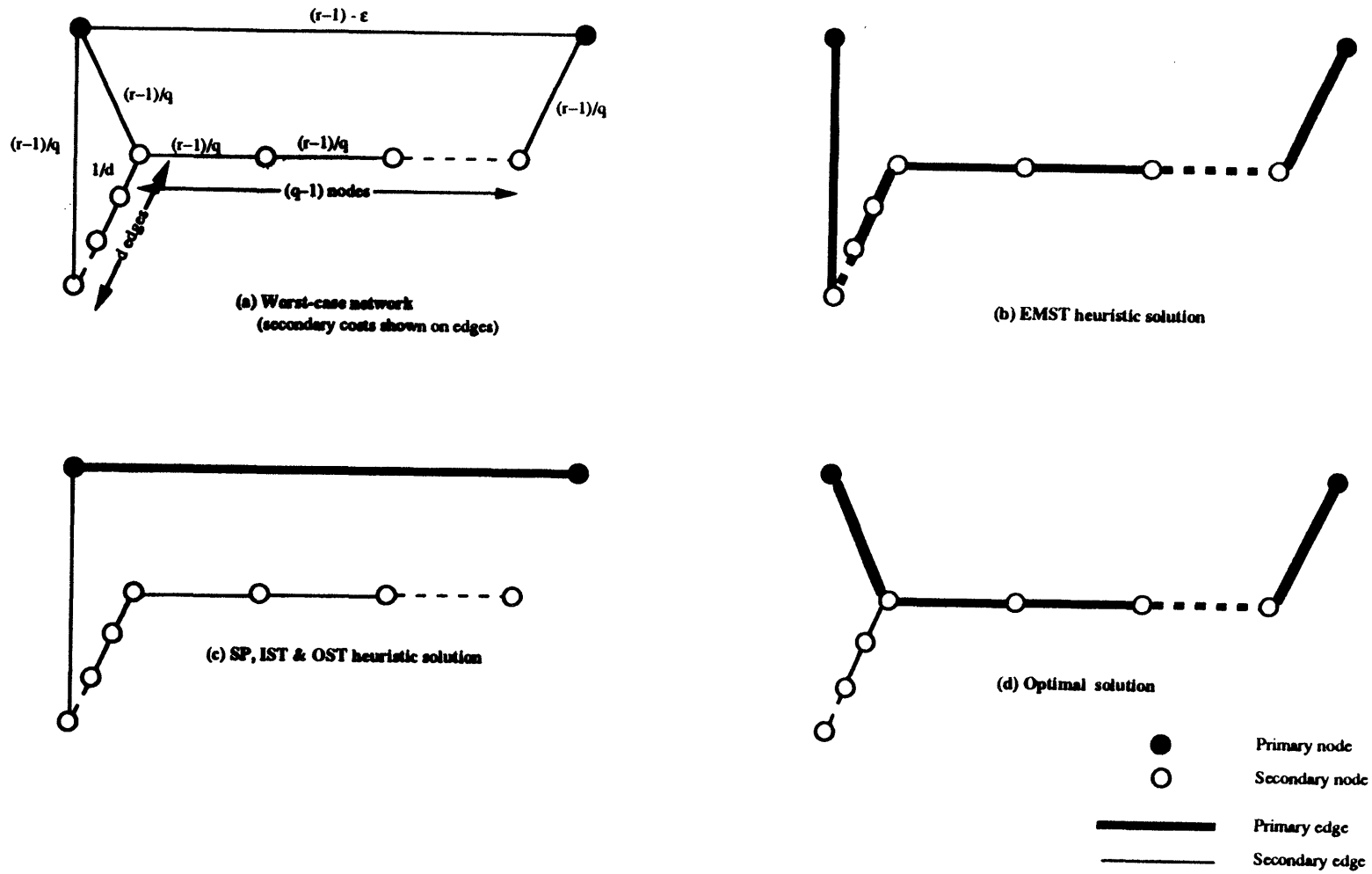
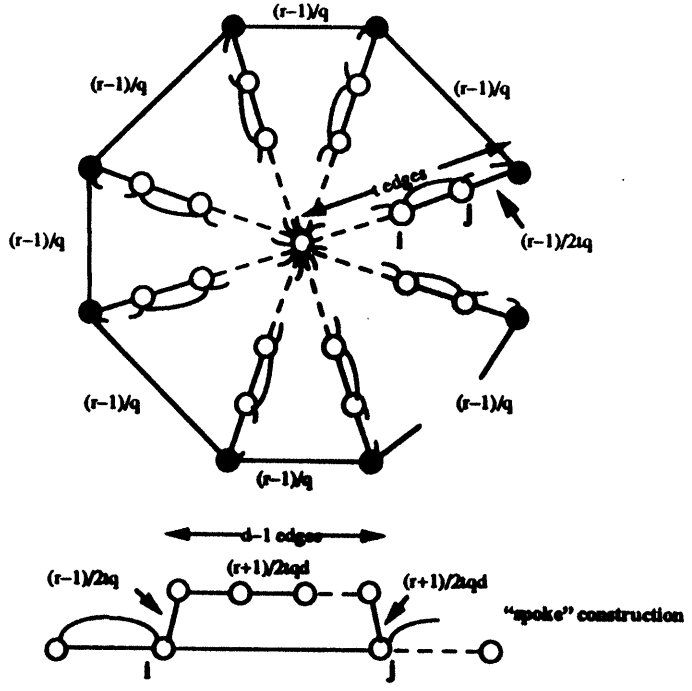
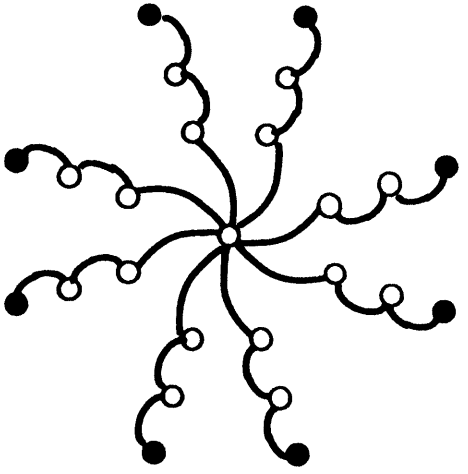


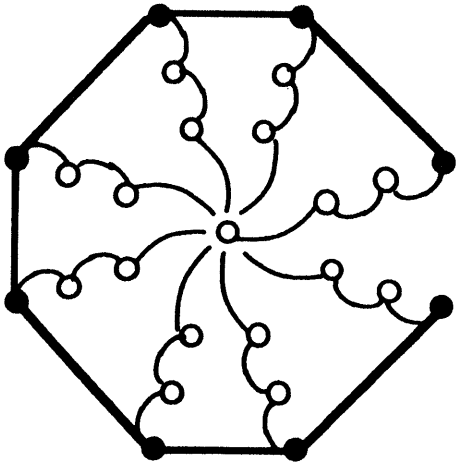
Figure 3: Worst-case example for HND problem with constant primary-to-secondary cost ratio



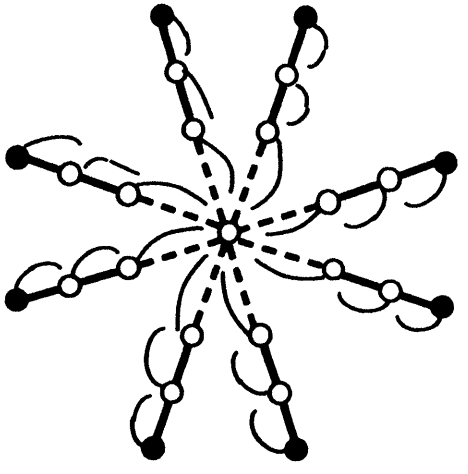
(a) Worst-case network
(secondary costs shown on edges)



(b) EMST heuristic solution



(c) FST, IST and OST heuristic solution



(d) Optimal solution

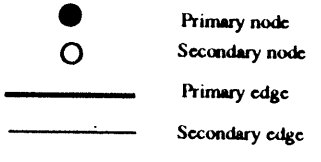


Figure 4: Worst-case example for TLND problem with constant primary-to-secondary cost ratio

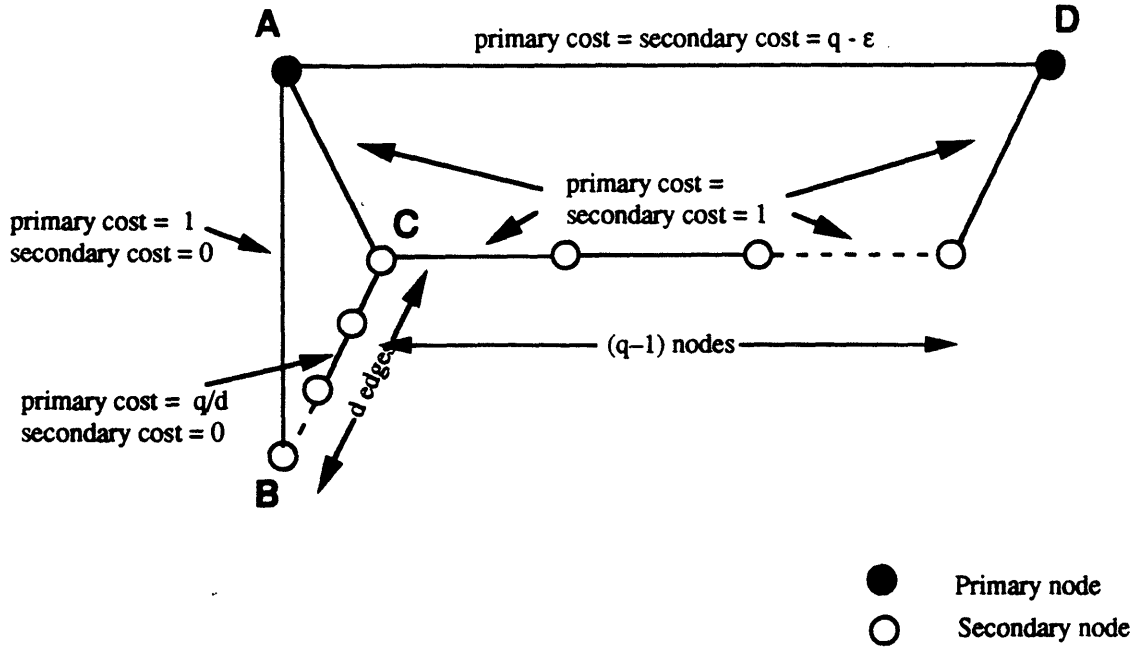


Figure 5: Worst-case example for HND problem with nonproportional costs

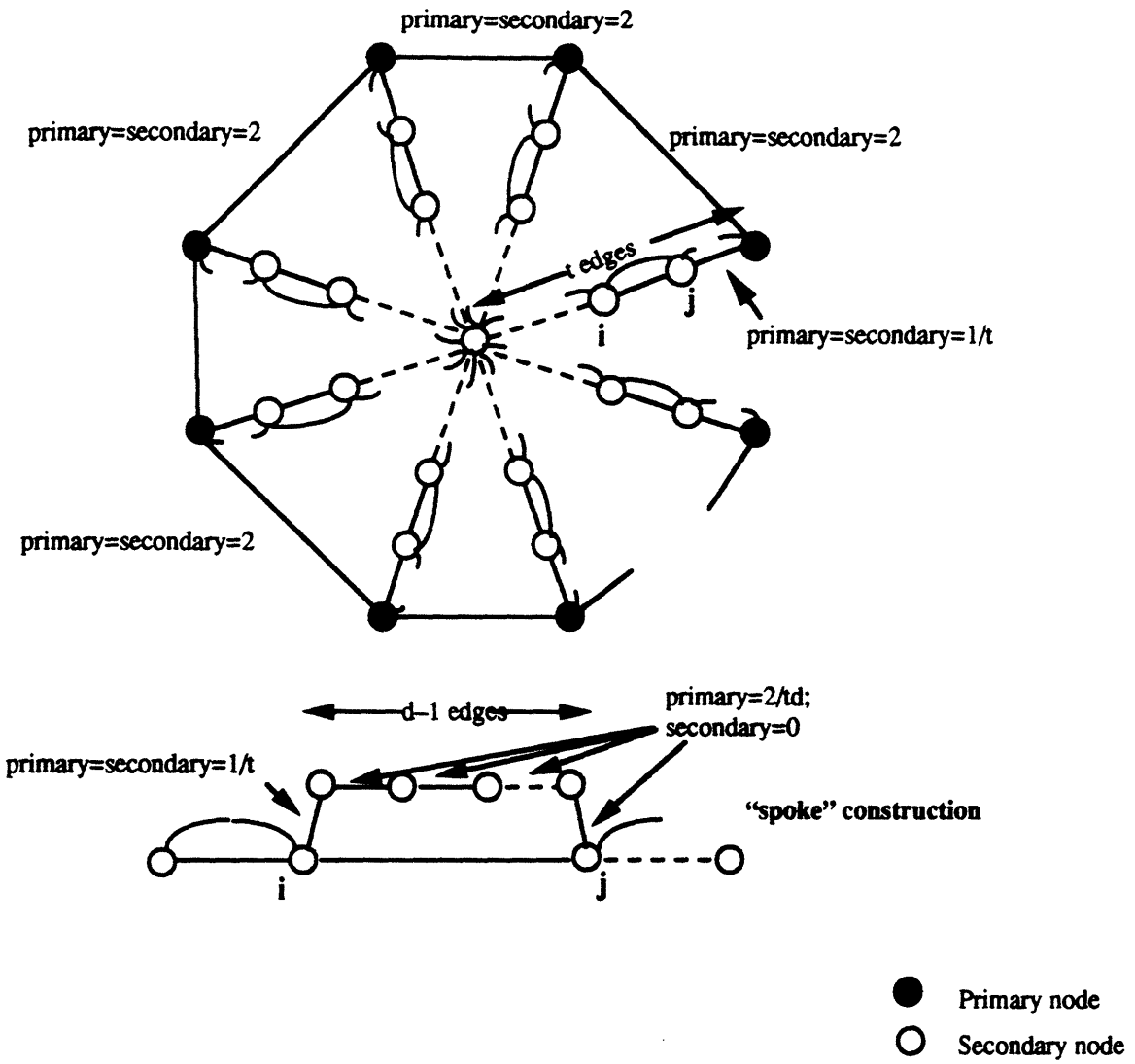


Figure 6: Worst-case example for TLND problem with nonproportional costs

References

AHUJA, R. K., T. L. MAGNANTI, and J. B. ORLIN. 1993. *Networks Flows: Theory, Algorithms and Applications*. Prentice Hall, New Jersey.

ANEJA, Y. P. 1980. An Integer Linear Programming Approach to the Steiner Problem in Graphs. *Networks* **10**, 167–178.

BALAKRISHNAN, A., T. L. MAGNANTI, and P. MIRCHANDANI. 1992. A Dual-based Algorithm for Multi-level Network Design. Working Paper, Operations Research Center, Massachusetts Institute of Technology, Cambridge.

BALAKRISHNAN, A., T. L. MAGNANTI, and R. T. WONG. 1989. A Dual–Ascent Procedure for Large–Scale Uncapacitated Network Design. *Opns. Res.* **37**, 716–740.

BEASLEY, J. E. 1984. An Algorithm for the Steiner Problem in Graphs. *Networks* **14**, 147–159.

BEASLEY, J. E. 1989. An SST–based Algorithm for the Steiner Problem in Graphs. *Networks* **19**, 1–16.

CHOPRA, S., and M. R. RAO. 1988a. The Steiner Tree Problem I: Formulations, Compositions and Extensions of Facets. Working Paper, Graduate School of Business Administration, New York University, New York.

CHOPRA, S., and M. R. RAO. 1988b. The Steiner Tree Problem II: Properties and Classes of Facets. Working Paper, Graduate School of Business Administration, New York University, New York.

CURRENT, J. R., C. S. REVELLE, and J. L. COHON. 1986. The Hierarchical Network Design Problem. *Eur. J. Oper. Res.* **27**, 57-66.

DREYFUS, S. E., and R. A. WAGNER. 1972. The Steiner Problem in Graphs. *Networks* **1**, 195–207.

DUIN, C. and A. VOLGENANT. 1989. Reducing the Hierarchical Network Design Problem. *Eur. J. Oper. Res.*, **39**, 332-344.

DUIN, C. and T. VOLGENANT. 1991. The Multi-Weighted Steiner Tree Problem. *Annals of Operations Research*, **33**, 451-469.

GAREY, M. R., and D. S. JOHNSON. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, San Francisco.

GOEMANS, M. X. 1992. Personal communication

GOEMANS, M. X., and D. J. BERTSIMAS. 1990. Survivable Networks, Linear Programming Relaxations and the Parsimonious Property. Working Paper OR 225–90, Operations Research Center, Massachusetts Institute of Technology, Cambridge.

GOEMANS, M. X., and MYUNG, Y. 1991. A Catalog of Steiner Tree Formulations. Working Paper, Operations Research Center, Massachusetts Institute of Technology, Cambridge.

IWAINSKY, A. 1985. Optimal Trees–A Short Overview on Problem Formulations. In *Optimization of Connection Structures in Graphs*, Central Institute of Cybernetics and Information Processes, Berlin.

KOU, L., G. MARKOWSKY, and L. BERMAN. 1981. A Fast Algorithm for Steiner Trees. *Acta Informatica*, **15**, 141-145.

MAGNANTI, T. L. and R. T. WONG. 1981. Network Design and Combinatorial Optimization. National Science Foundation Proposal.

MARTIN, R. K. 1986. A Sharp Polynomial Size Linear Programming Formulation of the Minimum Spanning Tree Problem. Working paper, University of Chicago, Chicago.

ORLIN, J. B. 1991. Personal communication.

PATEL, N. R. 1979. Locating Rural Social Service Centers in India. *Mgmt. Sci.* **25**, 22–30.

PIRKUL, H., J. CURRENT, and V. NAGARAJAN. 1991. The Hierarchical Network Design Problem: A New Formulation and Solution Procedures. *Trans. Sci.* **25**, 175-182.

PREPARATA, F. P., and M. I. SHAMOS. 1985. *Computational Geometry: An Introduction*, Springer-Verlag, New York.

PRODON, A., T. M. LIEBLING, and H. GRÖFLIN. 1985. Steiner's Problem on 2-trees. Research Report RO 850315, Ecole Polytechnique de Lausanne, Lausanne, Switzerland.

SHIER, D. 1991. A Heuristic for the Two-level Network Design Problem. Presented at the TIMS/ORSA Annual Meeting, Nashville.

SULLIVAN, G. F. 1982. Approximation Algorithms for Steiner Tree Problems. Technical Report 249, Yale University, Department of Computer Science, New Haven, CT.

TAKAHASHI, H., and A. MATSUYAMA. 1980. An Approximate Solution for the Steiner Problem in Graphs. *Math. Japonica* **24**, 573-577.

WINTER, P. 1987. Steiner Problem in Networks: A Survey. *Networks* **17**, 129–167.

WONG, R. T. 1984. Dual Ascent Approach for Steiner Tree Problems on a Directed Graph. ***Math. Prog.*** **28**, 271–287.