THE MULTI-LEVEL NETWORK DESIGN PROBLEM

Anantaram Balakrishnan
Thomas L. Magnanti
Prakash Mirchandani

OR 262-91 December 1991

The Multi-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

December, 1991

[†] Supported in part by a grant from the AT&T Research Fund

Abstract

This paper studies a new multi-facility network synthesis problem, called the Multi-level Network Design (MLND) problem, that arises in the topological design of hierarchical communication, transportation, and electric power distribution networks whose nodes have varying levels of importance: the more critical or higher level nodes require 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 MLND problem seeks a connected design that minimizes total fixed cost while spanning all the nodes, and connecting nodes at each level via facilities of the corresponding or higher type. This problem generalizes the well-known Steiner network problem and the hierarchical network design problem. In this paper, we describe alternative model formulations for this problem and analyze the worst-case performance for heuristics based upon Steiner and spanning tree computations. For one model that we consider, the heuristic worst-case bounds on the performance ratio are either 4/3 or the worst-case performance ratio ρ of the embedded Steiner tree heuristic. A companion paper develops and tests a dual ascent procedure that generates tight upper and lower bounds on the optimal value of the problem.

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

1. Introduction

This paper studies a new 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 generic MLND problem. In 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. In the MLND problem, we would like to select a connected subset of edges, and choose a facility type for each edge so that all nodes at any level communicate via the corresponding or higher grade facilities. 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.

Multi-level network design problems have considerable economic significance. Consider, for instance, the telecommunications context. With advances in fiber optic transmission technology and faced 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 1990, the regional telephone companies in the United States doubled their deployment of fiber optic equipment to an installed base of over 4 million fiber-kilometers. Total U. S. sales of fiber optic cables and systems was approximately $ 1.4 billion in 1990, 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 1991).

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). For instance, in a metropolitan network, the switching centers and certain critical customers (e.g., large businesses) might represent primary nodes, while household customers represent secondary nodes. Each link of the network can contain either fiber optic cables (primary facilities) or copper cables (secondary facilities, having lower cost, but also lower bandwidth). Because primary nodes have greater traffic volume and higher transmission frequency, the network must connect them using fiber optic systems. 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 variable or volume-dependent costs), this network configuration problem reduces to the MLND 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 MLND problem might not completely capture all the complexities of the actual design task. For instance, the model incorporates only a very coarse and aggergate representation of capacity constraints via the discrete facility types. However, MLND solutions can provide insights and principled starting points for an overall network design exercise.

The MLND model also has theoretical significance because it generalizes several classical discrete and network optimization models. The TLND problem generalizes the **Hierarchical Network Design** (HND) problem, defined by Current et al. [1986], which designates exactly two nodes of the network as primary nodes. Thus, all the edges on the path connecting these two nodes must contain primary facilities; 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 as a TLND problem, we treat the terminal nodes of the Steiner network problem as primary nodes, designate the potential Steiner nodes as secondary nodes, and use zero secondary costs for all edges (the primary costs are the original arc lengths in the Steiner problem). Deleting all the secondary edges from the optimal TLND solution gives the minimum cost Steiner network.

The multi-level and two-level network design problems, in their general form (i.e., with more than two primary nodes, and at least one secondary node), are new, and to our knowledge have not been previously addressed in the management science/operations research literature. However, researchers have extensively studied important special cases such as the Steiner network problem and the Hierarchical network design 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 Groflin [1985], Chopra and Rao [1988a] [1988b]), worst-case analysis of heuristics (Takahashi and Matsuyama [1980], Goemans and Bertsimas [1990]), and computational testing of optimization-based solution methods (e.g., Wong [1984], Beasley [1984], [1989], Chopra, Gorres and Rao [1990]). The literature on the Hierarchical network design 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.

The MLND 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 MLND problem, and develops worst-case bounds for a combined heuristic based on Steiner and spanning tree solutions. A companion paper (Balakrishnan, Magnanti and Mirchandani [1991]) 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 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. For expositional convenience, almost all of our subsequent discussion focuses on the TLND problem. However, our model formulation (and the dual ascent solution methodology) also applies to the more general multi-level problem. Section 2 introduces the 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 considerably strengthen the linear programming relaxation. In Section 3, we consider a directed version of the problem which has a more compact formulation, and show that this formulation has the same optimal linear programming value as the enhanced undirected formulation. Section 4 describes two natural heuristic strategies based upon minimum spanning tree and Steiner tree solutions, and derives worst-case performance bounds for a combined TLND heuristic. For problems with proportional primary and secondary costs, the method's worst case bound is 4/3 if we solve an embedded Steiner tree problem exactly, and is ρ if we solve the embedded Steiner tree problem by a heuristic that has a worst-case bound of $\rho \geq 2$. We also provide worst-case examples to show that the 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 $a_{ij} \geq b_{ij} \geq 0$. Note that we incur no loss of generality by assuming 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 only contain primary facilities.

The TLND problem seeks a tree that spans all the nodes and containing 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 the problem formulation. These model enhancements are critical because our dual ascent solution method (described in Balakrishnan et al. [1991]) relies on generating good linear programming-based lower bounds on the optimal value. Using a small example, in Section 2.4 we demonstrate 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.

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 (i. e., the subgraph defined by edges that contain either a primary or a secondary facility) is a spanning tree of the original graph G (since all costs are nonnegative), and

(ii) the primary subnetwork (i.e., the edges containing primary facilities) is a Steiner tree (with the primary nodes as terminals) embedded in the spanning tree.

Correspondingly, we have two sets of binary decision variables:

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

$$w_{ij} = \begin{cases} 1 & \text{if edge } \{i,j\} \text{ belongs to the optimal design, and} \\ 0 & \text{otherwise.} \end{cases}$$

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, with $w_{ij} = 1$ if edge $\{i,j\}$ belongs to the spanning tree. The TLND problem then has the following **Steiner–Spanning tree (S–ST)** formulation:

[S–ST]

$$\text{minimize} \quad \sum_{\{i,j\}\in E} (e_{ij}\, u_{ij} + b_{ij}\, w_{ij}) \tag{2.1}$$

subject to

Steiner tree constraints:

$$u \in U, \tag{2.2}$$

Spanning tree constraints:

$$w \in W, \tag{2.3}$$

Linking constraints:

$$u_{ij} \leq w_{ij} \quad \text{for all } \{i,j\} \in E, \text{ and} \tag{2.4}$$

Integrality constraints:

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

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 $l = 1, 2, \dots, L$ 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 $l = 1, 2, \dots, L$. The linking constraints are:

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

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 paths in the network connecting the two primary nodes. Observe, further, that 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 worst-case bounds for some heuristic methods based upon minimum spanning and Steiner trees.

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 indicating that we need to 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*. We denote the set of primary and secondary commodities as P and S. 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). Note that, although the problem is defined over an undirected graph, we use directed flow variables to distinguish the direction of flow. We next expand the Steiner tree and spanning tree constraints (2.2) and(2.3) using these flow variables and the previous 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]):

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:

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

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:

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

Spanning tree forcing constraints:

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

Nonnegativity constraints:

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

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 $\{i,j\}$, 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

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

These constraints specify that a secondary commodity k can flow from node i to node j or vice versa only if edge {i,j} contains either a primary or a secondary facility. Finally, with the change of variables, the primary cost a_{ij} replaces 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 **B̲asic U̲ndirected F̲low-based formulation**, which we denote as **[BUF]**.

[BUF]

$$\text{minimize} \quad \sum_{\{i,j\} \in E} a_{ij} u_{ij} + \sum_{\{i,j\} \in E} b_{ij} v_{ij} \tag{2.6}$$

subject to

Commodity flow conservation:

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

Primary forcing constraints:

$$f_{ij}^{k} \leq u_{ij}, \text{ and} \tag{2.8a}$$

$$f_{ji}^{k} \quad \leq \quad u_{ij} \qquad \text{for all } \{i,j\}\in E,\ k\in P, \qquad (2.8b)$$

Secondary forcing constraints:

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

Nonnegativity, integrality:

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

Model extensions:

The flow-based model easily accommodates variable (flow-dependent) costs; these costs appear as objective function coefficients for the flow variables. As before, the formulation extends readily to the more general multi-level network design problem. If we let u_{ij}^{l} be a binary variable indicating whether (u_{ij}^{l} = 1) or not (u_{ij}^{l} = 0) we install a type l facility on edge $\{i,j\}$, then for all nodes N_l at level l, the constraints (2.9a) and (2.9b) for level l flow become

$$f_{ij}^{k} \quad \leq \quad u_{ij}^{1} + u_{ij}^{2} + \ldots + u_{ij}^{l}\text{, and}$$
$$f_{ji}^{k} \quad \leq \quad u_{ij}^{1} + u_{ij}^{2} + \ldots + u_{ij}^{l} \qquad \text{for all } \{i,j\}\in E,\ k \in N_l\,.$$

Alternative formulations:

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). However, 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. The dual ascent solution method that we describe in Balakrishnan et al. [1991] uses the single-source commodity flow formulation. 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 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. 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*

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

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

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

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

The bidirectional commodity-pair forcing constraints 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. [1991]), formulation [EUF] contains more than 450 million constraints.

The bidirectional commodity-pair forcing constraints are not new. Magnanti and Wong [1981] proposed similar constraints for the uncapacitated network design problem, and Balakrishnan, Magnanti and Wong [1989] incorporated them 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). Whereas the uncapacitated network design and minimum spanning tree models require only a single commodity type, the MLND model introduces the P-P, P-S, and S-S forcing constraints (2.11) - (2.13) for L different commodity types.

Using a simple example, we next illustrate the impact (on the linear programming lower bound) of successively adding the commodity–pair forcing constraints. Subsequently, we prove that the optimal value of the linear programming relaxation for the enhanced undirected formulation [EUF] equals the optimal LP value of a more compact directed formulation.

2.4 Impact of Adding the Bidirectional Commodity-pair Forcing Constraints

To illustrate the impact of the commodity-pair forcing inequalities on the linear programming lower bound, consider the six-node, complete network shown in Figure 1. 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 the secondary costs. Primary nodes are shaded circles, and secondary nodes are hollow circles. Dark and light lines represent, respectively, primary and secondary edges with positive LP solution values.

Figures 2(a) through 2(e) depict the optimal linear programming solutions as we progressively strengthen the basic flow-based formulation [BUF] by adding the P-P forcing constraints, the S-S forcing constraints, and the P-S forcing constraints. Solving the LP relaxation of the basic formulation [BUF] gives the solution shown in Figure 2(a) with a cost of 78.5; however, 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) (after adding the P-P forcing constraints 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 also integral; 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). Note that 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 comprised 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 these arcs have 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 both the primary and secondary arc costs are 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 from the undirected to the directed version of the problem is valid only for problems with a single-source (or single-destination) commodity flow pattern. For problem contexts requiring multicommodity flows between multiple origins and destinations, we cannot perform this transformation since the directed model double counts the fixed cost for edges that carry flow in both directions. In the remainder of this section, we assume that the problem context does not necessitate multi-origin flows.

3.1 The Directed Flow-based Formulation

To formulate the directed TLND problem as a mixed-integer program, we use the same commodity flow pattern as the undirected problem (i.e., 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 as [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]

$$\text{minimize} \quad \sum_{(i,j)\in A} a_{ij}x_{ij} + \sum_{(i,j)\in A} b_{ij}y_{ij} \qquad (3.1)$$

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

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

Secondary forcing constraints

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

Nonnegativity, integrality:

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

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

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, Geomans 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 is a projection of the polyhedron determined by the linear programming relaxation of the directed formulation if we remove the constraint $u_{ij} \leq 1$ from the undirected formulation. Their result 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, 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.

We denote the linear programming relaxations of [DF] and [EUF] as [LDF] and [LEUF]. We obtain the relaxations 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 $0 \leq u_{ij} \leq 1$).

Theorem 1: *For problems with nonnegative primary and secondary costs, the linear programming relaxations of formulations [EUF] and [DF] are equivalent, i. e., they 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 flow from 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]:

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

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 while still satisfying 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 to accomodate the given flows.

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

$$u_{ij} = \max \{ f^k_{ij} + f^h_{ji}: k,h \in P \} = F^P_{ij} + F^P_{ji}, \text{ and} \tag{3.7a}$$
$$v_{ij} = \max \{ f^k_{ij} + f^h_{ji}: k,h \in P \cup S \} - u_{ij} = F^{PS}_{ij} + F^{PS}_{ji} - u_{ij} \geq 0. \tag{3.7b}$$

Again, the undirected design variables satisfy nonnegativity requirements, and 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 can accomodate the given flows.

We refer to directed and undirected design values satisfying (3.6) or (3.7) as *tight design values*. Observe that tight directed and undirected design values defined by the same flows f^k_{ij} satisfy the following relationships:

$$u_{ij} = x_{ij} + x_{ji}, \text{ and} \tag{3.8a}$$
$$v_{ij} = y_{ij} + y_{ji} \qquad \text{for all } \{i,j\} \in E. \tag{3.8b}$$

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 a tight design to the other model satisfying (3.8). Since $c_{ij} = c_{ji}$ in the directed problem, 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 using the flow decomposition property (see, for example, Ahuja, Magnanti, and Orlin [1992]), establishes that the reverse transformation from [LDF] to [LEUF] also gives an undirected design solution that 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. ◆

In Balakrishnan et al. [1991], we describe a dual ascent algorithm that approximately solves the dual of the linear programming relaxation. The LP-equivalence of the directed formulation [DF] and the enhanced undirected formulation [EUF] enables us to develop and apply the dual ascent algorithm for directed problems. This algorithm is much easier to describe and implement with the directed model rather than its equivalent undirected formulation. 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 does not increase the total flow cost of the optimal LP solution. We next describe and analyze the performance of some natural heuristic solution methods for the TLND problem.

4. Worst-case Analysis of TLND Heuristics

This section analyzes the worst-case performance of two heuristic methods for the undirected TLND problem. These heuristics are based, respectively, on minimum spanning tree and Steiner network solutions. The latter heuristic is a two-stage method that first finds an exact or approximate Steiner tree, and then adds secondary edges to obtain a feasible TLND solution. To analyze the worst-case performance of these heuristics, we first consider a special class of TLND problems with proportional primary-to-secondary costs, i.e., the ratio of primary-to-secondary costs is the same for all edges. For this case, we show that a *composite* heuristic that selects the better of the minimum spanning tree and a solution based on the optimal Steiner network has a worst-case performance ratio of 4/3. If we use an approximate method to solve the Steiner problem, the TLND heuristic has the same worst-case ratio, say ρ, as the Steiner network heuristic. As we note in our development, we can interpret the lower bounds used to compute the worst-case ratios as optimal values for certain *relaxations* of the Steiner–Spanning tree (S–ST) formulation. For problems that do not satisfy the proportional cost condition (i.e., if the primary-to-secondary cost ratios vary across edges), the worst-case ratio increases to $(\rho + 1)$. Finally, we consider "reverse" heuristics that first use the secondary costs to connect the secondary nodes (and possibly some or all primary nodes), and subsequently use incremental costs to complete the primary subnetwork. We show that, even if we include the best "reverse" solution in the composite heuristic, the worst-case performance bounds do not improve.

Our results for the proportional costs case are motivated by Orlin's [1991] worst-case analysis of the shortest path and minimum spanning tree heuristics for the HND special case. Orlin showed that, when all edges have the same primary-to-secondary cost ratio, the combined strategy of selecting the better of the two heuristic solutions has a worst-case performance bound no greater than 1.618.

4.1 The Minimum Spanning Tree and Steiner Tree Heuristics

The *Minimum Spanning Tree (MST) heuristic* constructs a feasible network design by selecting the edges of the minimum tree spanning *all* the nodes of the original graph G (using primary edge costs), and installing primary facilities on all the edges.

The *Steiner Tree heuristic* 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 then apply the following *optimal secondary completion procedure* to identify the secondary edges of the heuristic solution:

> *Optimal secondary completion procedure:*
> Create a condensed graph by aggregating all nodes spanned by the primary subtree into a single node, say node 0. 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 edge costs. Install secondary facilities on the edges of this subtree.

Note that since the Steiner network problem is itself NP-hard, finding a Steiner tree-based heuristic solution to the TLND problem in polynomial time would entail using a Steiner tree heuristic to identify the primary subtree in the first step. When we use a heuristic method to construct the Steiner tree, we will refer to the overall heuristic strategy for the TLND problem as the *Approximate Steiner Tree (AST) heuristic*. In contrast, the *Exact Steiner Tree (EST) heuristic* solves the Steiner network problem exactly, and then applies the optimal secondary completion procedure to construct a feasible two-level network design.

Note that the AST heuristic includes the following spanning tree-based method as a special case:

- find the minimum tree spanning only the primary nodes, and
- apply the optimal secondary completion procedure relative to this primary minimum spanning tree to determine the secondary edges.

We refer to this special case as the *Primary Node Consolidation (PNC) heuristic*.

For the HND special case (with only two primary nodes), finding the optimal Steiner network corresponds to finding the shortest path between the two primary nodes (using the primary edge costs). We refer to this specialization of the EST heuristic as the *Shortest Path heuristic*. We will distinguish this case when we discuss worst-case examples.

To analyze the worst-case performance of the AST heuristic, we will assume that the embedded Steiner network heuristic has a known worst-case performance ratio of ρ. For instance, when the primary edge costs satisfy the triangle inequality, the primary minimum spanning tree is at most twice the cost of the optimal Steiner network solution (see Takahashi and Matsuyama [1980] and Goemans and Bertsimas [1990]). Thus, $\rho = 2$ for the PNC heuristic. Note that the EST heuristic has $\rho = 1$. Finally, we note that we can interpret the MST and PNC heuristics as optimal methods for solving *restricted* versions of the TLND problem formulation, obtained by fixing certain design variables or flow variables to value zero (e.g., the MST heuristic solves the restricted problem with $v_{ij} = 0$ for all edges). In the following discussion, we let T(G) denote the minimum tree (using secondary costs) spanning all the nodes of the graph G.

4.2 TLND with Proportional Primary-to-Secondary costs

We now focus on a special class of TLND problems having the same primary-to-secondary cost ratio, say r, for all edges, i.e.,

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

Note that, in this case, the incremental cost e_{ij} of edge $\{i,j\}$ equals $(r-1)\, b_{ij}$. 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 MST and Steiner tree heuristics, we first develop some lower bounds (in terms of the Steiner tree cost s, the fixed ratio r, and the unit 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 both methods and selects the better 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***. Note that by using non-zero values for the multipliers μ_{ij}, we might be able to improve upon the lower bound $\underline{Z}_1$.

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 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

The *Minimum Spanning Tree heuristic* selects the minimum tree spanning all the nodes, and installs a primary facility on each edge of this tree. Since the primary-to-secondary cost ratio is fixed at r for all edges, and since the secondary minimum spanning tree has unit cost, the cost of the MST heuristic solution is

$$Z_{MST} = r.$$

The *Approximate Steiner Tree heuristic* first finds an approximate Steiner tree solution. By assumption, the primary cost of this solution is no more than ρ times the primary cost (= r s) 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 (with unit cost) for the original graph G. Therefore,

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

4.2.3 Worst-case performance ratio

Let ω_{MST} and ω_{AST} represent, respectively, the worst-case performance ratios (i.e., ratio of heuristic solution cost to optimal TLND value) of the MST and AST heuristics.

For the *MST heuristic,* the performance ratio ω_{MST} has the following upper bound:

$$\omega_{MST} \leq Z_{MST}/\underline{Z}_1 = r / \{(r-1)s + 1\}.$$

For any given value of $r \geq 1$, this function is decreasing in s; as $s \to 0$, the right-hand side of this equation tends to r.

On the other hand, for the *AST heuristic,*

$$\omega_{AST} \leq Z_{AST}/\underline{Z}_1 = \{\rho rs + 1\} / \{(r-1)s + 1\}.$$

With $r \geq 1$, this function increases with s. Since $s \leq 1$, ω_{AST} has an upper bound of $(\rho + 1/r)$.

Since ω_{MST} decreases and ω_{AST} increases with s, we consider a *Composite heuristic* that selects the better of the two heuristic solutions obtained by the MST and AST heuristics. Let $\hat{Z} = \min \{Z_{MST}, Z_{AST}\}$ 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 worst-case performance ratio ω is bounded by

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

For fixed r, the right-hand side achieves its maximum value when $s = (r-1)/\rho r$. At this value of s, the performance ratio has the upper bound

$$\omega \leq \rho / \{1 + (\rho-2)/r + 1/r^2\}. \qquad (4.1)$$

If $\rho \geq 2$, then $\omega < \rho$, and $\omega \to \rho$ as $r \to \infty$. On the other hand, if $\rho < 2$, then $\omega > \rho$. The ratio ω has its maximum value when $r = 2/(2-\rho)$. Thus, for $\rho < 2$,

$$\omega \leq 4 / (4-\rho).$$

These arguments establish the following theorem.

Theorem 2:

If the ratio of primary-to-secondary costs is constant for all edges, then the worst-case performance ratio of the composite heuristic is

$$\omega \le 4/(4-\rho) \quad \text{if } \rho < 2, \text{ and}$$
$$\le \rho \quad \text{if } \rho \ge 2.$$

Observe that, for the *Exact Steiner Tree heuristic* (with $\rho = 1$), Theorem 2 implies a worst-case bound of 4/3. In particular, for the HND problem (with only two primary nodes) the shortest path heuristic, which consists of finding the shortest primary path and applying optimal secondary completion, produces a solution that is at most $33\frac{1}{3}\%$ more expensive than the optimal solution.

4.2.4 Worst-case examples

We next show that the bounds of Theorem 2 are tight for two cases:

(i) for the shortest path heuristic ($\rho = 1$) applied to the HND problem (with two primary nodes); and,

(ii) for the primary node consolidation (PNC) heuristic (that uses the primary spanning tree as the approximate Steiner network solution) which has a worst-case performance ratio ρ of 2.

To prove the tightness of bounds in Theorem 2, we show that the upper bound (4.1) on ω is achievable for arbitrary values of r. In particular, for $\rho = 1$, we show an example with $\omega = r^2/(r^2-r+1)$ that satisfies (4.1) as an equality. Similarly, for $\rho = 2$, our worst-case example achieves $\omega = 2r^2/(r^2+1)$.

Figure 3(a) shows the HND worst-case example. For this discussion, assume that the parameter d has value 1. (In Section 4.4.2 we use the same example with a higher value of d.) This network has two primary nodes (shown as solid circles), and q secondary nodes (the hollow circles). The parameter ε is has a very small positive value. The number on each edge denotes its secondary cost; the primary cost is r times this value. For this example,

(i) the MST heuristic (Figure 3(b)) selects the primary edges on the lower path, with a total cost of

$$Z_{MST} = r\{q[r-1]/rq + 1/r\} = r;$$

(ii) the EST 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

$$Z_{EST} = r(r-1)/r + 1/r + (q-1)(r-1)/rq = r - (r-1)/rq.$$

Note that $Z_{EST} \rightarrow r$ as $q \rightarrow \infty$;

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

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

Consequently, as $\varepsilon \rightarrow 0$, ω 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 shortest path heuristic.

Figure 4(a) shows the worst-case example for the PNC heuristic with $\rho = 2$. This example has q primary nodes (solid circles) on the circumference; the edges connecting each primary node to its neighbors by edges have a secondary cost of $(r-1)/rq$. The network connects a central secondary node to each primary node via a radial spoke containing $(t-1)$ secondary nodes. Finally, the network contains a pendant secondary node attached to one of the primary nodes with an edge having a secondary cost of $(r+1)/2r$. The parameters q and t are both large integers. For this example,

(i) the MST heuristic solution, shown in Figure 4(b), has total cost

$$Z_{MST} = rqt(r-1)/2rqt + r(r+1)/2r = r;$$

(ii) the PNC heuristic solution, shown in Figure 4(c), has cost

$$Z_{AST} = r(q-1)(r-1)/rq + q(t-1)(r-1)/2rqt + (r-1)/2rqt + (r+1)/2r,$$

which approaches r as q and $t \to \infty$; and,

(iii) the optimal solution (Figure 4(d)) has cost

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

Therefore, for this example,

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

Again, as $r \to \infty$, $\omega \to 2$ as indicated in Theorem 2.

4.3 TLND with Varying Primary-to-Secondary Cost Ratios

We now develop tight upper bounds on the performance ratio of the Composite heuristic when the ratio of primary to secondary costs varies by edge.

Theorem 3:

Let ρ be the worst-case performance ratio of the Steiner network heuristic. For general primary and secondary costs, the Approximate Steiner Tree heuristic has a worst-case performance ratio of $\omega = \rho + 1$.

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 the secondary costs. Since both the Steiner tree and secondary minimum spanning tree problems are relaxations of TLND (with zero secondary costs, and primary costs equal to secondary costs, respectively),

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

But, the Approximate Steiner Tree heuristic finds a primary tree with a cost that is at most ρ times the optimal Steiner tree cost; and this tree's secondary completion has a cost less than the secondary minimum spanning tree. Thus,

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

From the previous inequalities,

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

Therefore,

$$\omega = Z_{AST}/Z^{*}$$
$$\leq (\rho + 1). \qquad \blacklozenge$$

Corollary:

For TLND problems with nonproportional costs, the Exact Steiner Tree heuristic has a worst-case performance ratio of 2.

4.3.1 Worst-case examples

We next show that, with varying primary-to-secondary cost ratios, the worst-case bound of $\rho + 1$ is tight for two scenarios–the HND problem with two primary nodes, and the general TLND problem with more than two primary nodes. For the HND problem we use the primary shortest path heuristic; with more than two primary nodes, we apply the PNC heuristic to find the approximate Steiner tree. For both scenarios, the worst-case examples (Figures 5(a) and 6(a)) have similar network structures to our previous examples (Figures 3(a) and 4(a)); however, the edge costs are different. For both examples, the primary and secondary costs are equal for all edges except the pendant edge which has a secondary cost of 0. The pendant edge has a primary cost of q for the HND worst-case example, and 2q for the TLND worst-case example.

First, consider the HND problem with nonproportional costs. The example in Figure 5(a) shows that the bound of $\rho + 1 = 2$ is tight for this problem. For this example, the cost of the MST solution (Figure 5(b)) is

$$Z_{MST} = q + q = 2q.$$

The EST solution, obtained by first finding the shortest path between the two primary nodes, has cost

$$Z_{EST} \;=\; q + (q-1) - \varepsilon \;=\; 2q - 1 - \varepsilon.$$

The optimal solution, shown in Figure 5(d), costs (q + 0) = q. Thus, the performance ratio for the composite heuristic is arbitrarily close to 2 as desired.

Now consider the general TLND problem with more than 2 primary nodes and nonproportional costs. For the worst-case example shown in Figure 6(a), the cost of the MST solution (Figure 6(b)) is

$$Z_{MST} \;=\; qt/t \;+\; 2q \;=\; 3q.$$

The cost of the AST solution, using the PNC heuristic to construct the approximate Steiner tree, is

$$Z_{AST} \;=\; 2(q-1) \;+ q(t-1)/t \;+\; 1/t \;+ 0,$$

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

$$Z^* \;=\; qt/t \;+\; 0 \;=\; q.$$

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

4.4 Worst-case performance of Reverse Heuristics

The MST, EST, and AST heuristics first connect the primary nodes, and then choose secondary edges (if required) using the optimal secondary completion. 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 is intuitively appealing, especially for problem instances with secondary costs close to primary costs. However, by modifying our previous worst-case examples, we show that these reverse heuristics do not improve the composite method's worst-case performance.

First, let us describe a few alternative implementations of the reverse strategy.

(i) **Secondary Spanning Tree (SST) heuristic:**
Find the minimum spanning tree T(G) connecting all nodes N using secondary costs. Let T(P) denote the subtree connecting the primary nodes. Install primary facilities on all edges of T(P) and secondary facilities on the remaining edges.

(ii) **Incremental Steiner Tree (IST) heuristic:**
Using secondary costs, find the minimum tree spanning the node set $S \cup \{1\}$ (recall that primary node 1 is the root node). Using incremental costs for the edges of this tree, and the original primary costs on the remaining edges of G, construct either an exact or approximate Steiner tree (having worst-case ratio ρ) with primary nodes as terminals. Add the edges of the Steiner tree to the original subtree, and successively drop edges to eliminate any cycles. In this solution, the edges of the subtree spanning the primary nodes contain primary facilities; all other edges contain secondary facilities.

(iii) **Overlay Steiner Tree (OST) heuristic:**
This heuristic combines underlying principles from the previous two heuristics. First, find the minimum spanning tree T(G) connecting all the nodes. Instead of installing primary facilities on the primary subtree T(P) (as in the SST method), we now construct a Steiner tree (exact or approximate) using incremental edge costs (similar to the IST heuristic). As before, we add the edges of the Steiner tree (containing primary facilities) to the spanning tree, and eliminate cycles by successively dropping secondary edges.

Let us now determine upper bounds on the cost of these reverse heuristic solutions, assuming proportional costs, i.e., with the same primary-to-secondary cost ratio r for all edges. Again, we scale the costs so that the minimum spanning tree has a secondary cost of 1, and we let s (≤ 1) denote the secondary cost of the minimum Steiner tree with primary nodes as terminals. Let Z_{SST}, Z_{IST}, and Z_{OST} denote the costs of the SST, IST, and OST heuristic solutions. Note that:

(i) for the SST heuristic, since the primary subtree T(P) might potentially coincide with the (primary) minimum spanning tree T(G) which has cost r,

$$Z_{SST} \leq r;$$

(ii) the IST heuristic incurs a secondary cost of at most 1 unit in the first step, and a maximum primary cost of ρ r s in the second step (for the approximate Steiner tree connecting the primary nodes). Therefore,

$$Z_{IST} \leq \rho\, r\, s + 1; \text{ and,}$$

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

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

Observe that our upper bounds for the reverse heuristics are the same as the bounds for the forward heuristics (recall that the MST heuristic has an upper bound of r, and the AST heuristic cost does not exceed ρ r s + 1). Therefore, our previous worst-case bound (Theorem 2) is also valid for the reverse heuristics. Next, we show using an example that this bound is tight. In fact, we show that the *Expanded Composite heuristic* that selects the best solution from the two forward and three reverse heuristic solutions does not have better worst-case performance than the previous "forward" composite heuristic.

4.4.1 Worst-case examples

Again, we separately consider the HND problem for which we can find the exact Steiner tree (the shortest path connecting the two primary nodes), and the general TLND problem (containing more than two primary nodes) which requires the PNC heuristic.

Consider the example in Figure 3(a) with a sufficiently large integer value for d so that $d \geq q/(r-1)$. Choosing this high value ensures that any minimum spanning tree selects only one of the two nonhorizontal edges incident to the primary node on the left. We will argue that, for this example, the Expanded Composite heuristic has the same performance ratio of $r^2/(r^2 - r + 1)$ that we established previously (see Section 4.2.4). First, we can easily verify that the MST and the shortest path heuristics construct the solutions shown in Figure 3(b) and Figure 3(c). The SST heuristic generates the solution in Figure 3(b). The IST heuristic first selects the lower path as the minimum tree spanning the secondary nodes. The method then chooses the direct edge connecting the two primary nodes, and installs a primary facility on this edge. Figure 3(c) shows the resulting IST solution. Finally, the OST heuristic first connects the two primary nodes via the lower path, and then installs a primary facility (with an incremental cost of $r-1$) on the direct edge connecting the two primary nodes. The drop phase of the heuristic deletes one of the edges in the lower path, resulting in the design shown in Figure 3(c). Figure 3(d) contains the optimal solution for this problem instance. Comparing the cost of the heuristic solutions in Figures 3(b) and 3(c) with the cost of Figure 3(d) verifies that the performance ratio of the enhanced composite heuristic approaches $r^2/(r^2 - r + 1)$.

We now consider the PNC-based composite heuristic for the general TLND problem with more than two primary nodes and proportional primary-to-secondary costs. The example in Figure 7 shows that, even in this case, including the reverse heuristics does not improve the worst-case performance ratio. The network of Figure 7(a) contains q primary nodes, and each primary node is connected to the central secondary node via a "direct" path containing t edges (similar to Figure 4(a)). In addition, each primary node is also connected via the following "indirect" hopping path to the secondary node. Consider any two adjacent nodes, say nodes i and j, in the direct path. We construct an alternate path containing $(d + 1)$ edges (with d sufficiently large) between these two nodes. The first edge on this indirect path has a secondary cost of $(r-1)/2rqt$ (the direct edge from node i to node j also has the same secondary cost); all remaining edges have secondary costs of $(r+1)/2rdqt$. Thus, the cost of this alternate path between nodes i and j is

$$(r-1)/2rqt + d(r+1)/2rdqt = (r-1)/2rqt + (r+1)/2rqt.$$

Since the network contains qt pairs of adjacent nodes on the direct paths, the total secondary cost of all alternate paths between the primary nodes and the secondary node is

$$(r-1)/2r + (r+1)/2r.$$

Observe that the second component of this total cost equals the secondary cost of the pendant edge in Figure 4(a). Consequently, the costs of the solutions shown in Figures 7(b), 7(c), and 7(d) follow from similar calculations of MST, AST and optimal solutions in Figures 4(c), 4(b), and 4(c). Figure 7(b) shows the MST and the SST solution. Our previous observation concerning the total cost of alternate paths shows that the MST solution has a total cost of

$$r\{(r-1)/2r + (r+1)/2r)\} = r.$$

Similarly, the solution in Figure 7(c) has a cost that is arbitrarily close to r, while the cost of the optimal solution (Figure 7(d)) approaches

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

Hence, including the reverse heuristics does not improve the worst-case ratio.

We can similarly modify the examples shown in Figures 5(a) and 6(a) to prove that the worst-case performance of the Expanded Composite heuristic does not improve even for problems with varying primary-to-secondary cost ratios.

4.5 Summary of Worst-case Results

This section has described several intuitive heuristics 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 bounds would depend on the values of the parameters r_l and r_u. The following table summarizes the worst-case performance bounds for the different heuristics under various problem scenarios. Although we have not proved all the bounds shown in the table, they follow easily from our results.

Worst-case Performance Ratios for TLND heuristics

Heuristic Method	Proportional costs	Nonproportional costs
FORWARD heuristics:		
MST heuristic	r	∞
AST heuristic	$\rho + 1/r$	$\rho + 1$
REVERSE heuristics:		
SST heuristic	r	∞
IST heuristic	$\rho + 1/r$	$\rho + 1$
IST heuristic	$\rho + 1/r$	$\rho + 1$
Expanded 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 using composite 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 either heuristic alone. The bound of 4/3 for the Exact Steiner tree heuristic is one example. Our analysis leading to the results 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 better of two heuristic solutions based on 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 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. [1991]), 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 (guaranteed to be within 0.9% from optimality).

Acknowledgments: We are indebted to Professor James Orlin for illuminating discussions about heuristic worst-case analysis. Our results concerning the relationship between directed and undirected formulations are rooted in many discussions about network design with Dr. Richard Wong and the insights we have gained from him.

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.,

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

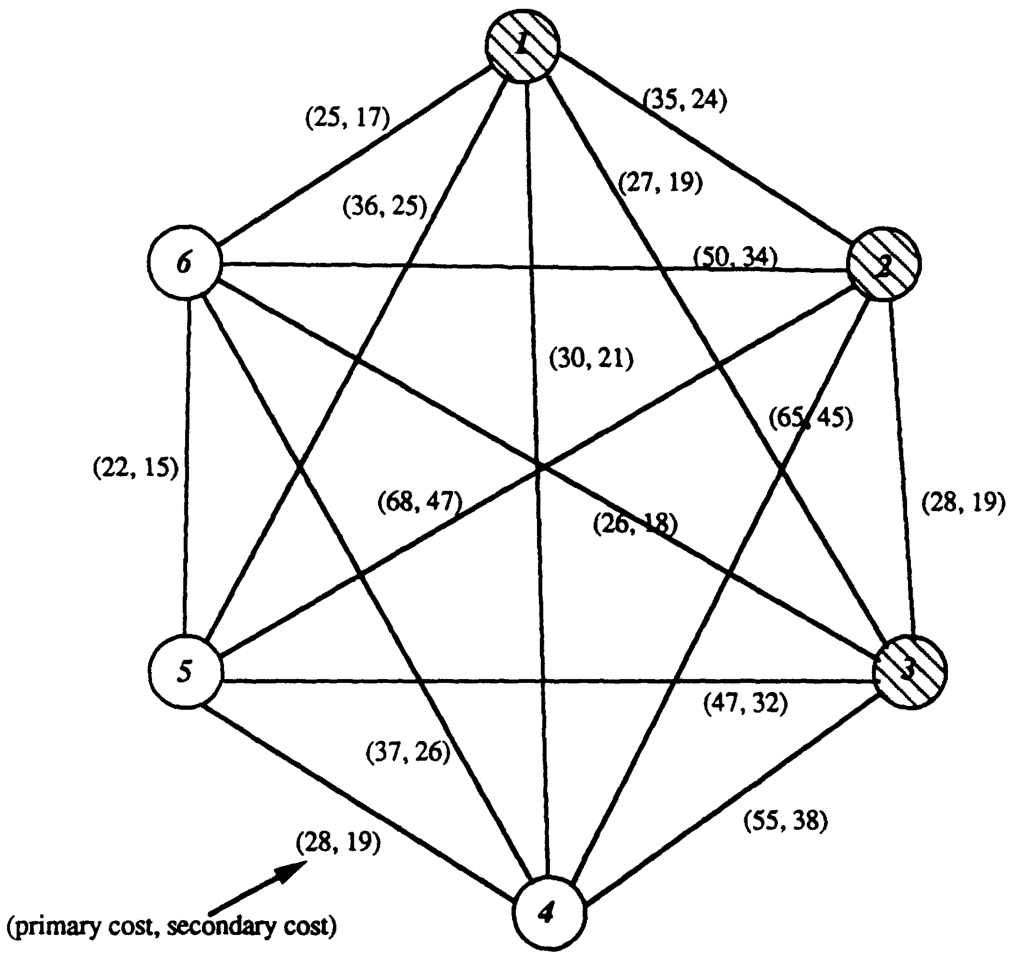
Let Π_i denote the set of paths from the root node 1 to node i *not containing node j*. Similarly, let Π_j denote the set of paths from node 1 to node j not containing node i. We will maintain commodity k's current flow into node i by increasing its flow on paths Π_i by $\phi = (f_{ij}^{h} + f_{ji}^{k} - 1)$ units, and correspondingly decreasing its flow on paths Π_j and arc (j,i). Observe that $0 < \phi \le f^k_{ji}$. 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^h_{ij} units on the paths Π_i. We can, therefore, increase commodity k's flow on these paths by $\varepsilon = (f^h_{ij} - f^k_{ij}) > 0$. Since $f^k_{ij} \le 1 - f^k_{ji}$, we have $\varepsilon \ge (f^h_{ij} + f^k_{ji} - 1) = \phi$. 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 $f'^k_{ji} = f^k_{ji} - \phi \ge 0$ 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:

$$\begin{aligned} f^h_{ij} + f'^k_{ji} &= f^h_{ij} + f^k_{ji} - \phi \\ &\le 1. \end{aligned}$$

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 $y_{ij} + y_{ji} \le 1$. ♦



▨ Primary node

○ Secondary node

Figure 1: 6-node TLND Example

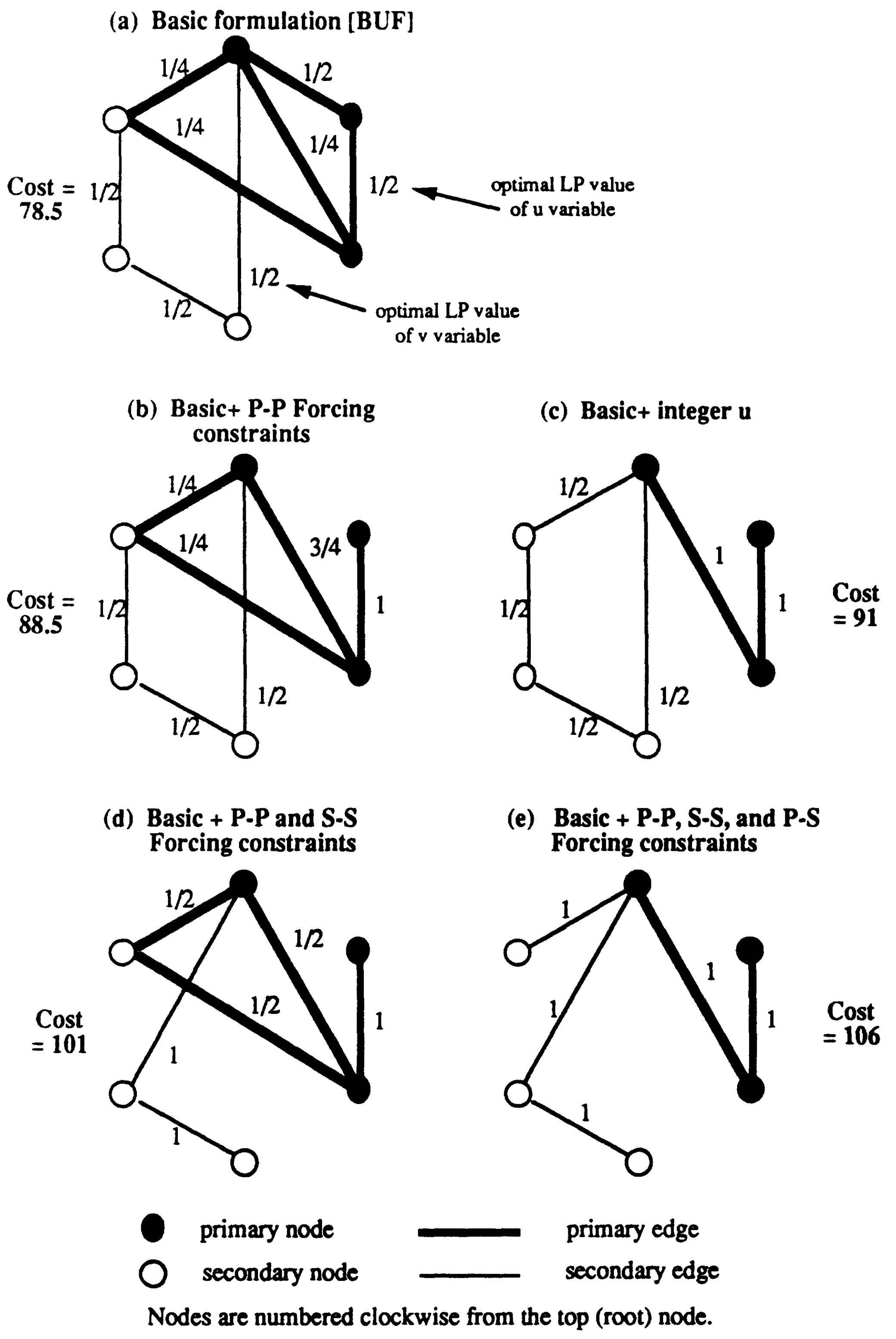

(a) Basic formulation [BUF]
1/4
1/2
1/4
1/4
Cost = 78.5
1/2
1/2
optimal LP value of u variable
1/2
1/2
optimal LP value of v variable
(b) Basic+ P-P Forcing constraints
1/4
1/4
3/4
Cost = 88.5
1/2
1
1/2
1/2
(c) Basic+ integer u
1/2
1
1/2
1
Cost = 91
1/2
1/2
(d) Basic + P-P and S-S Forcing constraints
1/2
1/2
1/2
Cost = 101
1
1
1
(e) Basic + P-P, S-S, and P-S Forcing constraints
1
1
1
1
Cost = 106
1
primary node
primary edge
secondary node
secondary edge
Nodes are numbered clockwise from the top (root) node.

Figure 2: LP solutions for 6-node Example

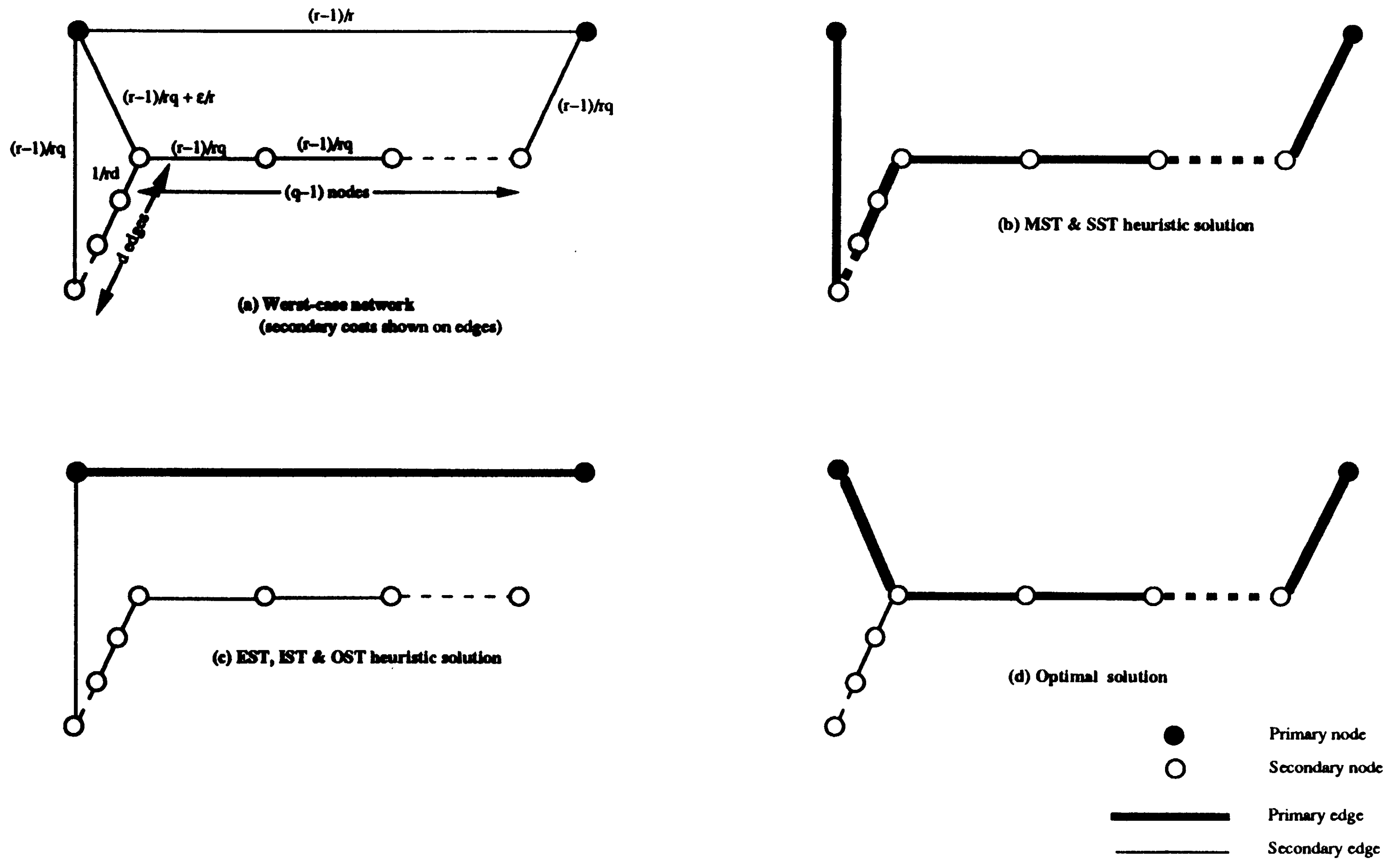
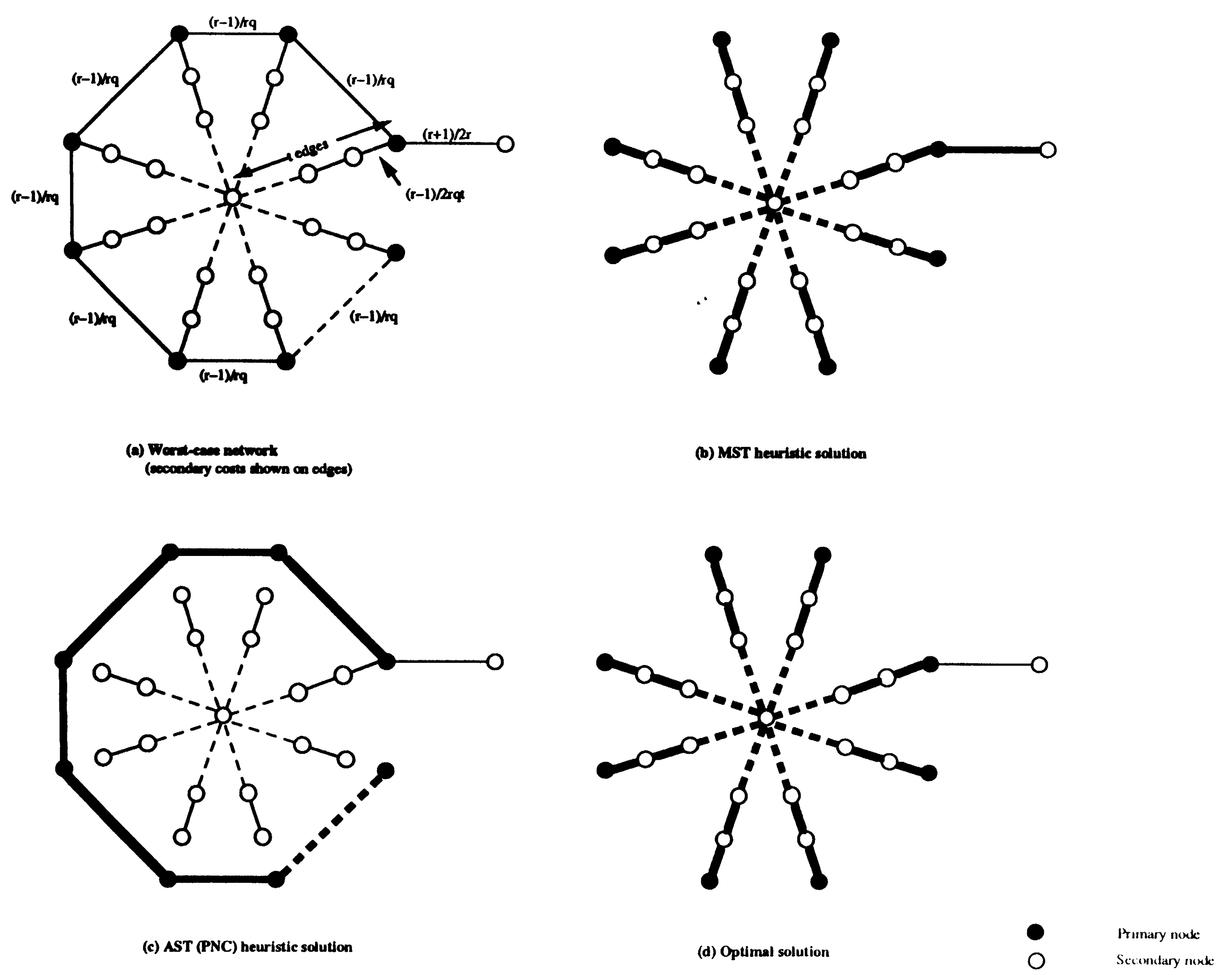


Figure 3: Worst-case example for HND problem ($\rho = 1$) with constant primary-to-secondary cost ratio r



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

(b) MST heuristic solution

(c) AST (PNC) heuristic solution

(d) Optimal solution

●	Primary node
○	Secondary node
▬	Primary edge
―	Secondary edge

Figure 4. Worst-case example for TLND problem ($\rho = 2$)
with constant primary-to-secondary cost ratio r

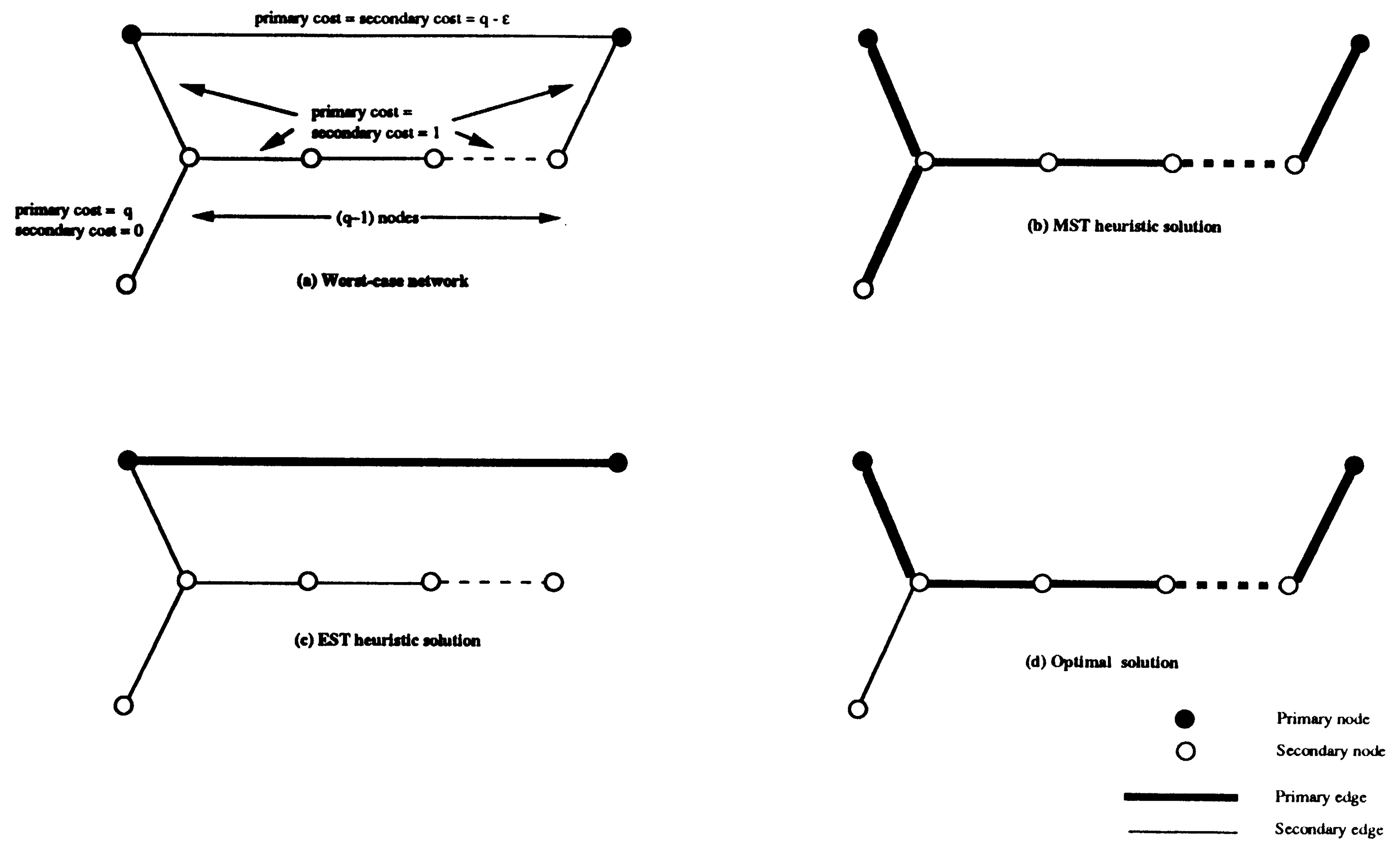


Figure 5: Worst-case example for HND problem ($\rho = 1$) with varying primary-secondary cost ratios

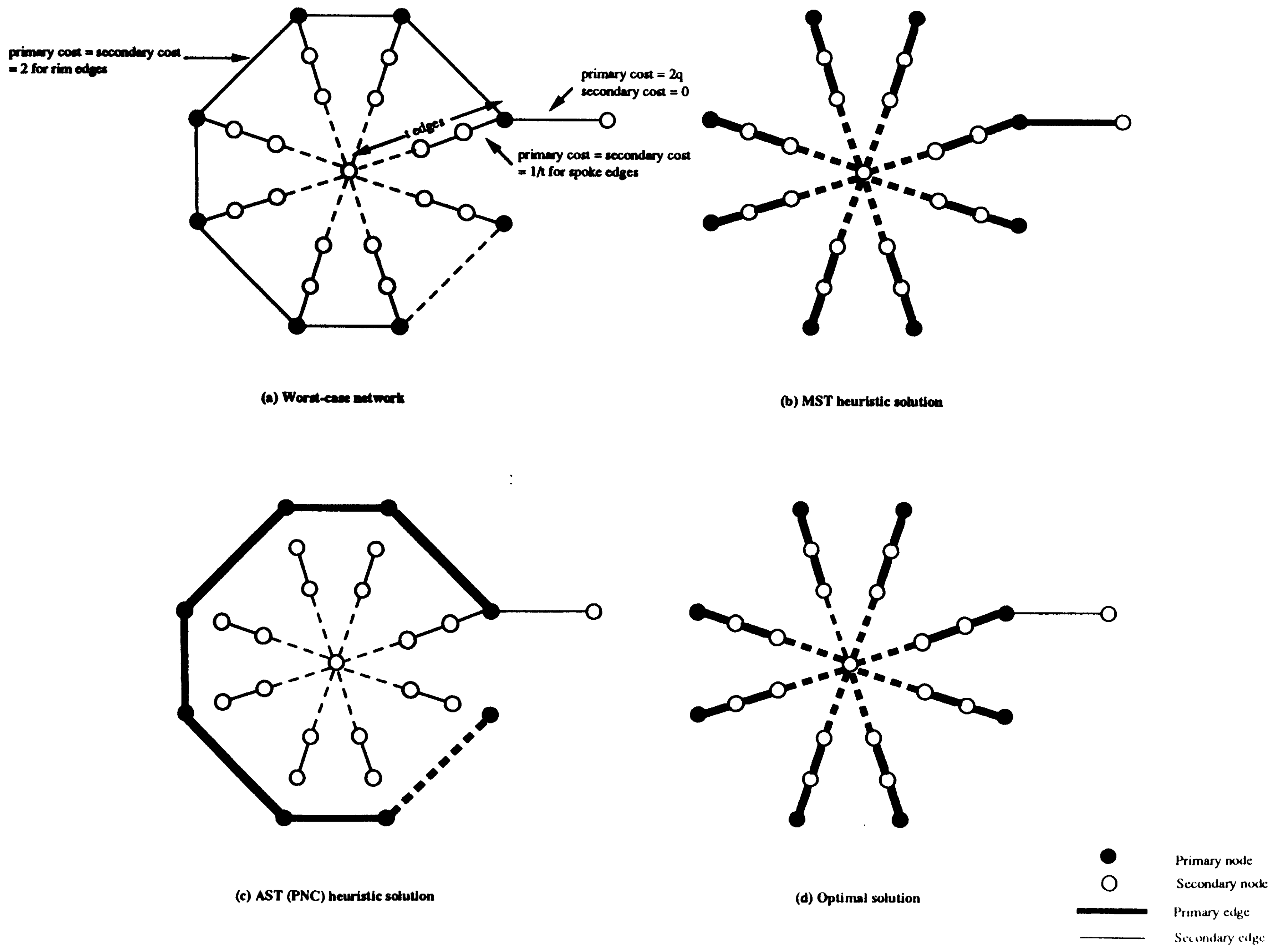
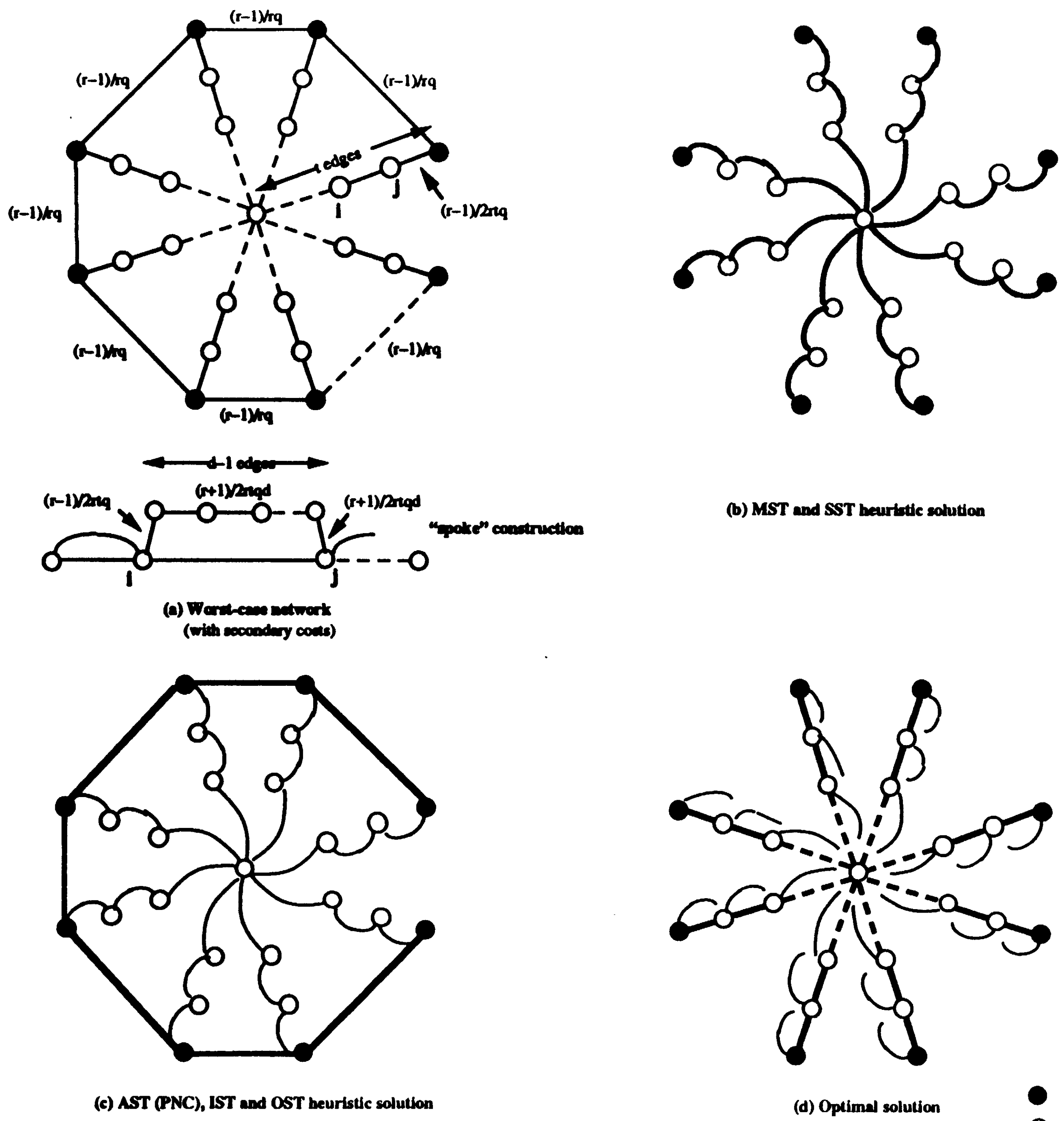

primary cost = secondary cost = 2 for rim edges
t edges
primary cost = 2q
secondary cost = 0
primary cost = secondary cost = 1/t for spoke edges
(a) Worst-case network
(b) MST heuristic solution
(c) AST (PNC) heuristic solution
(d) Optimal solution
Primary node
Secondary node
Primary edge
Secondary edge

Figure 6: Worst-case example for TLND problem ($\rho = 2$) with varying primary-to-secondary cost ratios



●	Primary node
○	Secondary node
▬	Primary edge
—	Secondary edge

Figure 7: Worst-case example for TLND problem ($\rho = 2$) with reverse heuristics with constant primary-to-secondary cost ratio r

References

AHUJA, R. K., T. L. MAGNANTI, and J. B. ORLIN. 1992. *Networks Flows: Theory, Algorithms and Applications.* Prentice Hall, in press.

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. 1991. 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., E. GORRES, and M. R. RAO. 1990. Solving the Steiner Tree Problem on a Graph using Branch and Cut. Working Paper, Northwestern University, Evanston.

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.

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., 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.

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.

PRODON, A., T. M. LIEBLING, and H. GROFLIN. 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.

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.