Strong Formulations for Network Design Problems with Connectivity Requirements

Thomas L. Magnanti
Sloan School of Management and Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

S. Raghavan
The Robert H. Smith School of Business and the Institute for Systems Research, Van Munching Hall, University of Maryland, College Park, Maryland 20742

The network design problem with connectivity requirements (NDC) includes as special cases a wide variety of celebrated combinatorial optimization problems including the minimum spanning tree, Steiner tree, and survivable network design problems. We develop strong formulations for two versions of the edge-connectivity NDC problem: unitary problems requiring connected network designs, and nonunitary problems permitting nonconnected networks as solutions. We (1) present a new directed formulation for the unitary NDC problem that is stronger than a natural undirected formulation; (2) project out two classes of valid inequalities—partition inequalities, and combinatorial design inequalities—that generalize known classes of valid inequalities for the Steiner tree problem to the unitary NDC problem; and (3) show how to strengthen and direct nonunitary problems. Our results provide a unifying framework for strengthening formulations for NDC problems, and demonstrate the power of flow-based formulations for network design problems with connectivity requirements. **NETWORKS, Vol. 45(2), 61–79 2005**

Keywords: network design; strong formulations; survivability; steiner forest problem; directed formulations; valid inequalities; projection

1. INTRODUCTION

Network Design Problems with Connectivity Requirements (*NDC*) arise in a wide variety of application domains including VLSI design and telecommunication network design. The increasing reliance on communication networks (and expectations of a digital future) places an enormous importance on the reliability of such networks. Given the enormous bandwidth capabilities of communication networks, and the increasing array of services provided over them, the failure of any link in such a network can have significant, perhaps even catastrophic consequences.

In this article, we consider network design problems with edge connectivity requirements. Informally, given requirements for the number of edge–disjoint paths between every pair of nodes, we wish to design a minimum cost network that satisfies these requirements. To set notation, and define the class of problems we consider, we formally state the NDC problem (also described as the Generalized Steiner Problem by Winter [33, 34]) as follows.

Network Design Problem with Connectivity Requirements (NDC): We are given an *undirected* graph $G = (N, E)$, with node set N and edge set E, and a cost vector $c \in \mathcal{R}_+^{|E|}$ on the edges E. We are also given a symmetric $|N| \times |N|$ requirement matrix $\mathbf{R} = [r_{ij}]$. The entry r_{ij} prescribes the minimum number of edge-disjoint paths needed between nodes i and j. We wish to select a set of edges that satisfy these requirements at minimum cost, as measured by the sum of costs of edges we choose.

The NDC problem models a wide variety of combinatorial optimization problems including the classical minimum spanning tree and Steiner tree problems. One important specialization of the NDC problem that arises in the design of telecommunications networks (see [7]) is the *Survivable Network Design Problem* (*SND*). In this application, each node v in the graph has a connectivity requirement r_v and the connectivity requirements between nodes s and t are given by $r_{st} = \min\{r_s, r_t\}$. Table 1 shows several other noteworthy cases of the NDC problem.

A few observations concerning the entries in Table 1 are worth making. The k-edge disjoint path problem seeks, at

Received January 2002; accepted October 2004
Correspondence to: S. Raghavan; e-mail: raghavan@isr.umd.edu
Contract grant sponsor: Singapore MIT Alliance (to T.L.M.).
Contract grant sponsor: Department of Defense, Laboratory for Telecommunications Sciences (to S.R.).
DOI 10.1002/net.20046
Published online in Wiley InterScience (www.interscience.wiley.com).

TABLE 1. Specializations of network design problems with connectivity constraints.

Problem type	SND or NDC	Connectivity requirements
Minimum spanning tree problem	SND	$r_v = 1$ for all nodes v.
Steiner tree problem	SND	$r_v = 1$ for all nodes v required in the tree; $r_v = 0$ for all other nodes.
k-Edge disjoint path problem	SND	$r_s = r_t = k$; $r_v = 0$ for all other nodes.
Minimum cost k-edge–connected spanning subgraph problem	SND	$r_v = k$ for all nodes v.
Minimum cost Steiner k-edge–connected spanning subgraph problem	SND	$r_v = k$ for all required nodes v; $r_v = 0$ for all other nodes.
Network design with low connectivity requirements (NDLC)	SND	$r_v \in \{0, 1, 2\}$ for all nodes v.
Point to point connection problems	NDC	$r_{s_it_i} = 1$ for given source sets $\{s_1, s_2, \ldots, s_P\}$and terminal sets $\{t_1, t_2, \ldots, t_P\}$; $r_{ij} = 0$ otherwise
Steiner forest problem	NDC	$r_{ij} = 1$ if $i \in T_q$ and $j \in T_q$ for some pairwise disjoint node sets in $T_1, T_2, \ldots, T_P$; $r_{ij} = 0$ otherwise.

minimum cost, k-edge disjoint paths between specified nodes s and t. The minimum cost k-edge connected spanning subgraph problem is an SND problem with $r_v = k$ for all nodes. The network design problem with low connectivity requirements (NDLC) is of particular interest to local telephone companies (see [7]). In this special case of the SND problem, the connectivity requirements are restricted to $\{0, 1, 2\}$. (Because most local telephone companies believe it is sufficient to protect against single link failures in the local loop, this problem is of significant importance to them.) In the Steiner forest problem, we are given a graph $G = (N, E)$ and node sets $T_1, T_2, \ldots, T_P$ with $T_i \cap T_j = \phi$ for all node set pairs i, j. We wish to design a graph at minimum cost that connects all the nodes in each node set. The point to point connection problem is a *special case* of the Steiner forest problem with $T_i = \{s_i, t_i\}$ for $i = 1, \ldots, P$.

NDC problems can be classified in two ways. If the connectivity requirements imply that all nodes with a (positive) connectivity requirement must be connected, we say the problem is a *unitary NDC problem.* Otherwise, it is a *nonunitary NDC problem.* For example, the SND problem is a unitary NDC problem, while the Steiner forest problem is a nonunitary NDC problem.

The examples in Table 1 show that the NDC problem models a very wide variety of connectivity problems on graphs. These problems appear both as stand alone problems and as subproblems in more complex network design applications (like VLSI design and telecommunications network design and management). Consequently, techniques for modeling and solving NDC problems have widespread applicability.

Considerable accumulated experience in the optimization literature has demonstrated the value of developing good linear programming relaxations (strong formulations) of combinatorial optimization problems. Strong formulations are very useful in developing exact algorithms solution methods (branch and bound, branch and cut, column generation) because their use rapidly accelerates these solution techniques. Strong formulations can also provide good bounds on the optimal solution and so are useful in assessing heuristic solution methods. In particular, dual-ascent heuristic techniques (that generate both lower bounds on the optimal solution value and feasible solutions to the combinatorial optimization problem) based upon strong formulations typically provide better solutions than those based upon weaker linear programming relaxations. The development in this article is motivated by a desire to develop *better linear programming relaxations* for NDC problems, and to provide a *unifying strengthening approach applicable to all NDC problems.*

Because the NDC problem models a wide variety of combinatorial optimization problems, the polyhedral structure of many special cases of the NDC problem have been well studied. Over the past 20 years, researchers have proposed a large number of formulations (and solution method based on them) for the Steiner tree problem. Most noteworthy among these are the articles by Wong [35], proposing a (bi)directed model for the undirected Steiner tree problem; Chopra and Rao [10, 11], examining the facial structure of the undirected Steiner tree polyhedron and its relationship to a directed formulation for the Steiner tree problem; Goemans [15], investigating extended formulations with node and edge variables for the Steiner tree problem and introducing combinatorial design inequalities for the Steiner tree problem; and Goemans and Myung [17], establishing the relationship between several formulations for the Steiner tree problem.

Several researchers have examined special cases of unitary NDC problems with higher connectivity requirements (i.e., greater than 1). For series-parallel graphs, Mahjoub [27] and Baïou and Mahjoub [1] provide complete descriptions of the 2-edge-connected spanning subgraph polytope and the Steiner 2-edge-connected spanning subgraph polytope, respectively. Didi Biha and Mahjoub [5] provide a complete description of the k-edge-connected polytope on series-parallel graphs. Boyd and Hao [6] introduce a class of valid inequalities for the 2-edge-connected spanning subgraph polytope and describe necessary and sufficient conditions for these valid inequalities to define facets. Based on a result by Robbins, Chopra [9] describes a directed formulation for the NDLC problem in a model that permits unlimited edge replication. Using a result due to Nash-Williams [29], a generalization of Robbins theorem, Goemans [14] shows how to strengthen a well-known cutset formulation for the SND problem with connectivities $r_v \in \{0, 1, \text{even}\}$ in a model that permits unlimited edge replication. Grötschel et al. [21–23, 32] investigate the polyhedral structure of both the edge- and node-connectivity versions of the SND problem. One of these articles [21] investigates the polyhedral structure of the NDLC problem, while another [23] examines the polyhedral structure of SND problems whose highest connectivity requirements are three or more. The other two articles, [22] and [32], contain comprehensive summaries of polyhedral results for the SND problem. Recently, Balakrishnan et al. [2] propose a connectivity splitting-model for the NDC that strengthens a well-known cutset formulation for the problem. Their model is particularly useful for worst-case analysis of heuristics for the NDC problem.

Researchers have proposed many solution methods (both exact and approximate) for the NDC problem and its specializations. Our discussion has focused on polyhedral research in this area. Survey articles by Grötschel et al. [22], Raghavan and Magnanti [31], and Frank [12] provide more comprehensive reviews of research on the NDC and its specializations.

In this article, we develop strong formulations for both unitary and nonunitary NDC problems. Our work differs from earlier research in several ways. Goemans [14] and Grötschel et al. [22] have shown in various forms how to use a result due to Nash-Williams to obtain stronger models for the SND problem with connectivities $r_v \in \{0, 1, \text{even}\}$. We show that although the Nash-Williams theorem is useful to motivate the directing procedure, it does not play a role in strengthening the formulation (i.e., it is not necessary!). Consequently, we are able to generalize the directing procedure to strengthen formulations for *all unitary NDC problems.*

Next, by projecting from a strengthened (extended) formulation for the unitary NDC problem, we develop two classes of valid inequalities that are generalizations of facet-defining valid inequalities for the Steiner tree problem. For special cases of the unitary NDC problems, several researchers [10, 11, 15, 22] have shown how to project these inequalities from extended formulations that are equivalent to the flow-based formulation we have used for the NDC problem. We develop the projection from the flow-based formulation for three reasons. First, several extended formulations that are equivalent to the flow formulation for the Steiner tree problem (e.g., node weighted extended formulations for the Steiner tree problem [15]) do not generalize to the NDC problem. Second, the understanding of the flow-based formulation and its relationship to the cutset formulation will permit us to develop a directing and strengthening technique for nonunitary problems that requires flow variables. Third, we believe this article is the first to explicitly show how to project from this flow-based formulation.

Finally, we show how to direct nonunitary NDC problems. In the literature, these problems appear to have received significantly less attention. We implement our directing procedure using flow variables to obtain strengthened (flow-based) formulations for nonunitary NDC problems. It it not obvious how to implement the directing procedure without using flow variables.

The rest of this article is organized as follows. In Section 2 we review two well-known formulations for the NDC problem—a natural formulation with edge variables, and an extended formulation containing both flow and edge variables. Next, in Section 3, we first motivate the directing procedure using a result by Nash-Williams that applies to unitary NDC problems with restricted connectivities. We then develop a strengthened formulation of all unitary NDC problems, *without* using the Nash-Williams result. Section 4 deals with the strength of the improved formulation. First, we provide some preliminary results regarding the projection of the improved flow formulation onto the space of the edge variables. Next, we show how to project both partition inequalities and combinatorial design inequalities (which includes the special case of odd-hole inequalities) from the improved flow formulation. In Section 5 we examine nonunitary NDC problems. We first show how to strengthen a formulation of the Steiner forest problem by applying a new directing technique. In Section 6 we use this technique to strengthen formulations for all NDC problems. Finally, in Section 7, we provide some concluding remarks.

1.1. Notation

We assume familiarity with standard graph theory terminology. We work with undirected graphs, denoted by $G = (N, E)$, and directed graphs, denoted by $D = (N, A)$, which we refer to as *graphs* and *digraphs.* To distinguish between directed and undirected graphs, we refer to undirected graphs as *graphs,* undirected edges as *edges,* directed graphs as *digraphs,* and directed edges as *arcs.* We use braces to denote an edge between nodes i and j, that is, $\{i, j\}$, and parentheses to denote a directed arc from node i to node j, that is, (i, j). $\text{econ}(T) := \max\{r_{ij} | j \in T, i \in N \backslash T\}$ denotes the *edge-connectivity* requirements of a set of nodes $T \subset N$. It is the maximum edge-connectivity requirement between any node in T and its complement. For NDC problems we refer to $\text{econ}(i)$ as the *maximum connectivity* requirement of node i. If $\text{econ}(i) > 0$, we say node i is a *required* node. In models that permit parallel edges, we let b_{ij} represent the number of parallel edges allowed between nodes i and j. For example, if a model permits two edges between nodes i and j, then the graph G contains the edge $\{i, j\}$ and $b_{ij} = 2$. In an undirected graph, any set of nodes $T \subset N$ defines a *cut* $\delta(T) = \{\{i, j\} : i \in N \backslash T, j \in T, \{i, j\} \in E\}$. Similarly, any set of nodes $T \subset N$ in a directed graph defines a *dicut* $\delta^-(T) = \{(i, j) : i \in N \backslash T, j \in T, (i, j) \in A\}$ of arcs directed into the node set T and a dicut $\delta^+(T) = \{(i, j) : i \in T, j \in N \backslash T, (i, j) \in A\}$ of arcs directed out of T. An *$s - t$ cut* is a cut $\delta(T)$ with $s \notin T$ and $t \in T$. Similarly, an *$s - t$ dicut* is a dicut, say $\delta^-(T)$, with $s \notin T$ and $t \in T$. When we consider flow formulations, the *capacity* of an edge or arc represents the maximum flow of any commodity that may be sent on that edge or arc. The *capacity of a cut* $\delta(W)$ is the sum of the capacity of the edges in the cut, and the *capacity of a dicut* $\delta^-(W)$ is the sum of the capacities of the arcs in the dicut.

Sometimes we will want to eliminate the variables from an "extended" formulation of a problem. Let $\mathbf{A}$ and $\mathbf{B}$ be two given matrices and $\mathbf{d}$ be a column vector, all with the same number of rows. Consider the polyhedron $P = \{(\mathbf{x}, \mathbf{f}) : \mathbf{Ax} + \mathbf{Bf} \geq \mathbf{d}\}$. The polyhedron $Q = \{\mathbf{x} : \mathbf{Ax} + \mathbf{Bf} \geq \mathbf{d}$ for some vector $\mathbf{f}\}$ obtained by eliminating the $\mathbf{f}$ variables is the *projection* of the polyhedron P onto the space of the "natural" $\mathbf{x}$ variables. We denote it as $\mathrm{Proj}_{\mathbf{x}}(P)$. Suppose we have two formulations for a problem with the same set of "natural variables" (in this article, the natural variables are edge variables x_{ij}). We say the two formulations are *equivalent* if they provide the same objective value, when solved as linear programs, for all objective function coefficients of the natural variables (the objective function coefficients for the additional variables are zero). In other words, two formulations are equivalent if their projection onto the space of the natural variables is identical. We say that adding an inequality $\mathcal{I}$ *strengthens* a formulation of a (mixed) integer programming problem if it is valid and adding it to the formulation improves the objective value of the linear programming relaxation of the formulation for some choice of the objective function coefficients. We say that a formulation $\mathcal{P}_1$ is *stronger* than a formulation $\mathcal{P}_2$ if, when solved as linear programs, the objective value of $\mathcal{P}_1$ is always as good as the objective value of $\mathcal{P}_2$, and in at least one instance is strictly better than the objective value of $\mathcal{P}_2$.

2. FORMULATIONS FOR THE NDC PROBLEM

In this section we describe two well-known models for the NDC problem—one a cutset model, and the other a multicommodity flow-based model. For the flow-based model we also show how to minimize the number of commodities, a method that proves invaluable in our subsequent discussions.

Menger's theorem [28] states that the number of edge-disjoint paths between a pair of nodes, say s and t, is equal to the minimum number of edges across any cut between them, that is, any $s - t$ cut. Consequently, the following well-known "cutset" formulation, with x_{ij} representing the number of copies of edge $\{i, j\}$ in the network, is a valid representation of the NDC problem.

Cutset Formulation for the NDC Problem:

$$\text{Minimize} \quad \sum_{\{i,j\} \in E} c_{ij} x_{ij} \tag{1a}$$

$$\text{subject to:} \quad \sum_{\{i,j\} \in \delta(S)} x_{ij} \geq \mathrm{econ}(S)$$

$$\text{for all nodes sets } S \text{ with } \phi \neq S \neq N, \tag{1b}$$

$$x_{ij} \leq b_{ij} \quad \text{for all } \{i, j\} \in E, \tag{1c}$$

$$x_{ij} \geq 0 \quad \text{and integer, for all } \{i, j\} \in E. \tag{1d}$$

An alternative way to formulate the problem is to enforce the connectivity requirements of the matrix $\mathbf{R}$ using commodity flows. For each pair $\{s, t\}$ of nodes, with $r_{st} \geq 1$, create a commodity, arbitrarily choosing one of the nodes as the origin of the commodity and the other nodes as the destination. Let K denote the set of commodities and let q_k, for each $k \in K$, denote the edge-connectivity requirement between the origin and destination of commodity k: if $r_{st} = 3$, then $q_k = 3$ for the commodity k corresponding to the node pair $\{s, t\}$. The following mixed integer program, with x_{ij} representing the number of copies of edge $\{i, j\}$ in the network and f^k_{ij} the flows, is a valid formulation for the NDC problem.

Undirected Flow Formulation for the NDC Problem:

$$\text{Minimize} \quad \sum_{\{i,j\} \in E} c_{ij} x_{ij} \tag{2a}$$

$$\text{subject to:} \quad \sum_{j \in N} f^k_{ji} - \sum_{l \in N} f^k_{il} = \begin{cases} -q_k & \text{if } i = O(k), \\ q_k & \text{if } i = D(k), \\ 0 & \text{otherwise}, \end{cases}$$

$$\text{for all } i \in N \text{ and } k \in K, \tag{2b}$$

$$\left.\begin{matrix} f^k_{ij} \\ f^k_{ji} \end{matrix}\right\} \leq x_{ij} \quad \text{for all } \{i, j\} \in E \text{ and } k \in K, \tag{2c}$$

$$f^k_{ij}, f^k_{ji} \geq 0 \quad \text{for all } \{i, j\} \in E \text{ and } k \in K, \tag{2d}$$

$$x_{ij} \leq b_{ij} \quad \text{for all } \{i, j\} \in E, \tag{2e}$$

$$x_{ij} \geq 0 \quad \text{and integer, for all } \{i, j\} \in E. \tag{2f}$$

Let $P^{(1)}_{\mathrm{cut}}$ denote the polyhedron defined by the linear relaxation of constraints (1b)–(1d). Let $P^{(2)}_{\mathrm{flo}}$ denote the polyhedron defined by the linear relaxation of constraints (2b)–(2f). The max-flow min-cut theorem implies that the cutset and flow formulations are equivalent in the following sense.

Lemma 2.1. $P_{cut}^{(1)} = Proj_{\mathbf{x}}(P_{flo}^{(2)})$.

Notice that the cutset formulation is of exponential size, while the flow formulation is compact: it has $\mathbb{O}(|K|(|E| + |N|))$ constraints and $\mathbb{O}(|K|\,|E|)$ variables.

A simple, and naive, way to determine the number of commodities in the flow formulation is to create a commodity for every pair of nodes with a connectivity requirement. For an underlying graph with 100 nodes and 1000 edges and positive connectivity requirements between all nodes, this approach would create a commodity for every node pair and so 4950 commodities. Consequently, the model would contain 495,000 flow balance constraints, 9,900,000 constraints of type (2c), 9,900,000 nonnegativity constraints for the flow variables, 1000 constraints of type (2e), and 1000 nonnegativity and integrality constraints for the edge variables. As this example shows, the flow formulation can be very large.

By using fewer commodities, if possible, we could reduce the size of the formulation. To accomplish this objective, we can use an idea that Gomory and Hu [18] used when solving the classical network synthesis problem. Given the connectivity requirements matrix $\mathbf{R}$, create a "requirement" graph G^R on the node set N, giving edge $\{i, j\}$ between nodes i and j in G^R a weight r_{ij}. Gomory and Hu [18] showed that it is sufficient to consider the connectivity requirements only for the edges on a maximum spanning tree of this graph. It is easy to verify this result using the max-flow min-cut theorem and the maximum spanning tree optimality conditions. As is well known, a spanning tree is a maximum spanning tree if and only if it satisfies the following optimality condition: for every nontree edge $\{k, l\}$ of G^R, $r_{ij} \geq r_{kl}$ for every edge $\{i, j\}$ contained in the (tree) path on the maximum spanning tree connecting nodes k and l. As a result, any network that satisfies the requirements of the maximum spanning tree has sufficient capacity to satisfy the requirements of nontree edges.

Gomory and Hu's [18] result permits us to model the edge-connectivity requirements in any NDC problem with $|N| - 1$ or fewer commodities. We simply compute the maximum spanning tree of the requirement graph, which we now refer to as the *requirement spanning tree,* and create commodities only for those edges of the maximum spanning tree with nonzero weight. Because the requirement spanning tree has $|N| - 1$ edges and finding it requires $\mathbb{O}(|E| + |N|\log|N|)$ time [13], this procedure creates at most $|N| - 1$ commodities (we will not create commodities for zero weight edges of the requirement spanning tree) and requires $\mathbb{O}(|E| + |N|\log|N|)$ time.

The cutset formulation (1) and the undirected flow formulation (2) for the NDC problem are known to be weak. Computational experiments reported by Grötschel et al. [20, 23] confirm this result, particularly when the requirement spanning tree has edges with connectivity requirement 1. Jain (see [24]) provides some theoretical evidence to support these results. He shows that the worst-case ratio of the optimal value of the integer program to the optimal value of the linear programming relaxation of the cutset formulation is 2.

3. STRONGER FORMULATIONS FOR UNITARY NDC PROBLEMS

In this section we first describe a procedure for directing unitary NDC problems for situations when the connectivity requirements are all even or 1. We then generalize this result, developing a strong (i.e., directed) formulation for any unitary NDC problem (i.e., even those with odd connectivity requirements). For ease of exposition, for the rest of this article we assume that the model *does not permit edge replication.* It is straightforward to verify that the results apply to models that permit edge replication.

3.1. Directing the Unitary NDC Problem

The following result due to Nash-Williams [29] provides a key ingredient for transforming the undirected formulation to a directed one.

Theorem 3.1 (Nash-Williams). *Suppose G is an undirected graph with r_{xy} edge-disjoint paths connecting each pair x and y of its nodes. Then it is possible to direct the graph (i.e., orient its edges) so that the resulting digraph contains $\lfloor r_{xy}/2 \rfloor$ arc-disjoint paths from node x to node y and $\lfloor r_{xy}/2 \rfloor$ arc-disjoint paths from node y to node x.*

Consider any unitary NDC problem whose connectivity requirements r_{st} are even or 1. We can view any feasible integer solution to this problem as follows: it is connected and contains several 2-edge-connected components. If we contract the 2-edge-connected components, the solution becomes a tree. The edges on the tree are the *bridge* edges in the feasible solution before we contracted the 2-edge-connected components; that is, removing these edges disconnects the graph defined by that solution.

The Nash-Williams theorem permits us to direct the edges of each 2-edge-connected component so that for any pair of nodes i and j with $r_{ij} \geq 2$ (by assumption these requirements must be even), the network contains $r_{ij}/2$ directed arc-disjoint paths from node i to node j, and $r_{ij}/2$ directed arc-disjoint paths from node j to node i. Once oriented, each 2-edge-connected component contains a directed path between every pair of its nodes. Therefore, if nodes i and j belong to the same 2-edge-connected component and $r_{ij} = 2$, the oriented network contains a directed path from node i to node j and a directed path from node j to node i. To direct the bridges, consider the tree obtained by contracting each 2-edge-connected component of the solution. Select any one of the nodes created by the con-

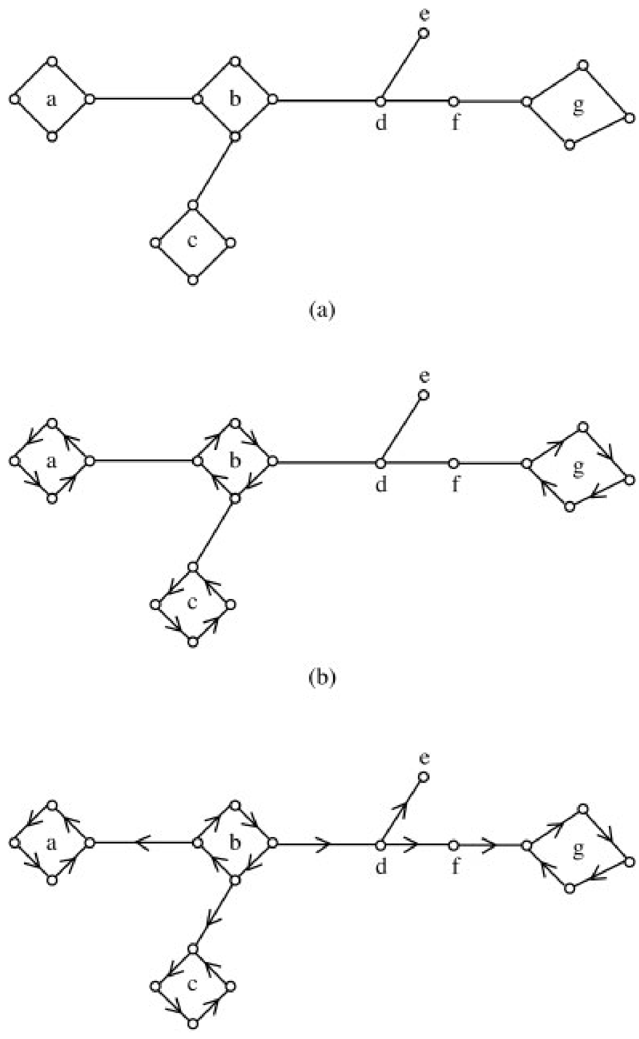


FIG. 1. Directing the bridges of a feasible solution to the unitary NDC problem. (a) Feasible solution. (b) Direct the edges of each 2-edge-connected component. (c) Direct the bridges away from component b.

traction as a root node and direct the tree away from this node.

Figure 1 illustrates this directing procedure. In this example a, b, c and g are 2-edge-connected components. Between every pair of nodes s and t in these components $r_{st} = 2$. We orient the edges of each component (see Fig. 1b) so that it contains a directed path between every pair of nodes in each 2-edge-connected component. Next, we select the node created by contracting component b as the root node and direct the tree edges (i.e., the bridges of the solution) away from node b. Figure 1c shows the graph at the conclusion of the directing procedure.

These observations permit us to formulate the unitary NDC problem as follows. Let y_{ij} be 1 if edge $\{i, j\}$ is oriented from node i to node j in the directing procedure applied to the optimal solution [i.e., the oriented network contains arc (i, j)] and be 0 otherwise.

Directed Cut Formulation for the Unitary NDC Problem ($r_{st} \in \{0, 1, \text{even}\}$):

$$\text{Minimize} \quad \sum_{\{i,j\} \in E} c_{ij} x_{ij} \tag{3a}$$

$$\text{subject to:} \quad \sum_{(i,j) \in \delta^-(S)} y_{ij} \geq \frac{\text{econ}(S)}{2}$$

$$\text{if econ}(S) \geq 2, \text{ for all } S \subseteq N, \tag{3b}$$

$$\sum_{(i,j) \in \delta^-(S)} y_{ij} \geq 1 \quad \text{if econ}(S) = 1, \text{ for all } S, \text{ root} \notin S, \tag{3c}$$

$$y_{ij} + y_{ji} \leq x_{ij} \quad \text{for all } \{i, j\} \in E, \tag{3d}$$

$$x_{ij} \leq 1 \quad \text{for all } \{i, j\} \in E, \tag{3e}$$

$$y_{ij}, y_{ji}, x_{ij} \geq 0 \quad \text{and integer, for all } \{i, j\} \in E. \tag{3f}$$

Because $\text{econ}(S) \equiv \max\{r_{ij} | j \in S; i \in N \backslash S\}$, constraint (3b) ensures that for every pair of nodes s and t with $r_{st} \geq 2$, every $s - t$ dicut contains at least $r_{st}/2$ arcs and every $t - s$ dicut contains at least $r_{st}/2$ arcs. Menger's theorem ensures that the oriented network contains at least $r_{st}/2$ arc-disjoint paths from node s to node t and $r_{st}/2$ arc-disjoint paths from node t to node s. Similarly, constraints (3b) and (3c) ensure that the oriented network contains a directed path from the root to every required node. Constraint (3d) ensures that the oriented network contains at most one of the arcs (i, j) and (j, i).

Proposition 3.2. *The directed cut model is a valid formulation for the unitary NDC problem when $r_{st} \in \{0, 1, even\}$ in the sense that* $(\mathbf{x}, \mathbf{y})$ *is a feasible solution to this model if and only if* $\mathbf{x}$ *is an incidence vector of a feasible NDC design* [*that is*, $\mathbf{x}$ *is a feasible solution to the cutset model* (1)].

Proof. Suppose $\mathbf{x}$ is an incidence vector of a feasible NDC design. The argument following Theorem 3.1 shows how to construct an integer vector $\mathbf{y}$ so that $(\mathbf{x}, \mathbf{y})$ is a valid solution for the directed cut model.

To establish the converse, suppose $(\mathbf{x}, \mathbf{y})$ is a feasible solution to the directed cut model. Let Q be any node set with $\text{econ}(Q) = \text{econ}(N \backslash Q) \geq 2$. Combining inequality (3b) for $S = Q$ and $S = N \backslash Q$ and the inequalities (3d) summed over all $\{i, j\} \in \delta(Q)$ gives

$$\sum_{\{i,j\} \in \delta(Q)} x_{ij} \geq \sum_{(i,j) \in \delta^+(Q)} y_{ij} + \sum_{(j,i) \in \delta^-(Q)} y_{ji} \geq \text{econ}(Q).$$

If Q is any node set with $\text{econ}(Q) = \text{econ}(N \backslash Q) = 1$, assume without loss of generality that the root node is not in Q. Then the inequality (3c) with $S = Q$ and the inequality (3d) summed over $\{i, j\} \in \delta(Q)$ imply that

$$\sum_{\{i,j\}\in\delta(Q)} x_{ij} \geq 1.$$

Therefore, $\mathbf{x}$ is a feasible solution to the cutset formulation (1). ∎

In the next section we describe a relaxation of the directed cut model (3) that applies to all unitary NDC problems and obviates the need for the Nash-Williams theorem.

3.2. Generalizing the Directing Procedure

The directed cut formulation (3) is not valid for unitary NDC problems with odd connectivity requirements. As an example, consider an SND problem defined on K_4, the complete graph on four nodes, assuming each node has a connectivity requirement of 3. The optimal solution for this problem is K_4. For any node i in K_4, there is no way to direct the edges so that both $\delta^+(i)$ and $\delta^-(i)$ are at least 1.5.

Suppose, however, that in the directed cut formulation we relax the integrality constraints imposed upon the y_{ij} variables, and interpret y_{ij} as the capacity on the flow of any commodity on arc (i, j). We will show that this formulation is a valid mixed integer program for any unitary NDC problem. Consider any feasible solution to an unitary NDC problem. As we noted previously, the solution is a connected graph consisting of 2-edge-connected components and bridges. Suppose (1) we set $y_{ij} = y_{ji} = 1/2$ for each edge on the 2-edge-connected components, and (2) direct the bridges away from the component that contains the root node, setting y_{ij} to 1 if edge $\{i, j\}$ is oriented from node i to node j, and 0 otherwise (i.e., once we contract the 2-edge-connected components the directing procedure for the bridges is similar to the directing procedure for the Steiner tree). The resulting solution $(\mathbf{x}, \mathbf{y})$ is feasible in the directed cut formulation if we relax the integrality condition on $\mathbf{y}$. Therefore, the following directed cut formulation is valid for all unitary NDC problems.

Directed Cut Formulation for the Unitary NDC Problem:

$$\text{Minimize} \quad \sum_{\{i,j\}\in E} c_{ij}x_{ij} \tag{4a}$$

$$\text{subject to:} \quad \sum_{(i,j)\in\delta^-(S)} y_{ij} \geq \frac{\text{econ}(S)}{2}$$

$$\text{if econ}(S) \geq 2, \text{ for all } S \subseteq N, \tag{4b}$$

$$\sum_{(i,j)\in\delta^-(S)} y_{ij} \geq 1 \quad \text{if econ}(S) = 1, \text{ for all } S, \text{ root} \notin S, \tag{4c}$$

$$y_{ij} + y_{ji} \leq x_{ij} \quad \text{for all } \{i, j\} \in E, \tag{4d}$$

$$x_{ij} \leq 1 \quad \text{for all } \{i, j\} \in E, \tag{4e}$$

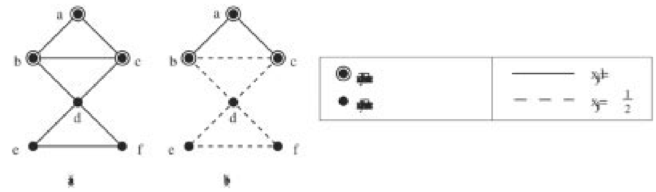


FIG. 2. The directed cut formulation is stronger than the cutset formulation. (a) Underlying graph. (b) Fractional solution to the cutset formulation that is cut off by the directed cut formulation.

$$y_{ij}, y_{ji} \geq 0 \quad \text{for all } \{i, j\} \in E, \tag{4f}$$

$$x_{ij} \geq 0 \quad \text{and integer, for all } \{i, j\} \in E. \tag{4g}$$

Proposition 3.3. *The directed cut model* (4) *is a valid formulation for the unitary NDC problem in the sense that* $(\mathbf{x}, \mathbf{y})$ *is a feasible solution to this model if and only if* $\mathbf{x}$ *is an incidence vector of a feasible NDC design.*

Proof. Similar to the proof of Proposition 3.2. ∎

Let $P^{(4)}_{\text{dcut}}$ denote the polyhedron defined by the linear relaxation of constraints (4b)–(4g). We can now make a stronger claim.

Lemma 3.4. $Proj_{\mathbf{x}}(P^{(4)}_{dcut}) \subseteq P^{(1)}_{cut}$.

Proof. Similar to the proof of the converse in Proposition 3.2. ∎

To see that there are instances where $\text{Proj}_{\mathbf{x}}(P^{(4)}_{\text{dcut}}) \subset P^{(1)}_{\text{cut}}$, consider the NDLC example shown in Figure 2. Nodes a, b, and c have a connectivity requirement of 2. Nodes d, e, and f have a connectivity requirement of 1. The solution $x_{ab} = x_{ac} = 1$ and $x_{bc} = x_{bd} = x_{cd} = x_{de} = x_{df} = x_{ef} = 0.5$ is feasible to $P^{(1)}_{\text{cut}}$, but not contained in $\text{Proj}_{\mathbf{x}}(P^{(4)}_{\text{dcut}})$.

3.2.1. Flow Formulation. The max-flow min-cut theorem permits us to formulate an improved flow model, with multiple commodities K to be defined below, that is equivalent to the directed cut model (4).

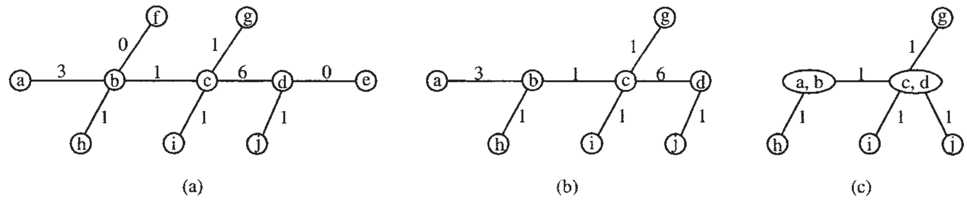
Improved Undirected Flow Formulation for the Unitary NDC Problem:

$$\text{Minimize} \quad \sum_{\{i,j\}\in E} c_{ij}x_{ij} \tag{5a}$$

$$\text{subject to:} \quad \sum_{j\in N} f^k_{ji} - \sum_{l\in N} f^k_{il} = \begin{cases} -q_k & \text{if } i = O(k), \\ q_k & \text{if } i = D(k), \\ 0 & \text{otherwise,} \end{cases}$$

$$\text{for all } i \in N \text{ and } k \in K, \tag{5b}$$

$$f^k_{ij} + f^h_{ji} \leq x_{ij} \quad \text{for all } \{i, j\} \in E \text{ and } k, h \in K, \tag{5c}$$



Commodity Origin	Commodity Destination	Commodity Requirement
a	b	1.5
b	a	1.5
c	d	3
d	c	3
a	c	1
a	g	1
a	h	1
a	i	1
a	j	1

(d)

FIG. 3. Commodity selection procedure for the unitary NDC problem. (a) Requirement spanning tree. (b) Tree obtained by deleting edges $\{s, t\}$ with $r_{st} = 0$. (c) Tree obtained by contracting edges $\{s, t\}$ with $r_{st} \geq 2$. (d) Commodities in improved flow formulation (5) with node a selected as the root.

$$f_{ij}^k, f_{ji}^k \geq 0 \quad \text{for all } \{i, j\} \in E \text{ and } k \in K, \tag{5d}$$

$$x_{ij} \leq 1 \quad \text{for all } \{i, j\} \in E, \tag{5e}$$

$$x_{ij} \geq 0 \quad \text{and integer, for all } \{i, j\} \in E. \tag{5f}$$

Using the procedure described in Section 2, we can create the commodities (K) as follows.

Commodity Selection Procedure for Improved Flow Formulation (5):

1. Find the requirement spanning tree.
2. Delete all edges with $r_{st} = 0$ from the requirement spanning tree. The resulting tree is connected because, by assumption, the NDC problem is unitary.
3. For each edge $\{s, t\}$ of the requirement spanning tree with $r_{st} \geq 2$, create two commodities: one with origin node s and destination node t, and the other with origin node t and destination node s; each of these commodities has a flow requirement of $r_{st}/2$.
4. Contract each edge $\{s, t\}$ with $r_{st} \geq 2$ in the requirement spanning tree, creating a contracted requirement spanning tree T with $r_{ij} = 1$ for all edges $\{i, j\}$. We distinguish nodes created by the contraction from the original nodes by calling them *components.* We denote a component by any one of the nodes it contains in the original requirement spanning tree (e.g., if we create a component by contracting nodes s and t, then we denote the component as s). Select a component i in T as the root node (if T does not contain any components, then select any node as the root node arbitrarily). Create a commodity for every node j in T other than the root node, with node i as its origin (in the original graph), and node/component j as its destination (in the original graph), with a requirement of 1.

Figure 3 illustrates this procedure. The following useful property is a consequence of the commodity selection procedure.

Proposition 3.5. *For any node set S,*

1. If $econ(S) \geq 2$, *then the improved flow formulation contains a commodity k whose flow requirement is $econ(S)/2$, origin is in $N \backslash S$, and destination is in S.*
2. If $econ(S) = 1$ *and root $\notin S$, then the improved flow model contains a commodity whose flow requirement is 1, origin is the root node, and destination is in S.*
3. If $econ(S) = 1$ *and root $\in S$, then no commodity in the improved flow model has its origin in $N \backslash S$, and destination in S.*

Proof. This result follows from the commodity selection procedure and the fact that $r_{st} = \max\{r_{ij} | j \in S;\ i \in N \backslash S\} \equiv \text{econ}(S)$ for any edge $\{s, t\}$ in the requirement spanning tree. ■

We now establish the validity of the improved undirected flow formulation for the unitary NDC problem by showing that this formulation and the directed cut formulation are equivalent. Let $P_{\text{iflo}}^{(5)}$ denote the polyhedron defined by the linear relaxation of constraints (5b)–(5f).

Lemma 3.6. *$Proj_{\mathbf{x}}(P_{dcut}^{(4)}) = Proj_{\mathbf{x}}(P_{iflo}^{(5)})$ whenever the root nodes selected in both formulations are in the same component of the contracted requirement spanning tree.*

Proof. First, consider any feasible solution ($\mathbf{x}^*$, $\mathbf{y}^*$) to the directed cut formulation. If we interpret y_{ij}^* as a capacity imposed upon the flow from node i to node j, the max-flow min-cut theorem implies that we can (1) send $r_{st}/2$ units of flow between any pair of nodes s and t in the requirement spanning tree with $r_{st} \geq 2$, and (2) send one unit of flow from the root component in the contracted requirement spanning tree to any other node/component in the contracted requirement spanning tree. Furthermore, the constraint $y_{ij}^* + y_{ji}^* \leq x_{ij}^*$ implies that we can fulfill conditions (1) and (2) while ensuring that for each edge $\{i, j\}$, the sum of the maximum flow sent (on the edge $\{i, j\}$) from node i to node j, and the maximum flow sent from node j to node i does not exceed x_{ij}. These arguments show that we can find flow variables $\mathbf{f}^*$ so that ($\mathbf{x}^*$, $\mathbf{f}^*$) is feasible in the improved undirected flow formulation. Thus, $\text{Proj}_{\mathbf{x}}(P_{\text{dcut}}^{(4)}) \subseteq \text{Proj}_{\mathbf{x}}(P_{\text{iflo}}^{(5)})$.

Suppose ($\bar{\mathbf{x}}$, $\bar{\mathbf{f}}$) is a feasible solution to the improved flow formulation. For each edge $\{i, j\}$, set $\bar{y}_{ij} = \max_{k\in K}\bar{f}_{ij}^k$ and $\bar{y}_{ji} = \max_{k\in K}\bar{f}_{ji}^k$. We claim the solution ($\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$) is feasible in the directed cut formulation. Whenever edge $\{s, t\}$ is in the requirement spanning tree and $r_{st} \geq 2$, the improved undirected flow formulation sends $r_{st}/2$ units of flow from node s to node t and $r_{st}/2$ units of flow from node t to node s. Consequently, if edge $\{s, t\}$ is in the requirement spanning tree and $r_{st} \geq 2$, the capacity of every $s - t$ dicut and every $t - s$ dicut is at least $r_{st}/2$ [for the solution ($\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$)]. For any node set S, the requirement spanning tree contains an edge $\{s, t\}$ in $\delta(S)$ with $\text{econ}(S) = r_{st}$. Therefore, whenever $\text{econ}(S) \geq 2$, the capacity of the dicut $\delta^-(S)$ is at least $\text{econ}(S)/2$. The improved undirected flow formulation sends 1 unit of flow from the root component to every node/component in the contracted requirement spanning tree. Therefore, the capacity of every dicut $\delta^-(S)$ with root $\notin S$ and $\text{econ}(S) = 1$ is at least 1. The constraint $\bar{f}_{ij}^k + \bar{f}_{ji}^h \leq \bar{x}_{ij}$ implies that for every edge, $\bar{y}_{ij} + \bar{y}_{ji} \leq \bar{x}_{ij}$. Consequently, ($\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$) is feasible for the directed cut model. Thus, $\text{Proj}_{\mathbf{x}}(P_{\text{iflo}}^{(5)}) \subseteq \text{Proj}_{\mathbf{x}}(P_{\text{dcut}}^{(4)})$. ∎

The preceding discussion showed that $\text{Proj}_{\mathbf{x}}(P_{\text{dcut}}^{(4)}) = \text{Proj}_{\mathbf{x}}(P_{\text{iflo}}^{(5)})$ when both the directed cut model (4) and the improved flow model (5) choose the root node from the same component of the contracted requirement spanning tree. Because the contracted requirement spanning tree of the SND problem has a single component, in this case $\text{Proj}_{\mathbf{x}}(P_{\text{dcut}}^{(4)})$ and $\text{Proj}_{\mathbf{x}}(P_{\text{iflo}}^{(5)})$ are independent of the choice of root node. (In the special case of the Steiner tree problem, Goemans and Myung [17] establish this result.) This result is also a consequence of the fact that the choice of root node within a component of the contracted requirement spanning tree does not affect constraint (4c). We conjecture that for the unitary NDC problem $\text{Proj}_{\mathbf{x}}(P_{\text{dcut}}^{(4)})$ and $\text{Proj}_{\mathbf{x}}(P_{\text{iflo}}^{(5)})$ are independent of the choice of root node as well.

Taken together, Lemmas 2.1, 3.4, and 3.6 imply the following relationships between the formulations (the first equality assumes the root node in both models are in the same component of the contracted requirement spanning tree)

$$\text{Proj}_{\mathbf{x}}(P_{\text{iflo}}^{(5)}) = \text{Proj}_{\mathbf{x}}(P_{\text{dcut}}^{(4)}) \subseteq P_{\text{cut}}^{(1)} = \text{Proj}_{\mathbf{x}}(P_{\text{flo}}^{(2)}).$$

Before concluding this section, we note the directed cut model (4) and the improved flow model (5) are stronger, as linear programs, than the cutset (1) and the undirected flow (2) model only if the requirement spanning tree contains an edge $\{s, t\}$ with $r_{st} = 1$. To see this result, observe that if no pair of nodes i and j has a connectivity requirement of 1, then for all node sets S, $\text{econ}(S) \neq 1$. But then, if $\bar{\mathbf{x}}$ is any feasible solution to the cutset formulation, the vector ($\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$), with $\bar{y}_{ij} = \bar{y}_{ji} = \bar{x}_{ij}/2$, is feasible in the directed cut formulation. As we have shown before, if ($\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$) is any feasible solution to the directed cut formulation, then $\bar{\mathbf{x}}$ is feasible in the cutset formulation. Therefore, in this case, the two models are equivalent.

Finally, we note that a simple modification of the formulations we have considered permits us to model situations that allow edge replication: we just replace the constraint $x_{ij} \leq 1$ by the constraint $x_{ij} \leq b_{ij}$ throughout our discussion.

4. PROJECTING FROM THE IMPROVED FLOW FORMULATION

To compare the improved flow formulation and the cutset formulation, we would like to project out the flow variables from the improved flow formulation so that the resulting models have the same set of variables. An elegant method for projection, proposed by Balas and Pulleyblank [3], and implicit in the work of Benders [4], is based upon a theorem of the alternatives.

Theorem 4.1 (Projection Theorem). *The projection of the set $P = \{(\mathbf{x}, \mathbf{f}) \in \mathcal{R}^{n+m} | \mathbf{Ax} + \mathbf{Bf} \geq \mathbf{d}\}$ onto the space of the x variables is*

$$Proj_{\mathbf{x}}(P) = \{\mathbf{x} \in \mathcal{R}^n | (\mathbf{g}^j)^T\mathbf{Ax} \geq (\mathbf{g}^j)^T\mathbf{d}, \quad for\ j = 1, 2, \ldots, J\},$$

which is defined by a finite set of generators $\{\mathbf{g}^j | j = 1, \ldots, J\}$ of the cone $C = \{\mathbf{g} | \mathbf{B}^T\mathbf{g} = \mathbf{0};\ \mathbf{g} \geq 0\}$.

The cone C in the statement of Theorem 4.1 is just the linear programming dual to the feasibility problem obtained by deleting the $\mathbf{x}$ variables and setting the right-hand side to zero in the inequality $Ax + Bf \geq d$. When the polyhedron P is defined by equality as well as inequality constraints, as in the improved undirected formulation, Theorem 4.1 assumes the following form.

Corollary 4.2. *The projection of the set* $P = \{(\mathbf{x}, \mathbf{f}) \in \mathfrak{R}^{n+m} | \mathbf{Ax} + \mathbf{Bf} \geq \mathbf{d}; \mathbf{A}'\mathbf{x} + \mathbf{B}'\mathbf{f} = \mathbf{d}'; \mathbf{x} \in \mathbf{X}; \mathbf{f} \geq \mathbf{0}\}$ *onto the space of the x variables is*

$$Proj_{\mathbf{x}}(P) = \{\mathbf{x} \in \mathbf{X} | (\mathbf{g}_u^j)^T\mathbf{Ax} + (\mathbf{g}_v^j)^T\mathbf{A}'\mathbf{x} \geq (\mathbf{g}_u^j)^T\mathbf{d} + (\mathbf{g}_v^j)^T\mathbf{d}', \quad for\ j = 1, 2, \ldots, J\},$$

which is defined by a finite set of generators $\{(\mathbf{g}_u^j, \mathbf{g}_v^j) | j = 1, \ldots, J\}$ *of the cone* $C = \{(\mathbf{u}, \mathbf{v}) | \mathbf{B}^T\mathbf{u} + \mathbf{B}'^T\mathbf{v} \leq \mathbf{0}; \mathbf{u} \geq 0;\ \mathbf{v}$ *unrestricted*$\}$.

If we can identify a set of finite generators of the cone C, then we obtain the projection of the set P. The Projection Theorem has the additional advantage that every member $(\mathbf{u}, \mathbf{v})$ of the cone C defines a valid inequality $(\mathbf{u}^TA + \mathbf{v}^TA')\mathbf{x} \geq \mathbf{u}^T\mathbf{d} + \mathbf{v}^T\mathbf{d}'$ for $\text{Proj}_{\mathbf{x}}(P)$. As a consequence, even if we cannot characterize the generators of the cone, we can still use the cone to obtain valid inequalities for $\text{Proj}_{\mathbf{x}}(P)$.

In Sections 4.1 and 4.2 we will use the Projection Theorem to show that the improved flow formulation (5) implies two classes of valid inequalities for the cutset formulation. To develop these results, we need to find generators for the following projection cone:

$$\left.\begin{array}{l} \sum_{h\in K} u_{ij}^{kh} + v_i^k - v_j^k \\ \sum_{h\in K} u_{ij}^{hk} + v_j^k - v_i^k \end{array}\right\} \geq 0 \quad \forall \{i, j\} \in E,\ \forall k \in K, \tag{6a}$$

$$u_{ij}^{kh} \geq 0 \quad \forall \{i, j\} \in E,\ \forall k, h \in K. \tag{6b}$$

In these inequalities, v_i^k is the dual variable corresponding to the flow balance equation at node i for commodity k, and u_{ij}^{kh} is the dual variable corresponding to the forcing constraint $f_{ij}^k + f_{ji}^h \leq x_{ij}$. Note that for any edge $\{i, j\}$ and any pair of commodities k and h, the model contains two forcing constraints $f_{ij}^k + f_{ji}^h \leq x_{ij}$ and $f_{ij}^h + f_{ji}^k \leq x_{ij}$. We identify the dual variable u_{ij}^{kh} with the constraint $f_{ij}^k + f_{ji}^h \leq x_{ij}$ and the dual variable u_{ij}^{hk} with the constraint $f_{ij}^h + f_{ji}^k \leq x_{ij}$. By convention, the dual variables obtained by reversing the indices, that is, u_{ij}^{kh} and u_{ji}^{hk}, are the same.

Because one flow balance equation for each commodity is redundant, we can set, for each commodity k, $v_{O(k)}^k$ to value zero. Using Corollary 4.2 for any member of this cone, we obtain a valid inequality of the form

$$\sum_{\{i,j\}\in E} \left(\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh} \right) x_{ij} \geq \sum_{k\in K} q_k v_{D(k)}^k. \tag{7}$$

In this expression, q_k is the number of units of commodity k sent from commodity k's origin to its destination. We refer to the coefficient of x_{ij} in this inequality as π_{ij} and the right-hand side coefficient as π_0.

Given some choice of the variables v_i^k for all the nodes i and commodities k, there are a number of choices for u_{ij}^{hk}. How do we determine the best such choice? Because the coefficient π_{ij} of x_{ij} in the inequality $\pi\mathbf{x} \geq \pi_0$ is $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh}$, we would like $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh}$ to be as small as possible for each edge. The following theorem, which we will use in our derivation of valid inequalities, describes the choice of u_{ij}^{hk} that minimizes $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh}$. We give a constructive proof that also shows how to determine the u_{ij}^{hk} values.

Theorem 4.3. *Suppose we are given values for v_i^k for all nodes i and all commodities k. For any edge $\{i, j\}$, let $t_{ij}^k = \max(0, v_j^k - v_i^k)$ and $t_{ji}^k = \max(0, v_i^k - v_j^k)$. Define $t_{ij} = \sum_{k\in K} t_{ij}^k$ and $t_{ji} = \sum_{k\in K} t_{ji}^k$. Then $\max(t_{ij}, t_{ji})$ is the minimum value of $\sum_{k\in K} \sum_{h\in k} u_{ij}^{kh}$ in inequality (7) over all feasible u_{ij}^{kh} values in the projection cone.*

Proof. We will establish this result for each edge $\{i, j\}$. We first show that $\max(t_{ij}, t_{ji})$ is a lower bound on the value of $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh}$ in any feasible solution to the projection cone (i.e., inequalities (6a)–(6b)). Let I be the set of all commodities with $t_{ij}^k > 0$ and J be the set of commodities with $t_{ji}^k > 0$. Equation (6a) states $\sum_{h\in K} u_{ij}^{kh} \geq v_j^k - v_i^k$. Summing over all commodities in the set I gives $\sum_{k\in I} \sum_{h\in K} u_{ij}^{kh} \geq \sum_{k\in I} (v_j^k - v_i^k) = t_{ij}$. Similarly, by considering the commodities in the set J, we obtain $\sum_{k\in J} \sum_{h\in K} u_{ij}^{hk} \geq \sum_{k\in J} (v_i^k - v_j^k) = t_{ji}$. But then the inequalities $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh} \geq \sum_{k\in I} \sum_{h\in K} u_{ij}^{kh}$ and $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh} \geq \sum_{k\in J} \sum_{h\in K} u_{ij}^{hk}$ imply that $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh} \geq \max(t_{ij}, t_{ji})$.

We next prescribe feasible values for the variables u_{ij}^{kh} that achieve the lower bound $\max(t_{ij}, t_{ji})$. Initially, each $u_{ij}^{kh} = 0$ and $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh} = 0$. Select a commodity l from I and a commodity m from J. Set $u_{ij}^{lm} = \min\{t_{ij}^l, t_{ji}^m\}$. If $t_{ij}^l \geq t_{ji}^m$, delete m from J, and if $t_{ij}^l \leq t_{ji}^m$, delete l from I. Set $t_{ij}^l = t_{ij}^l - u_{ij}^{lm}$ and $t_{ji}^m = t_{ji}^m - u_{ij}^{lm}$. Repeat this procedure until one of the two sets I and J, say J, is empty. Note that at this point $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh} = \min(t_{ij}, t_{ji})$ and the $\mathbf{u}$ and $\mathbf{v}$ variables satisfy inequalities (6a)–(6b) for every commodity we have deleted from I and J. For the remaining commodities $l \in I$, let m be any commodity in K and set $u_{ij}^{lm} = u_{ij}^{lm} + t_{ij}^l$. Thus, $\sum_{k\in K} \sum_{h\in K} u_{ij}^{kh} = \min(t_{ij}, t_{ji}) + (\max(t_{ij}, t_{ji}) - \min(t_{ij}, t_{ji})) = \max(t_{ij}, t_{ji})$. By construction, this choice of u_{ij}^{hk} satisfies the inequalities of the cone. ∎

With the aid of Theorem 4.3, we will now derive two classes of valid inequalities—partition inequalities, and combinatorial design inequalities—that generalize known classes of valid inequalities for the Steiner tree problem to the unitary NDC problem.

4.1. Partition Inequalities

Suppose we partition the node set N into disjoint node sets $N_0, N_1, \ldots, N_p$ satisfying the property that each node set has $\text{econ}(N_i) > 0$. Let, $I_1 = \{i : \text{econ}(N_i) = 1\}$ and I_2

$= \{i : i \in \text{econ}(N_i) \geq 2\}$. A *partition inequality* for the unitary NDC problem is an inequality of the form:

$$\frac{1}{2}\sum_{k=0}^{k=p}\sum_{\delta(N_k)} x_{ij} \geq \begin{cases} p & \text{if } I_2 = \phi, \\ \left\lceil \frac{1}{2}\sum_{i\in I_2} \text{econ}(N_i) \right\rceil + |I_1| & \text{otherwise.} \end{cases} \tag{8}$$

Chopra [8] and Magnanti and Wolsey [26] show that the partition inequalities describe the dominant of the spanning tree polytope. Chopra and Rao [10] and Grötschel and Monma [19] show that under appropriate conditions, partition inequalities are facet defining for the Steiner tree problem and for the NDLC problem, respectively. Stoer [32] shows that under appropriate conditions they are facet defining for the SND problem. Our derivation will show that these inequalities are valid for the more general unitary NDC problem.

Consider a partition $N_0, \ldots, N_p$. Without loss of generality assume that the root node is a node in N_0. Consider any set N_i of the partition. If $\text{econ}(N_i) \geq 2$, Proposition 3.5 implies the improved flow formulation must contain two commodities: one with origin $n_i \in N_i$ and destination some node $m_i \notin N_i$, and one with destination $n_i \in N_i$ and origin $m_i \notin N_i$, both with a flow requirement of $\text{econ}(N_i)/2$. Let i denote the commodity with destination node $n_i \in N_i$ (and origin $m_i \notin N_i$) and a requirement of $\text{econ}(N_i)/2$. Similarly, if $\text{econ}(N_i) = 1$ and $i \neq 0$, the improved flow formulation must contain a commodity i with destination a node $n_i \in N_i$ (the origin would be the root node) and a requirement of 1.

For commodities $k = 0, \ldots, p$, we set

$$v_i^k = \begin{cases} 1 & \text{if } i \in N_k, \\ 0 & \text{otherwise,} \end{cases}$$

and set $v_i^k = 0$ otherwise.

With this choice of values for the v_i^k variables, we ensure that for all edges $\{i, j\}$ across the partition $\sum_{k\in K} \max(0, v_j^k - v_i^k) = \sum_{k\in K} \max(0, v_i^k - v_j^k) = 1$ and for edges $\{i, j\}$ not in the partition $\sum_{k\in K} \max(0, v_j^k - v_i^k) = \sum_{k\in K} \max(0, v_i^k - v_j^k) = 0$. Thus, by choosing the values of u_{ij}^{kh} as indicated by Theorem 4.3, we find that $\pi_{ij} = \sum_{k\in K}\sum_{h\in K} u_{ij}^{kh}$ is 1 if i and j are in different sets of the partition and $\pi_{ij} = 0$ otherwise. Also, $\pi_0 = \sum_{k\in K} q_k v_{D(k)}^k = \frac{1}{2}\sum_{i\in I_2} \text{econ}(N_i) + |I_1|$, because $v_{D(k)}^k = 1$, $q_k = 1$ if $\text{econ}(N_k) = 1$, and $q_k = \text{econ}(N_k)/2$ if $\text{econ}(N_k) \geq 2$, for commodities $0, \ldots, p$; and $v_{D(k)}^k = 0$ otherwise.

If $I_2 = \phi$, by Proposition 3.5, the improved flow formulation does not contain any commodity with origin outside N_0 and destination in N_0. Consequently, we select p commodities with destinations in $N_1, \ldots, N_p$ and origin in N_0. With the same choice of v_i^k for $k = 1, \ldots, p$ (i.e., $v_i^k = 1$ if $i \in N_k$ and 0 otherwise), for all edges $\{i, j\}$ across the partition, $\max(\sum_{k\in K} \max(0, v_j^k - v_i^k), \sum_{k\in K} \max(0, v_i^k - v_j^k)) = 1$; and for edges $\{i, j\}$ not in the partition, $\sum_{k\in K} \max(0, v_j^k - v_i^k) = \sum_{k\in K} \max(0, v_i^k - v_j^k) = 0$. Thus, in accordance with Theorem 4.3, the edges across the partition have coefficient 1; all others have coefficient 0. The resulting right-hand side is $\pi_0 = \sum_k v_{D(k)}^k = p$.

We have thus obtained the following valid inequality:

$$\frac{1}{2}\sum_{k=0}^{k=p}\sum_{\delta(N_k)} x_{ij} \geq \begin{cases} p & \text{if } I_2 = \phi, \\ \frac{1}{2}\sum_{i\in I_2} \text{econ}(N_i) + |I_1| & \text{otherwise.} \end{cases} \tag{9}$$

But since, in the mixed integer program, the variables x_{ij} are either 0 or 1, we can round up the right-hand side in this inequality and still maintain feasibility to obtain (8).

If the number of sets in the partition inequality with odd connectivity requirement greater than one is odd, then the improved undirected formulation implies a weaker form of the partition inequality that we refer to as weak partition inequalities [that is inequality (9)]. Otherwise, the formulation implies the partition inequality. Note that the weak partition inequalities are stronger than the cutset formulation (as long as $I_1 \neq \phi$).

Because the flow model implies the weak partition inequalities, and does not always imply the partition inequalities, we might like to characterize, in a certain sense, how much stronger a model containing the partition inequalities would be compared to a model containing the weak partition inequalities.

To compare two classes of valid inequalities, we use the following notion previously introduced by Goemans [16]. Let $\mathcal{X}_1$ and $\mathcal{X}_2$ denote two finite classes of valid inequalities and $\boldsymbol{P}(\mathcal{X}_1)$ and $\boldsymbol{P}(\mathcal{X}_2)$ the feasible sets associated with them. Then, the *relative strength* of the class of the valid inequalities $\mathcal{X}_1$ to the class of the valid inequalities $\mathcal{X}_2$ is defined as

$$\max_{\mathbf{c}\in\mathcal{R}_+^{|E|}}\left\{\frac{\min\{\mathbf{cx} : \mathbf{x} \in \mathcal{R}_+^{|E|}; \mathbf{x} \in \boldsymbol{P}(\mathcal{X}_1); \mathbf{x} \in \boldsymbol{P}(\mathcal{X}_2)\}}{\min\{\mathbf{cx} : \mathbf{x} \in \mathcal{R}_+^{|E|}; \mathbf{x} \in \boldsymbol{P}(\mathcal{X}_2)\}}\right\},$$

with the convention $\frac{0}{0} = 1$. The relative strength measures how much, in the best case, the objective function of a linear program that contains the class of valid inequalities $\mathcal{X}_2$ improves by adding to it the class of valid inequalities $\mathcal{X}_1$.

The following result characterizes the relative strength of the partition inequalities with respect to the weak partition inequalities when (unlimited) edge replication is permitted.

Theorem 4.4. *When edge replication is permitted, the relative strength of the class of partition inequalities $\mathcal{X}_1$ with respect to the class of weak partition inequalities $\mathcal{X}_2$ is at most $\frac{10}{9}$.*

Proof. We will show that by multiplying any feasible solution (including any optimal solution) to the linear program $\min\{\mathbf{cx} : \mathbf{x} \in \mathcal{R}_+^{|E|}; \mathbf{x} \in \boldsymbol{P}(\mathcal{X}_2)\}$ (we refer to this linear program as LP2) by $\frac{10}{9}$ gives a feasible solution to the linear program $\min\{\mathbf{cx} : \mathbf{x} \in \mathcal{R}_+^{|E|}; \mathbf{x} \in \boldsymbol{P}(\mathcal{X}_1); \mathbf{x} \in \boldsymbol{P}(\mathcal{X}_2)\}$ (we refer to this linear program as LP1). This result implies that the optimal value to LP1 is at most $\frac{10}{9}$ times the optimal value to LP2. Note that the weak partition inequalities are implied by the partition inequalities. Consequently, we can delete $\mathbf{x} \in \boldsymbol{P}(\mathcal{X}_2)$ from LP1.

LP1 and LP2 differ when the right-hand side of the weak partition inequalities is fractional. If we show the maximum ratio between the right-hand side of any partition inequality and its corresponding weak partition inequality is $\frac{10}{9}$, we have shown that the relative strength of the partition inequalities with respect to the weak partition inequalities is $\frac{10}{9}$. (Because multiplying any solution that satisfies the weak partition inequalities by $\frac{10}{9}$ gives a solution that satisfies the partition inequalities.)

The right-hand side of the weak partition inequalities and the partition inequalities differ by at most 0.5. This occurs when the cardinality of the set $\{i : \text{econ}(N_i) \text{ odd; and } \text{econ}(N_i) \geq 3\}$ is odd [that is for an odd number of sets, $\text{econ}(N_i)$ is (1) odd, and (2) greater than or equal to 3]. Noting that the partition contains at least two sets with the highest value of $\text{econ}(N_i)$, we find that the maximum ratio is obtained by considering a partition with three sets, each with connectivity 3. The right-hand side for the weak partition inequality is 4.5, and the right-hand side for the partition inequality is 5. The ratio is $\frac{10}{9}$. ■

4.2. Combinatorial Design Inequalities

For the Steiner tree problem Goemans [15] introduced a new class of facet defining valid inequalities called *combinatorial design inequalities.* These inequalities generalize a class of inequalities called odd-hole inequalities [10]. He showed that under appropriate conditions combinatorial design inequalities are facet defining for the Steiner tree problem. He derives the combinatorial design inequalities by projecting from a node weighted (undirected) extended formulation for the Steiner tree problem. We show how to project out the combinatorial design inequalities from the improved undirected flow formulation (5), and as a result generalize the combinatorial design inequality to the unitary NDC problem, obtaining a new class of valid inequalities for this problem.

The description of the combinatorial design inequality is fairly involved. Let $T_p = \{a_0, \ldots, a_p\}$ be the set of nodes with nonzero connectivity requirements. $Z_q = N - T_p = \{b_0, \ldots, b_q\}$ is the set of nodes with zero connectivity requirements. Associate with each node a_i of T_p a subset Z_{a_i} containing elements of Z_q. Based on these subsets, we also define sets T_{b_i} associated with each node b_i in Z_q. T_{b_i} contains those elements of T_p whose associated subset contains the node b_i.

Define the $(q + 1) \times (p + 1)$ matrix $\mathbf{D} = [d_{ij}]$ with $d_{ij} = 1$ if $a_j \in T_{b_i}$ and $d_{ij} = 0$ otherwise. Impose the following two conditions on $\mathbf{D}$: (1) $\text{rank}(\mathbf{D}) = p + 1$, and (2) the unit vector e belongs to the cone generated by the columns of $\mathbf{D}$; i.e., $\mathbf{Dy} = \mathbf{e}$ for some vector $\mathbf{y} \in \mathcal{R}_+^{(p+1)}$. For any fixed $d > 0$, if we set $\beta = d\mathbf{y}$, we see that $\mathbf{D}\beta = d\mathbf{e}$. Letting β_j denote the jth component of β (i.e., $\beta_j = dy_j$), we see that

$$\sum_{a_j \in T_{b_i}} \beta_j = d \quad \text{for all } i = 0, 1, \ldots, q. \tag{10}$$

If we select d so that the greatest common divisor of β_0, $\beta_1, \ldots, \beta_p$, and d is 1, the coefficients of x_{ij} in the following combinatorial design inequalities will be integer and as small as possible. For every edge $\{s, t\}$, define

$$d_{st} = \begin{cases} \sum_{k: a_k \in (T_{b_i} \cap T)} \beta_k & \{s, t\} = \{b_i, b_j\} \text{ where } b_i, b_j \in Z_q; \\ \beta_j & \{s, t\} = \{a_j, b_i\} \text{ with } b_i \in Z_{a_j}; \\ 0 & \text{otherwise.} \end{cases}$$

For the Steiner tree problem, the inequality

$$\sum_{\{i,j\} \in E} (d - d_{ij}) x_{ij} \geq dp$$

is a combinatorial design inequality. Goemans [15] provides some graphical examples of combinatorial design inequalities for the Steiner tree problem.

We extend the definition of the combinatorial design inequality to obtain the following class of valid inequalities for the unitary NDC problem. Let $L_1 = \{i : i \in T_p, \text{econ}(i) = 1\}$ and $L_2 = \{i : i \in T_p, \text{econ}(i) \geq 2\}$.

$$\sum_{\{i,j\} \in E} (d - d_{ij}) x_{ij} \geq \begin{cases} dp & \text{if } L_2 = \phi, \\ \left(\left\lceil \frac{d}{2} \sum_{l \in L_2} \text{econ}(l) \right\rceil + d \left| L_1 \right| \right) & \text{otherwise.} \end{cases} \tag{11}$$

We now show the validity of the combinatorial design inequalities for the unitary NDC problem by projecting them from the improved undirected flow formulation. In the improved undirected flow formulation, let node a_0 be the root node. For each node a_l, $l = 0, \ldots, p$, we select a commodity l with destination a_l as follows. If $\text{econ}(a_l) \geq 2$, we select a commodity l with destination node a_l and flow requirement $\text{econ}(a_l)/2$. If $\text{econ}(a_l) = 1$, we select a commodity l with destination node a_l and flow requirement 1. For these commodities $(k = 0, \ldots, p)$, we set

$$v_i^k = \begin{cases} d & \text{if } i = a_k; \\ \beta_k & \text{if } i \in Z_{a_k}; \\ 0 & \text{otherwise.} \end{cases}$$

For all other commodities k, set v_i^k to zero.

Consider an edge $\{s, t\} = \{a_j, b_i\}$ with $b_i \in Z_{a_j}$. When a_j is the destination node, $v_{a_j}^j = d$ and $v_{b_i}^j = \beta_j$ if $i \in Z_{a_j}$.

Consider any node $a_k \in T_{b_i} \backslash \{a_j\}$. $v^l_{a_k} \neq 0$ only if a_k is a destination node for commodity l, that is, $l = k$ and, in this case, $v^k_{b_i} = \beta_k$ and $v^k_{a_j} = 0$. Note that, $v^k_{b_i} - v^k_{a_j} = \beta_k$. Thus,

$$\sum_{a_k \in \{Z_{b_i} \backslash \{a_j\}\}} v^k_{b_i} - v^k_{a_j} = \sum_{a_k \in \{Z_{b_i} \backslash \{a_j\}\}} \beta_k \overset{(10)}{=} d - \beta_j = v^j_{a_j} - v^j_{b_i}.$$

Here the notation $\overset{(10)}{=}$ means that the equality follows from expression (10). If $a_k \notin T_{b_i}$, then $v^k_{a_j} = v^k_{b_i} = 0$. Therefore, $\sum_{k \in K} \max(0, v^k_{a_j} - v^k_{b_i}) = \sum_{k \in K} \max(0, v^k_{b_i} - v^k_{a_j}) = d - \beta_j$. By Theorem 4.3, in the projected inequality, $d - \beta_j$ is the coefficient of an edge $\{s, t\} = \{a_j, b_i\}$ with $b_i \in Z_{a_j}$.

Consider an edge $\{s, t\} = \{b_i, b_j\}$. For any commodity k with destination a_k, $v^k_{b_i}$ and $v^k_{b_j}$ differ only if exactly one of b_i and b_j belongs to Z_{a_k}. If b_i belongs to Z_{a_k}, then $v^k_{b_i} = \beta_k$ and $v^k_{b_j} = 0$. Thus, $v^k_{b_i} - v^k_{b_j} = \beta_k$. Summing over all sets Z_{a_k} that contain node b_i but not node b_j, we find that

$$\sum_{k: b_i \in Z_{a_k}; b_j \notin Z_{a_k}} v^k_{b_i} - v^k_{b_j} = \sum_{k: b_i \in Z_{a_k}; b_j \notin Z_{a_k}} \beta_k$$

$$\overset{(10)}{=} d - \sum_{k: b_i \in Z_{a_k}; b_j \in Z_{a_k}} \beta_k.$$

Similarly, summing up over all sets Z_{a_k} that contain node b_j but not node b_i, we find that

$$\sum_{k: b_j \in Z_{a_k}; b_i \notin Z_{a_k}} v^k_{b_j} - v^k_{b_i} = \sum_{k: b_j \in Z_{a_k}; b_i \notin Z_{a_k}} \beta_k$$

$$\overset{(10)}{=} d - \sum_{k: b_j \in Z_{a_k}; b_i \in Z_{a_k}} \beta_k.$$

Therefore, $\sum_{k \in K} \max(0, v^k_{b_j} - v^k_{b_i}) = \sum_{k \in K} \max(0, v^k_{b_i} - v^k_{b_j}) = d - \sum_{k: b_j \in Z_{a_k}; b_i \in Z_{a_k}} \beta_k$. By Theorem 4.3, $d - \sum_{k: b_j \in Z_{a_k}; b_i \in Z_{a_k}} \beta_k$ is the coefficient of an edge $\{s, t\} = \{b_i, b_j\}$ in the projected inequality.

All the other edges are either of the form $\{s, t\} = \{a_i, a_j\}$ or $\{s, t\} = \{a_j, b_i\}$ with $b_i \notin Z_{a_j}$. For any edge of the form $\{a_i, a_j\}$, $v^j_{a_j} - v^j_{a_i} = v^i_{a_i} - v^i_{a_j} = d$ for commodities i and j and $v^k_{a_i} = v^k_{a_j} = 0$ otherwise. Consider an edge of the form $\{s, t\} = \{a_j, b_i\}$ with $b_i \notin Z_{a_j}$. For all commodities k with $a_k \in T_{b_i}$, that is, a_k is the destination, $v^k_{b_i} - v^k_{a_i} = \beta_k$. For commodity j, a_j is the destination and $v^j_{a_j} - v^j_{b_i} = d$. For all other commodities k, $v^k_{a_j} = v^k_{b_i} = 0$. Therefore, for both edges of the form $\{s, t\} = \{a_i, a_j\}$ and $\{s, t\} = \{a_j, b_i\}$ with $b_i \notin Z_{a_j}$, we find that $\sum_{k \in K} \max(0, v^k_s - v^k_t) = \sum_{k \in K} \max(0, v^k_s - v^k_t) = d$. By Theorem 4.3, d is the coefficient of these edges in the projected inequality.

The right-hand side of the projected inequality is $\pi_0 = \sum_k q_k v^k_{D(k)} = d(\frac{1}{2} \sum_{l \in L_2} \text{econ}(l) + |L_1|)$ because $v^k_{D(k)} = d$ for commodities $0, \ldots, p$, and $v^k_{D(k)} = 0$ otherwise. If the problem has exactly one node or no nodes with econ$(i) \geq 2$, with the same choice of v^k_i variables, we obtain the same coefficients for x_{ij} and a right-hand side of dp (there will be no commodity with destination the root node a_0).

Thus, the projected inequality is

$$\sum_{\{i,j\} \in E} (d - d_{ij}) x_{ij} \geq \begin{cases} dp & \text{if } L_2 = \phi, \\ d\left(\frac{1}{2} \sum_{l \in L_2} \text{econ}(l) + |L_1|\right) & \text{otherwise.} \end{cases} \tag{12}$$

Once again, noting that the left-hand side should be integer if the x variables are integer, we can round up the right-hand side, giving inequality (11).

We refer to inequalities (12) as weak combinatorial design inequalities. Noting that $d \geq 1$, it is easy to prove a result similar to Theorem 4.4—namely, if edge replication is permitted, the relative strength of the class of combinatorial design inequalities with respect to the class of weak combinatorial design inequalities is at most $\frac{10}{9}$.

To close this section, we recall that odd-hole inequalities are special cases of combinatorial design inequalities (see [15]), so these results also show that by projecting out the flow variables from the formulation we obtain the odd-hole inequalities.

5. NONUNITARY PROBLEMS

So far we have restricted our attention to unitary problems. We now examine nonunitary NDC problems. Our starting point will be the special case of the Steiner forest problem. Recall that in the Steiner forest problem we are given a graph $G = (N, E)$ and node sets $T_1, T_2, \ldots, T_P$ with $T_i \cap T_j = \phi$ for all node set pairs $(i \neq j)$. We wish to design a graph at minimum cost that connects the nodes in each node set (possibly by including multiple node sets in any connected component of the graph).

For the unitary NDC problem, we derived a stronger formulation by generalizing a well-known directing procedure for the Steiner tree problem. The essential idea used was to direct the bridge edges of a solution to the unitary NDC problem in a manner akin to the directing procedure for the Steiner tree. Analogously, to strengthen the formulation for the nonunitary NDC problem we will first determine how to strengthen the Steiner forest problem—a nonunitary NDC problem with each $r_{st} \in \{0, 1\}$—by directing it.

We are unaware of any models stronger than the cutset model for the Steiner forest problem, and believe the model we present is the first directed model in the literature for this problem.

5.1. Directing the Steiner Forest Problem

For convenience, we once again describe the cutset and undirected flow formulations for the NDC problem as applied to the Steiner forest problem.

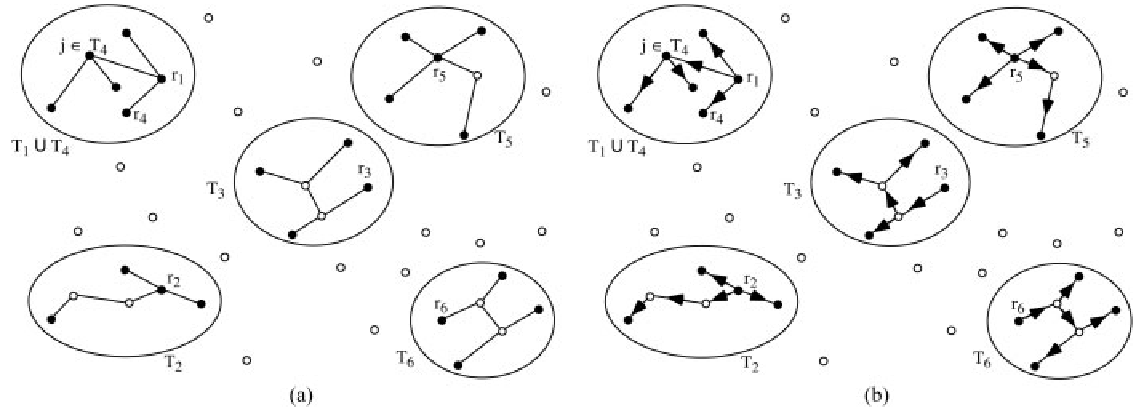


FIG. 4. Example of the directing procedure. Filled nodes are required nodes, unfilled nodes are Steiner nodes. (a) Undirected forest. (b) Direct each forest away from the lowest indexed root node that it contains.

Cutset Formulation for the Steiner Forest Problem:

$$\text{Minimize} \quad \sum_{\{i,j\}\in E} c_{ij}x_{ij} \tag{13a}$$

subject to:

$$\sum_{\{i,j\}\in\delta(S)} x_{ij} \geq 1 \quad \begin{array}{l}\text{for all } S,\ \phi \subset S \subset N,\\ S \cap T_i \neq \phi \text{ and}\\ (N\backslash S) \cap T_i \neq \phi \text{ for some } i,\end{array} \tag{13b}$$

$$x_{ij} \leq 1 \quad \text{for all } \{i, j\} \in E, \tag{13c}$$

$$x_{ij} \geq 0 \quad \text{and integer.} \tag{13d}$$

In the undirected flow formulation (2) for the Steiner forest problem, we select a root node for each node set and send one unit of flow from the root node of each node set to every node in that node set. Thus, $q_k = 1$ for all $k \in K$, and $b_{ij} = 1$ for all $\{i, j\} \in E$, in the undirected flow formulation.

If we assume each $c_{ij} \geq 0$, these formulations always have a Steiner forest as an optimal solution, and so each component of the forest is a tree. Nodes belonging to any node set T_i, for any i, lie in the same component. As an example, Figure 4a shows the optimal solution to a Steiner forest problem with five components. One component contains the two node sets T_1 and T_4. All the other components contain nodes from exactly one node set.

How might we direct the Steiner forest problem? Because each component in the optimal solution is a tree, we could arbitrarily choose a node in each component and direct each tree away from it. Unfortunately, before we solve the problem, although we know that nodes in each node set will lie in the same component, we do not know the number of components in the optimal solution and the node sets they contain. The problem is to determine, *a priori,* the root node for each component. For this reason, directing the Steiner forest problem raises difficulties not encountered in directing the Steiner tree (and the unitary NDC) problem.

To direct the Steiner forest, for each set T_i, we choose an arbitrary root node $r_i \in T_i$. We then direct each component (tree) away from the lowest indexed root node that it contains. In the example shown in Figure 4a, one component contains two node sets T_1 and T_4. Because T_1 is the lowest indexed node set in this component, we have directed the component away from the root node r_1 of node set T_1. All the other components contain nodes from only one node set T_i and we direct each of them away from the root node r_i of node set T_i. Figure 4b shows the forest after we have applied the directing procedure.

For notation, if $j \in T_i$, we let $\rho(j) = r_i$ denote the root node of the node set T_i that contains node j. We refer to r_i as node j's root node. We also define $T = T_1 \cup T_2 \cup \ldots T_P$, and let R be the set of all root nodes, that is, $R = \{r_1, r_2, \ldots, r_P\}$.

5.2. *Improved Flow Formulation for the Steiner Forest Problem*

We model the Steiner forest problem using multicommodity flows. Because the network we obtain after directing the Steiner forest contains a directed path from the lowest indexed root node in a component to all other nodes in that component, we can send a unit of flow from the root node of each directed component to every node in that component.

For each node $j \in T_i$, with $j \neq r_i$, and for each $p \leq i$, we define a commodity with origin node r_p and destination node j, and for each root node r_i and for each $p < i$, we define a commodity with origin node r_p and destination node r_i. In the optimal solution, it is possible to send a unit of flow from the lowest indexed root node of a component to each required node in that component. Let $CO(q)$ denote the set of all commodities that have node q as their origin and $CD(q)$ the set of all commodities that have node q as their destination. Let K denote the set of all commodities. We also define $\mathcal{H} = \{S : S \subset K$, and $|S \cap CO(r_j)| = 1$ for all $j = 1, \ldots, P\}$. Each member of $\mathcal{H}$ is a set of P

commodities satisfying the property that it contains a commodity originating at each one of the P root nodes.

Let x_{ij} be 1 if the network design contains edge $\{i, j\}$ and be 0 otherwise. The improved undirected flow formulation for the Steiner forest problem has the following form.

Improved Undirected Flow Formulation for Steiner Forest Problem:

$$\text{Minimize} \quad \sum_{(i,j)\in E} c_{ij} x_{ij} \tag{14a}$$

$$\text{subject to:} \quad \sum_{j\in N} f^k_{ji} - \sum_{l\in N} f^k_{il} \begin{cases} \leq -1 & \text{if } i = O(k), \\ \leq 1 & \text{if } i = D(k), \\ = 0 & \text{otherwise,} \end{cases} \left. \begin{matrix} \forall\, i \in N \\ \text{and} \\ k \in K, \end{matrix} \right\} \tag{14b}$$

$$\sum_{k\in CD(i)} \sum_{j\in N} f^k_{jD(k)} = 1 \quad \text{for all } i \in T\backslash R, \tag{14c}$$

$$\sum_{j\in N} f^k_{jD(k)} \leq \sum_{j\in N} f^{k*}_{jD(k^*)} \quad \begin{matrix} \forall\, i \in T\backslash R,\ \forall\, k \in CD(i), \\ \text{s.t. } O(k) = O(k^*), \text{ and} \\ D(k^*) = \rho(i), \end{matrix} \tag{14d}$$

$$\sum_{k\in H} f^k_{ij} + \sum_{k\in \bar{H}} f^k_{ji} \leq x_{ij} \quad \begin{matrix} \text{for all } \{i, j\} \in E, \text{ and} \\ \text{all } H, \bar{H} \text{ pairs in } \mathscr{H}, \end{matrix} \tag{14e}$$

$$\sum_{i\in N} \sum_{k\in H} f^k_{ij} \leq 1 \quad \begin{matrix} \text{for all } j \in N, \text{ and} \\ \text{all } H \text{ in } \mathscr{H}, \end{matrix} \tag{14f}$$

$$f^k_{D(k)l} = 0 \quad \begin{matrix} \text{for all } l \in N, \text{ and} \\ k \in K, \end{matrix} \tag{14g}$$

$$f^k_{ij} = 0 \quad \begin{matrix} \forall\, \{i, j\} \in E \text{ and any } k \\ \text{s.t. } O(k) \in T_v \text{ and} \\ i \text{ or } j \in T_u \text{ where } u < v, \end{matrix} \tag{14h}$$

$$\left. \begin{matrix} f^k_{ij} \\ f^k_{ji} \end{matrix} \right\} \geq 0 \quad \begin{matrix} \text{for all } \{i, j\} \in E, \text{ and} \\ k \in K, \end{matrix} \tag{14i}$$

$$x_{ij} \in \{0, 1\} \quad \text{for all } \{i, j\} \in E. \tag{14j}$$

Constraints (14b), (14c), and (14g) ensure that each node i in $T\backslash R$ obtains a unit of flow from either its root node, or the root node of a lower indexed node set. Constraints (14d) and (14g) ensure that if node $i \in T_j$, $i \neq r_j$, is supplied by a commodity k whose origin is not the root node of set T_j, then its root node also is supplied from the origin of commodity k (i.e., its root node belongs to the same component that it belongs to). Note that constraint (14g) simply states that flow of a commodity out of its destination node is zero, and so allows us to simplify notation in constraints (14c) and (14d) (they contain terms for flow only into the destination node). Constraint (14e) follows from the property that in an optimal solution, flow travels in only one direction across an edge, and all the flow across an edge originates from the same source (the root node of the component the edge belongs to). Constraint (14f) follows from the fact that flow into any node in a component originates from a single node (the root node of that component). Constraint (14h) eliminates a large number of flow variables from the formulation. It says that for any node there should not be any flow into or out of that node from the root of a higher indexed node set.

We now show that the flow formulation obtained by this directing procedure is stronger than the cutset formulation. Let $P^{(13)}_{\text{cut}}$ denote the polyhedron defined by the linear relaxation of constraints (13b)–(13d). Let $P^{(14)}_{\text{ifsf}}$ denote the polyhedron defined by the linear relaxation of constraints (14b)–(14j).

Lemma 5.1. *$Proj_{\mathbf{x}}(P^{(14)}_{ifsf}) \subseteq P^{(13)}_{cut}$.*

Proof. Consider any set S such that $S \cap T_m \neq \phi$ and $(N\backslash S) \cap T_m \neq \phi$ for some $m \in \{1, 2, \ldots, P\}$. Without loss of generality assume that $r_m \in N\backslash S$ and let l be a node in $S \cap T_m$. Consider any feasible solution $(\bar{\mathbf{x}}, \bar{\mathbf{f}})$ to $P^{(14)}_{\text{ifsf}}$. Select H and $\bar{H} \in \mathscr{H}$, such that $H \supseteq CD(l)$ and $\bar{H} \supseteq CD(r_m)$ [i.e., $CD(\rho(l))$]. Summing constraint (14e) over all edges $\{i, j\} \in \delta(S)$, we obtain

$$\sum_{\{i,j\}\in\delta(S)} \bar{x}_{ij} \geq \sum_{k\in H} \sum_{\{i,j\}\in\delta(S)} \bar{f}^k_{ij} + \sum_{k\in \bar{H}} \sum_{\{i,j\}\in\delta(S)} \bar{f}^k_{ji}$$
$$\geq \sum_{k\in CD(l)} \sum_{\{i,j\}\in\delta(S)} \bar{f}^k_{ij} + \sum_{k\in CD(r_m)} \sum_{\{i,j\}\in\delta(S)} \bar{f}^k_{ji}.$$

By flow balance,

$$\Sigma_{k\in CD(l)} \Sigma_{\{i,j\}\in\delta(S)} \bar{f}^k_{ij} \geq \Sigma_{k\in CD(l),\ \ (k)\in N\backslash S} \Sigma_{j\in N} \bar{f}^k_{jl}.$$

Constraint (14d) states $\Sigma_{j\in N} \bar{f}^k_{jl} \leq \Sigma_{j\in N} f^{k*}_{jr_m}$, $\forall k \in CD(l)$, $k^* \in CD(r_m)$, such that $O(k) = O(k^*)$. Thus, $\Sigma_{k\in CD(r_m)} \Sigma_{\{i,j\}\in\delta(S)} \bar{f}^k_{ji} \geq \Sigma_{k\in CD(r_m), O(k)\in S} \Sigma_{j\in N} \bar{f}^k_{jr_m} \geq \Sigma_{k\in CD(l), O(k)\in S} \Sigma_{j\in N} \bar{f}^k_{jl}$. In other words

$$\sum_{k\in CD(l)} \sum_{\{i,j\}\in\delta(S)} \bar{f}^k_{ij} + \sum_{k\in CD(r_m)} \sum_{\{i,j\}\in\delta(S)} \bar{f}^k_{ji}$$
$$\geq \sum_{k\in CD(l), O(k)\in N\backslash S} \sum_{j\in N} \bar{f}^k_{jl} + \sum_{k\in CD(l), O(k)\in S} \sum_{j\in N} \bar{f}^k_{jl} \overset{(14c)}{=} 1.$$

■

5.3 *Computational Results*

Table 2 describes some computational experiments comparing the improved flow formulation with the undirected flow formulation (or cutset formulation, because they are equivalent) for the Steiner forest problem. In the experiments reported in these tables, we generated problems with 10, 15, and 20 nodes. We then varied the number of node sets P (i.e., $T_1, \ldots, T_P$) and the number of edges in the graph. In the test problems $T_1 \cup T_2 \cup T_3 \cup \ldots \cup T_P$

TABLE 2. Computational experiments comparing linear programming relaxations of improved undirected flow formulation and cut-set/undirected flow formulation.

(a) Random edge costs

			Improved flow		Cutset
$\|N\|$	P	$\|E\|$	LP/IP ratio	Time (seconds)	LP/IP ratio
10	3	20	1.000	0.019	0.787
		30	1.000	0.025	0.770
		45	1.000	0.044	0.724
	4	20	1.000	0.025	0.843
		30	0.995	0.059	0.800
		45	1.000	0.091	0.746
15	4	30	1.000	0.672	0.767
		50	1.000	1.325	0.736
		105	1.000	6.669	0.693
	6	30	1.000	18.122	0.797
		50	1.000	36.791	0.752
		105	0.995	799.640	0.684
20	3	50	1.000	1.009	0.692
		100	1.000	14.181	0.683
		190	1.000	20.750	0.640
	5	50	1.000	627.942	0.733
		100	1.000	473.367	0.709

(b) Euclidean edge costs

			Improved flow		Cutset
$\|N\|$	P	$\|E\|$	LP/IP ratio	Time (seconds)	LP/IP ratio
10	3	20	1.000	0.025	0.818
		30	1.000	0.028	0.742
		45	1.000	0.056	0.702
	4	20	1.000	0.034	0.861
		30	1.000	0.053	0.815
		45	1.000	0.119	0.751
15	4	30	1.000	0.787	0.724
		50	1.000	1.506	0.719
		105	1.000	43.625	0.658
	6	30	1.000	20.725	0.787
		50	1.000	46.369	0.748
		105	1.000	1208.665	0.669
20	3	50	1.000	1.203	0.694
		100	1.000	24.294	0.691
		190	1.000	26.256	0.621
	5	50	1.000	311.191	0.722
		100	1.000	1880.696	0.703

$= N$. Further, we ensure that each node set T_j contains at least two nodes, and the underlying graph contains a tree on each node set. (We do so by randomly ordering the nodes in a node set and creating an edge between nodes that are adjacent to each other in this ordering.) For each combination of these parameters (number of nodes, number of edges, and number of node sets), we created five instances with random edge costs and five instances with Euclidean edge costs. For instances, with random edge costs, we randomly generated the edge lengths as integers between 5 and 50 and for the Euclidean test problems, we generated the nodes on a 35×35 square grid with the edge lengths the Euclidean distances rounded down to the nearest integers. We ran all the computations using CPLEX 8.1 on a WINDOWS XP machine with an Intel Xeon processor, at 2.66 GHz, and 2-GB of RAM.

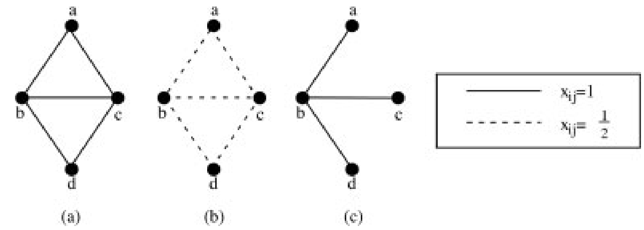


FIG. 5. (a) Graph with unit edge costs and two node sets $\{a, d\}$ and $\{b, c\}$. (b) Optimal solution to LP relaxation of improved undirected flow formulation when $T_1 = \{a, d\}$, and $T_2 = \{b, c\}$. (c) Optimal solution to LP relaxation of improved undirected flow formulation when $T_1 = \{b, c\}$, and $T_2 = \{a, d\}$.

Our results indicate that the improved undirected flow formulation is exceptionally strong. In 83 out of 85 random instances, the optimal solution to the linear programming relaxation is integral; while in all 85 Euclidean instances the optimal solution to the linear programming relaxation is integral. However, the size of this formulation grows drastically. The total number of commodities in the formulation is $|K| = (\sum_{i=1}^{P} i|T_i|) - P$ and the number of sets in $\mathcal{H}$ is $\prod_{j=1}^{P} ((\sum_{i=j}^{P} |T_i|) - 1)$. The number of constraints of type (14e) is $|E|\,|\mathcal{H}|^2$, which is extremely large although still polynomial for fixed P. Consequently, to use this formulation to solve large scale problems, we might need to judiciously add a subset of the constraints of type (14e) (adding others as necessary). This is a possible topic for future research.

A natural question to ask for the Steiner forest problem is whether (1) the choice of root node for each node set, and (2) the order of node sets, affects the optimal objective value of the linear programming relaxation of the improved flow formulation for the Steiner forest problems. The answer to the first question is open, and we conjecture that the choice of the root node within each node set does not affect the optimal objective value of the linear programming relaxation. However, as we show next, the order of node sets *does affect* the value of the optimal solution to the linear programming relaxation of the improved flow formulation for the Steiner forest problem.

As an example, consider the graph in Figure 5 with two node sets $\{a, d\}$ and $\{b, c\}$. Assume each edge in the graph has unit cost. In the improved undirected flow formulation, suppose we select $T_1 = \{a, d\}$ with node a as the root of node set T_1, and select $T_2 = \{b, c\}$ with node b as the root of node set T_2. This formulation contains four commodities. Commodities 1, 2, and 3 have origin node a and destinations nodes b, c, and d. Commodity 4 has origin node b and destination node c. The optimal solution to the linear programming relaxation of the improved undirected flow formulation sets $x_{ab} = x_{bd} = x_{dc} = x_{ca} = x_{bc} = 0.5, f^3_{ab} = f^3_{bd} = f^3_{ac} = f^3_{cd} = f^1_{ab} = f^2_{ac} = f^4_{bc} = 0.5$, with a cost of 2.5. If instead $T_1 = \{b, c\}$ and $T_2 = \{a, d\}$ with node b as the root of node set T_1 and node a as the root of node set T_2 then (1) commodities 1, 2, and 3 have origin node b and destinations nodes c, a, and d; and (2) commodity 4 has origin node a and destination node d. The optimal solution to the linear programming relaxation of the improved undirected flow formulation sets $x_{ab} = x_{bd} = x_{bc} = 1, f^1_{bc} = f^2_{ba} = f^3_{bd} = f^3_{cd} = 1$, with a cost of three units.

In this section we developed a directing procedure on an undirected graph by using directed commodity flows. We could instead implement the directing procedure by transforming the problem onto a directed graph. Raghavan [30] describes equivalent directed flow formulations for the Steiner forest problem. To conclude this section, we note that we modeled the directing procedure using commodity flows. Unlike the unitary NDC problem, there does not seem to be a straightforward way to formulate a directed cut model. This disparity demonstrates the flexibility and power of using flow models for modeling network design problems with connectivity constraints.

6. DIRECTING THE NDC PROBLEM

We now show how to generalize the directing procedure we have just presented for the Steiner forest problem to obtain a directed model for all NDC problems. As a result, we obtain a stronger formulation for the NDC model with edge-connectivity requirements.

To sketch the basic idea underlying the directing procedure, consider any solution to the NDC problem. It consists of one or more connected components. By following the procedure described in Section 3, we can direct each connected component of the integer solution to the NDC problem. However, like the Steiner forest problem, because the problem is nonunitary, we do not know *a priori* the number of connected components in the optimal solution and the required nodes they contain. By combining the directing procedure for the unitary NDC problem and the directing procedure for the Steiner forest problem, we obtain a directed model for the NDC problem.

The following commodity selection procedure outlines the essential idea of the directing procedure. We first use the directing procedure described in Section 3.2 to direct the problem for commodities with $r_{st} \geq 2$. We then apply the Steiner forest problem's directing procedure to direct the bridge edges in each component of the optimal integer solution to the NDC problem.

Commodity Selection Procedure for Improved Flow Formulation (15):

1. Find the requirement spanning tree.
2. Delete all edges with $r_{st} = 0$ from the requirement spanning tree. Delete all singleton nodes in the resulting forest.
3. For each edge $\{s, t\}$ of the requirement spanning tree with $r_{st} \geq 2$, create two commodities: one with origin node s and destination node t, and the other with origin node t and destination node s; each of these commodities has a flow requirement of $r_{st}/2$. Let L_2 denote this set of commodities.
4. Contract each edge $\{s, t\}$ with $r_{st} \geq 2$ in the requirement spanning tree, creating a forest F in which $r_{ij} = 1$ for all edges $\{i, j\}$. Identify the connected components $T_1, T_2, \ldots, T_P$ of this forest. Denote any node in F created by contraction by any of the nodes it contains in the original requirement spanning tree (e.g., if contracting nodes s and t creates a node in F, then we denote the contracted node by s). Select a contracted node in each set T_i as the root node r_i of the node set T_i. (If node set T_i does not contain a contracted node, then arbitrarily select any one of the nodes as the root node). Create commodities as described for the Steiner forest problem with node sets $T_1, T_2, \ldots, T_P$, and root nodes $r_1, r_2, \ldots, r_P$. Let L_1 denote this set of commodities.

Using this set of commodities $K = L_1 \cup L_2$, we obtain the following improved undirected flow formulation.

Improved Undirected Flow Formulation for NDC Problem:

$$\text{Minimize} \quad \sum_{(i,j)\in E} c_{ij}x_{ij} \tag{15a}$$

$$\text{subject to:} \quad \sum_{j\in N} f^k_{ji} - \sum_{l\in N} f^k_{il} \begin{cases} \geq -1 & \text{if } i = O(k), \\ \leq 1 & \text{if } i = D(k), \\ =0 & \text{otherwise,} \end{cases} \left. \begin{matrix} \forall\, i \in N \\ \text{and} \\ k \in L_1, \end{matrix} \right. \tag{15b}$$

$$\sum_{j\in N} f^k_{ji} - \sum_{l\in N} f^k_{il} = \begin{cases} -q_k & \text{if } i = O(k), \\ q_k & \text{if } i = D(k), \\ 0 & \text{otherwise,} \end{cases} \left. \begin{matrix} \forall\, i \in N \\ \text{and} \\ k \in L_2, \end{matrix} \right. \tag{15c}$$

$$\sum_{k\in CD(i)} \sum_{j\in N} f^k_{jD(k)} = 1 \quad \text{for all } i \in T\backslash R, \tag{15d}$$

$$\sum_{j\in N} f^k_{jD(k)} \leq \sum_{j\in N} f^{k^*}_{jD(k^*)} \quad \begin{matrix} \forall\, i \in T\backslash R,\ \forall\, k \in CD(i), \\ \text{s.t. } O(k) = O(k^*), \text{ and} \\ D(k^*) = \rho(i), \end{matrix} \tag{15e}$$

$$\sum_{k\in H} f^k_{ij} + \sum_{k\in \bar{H}} f^k_{ji} \leq x_{ij} \quad \begin{matrix} \text{for all } \{i, j\} \in E, \text{ and} \\ \text{all } H, \bar{H} \text{ pairs in } \mathcal{H}, \end{matrix} \tag{15f}$$

$$\sum_{i\in N} \sum_{k\in H} f^k_{ij} \leq 1 \quad \begin{matrix} \text{for all } j \in N, \text{ and} \\ \text{all } H \text{ in } \mathcal{H}, \end{matrix} \tag{15g}$$

$$f^k_{ij} + f^h_{ji} \leq x_{ij} \quad \text{for all } k, h \in K, \tag{15h}$$

$$f^k_{D(k)l} = 0 \quad \begin{matrix} \text{for all } l \in N \text{ and } k \in L_1, \\ \forall\, \{i, j\} \in E \text{ and any } k \in L_1 \end{matrix} \tag{15i}$$

$$f^k_{ij} = 0 \quad \begin{matrix} \text{s.t. } O(k) \in T_v \text{ and} \\ i \text{ or } j \in T_u \text{ where } u < v, \end{matrix} \tag{15j}$$

$$x_{ij} \leq 1 \quad \text{for all } \{i, j\} \in E, \tag{15k}$$

$$\left.\begin{matrix} f_{ij}^k \\ f_{ji}^k \end{matrix}\right\} \geq 0 \quad \begin{matrix} \text{for all } \{i, j\} \in E, \text{ and} \\ k \in K, \end{matrix} \tag{15l}$$

$$x_{ij} \geq 0 \quad \text{and integer, for all } \{i, j\} \in E. \tag{15m}$$

In this formulation, $T = T_1 \cup \ldots \cup T_P$, $R = \{r_1, \ldots, r_P\}$, and $\rho(j)$ denotes node j's root node. $CO(k)$, $CD(k)$, and $\mathcal{H}$ are defined as for the Steiner forest problem for the commodities $k \in L_1$.

Let $P_{\text{ifndc}}^{(15)}$ denote the polyhedron defined by the linear relaxation of constraints (15b)–(15m). Using an identical argument as in the proof of Lemma 5.1 establishes the following result.

Lemma 6.1. $Proj_{\mathbf{x}}(P_{ifndc}^{(15)}) \subseteq P_{cut}^{(13)}$.

If the requirement spanning tree contains no edge $\{s, t\}$ with $r_{st} = 1$, the improved formulation for the NDC problem (15) contains no commodities in L_1, and so the model contains only constraints (15c), (15h), (15k), (15l), and (15m). In this case, arguing as in Section 3.2 shows that this formulation is equivalent to the undirected flow formulation. Consequently, the improved formulation for the NDC problem is stronger than the undirected flow formulation only when the maximum spanning tree of the requirement graph contains some edge $\{s, t\}$ with $r_{st} = 1$.

7. CONCLUDING REMARKS

We have shown how to improve formulations for network design problems with connectivity requirements (NDC problems) by generalizing directing procedures for versions of these problems with connectivity requirements r_{st} of 0 or 1. For unitary NDC problems, we developed strong models by generalizing the directing procedure for the Steiner tree problem. We also developed a new directing procedure for the Steiner forest problem, and generalized it to create strong models for nonunitary NDC problems. Unlike the known use of the Nash-Williams procedure for orienting undirected graphs into directed paths, our method uses an orientation based upon fractional paths, permitting us to strengthen the underlying optimization models for problems with arbitrary connectivity requirements (and so not limited to connectivity requirements that are either one or even).

For unitary NDC problems, we also showed that the projection of the new formulations onto the space of the edge design variables contains two classes of valid inequalities (partition, and combinatorial design) that are generalizations of valid inequalities for the Steiner tree problem. For nonunitary NDC problems, we have not fully investigated the projection of the improved models (the projection cones are quite complex) to determine valid inequalities implied by them in the space of the original edge variables. This investigation is a potential direction of future research.

We have not considered node-connectivity requirements. Raghavan [30] describes formulations and algorithms for NDC problems with node-connectivity requirements, as well as NDC problems with both edge- and node-connectivity requirements. However, because node connectivity implies edge connectivity, all the valid inequalities derived for the edge-connectivity version of the NDC problem are valid for the node-connectivity version. Consequently, the partition, and combinatorial design inequalities are valid for the node-connectivity version of the unitary NDC problem, and in some instances they can be facet defining. For instance, Stoer [32] shows that under certain conditions, the partition inequalities are facet defining for the node-connectivity version of the SND problem.

In a related article [25] we consider the NDLC problem and derive a dual-ascent algorithm using a directed flow model on a directed graph. Computational experiments reported in that article show that the dual ascent algorithm applied to the directed flow model is able to solve problems with up to 300 nodes and 3000 edges to within a few percent of optimality, indicating that the linear programming relaxation of the improved flow formulation provides a good approximation to this mixed integer program model.

Acknowledgment

We thank Daliborka Stanojević for assistance with the computational experiments. We also thank the referees for their detailed remarks on earlier versions of this article.

REFERENCES

[1] A.M. Baïou and A.R. Mahjoub, Steiner 2-edge connected subgraph polytope on series-parallel graphs, SIAM J Discrete Math 10 (1997), 505–514.

[2] A. Balakrishnan, T.L. Magnanti, and P. Mirchandani, Connectivity-splitting models for survivable network design, Networks 43 (2004), 10–27.

[3] E. Balas and W.R. Pulleyblank, The perfectly matchable subgraph polytope of a bipartite graph, Networks 13 (1983), 495–516.

[4] J.F. Benders, Partitioning procedures for solving mixed variables programming problems, Numer Math 4 (1962), 238–252.

[5] M. Dida Biha and A.R. Mahjoub, k-edge connected polyhedra on series-parallel graphs, Oper Res Lett 19 (1996), 71–78.

[6] S.C. Boyd and T. Hao, An integer polytope related to the design of survivable communication networks, SIAM J Discrete Math 6 (1993), 612–630.

[7] R.H. Cardwell, C.L. Monma, and T.H. Wu, Computer-aided design procedures for survivable fiber optic networks, IEEE J Selected Areas Commun 7 (1989), 1188–1197.

[8] S. Chopra, On the spanning tree polyhedron, Oper Res Lett 8 (1989), 25–29.

[9] S. Chopra, Polyhedra of the equivalent subgraph problem

and some edge connectivity problems, SIAM J Discrete Math 5 (1992), 321–337.

[10] S. Chopra and M.R. Rao, The Steiner tree problem I: Formulations, compositions and extensions of facets, Math Program 64 (1994), 209–229.

[11] S. Chopra and M.R. Rao, The Steiner tree problem II: Properties and classes of facets, Math Program 64 (1994), 231–246.

[12] A. Frank, "Connectivity augmentation problems in network design," Mathematical Programming: State of the Art 1994, J.R. Birge and K.G. Murty (Editors), The University of Michigan, East Lansing, MI, 1994.

[13] M.L. Fredman and R.E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, J ACM 34 (1987), 596–615.

[14] M.X. Goemans, Analysis of linear programming relaxations for a class of connectivity problems, PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1990 (also Working paper OR 233-90, Operations Research Center, M.I.T.).

[15] M.X. Goemans, The Steiner tree polytope and related polyhedra, Math Program 63 (1994), 157–182.

[16] M.X. Goemans, Worst-case comparison of valid inequalities for the TSP, Math Program 69 (1995), 335–349.

[17] M.X. Goemans and Y.-S. Myung, A catalog of Steiner tree formulations, Networks 23 (1993), 19–28.

[18] R.E. Gomory and T.C. Hu, Multi-terminal network flows, SIAM J Appl Math 9 (1961), 551–570.

[19] M. Grötschel and C.L. Monma, Integer polyhedra arising from certain network design problems with connectivity constraints, SIAM J Discrete Math 3 (1990), 502–523.

[20] M. Grötschel, C.L. Monma, and M. Stoer, Computational results with a cutting plane algorithm for designing communication networks with low-connectivity constraints, Oper Res 40 (1992), 309–330.

[21] M. Grötschel, C.L. Monma, and M. Stoer, Facets for polyhedra arising in the design of communication networks with low-connectivity requirements, SIAM J Optimizat 2 (1992), 474–504.

[22] M. Grötschel, C.L. Monma, and M. Stoer, "Design of survivable networks," Handbooks in Operations Research and Management Science, M. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser (Editors), North-Holland, Amsterdam, 1995, pp. 617–672.

[23] M. Grötschel, C.L. Monma, and M. Stoer, Polyhedral and computational investigations for designing communication with high survivability requirements, Oper Res 43 (1995), 1012–1024.

[24] K. Jain, A factor 2 approximation algorithm for the generalized Steiner network problem, Combinatorica 21 (2001), 39–60.

[25] T.L. Magnanti and S. Raghavan, A dual-ascent algorithm for low-connectivity network design, Technical report, Robert H. Smith School of Business, University of Maryland, College Park, MD, 2004.

[26] T.L. Magnanti and L.A. Wolsey, Optimal trees, Handbooks in Operations Research and Management Science, M. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser (Editors), North-Holland, Amsterdam, 1995, pp. 503–615.

[27] A.R. Mahjoub, Two-edge connected spanning subgraphs and polyhedra, Math Program 64 (1994), 199–208.

[28] K. Menger, Zur allgemeinen kurventheorie, Fundam Math 10 (1927), 96–115.

[29] C.St.J.A. Nash-Williams, On orientations, connectivity, and odd vertex pairings in finite graphs, Can J Math 12 (1960), 555–567.

[30] S. Raghavan, Formulations and algorithms for network design problems with connectivity requirements, PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1995.

[31] S. Raghavan and T.L. Magnanti, "Network connectivity," Annotated Bibliographies in Combinatorial Optimization, M. Dell'Amico, F. Maffioli, and S. Martello (Editors), John Wiley & Sons, New York, 1997, pp. 335–354.

[32] M. Stoer, "Design of survivable networks," Lecture Notes in Mathematics, Springer-Verlag, Berlin, 1992.

[33] P. Winter, Generalized Steiner problems in series-parallel networks, J Algorithms 7 (1986), 549–566.

[34] P. Winter, Steiner problems in networks: A survey, Networks 17 (1987), 129–167.

[35] R.T. Wong, A dual ascent approach for Steiner tree problems on a directed graph, Math Program 28 (1984), 271–287.