# Reload Cost Trees and Network Design

**Ioannis Gamvros**
*IBM Software Group, ILOG Optimization, 4400 North First Street, San Jose, California 95134*

**Luis Gouveia**
*Departamento de Estatística e Investigação Operacional, Faculdade de Ciências da Universidade de Lisboa, Centro de Investigação Operacional Bloco C6, Piso 4, 1749-016 Lisboa, Portugal*

**S. Raghavan**
*Robert H. Smith School of Business & Institute for Systems Research, University of Maryland, College Park, Maryland 20742*

**In this article, we consider the notion of "reload costs" in network design. Reload costs occur naturally in many different settings including telecommunication networks using diverse technologies. However, reload costs have not been studied extensively in the literature. Given that reload costs occur naturally in many settings, we are motivated by the desire to develop "good" models for network design problems involving reload costs. In this article, and as a first step in this direction, we propose and discuss the reload cost spanning tree problem (RCSTP). We show that the RCSTP is NP-complete. We discuss several ways of modeling network design problems with reload costs. These involve models that expand the original graph significantly—to a directed line graph and a colored graph—to model reload costs. We show that the different modeling approaches lead to models with the same linear programming bound. We then discuss several variations of reload cost spanning tree and network design problems, and discuss both their complexity and models for these variations. To assess the effectiveness of the proposed models to solve RCSTP instances, we present results taken from instances with up to 50 nodes, 300 edges, and nine technologies for several variations of the problem.** © 2011 Wiley Periodicals, Inc. NETWORKS, Vol. 59(4), 365–379 2012

**Keywords:** reload costs; network design; integer programming formulations; spanning trees; Steiner variants; diameter/hop constraints

Received January 2010; accepted March 2011
*Correspondence to*: S. Raghavan; e-mail: raghavan@umd.edu
Contract grant sponsor: Fundação para a Ciência e Technologia grant POCTI-ISFL-1-152 (to Luis Gouveia).
DOI 10.1002/net.20443
Published online 1 August 2011 in Wiley Online Library (wileyonlinelibrary.com).


## 1. INTRODUCTION

In this article, we consider the notion of "reload costs" in network design. Reload costs occur in telecommunication networks using diverse technologies. For example, one of the products offered by commercial satellite service providers and their partners is a virtual private network (VPN) that can offer voice, video, and data connectivity between all of the geographically dispersed locations of large corporate, government, and military organizations. Typically, these VPNs are made up of satellite links and fiber optic cables. The use of diverse technologies at different junctions of the VPN results in an extra cost component (i.e., in addition to typical routing costs) that is associated with the equipment required to seamlessly bind them together. These interface costs are referred to as reload costs and depend on the technologies being connected. Moreover, these costs can sometimes dominate other costs such as the regular routing costs.

Reload costs can appear under many different contexts. In the telecommunications industry, any network design that incorporates different technologies will contain reload costs. Even in cases where the technology remains the same but there are many different telecommunications providers that participate in the complete network, switching between the networks of different providers might entail reload costs. In the transportation industry, intermodal freight is a fast growing and very successful business. In these transportation networks, the unloading of freight from one type of carrier to another results in significant reload costs. In the energy industry, reload costs can capture the losses associated with the interfaces used to transfer energy from one type of carrier to another.

Given that reload costs occur naturally in many settings, we are motivated by the desire to develop "good" models for network design problems involving reload costs. As a first

step in this direction, this article proposes and discusses the (minimum) reload cost spanning tree problem (RCSTP). Formally, we are given a graph $G_R = (V_R, E_R)$, a color $C_{ij}$ for each $\{i,j\} \in E_R$ (the colors represent different technologies in the telecommunications industry context or different carriers in the transportation industry context), a per unit (nonnegative) flow (i.e., variable) reload cost[1] $R^i_{nm}$ for each pair of colors $(n,m)$ at node $i$, and a set of (nonnegative) demands $d_{st}$ between all pairs of nodes $(s,t)$ in $V_R$. Given a spanning tree $T$, we define the reload cost between nodes $s$ and $t$ as the sum of the reloads in the unique path in $T$ between $s$ and $t$multiplied by the demand $d_{st}$. The total reload cost of a spanning tree $T$ is given by the sum of the reload costs for all pairs of nodes. We wish to build a spanning tree over the nodes in $V_R$ that has minimum total reload cost. We call this problem the RCSTP. For the moment, especially given our desire to better understand modeling issues related to reload costs, we ignore other types of costs in our models. Later, we explain how to incorporate additional costs within our models.

A problem related to the RCSTP is the Quadratic Spanning Tree Problem (QSTP) [2]. In the QSTP, we also wish to build a minimum cost spanning tree, where the costs are associated with pairs of edges. A version of the QSTP that is more closely related to the reload cost problem studied in this article is the so-called adjacent QSTP (AQSTP), where only adjacent pairs of edges have nonzero costs (see [2]). Notice that the distinction between the costs in the AQSTP and the reload costs incorporated in the RCSTP is that the former are fixed costs associated with the selection of adjacent edges, whereas the latter are variable (per-unit of flow) costs associated with flow on the edges. If the underlying graph has colored edges as we have in the RCSTP, a "colored" variant of the AQSTP is obtained when we consider costs associated with adjacent edges that have different colors. Here, a fixed reload cost is associated with every adjacent pair of edges on the tree with different colors. We call this problem the fixed RCSTP (FRCSTP) (see Section 6 for a discussion of the FRCSTP) which is also NP-complete because it generalizes the AQSTP (consider a graph with all edges with different colors). The difference between the FRCSTP and the RCSTP is identical to the difference between the classic Minimum Spanning Tree Problem (MSTP) and the Optimal Communications Spanning Tree Problem (OCSTP) [14].

Despite their apparent usefulness in modeling complex cost structures in both the telecommunications and transportation industry, reload costs have not been studied extensively in the literature. Specifically, the only article in which reload costs appear (and were first introduced) prior to the preliminary conference version of this article [8] is by Wirth and Steffan [17] who introduce minimum diameter spanning trees with reload costs. Their problem is similar to the RCSTP but with a different objective. They wish to build a tree network that spans all the nodes in the graph but has the smallest possible diameter with respect to the reload costs (i.e., they wish to minimize the maximum reload cost between any two nodes in the network). The authors show that the minimum diameter RCSTP is NP-Complete for graphs with an arbitrary node degree. They also present an approximation algorithm for graphs with maximum node degree equal to 5 and an exact algorithm for graphs with maximum node degree equal to 3. It is interesting to note that the difference between the RCSTP and the minimum diameter version studied by Wirth and Steffan [17] also has an analogue in more conventional spanning tree problems given by the previously mentioned OCSTP and the minimum diameter spanning tree problem [13], where one wishes to find a tree that minimizes the diameter (meaning the length of the longest path in the tree). We should note that except for the MSTP and the minimum diameter spanning tree problem all of the aforementioned problems are known to be NP-Complete. Not surprisingly, as we will show, the RCSTP is also NP-complete.

Subsequent to our presentation and introduction of the RCSTP in [8], other researchers have begun to consider reload costs in network design problems. In particular, we are aware of three new works that have appeared since 2008. Galbiati [6] considers the minimum diameter spanning tree with reload costs and solves an open question of Wirth and Steffan [17]. She shows that unless P=NP the problem cannot be approximated within any constant $\alpha < 2$ when reload costs are unrestricted and cannot be approximated within any constant $\beta < 5/3$ if the reload costs satisfy the triangle inequality. Amaldi et al. [1] discuss the complexity of some path, tour, and flow problems under variable reload costs. Gourvès et al. [11] focus on the complexity of the minimum reload cost $s-t$ path (which does not allow a node or edge to be revisited), trail (which allows a node to be revisited), and walk (which allows both a node and an edge to be revisited) problems under symmetric and asymmetric reload costs. None of the aforementioned works comprehensively studies tree problems nor do they consider modeling approaches to solve reload cost problems.

This article has several goals. In Section 2, we show that the RCSTP (and in fact, many special cases of the problem) is NP-complete. Thus, while reload costs are easy to state and conceptualize, even the simplest reload cost problems are challenging. Section 3 begins the discussion of how to effectively model problems with reload costs. Because reload costs only occur when there is a change of color on the edges on a route, a straightforward formulation to model the problem is a network flow problem with a quadratic objective function which is the first model presented and discussed in this study. Then, we present a linearized version of the quadratic model, and an enhanced version of this model that is obtained by generalizing the flow conservation constraints. This second model seems to be quite natural for the problem under study and can be seen as a straightforward linear network flow model in a specialized auxiliary graph, more precisely, a directed line graph of the original graph $G_R$. The line graph formulation has the disadvantage that it is completely blind to the number of different colors of a given instance. Section

[1] As the reload costs can be location dependent (e.g., in intermodal freight), to consider the most general version of this problem, we allow the reload cost to also depend on the node at which reload occurs.

4 presents a different formulation whose size depends on the number of colors of the problem. This formulation imitates, in a certain way, the networks using reload costs and views the given network as partitioned into layers, each one associated to a given color (technology or carrier). In general, this formulation that we refer to as the colored graph formulation also has (many) fewer constraints and variables than the line graph formulation. We also show that both formulations produce the same linear programming bound, thus making the colored graph formulation more attractive to use from a practical point of view (this will be confirmed by our computational experiments). Section 5 discusses the equivalence between the linear programming relaxations of the two models together with model enhancements. Section 6 describes several variants of the RCSTP. Adequate modifications of the models described for the RCSTP will also be described for these variants. Section 7 describes our computational experiments which include testing the models for the RCSTP as well as a directed model for a single-source variant of the problem. Section 8 provides concluding remarks.

## 2. COMPLEXITY OF THE RCSTP

The decision version of the RCSTP asks the question "Is there a spanning tree with total reload cost $\leq$ K?" It is easy to see that this decision version of the RCSTP is in NP. We now show that the RCSTP is NP-complete. To do so, we transform an instance of the 3-SAT problem to an instance of the RCSTP. Recall, an instance of 3-SAT is given by a set $X = \{x_1, x_2, x_3, \ldots .., x_n\}$ of true/false variables and a set $C = \{C_1, C_2, .. \ldots ., C_k\}$ of $k$ clauses over $X$, each containing 3 literals. The goal is to find an assignment of true/false values to the variables so that all clauses are truthfully satisfied. An instance of the RCSTP is constructed from an instance of the 3-SAT problem as follows. Create nodes $r, x_1, x_2, x_3, \ldots .., x_n, C_1, C_2, .. \ldots ., C_k$. Connect $r$ to node $x_i$ $(i = 1, \ldots, n)$ by two parallel edges: one with color $i$ and one with color $\bar{i}$. For every $i = 1, \ldots, n$ and $j = 1, \ldots, k$ connect $x_i$ to $C_j$ by an edge of color $i$ if $x_i$ is a literal in $C_j$, and connect $x_i$ to $C_j$ by an edge of color $\bar{i}$ if $\bar{x}_i$ is a literal in $C_j$. So far, the transformation is identical to that in Wirth and Stefan [17] and is illustrated by an example in Figure 1.

The reload costs are then set as follows. They are equal to 0 when there is no change in color, and equal to 1 when there is a change in color. The demands are set to 1 for every pair of nodes $(r, x_i)$ and $(r, C_i)$, and set to 0 to all other pairs. The resulting network can be viewed as a single source (as opposed to all pairs) RCSTP problem with unit reload costs. Observe that the reload costs satisfy the triangle inequality.

We note that it suffices to focus on spanning trees for the RCSTP where the clause nodes have unit degree. Suppose there is a clause node $C_j$ connected to node $x_i$ that has no edge to node $r$. Deleting $(x_i, C_j)$ and adding $(r, x_i)$ with color identical to $(x_i, C_j)$ results in a solution whose cost does not increase, and where the degree of clause node $C_j$ is reduced. When there is an assignment that satisfies all clauses truthfully, there will be no reloads in the paths from node $r$ to each of the other nodes in the graph, and thus the cost of the minimum reload cost tree will be 0. On the other hand, if there is no assignment of values to the variables that satisfies all clauses, then the minimum cost RCSTP must have a nonzero cost. (The corresponding decision version of the RCSTP asks the question whether there is a reload cost spanning tree with total reload cost $\leq 0$.) This shows that the single-source RCSTP with unit reload costs is NP-complete, and thus by extension the RCSTP is NP-complete.

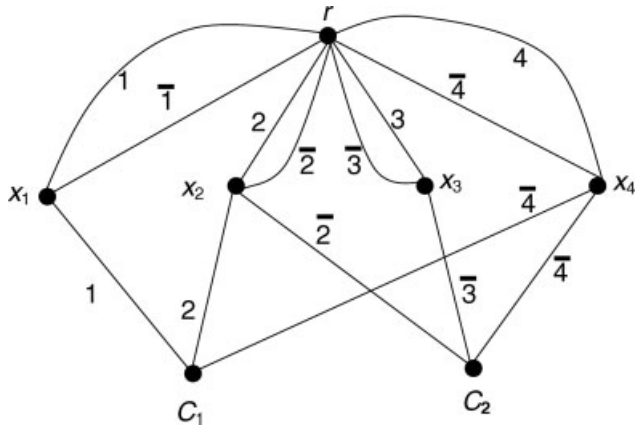


FIG. 1. Transformation of 3-SAT into the RCSTP.

## 3. THE LINEARIZED QUADRATIC FORMULATION AND THE LINE-GRAPH FORMULATION

In this section, we present a model for the problem that is obtained after linearizing the objective function of a straightforward network flow model (Subsection 3.1). Then, in Section 3.2, we present an enhancement of this model that is obtained by generalizing the flow conservation constraints and show that the new model has a natural interpretation when viewed in a special auxiliary graph.

In terms of notation, we note that, because the models that we will discuss have underlying directed shortest path problems, we shall define a set $A_R$ of arcs defined from the edge set $E_R$ (that is, for each edge $\{i, j\}$ in $E_R$, the set $A_R$ contains two arcs $(i, j)$ and $(j, i)$). We say that a path traverses arc $(i, j)$ if it traverses edge $\{i, j\}$ in the direction from $i$ to $j$.

### 3.1. The Linearized Formulation

Let $u_e$ be a binary variable indicating whether edge $e = \{i, j\}$ of the original graph is in the solution (we will use either the notation $u_e$ or $u_{\{ij\}}$) and let the binary variables $y_{ij}^{st}$ $(i \neq t$ and $j \neq s)$ indicate whether arc $(i, j)$ is in the path from node $s$ to node $t$). Then, the problem can be modeled as a straightforward network flow model with a quadratic objective function (to simplify the indexing of the summation terms, we assume that the models are defined on complete graphs):

$$\min \sum_{s,t \in V_R} d_{st} \sum_{i,j,k \in V_R} R^j_{C_{ij} C_{jk}} \left( y_{ij}^{st} * y_{jk}^{st} \right) \tag{3.1}$$

$$\sum_{i\in V_R} y^{st}_{ij} - \sum_{i\in V_R} y^{st}_{ji} = \begin{cases} 1 & j = t \\ 0 & j \neq s,t \\ -1 & j = s \end{cases} \quad \text{for all } s,t \in V_R, \tag{3.2}$$

$$y^{st}_{ij} + y^{st}_{ji} \leq u_{\{ij\}} \quad \text{for all } \{i,j\} \in E_R; s,t \in V_R, \tag{3.3}$$

$$\sum_{e\in E_R} u_e = |V_R| - 1, \tag{3.4}$$

$$y^{st}_{ij} \in \{0,1\} \quad \text{for all } (i,j) \in A_R, s,t \in V_R, \tag{3.5}$$

$$u_e \in \{0,1\} \quad \text{for all } e \in E_R. \tag{3.6}$$

We denote by QUAD the formulation presented above. The objective function is straightforward and considers all reload costs between arc transitions (note that this cost is zero when $C_{ij} = C_{jk}$). Constraints (3.2) are typical flow conservation constraints and guarantee that 1 unit of flow goes from node $s$ to node $t$, for each commodity pair $(s,t)$. Constraints (3.2) can also be viewed as modeling path problems, one for each pair of nodes $s$ and $t$. The forcing constraints (3.3) restrict the flow to be on edges that are in the spanning tree. Constraint (3.4) states that the spanning tree has the required number of edges and together with the flow conservation constraints and the forcing constraints ensures that the set of edges associated to variables $u_e$ equal to 1, form a spanning tree. Note that when the reload costs are symmetric (i.e., $R^i_{nm} = R^i_{mn}$), and there are no other routing costs it is possible to reduce the number of commodities by setting $d_{st} = d_{st} + d_{ts}$ for $s < t$, and $d_{ts} = 0$. We then only consider commodity pairs $(s,t)$ when $s < t$. In this way, we only include half of all the possible origin-destination pairs in our model and therefore we reduce the number of decision variables considered.

To linearize the quadratic objective function, consider the variables $f^{st}_{ijk}$ ($i,j \neq t$ and $j,k \neq s$) that indicate whether the path from $s$ to $t$ traverses arc $(j,k)$ immediately after traversing arc $(i,j)$. These variables permits us to write a model with the following linear objective function

$$\sum_{s,t\in V_R} d_{st} \sum_{i,j,k\in V_R} R^j_{C_{ij}C_{jk}} f^{st}_{ijk} \tag{3.7}$$

provided that we also introduce constraints linking the sets of flow variables. Here, we consider the following two sets of linking constraints:

Forward set:

$$y^{st}_{ij} = \sum_{k\in V_R} f^{st}_{ijk} \quad \text{for all } (i,j) \in A_R,\ s,t \in V_R, j \neq t, \tag{3.8a}$$

$$y^{st}_{it} = f^{st}_{itt'} \quad \text{for all } (i,t) \in A_R,\ s,t \in V_R. \tag{3.8b}$$

Backward set:

$$y^{st}_{ij} = \sum_{k\in V_R} f^{st}_{kij} \quad \text{for all } (i,j) \in A_R,\ s,t \in V_R, i \neq s, \tag{3.9a}$$

$$y^{st}_{sj} = f^{st}_{s'sj} \quad \text{for all } (s,j) \in A_R,\ s,t \in V_R. \tag{3.9b}$$

Note that when defining the forward linking constraints for the variables associated with arcs $(i,t)$, constraints (3.8b), we have also created linearized variables $f^{st}_{itt'}$ with zero reload costs. This is equivalent to adding a node $t'$ and an arc $(t,t')$ to the underlying shortest path problem from any node $s$ to node $t$ and considering a zero reload cost when changing from an arc $(j,t)$ to the new arc $(t,t')$. Similarly, in the backward linking constraints for the variables associated with arcs $(s,j)$, constraints (3.9b), we have also created linearized variables $f^{st}_{s'sj}$ with zero reload costs. This is equivalent to adding a node $s'$ and an arc $(s',s)$ to the underlying shortest path problem from any node $s$ to node $t$ and considering a zero reload cost when changing from the new arc $(s',s)$ to an arc $(s,i)$. Thus, a path from any node $s$ to any node $t$ in the original graph corresponds now to a path from the node $s'$ to the node $t'$ (which necessarily starts with arc $(s',s)$ and ends with arc $(t,t')$). We denote by LIN-QUAD the model obtained by replacing the objective function (3.1) in QUAD by (3.7), and by adding to QUAD the constraints (3.8a/b), (3.9a/b) and

$$f^{st}_{ijk} \in \{0,1\} \quad \text{for all } i,j,k \in V_R : (i,j),\ (j,k) \in A_R,\ s,t \in V_R. \tag{3.10}$$

Note, that we could have obtained two other valid linearized models by using only one set, forward or backward, of the linking constraints. However, it can easily be seen that such models would have a linear programming relaxation that is weaker than the linear programming relaxation of the LIN-QUAD model.

### *3.2. Enhancing the Linearized Model and the Line Graph Formulation*

Recall that when linearizing the quadratic model, we have introduced new variables that indicate the flow moving from one arc to another (adjacent) arc. To enhance the model, for each pair of source and destination nodes, we consider new constraints that are flow conservation constraints for each arc of the graph (as opposed to flow conservation constraints for each node as in the QUAD and LIN-QUAD models). Subsequently we shall denote by node flow conservation constraints, the flow conservation constraints (3.2) and by arc flow conservation constraints, the new flow conservation constraints (3.11).

The arc flow conservation constraints are as follows:

$$\sum_{i\in V_R} f^{st}_{ijk} - \sum_{i\in V_R} f^{st}_{jki} = \begin{cases} 1 & (j,k) = (t,t') \\ 0 & (j,k) \neq (s',s),(t,t') \\ -1 & (j,k) = (s',s) \end{cases} \quad \text{for all } s,t \in V_R. \tag{3.11}$$

These new constraints can be interpreted as follows: Suppose that the path from $s$ to $t$ contains arc $(j,k)$. Then, the constraints state that the path contains an arc entering node $j$ if and only if it contains an arc leaving node $k$.

We add these constraints to the LIN-QUAD model. Note that in the presence of these constraints for arcs $(j,k) \neq (s',s),(t,t')$, one of the sets of the linking constraints, (3.8a) or (3.9a), becomes redundant, and can be omitted. In the remainder of the text, we shall keep (3.8a). We shall use these linking constraints to replace "old" variables $y_{ij}^{st}$ by the new variables $f_{ijk}^{st}$ in the inequalities of the LIN-QUAD model. Thus, the previous linking constraints (3.3) become

$$\sum_{k\in V_R} f_{ijk}^{st} + \sum_{k\in V_R} f_{jik}^{st} \leq u_{\{ij\}} \quad \textit{for all } \{i,j\} \in E_R; s,t \in V_R. \tag{3.12}$$

We show next that the node flow conservation constraints (3.2) are redundant in the LIN-QUAD model augmented with the arc flow conservation constraints (3.11). First, we note that under (3.8b) and (3.9b), the node flow conservation constraints (3.2) for $j = s,t$ are the arc flow conservation constraints for $(j,k) = (t,t'),(s',s)$. With respect to (3.2) for $j \neq s,t$, note that after using the constraints (3.8a) linking the two sets of variables, the node flow conservation constraints (3.2) for $j \neq s,t$ become

$$\sum_{i\in V_R}\sum_{k\in V_R} f_{ijk}^{st} - \sum_{i\in V_R}\sum_{k\in V_R} f_{jik}^{st} = 0 \quad \textit{for all } s,t \in V_R; j \neq s,t. \tag{3.13}$$

Now, observe that if for a fixed $s$, $t$, and $j$, we add the arc flow conservation constraints (3.11) for all arcs $(j,k)$ we obtain the given transformed node flow conservation constraints (3.13). Thus, constraints (3.2) can also be omitted from the model. For clarity, we write the whole new enhanced and linearized model (without redundant inequalities) with the arc flow conservation constraints:

$$\min \sum_{s,t\in V_R} d_{st} \sum_{i,j,k\in V_R} R^{j}_{C_{ij}C_{jk}} f_{ijk}^{st} \tag{3.7}$$

$$\sum_{i\in V_R} f_{ijk}^{st} - \sum_{i\in V_R} f_{jki}^{st} = \begin{cases} 1 & (j,k) = (t,t') \\ 0 & (j,k) \neq (s',s),(t,t') \\ -1 & (j,k) = (s',s) \end{cases}$$
$$\textit{for all } s,t \in V_R, s < t \tag{3.11}$$

$$\sum_{k\in V_R} f_{ijk}^{st} + \sum_{k\in V_R} f_{jik}^{st} \leq u_{\{ij\}}$$
$$\textit{for all } \{i,j\} \in E_R; s,t \in V_R, s < t,\ i,j \neq s \tag{3.12}$$

$$\sum_{e\in E_R} u_e = |V_R| - 1 \tag{3.6}$$

$$f_{ijk}^{st} \in \{0,1\} \quad \textit{for all } i,j,k : (i,j),(j,k) \in A_R,\ s,t \in V_R, s < t \tag{3.10}$$

$$u_e \in \{0,1\} \quad \textit{for all } e \in E_R. \tag{3.4}$$

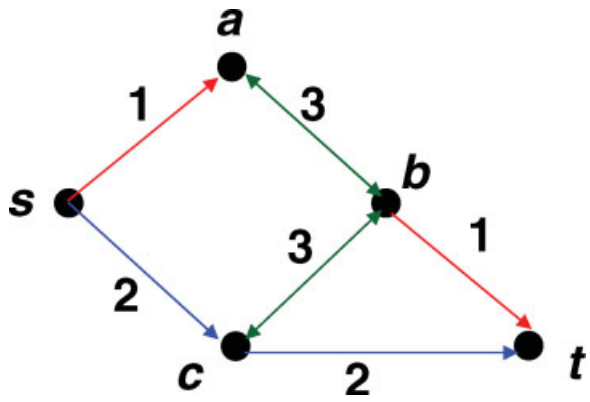


FIG. 2. The original graph (after directing) associated with commodity $(s,t)$. This example has three colors denoted by 1,2,3. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

For reasons to be specified next, we denote by LINE-GRAPH, the formulation just obtained. The steps leading from the LIN-QUAD model to the LINE-GRAPH model permit us to state the following result:

**Proposition 3.1.** *The linear programming bound of the LINE-GRAPH formulation is greater than or equal to the linear programming bound of the LIN-QUAD formulation.*

We are now ready to give a different and intuitive interpretation for the flow systems in the new formulation (and thus justify its designation). For each pair of nodes $s$ and $t$, the underlying "flow" problem can be interpreted as a "conventional" shortest path problem in a more complicated graph, a directed version of the so-called "line" graph (see [10]). The designation of the new formulation follows from this interpretation.

For each commodity $(s,t)$, consider an associated "directed" line graph $LG_{st} = (V_{st}, A_{st})$ defined as follows:

i. $V_{st} = \{(s',s),(t,t')\} \cup \{(x,y) : (x,y) \in A_R\}$
ii. $A_{st} = \{((a,b),(c,d)) : (a,b),(c,d) \in V_{st} \text{ and } b = c\}$

Thus, the node set $V_{st}$ contains nodes that correspond to the directed arcs in $A_R$. The two dummy nodes $(s',s)$ and $(t,t')$ corresponding to dummy arcs, as explained above. The arc set $A_{st}$ contains all arcs of the type $((x,y),(y,z))$ that correspond to two subsequent arcs in $A_R$. Note also that the arc flow conservation constraints (3.11) are nothing other than the normal node flow conservation constraints defined in $LG_{st}$.

Figures 2 and 3 illustrate, for a given commodity $(s,t)$, the original graph after directing it and the corresponding directed line graph. In Figure 2, the labels on the arcs denotes their color. For simplicity, for a given pair $i$ and $j$ of nodes such that the two arcs $(i,j)$ and $(j,i)$ exist, the figures include only one bi-directed edge. Note that in the first figure, we are only considering arcs leaving node $s$ and only arcs entering node $t$. A similar situation arises for the directed line graph shown in the second figure with respect to nodes $(s',s)$ and $(t,t')$. Note that the path $P = \{s,c,b,t\}$ (here, we only give

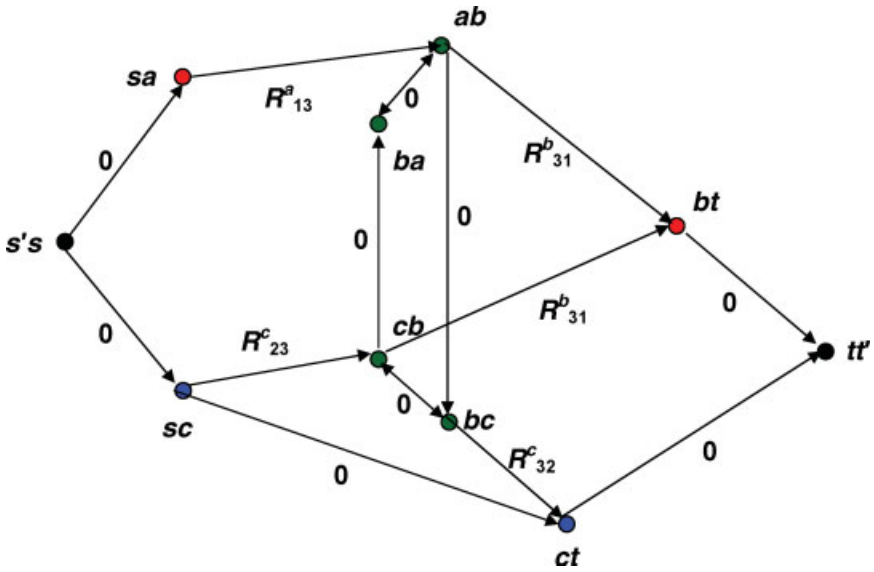


FIG. 3. The directed line graph obtained from the graph of Figure 2. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

the node set of the path) in the original graph corresponds to the path $P_{LG} = \{(s', s), (s, c), (c, b), (b, t), (t, t')\}$ in the line graph. The node transition costs of $P$ are modeled by the arc costs in $P_{LG}$. Note that for an undirected "reload" graph $G_R = (V_R, E_R)$, the associated directed line graph $LG_{st} = (V_{st}, A_{st})$ for a given commodity pair $(s, t)$ contains $2 + 2|E_R|$ nodes and approximately $\sum_{i \in V_R} \deg(i)^2$ arcs where $\deg(i)$ indicates the degree of node $i$ in the original graph $G_R$.

## 4. THE COLORED GRAPH FORMULATION

The LINE-GRAPH formulation has the disadvantage that it is completely blind to the number of different colors of a given instance. More precisely, for two instances which differ only in the number of colors of the edges included in the underlying graph, the LINE-GRAPH formulation has the same size for both instances and only differs in the number of positive reload costs associated with the flow variables. In this section, we present a different formulation whose size depends on the number of colors of the problem. Thus, in the same scenario discussed before, the formulation associated with the instance with fewer colors will have fewer variables and constraints—and thus fewer constraints and variables than the LINE-GRAPH formulation. We shall also show that both formulations produce the same linear programming bound, thus making the new formulation more attractive to use from a practical point of view (this will be confirmed by our computational experiments). We note, however, that this second formulation does not remove the interest of the line graph formulation. The reason for this is that the line graph formulation is more general in the sense that it allows reload costs that depend on the arcs entering and leaving the "reload" node. The formulation discussed in this section cannot model these situations.

### 4.1. The Colored Graph Formulation

The model proposed here imitates, in a certain way, the networks where reload costs occur and views the given network as partitioned into layers, each one associated with a given color (technology or carrier). Thus, for each color, the arcs and nodes associated with the color define one of these layers. Nodes where a reload can occur (change of color, technology, or carrier) are included in different layers and are connected by "reload" arcs. One example is given in Figures 4 and 5. Figure 4 depicts the original graph. A number associated with an edge indicates the color of the corresponding edge. Figure 5 shows the corresponding colored graph. As the original graph has three colors, the colored graph has three layers. For each node $x$ in the original graph, a copy $x_i$ is created in the layer associated with color "$i$." Every edge $e \in E_R$ is placed in the layer associated with its color. For example, edge $(s, a)$ is placed in layer 1 and becomes $(s_1, a_1)$. Between the copies of node $x$ across the different layers we add a clique representing reload cost edges. Going from one

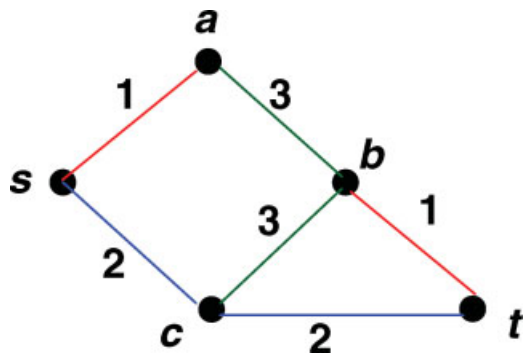


FIG. 4. A reload cost graph $G_R$ with three colors. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

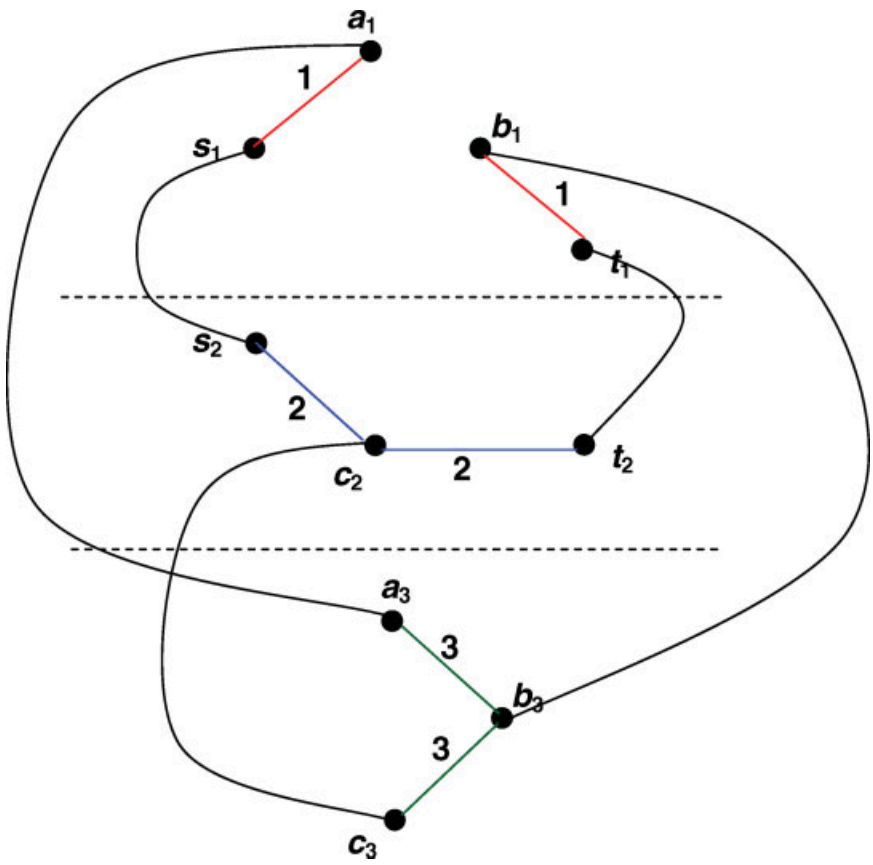


FIG. 5. The colored graph CG obtained from the graph of Figure 4. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

layer to another represents a color change and thus the occurrence of a reload cost. In Figure 5, the unlabeled edges in the colored graph are the reload edges.

Before presenting a more formal definition of the colored graph and of the corresponding model, we point out that for this approach to work the reload costs on any given node need to satisfy the triangle inequality; otherwise, the model would allow solutions that perform several reloads at a given node before changing from an arc to another. However, this assumption is quite reasonable and realistic in practice. As shown in our discussion of the complexity of the RCSTP, the problem remains NP-complete even when the reload costs satisfy the triangle inequality.

Let $C$ denote the colors in the original graph $G_R = (V_R, E_R)$. Consider a color $c \in C$, and let $G_c = (V_c, E_c)$ be the graph defining the corresponding layer, that is, $V_c$ denotes the set of nodes adjacent to edges with color $c$ in $E_R$ and $E_c$ the set of edges associated that have color $c$. The colored graph $CG = (V_{CG}, E_{CG})$ is defined as follows. $V_{CG} = \cup_{[c:c\in C]} V_c$ and $E_{CG} = \cup_{[c:c\in C]} E_c \cup R$, where $R$ denotes the set of reload edges. Edges in $R$ are of the type $\{i_c, i_p\}$ where node $i$ is in the layers associated with colors $c$ and $p$.

To define the model, we need to create commodities and flow variables for every pair of nodes $(s,t)$ with positive demand. We will denote by arc $(i_c, j_c)$ of the colored graph, the edge $\{i_c, j_c\}$ traversed in the direction from node $i_c$ to $j_c$. Thus, we consider binary variables $g^{st}_{i_c j_c}$ ($i \neq t$ and $j \neq s$) indicating whether arc $(i,j)$ of color $c$, is in the path from node $s$ to node $t$ (or alternatively whether we use "arc" $(i_c, j_c)$ of the colored graph in the path from $s$ to $t$). We also create binary variables $h^{st}_{j_c j_k}$ indicating whether we go from color $c$ to color $k$ at node $j$, on the path from $s$ to $t$ (or alternatively whether we use "reload arc" $(j_c, j_k)$ of the colored graph CG in the path from $s$ to $t$). We also use, as before, the binary variables $u_{\{i,j\}}$ indicating whether edge $e = \{i,j\} \in E_{CG} \backslash R$ of the original graph is in the solution (recall that the set of edges $E_{CG}\backslash R$ is identical to the set of edges $E_R$ of the original graph). In the formulation below, we denote by $C_j$ the set of colors associated with node $j$. We can now rewrite the model as follows (we use notation from the colored graph as well as from the original "reload" graph):

$$\min \sum_{s,t\in V_R} d_{st} \sum_{j\in V_R} \sum_{m,n\in C_j} R^j_{nm} h^{st}_{j_n j_m} \tag{4.1}$$

$$\sum_{k\in C_s} \sum_{j\in V_k} g^{st}_{s_k j_k} = 1 \quad \textit{for all } s,t \in V_R, s < t, \tag{4.2}$$

$$\sum_{k\in C_t} \sum_{i\in V_k} g^{st}_{i_k t_k} = 1 \quad \textit{for all } s,t \in V_R, s < t, \tag{4.3}$$

$$\sum_{i\in V_k} g^{st}_{i_k j_k} + \sum_{c:j\in V_c} h^{st}_{j_c j_k} - \sum_{i\in V_k} g^{st}_{j_k i_k} - \sum_{c:j\in V_c} h^{st}_{j_k j_c} = 0$$

$$\text{for all } j \neq s,t; k \in C : j \in V_k; s,t \in V_R, \tag{4.4}$$

$$g^{st}_{i_k j_k} + g^{st}_{j_k i_k} \leq u_{\{ij\}}$$

$$\text{for all } \{i,j\} \in E_R; k \in C : i,j \in V_k, s,t \in V_R, s < t, \tag{4.5}$$

$$\sum_{e\in E_R} u_e = |V_R| - 1, \tag{4.6}$$

$$g^{st}_{i_k j_k}, g^{st}_{j_k i_k} \in \{0,1\} \quad \text{for all } \{i,j\} \in E_R; k = C_{ij}; s,t \in V_R, \tag{4.7}$$

$$h^{st}_{i_k i_p} \in \{0,1\} \quad \text{for all } i \in V_R, k,p \in C_i, s,t \in V_R, \tag{4.8}$$

$$u_e \in \{0,1\} \quad \text{for all } e \in E_R. \tag{4.9}$$

Constraint set (4.2) states that for a given commodity $(s,t)$, one arc leaves one of the copies of node $s$. If the copy of this node is in colored layer $k$, then the arc leaving the node must use the same color. A similar interpretation is given to constraint set (4.3) but this time, with respect to the destination node $t$. Constraints (4.4) are the flow conservation constraints for every commodity pair $(s,t)$, for every node $j \neq s,t$ and every color associated to the node. Because every node in a layer has two types of edges adjacent to it, reload edges and edges within the layer, the flow balance constraints have two types of variables corresponding to them. Note that in terms of feasible solutions, these flow conservation constraints permit a sequence of reload edges on the same node and that is why we have imposed the triangle inequality assumption for reloads at each node. The fourth set (4.5) is the forcing constraints, and they are only defined for edges in $E_R$. Constraint (4.6) indicates that the solution has $|V_R| - 1$ edges in the different layers, and the remaining constrains are self-explanatory. We denote by COLORED-GRAPH the previous model.

We note that we could have defined the model in a slightly different way, which helps in proving the equivalence of the linear programming relaxation of the two models, LINE-GRAPH and COLORED-GRAPH. We could have added two dummy nodes $s$ and $t$ with arcs from node $s$ to every copy of node $s$ in the corresponding layers and with arcs from any copy of node $t$ to node $t$. Then, the first two flow conservation constraints (4.2 and 4.3) of the model COLORED-GRAPH can be rewritten as follows:

$$\sum_{k\in C_s} g^{st}_{s,s_k} = 1 \quad \text{for all } s,t \in V_R,$$

$$\sum_{k\in C_t} g^{st}_{t_k,t} = 1 \quad \text{for all } s,t \in V_R.$$

We conclude this section by noting that an alternative colored model has been studied in Gamvros [7]. In this approach, we do not have reload arcs of the type $(i_c, i_p)$ where node $i$ is in the layers associated with colors $c$ and $p$. Instead, we have arcs of the form $(i_c, j_p)$, where arc $(i,j)$ has color $c$ and a reload occurs at node $j$ from color $c$ to color $p$. Thus, an arc $(i_c, j_p)$ in this new layered color graph represents the sequence of two arcs $(i_c, j_c) - (j_c, j_p)$ in the previous colored graph. This would lead to a layered colored graph with many more arcs than the one we have developed in this article and that is why we are omitting the complete description of this approach from this article.

## 5. EQUIVALENCE OF THE CORRESPONDING LINEAR PROGRAMMING RELAXATIONS AND MODEL ENHANCEMENTS

In this section, we show that under the given assumptions, that is, reload costs depend solely on the change in color of the edges and satisfy the triangle inequality at every node, the LINE-GRAPH and COLORED-GRAPH formulations produce the same linear programming bound (this is shown in Subsection 5.1). In Subsection 5.2, we show how to significantly enhance the linear programming relaxation of the given models.

### 5.1. Equivalence of the Linear Programming Relaxations

To show that the two models, LINE-GRAPH and COLORED-GRAPH produce the same linear programming bound, we start by making the following simple observation.

Consider two nodes $s$ and $t$, and consider any path (which we can assume without loss of generality is loopless and elementary) between nodes $(s', s)$ and $(t, t')$ in the line graph. This path can easily be transformed into a path between a copy of node $s$ and a copy of node $t$ in the colored graph. We only need to distinguish two situations occurring in the path: (a) an arc $((i,j),(j,k))$ with positive cost (because there is a reload cost at node $j$ and arc $(i,j)$ has color $c1$ and arc $(j,k)$ has color $c2$) being included in the path, and (b) an arc $((i,j),(j,k))$ with zero cost (because there is no reload cost at node $j$ and arc $(i,j)$ has color $c1$ and arc $(j,k)$ has the same color $c1$) being included in the path. In the first case, the part of the path in the colored graph corresponding to this arc is composed of the three arcs $(i_{c1}, j_{c1}), (j_{c1}, j_{c2}), (j_{c2}, k_{c2})$ with a reload cost on the middle arc. In the second case, the part of the path in the colored graph corresponding to this arc is composed of the two arcs $(i_{c1}, j_{c1}), (j_{c1}, k_{c1})$. Note also that these arguments show that the two paths have the same cost. Consider now two nodes $s_c$ and $t_p$ (that is, the copy of node $s$ in the layer with color $c$ and the copy of node $t$ in the layer with color $p$), and consider an elementary path between $s_c$ and $t_p$ in the colored graph such that it does not do multiple reloads at the same node. By reversing the argument given before we see that this path can also be easily replicated in the line graph for commodity $(s,t)$ and that it has the same cost as the path in the colored graph.

Consider now a feasible solution for the linear programming relaxation of the LINE-GRAPH formulation. Let us consider any pair $s,t$ and the possible fractional solution corresponding to the 1 unit of flow between nodes $(s', s)$ and

$(t, t')$. As the underlying shortest path model is the standard unconstrained shortest path model in a special graph, the flow decomposition theorem (see [4]) permits us to state that this fractional solution can be decomposed into several paths, each one carrying a flow of less than one unit and such that the sum of the individual flows equals 1. Following the previous observation, each one of these fractional paths can be transformed into a fractional path, with the same cost, between a copy of node $s$ and a copy of node $t$ in the colored graph. Combining all of these fractional paths, we obtain one unit flow solution in the colored graph corresponding to the path between the copy of node $s$ and the copy of node $t$. As the line graph fractional solution satisfies the forcing constraints, the corresponding solution in the colored graph also satisfies the forcing constraints in the colored graph model. The reverse transformation is similar. Here, however, we start with an optimal solution for the linear programming relaxation of the colored graph formulation. Thus, any fractional path between any pair of nodes $s$ and $t$ does not include multiple reloads at any given node, and we can use the observation given before to obtain a solution feasible for the linear programming relaxation of the line graph formulation with the same cost.

The preceding argument shows that we can transform an optimal solution for the linear programming relaxation of one of the models into one for the linear programming relaxation of the other model with the same cost, leading to the statement.

**Proposition 5.1.** *The linear programming bound of the LINE-GRAPH formulation is equal to the linear programming bound of the COLORED-GRAPH formulation.*

### *5.2. Enhancing the Linking Constraints*

As our experiments will show, the model given above has a weak linear programming bound. In the context of Uncapacitated Network Design, Balakrishnan et al. [3] describe a way of strengthening the forcing constraints present in the model COLORED-GRAPH. The idea is that when edge $\{i,j\}$ is selected then all commodities flowing to (from) a given node "$s$" will flow either from $i$ to $j$ or from $j$ to $i$. We model this situation with the following set of constraints.

The first set, constraints (5.1), are defined for commodities flowing out of node "$s$" and the second set, constraints (5.2), are defined for commodities that terminate at node "$s$."

$$g^{si}_{i_k j_k} + g^{sj}_{j_k i_k} \leq u_{\{ij\}} \quad \textit{for all } \{i,j\} \in E_R; k \in C : i,j \in V_k, s \in V_R \quad (5.1)$$

$$g^{is}_{i_k j_k} + g^{js}_{j_k i_k} \leq u_{\{ij\}} \quad \textit{for all } \{i,j\} \in E_R; k \in C : i,j \in V_k, s \in V_R \quad (5.2)$$

We will denote by "Set1" the set of forcing constraints included in the COLORED-GRAPH model and by "Set2" the set of constraints (5.1), (5.2) together with the set of forcing constraints already included in the model. The previous constraints can be further generalized by taking into account all combinations of edges and commodities leading to

$$g^{sp}_{i_k j_k} + g^{sq}_{j_k i_k} \leq u_{\{ij\}} \quad \textit{for all } \{i,j\} \in E_R; k \in C : i,j \in V_k, s,p,q \in V_R : (s < p,q), \quad (5.3)$$

$$g^{ps}_{i_k j_k} + g^{qs}_{j_k i_k} \leq u_{\{ij\}} \quad \textit{for all } \{i,j\} \in E_R; k \in C : i,j \in V_k, s,p,q \in V_R : (p,q < s). \quad (5.4)$$

These constraints include the constraints (5.1), (5.2) as well as constraints (4.5) originally included in the model. We denote by "Set 3" the set defined by (5.3) and (5.4).

We note that similar forcing constraints could also have been suggested for the LINE-GRAPH model. This leads to enhanced versions of the LINE-GRAPH model and one can state propositions similar to Proposition 5.1 for the enhanced versions of the two models. However, due to the simplicity of the colored graph model, we focus our discussion on the colored graph model. Gamvros [7] describes the equivalent constraints for the directed LINE-GRAPH model.

## 6. VARIANTS OF RELOAD COST NETWORK DESIGN PROBLEMS

In this section, we discuss many different variants of the reload cost network design problem. For simplicity, we describe different specifications in each subsection, noting that more complex variations can be obtained by considering together specifications given in different subsections. Furthermore, we restrict our attention to modeling these variants on the COLORED-GRAPH model.

### *6.1. The Single-Source Multi-Destination Version of the RCSTP*

As noted in Section 2, the single-source multidestination version of the RCSTP is also NP-Complete. At first sight, this simpler version of the RCSTP appears not to deserve further discussion as it is enough to point out that, letting node 1 be the source, we can easily adapt the models previously stated by just considering commodities $(1,t)$ with $t \in V_R \backslash \{1\}$. For simplicity, because all paths originate at node 1, we drop the superscript "1" from the $g^{1p}_{i_k j_k}$ and $h^{1p}_{i_k i_q}$ variables. The forcing constraints in the model become

$$g^{p}_{i_k j_k} + g^{q}_{j_k i_k} \leq u_{\{ij\}} \quad \textit{for all } \{i,j\} \in E_R; k \in C : i,j \in V_k, p,q \in V_R, \quad (6.1)$$

However, the special structure of the problem (only one source) permits us to direct the tree away from the source

node, node 1, and we can model the problem by using directed arcs associated with the set $A_R$ instead of the undirected edge variables $u_{\{ij\}}$ associated with the set $E_R$. The reader is referred to Magnanti and Wolsey [15], where this directing technique is fully explained and exploited in terms of single-source multidestination tree problems. Let $x_{ij}$ be a binary variable indicating whether arc $(i,j)$ of the original graph is in the solution. This version of the problem can be modeled as follows.

$$\min \sum_{p\in V_R\setminus\{1\}} d_{1p} \sum_{j\in V_R} \sum_{m,n\in C_j} R^j_{nm} h^p_{j_n j_m}$$

$$\sum_{k\in C_s} \sum_{j\in V_k} g^p_{1_k j_k} = 1 \quad \textit{for all } p \in V_R\setminus\{1\}$$

$$\sum_{k\in C_t} \sum_{i\in V_k} g^p_{i_k p_k} = 1 \quad \textit{for all } p \in V_R\setminus\{1\}$$

$$\sum_{i\in V_k} g^p_{i_k j_k} + \sum_{c:j\in V_c} h^p_{j_c j_k} - \sum_{i\in V_k} g^p_{j_k i_k} - \sum_{c:j\in V_c} h^p_{j_k j_c} = 0$$

$$\textit{for all } j,p \in V_R\setminus\{1\}, j \neq p, k \in C : j \in V_k$$

$$g^p_{i_k j_k} \leq x_{ij} \quad \textit{for all } (i,j) \in A_R; k \in C : i,j \in V_k, p \in V_R\setminus\{1\} \tag{6.2}$$

$$\sum_{i\in V_R} x_{ij} = 1 \quad \textit{for all } j \in V_R\setminus\{1\} \tag{6.3}$$

$$g^p_{i_k j_k} \in \{0,1\} \quad \textit{for all } (i,j) \in A_R, k \in C : i,j \in V_k, p \in V_R\setminus\{1\}$$

$$h^p_{i_k i_q} \in \{0,1\} \quad \textit{for all } i \in V_R, k,q \in C_i, p \in V_R\setminus\{1\}$$

$$x_{ij} \in \{0,1\} \quad \textit{for all } (i,j) \in A_R.$$

The constraints that characterize this directed model are (6.2) and (6.3). It is interesting to relate the linear programming relaxation of this model with linear programming relaxation of the undirected model. For that, we consider the constraints

$$x_{ij} + x_{ji} = u_{\{ij\}} \quad \textit{for all } \{i,j\} \in E_R; i,j \neq 1, \tag{6.4}$$
$$x_{1j} = u_{\{1j\}} \quad \textit{for all } j \in V_R\setminus\{1\},$$

that link the two sets of variables, directed and undirected. Now, it is well known (and easy to see) that constraints (6.2) imply, under the linking constraints (6.4), the undirected forcing constraints (6.1) and that the indegree constraints (6.3) imply the edge cardinality constraint in the undirected model. Thus, the linear programming relaxation bound of the directed model is at least as good as its undirected counterpart. It is also possible to show (using ideas given, for instance, in Magnanti and Wolsey [15]) that the directed model and the undirected model, have equal linear programming bounds. This result exhibits the strength of the directed model, because at the cost of using twice as many design variables, it requires many fewer forcing constraints (of the order of |A||C||V|) than the ones needed in the undirected model (of the order of $|A||C||V|^2$) for obtaining an equivalent lower bound. We should note that when the $x_{ij}$ variables are defined as binary in the above directed model, the flow variables $g^p_{i_k j_k}$ and $h^p_{i_k i_q}$ may be relaxed, as the resulting constraint matrix is totally unimodular. (A similar observation can be made with regard to the $u_e$ variables and the LINE-GRAPH and COLORED-GRAPH models in Sections 3 and 4.)

### 6.2. The Fixed Cost RCSTP

As noted in the introduction, we can consider a variant of the problem that is similar to the AQSTP and where we only consider costs related to adjacent edges with different colors. We can model this variant by adding binary arc reload variables $v_{i_m i_n}$ associated with each reload edge, $i \in V_R, m,n \in C_i$ indicating whether there is a reload at node $i$ from color $m$ to color $n$. The objective function becomes

$$\min \sum_{j\in V_R} \sum_{m,n\in C_j} R^j_{nm} v_{j_n j_m}, \tag{6.5}$$

and we add the following constraints to the COLORED-GRAPH model described in Section 4, where we have demands between all pairs of nodes.

$$h^{st}_{i_m i_n} \leq v_{i_m i_n} \quad \textit{for all } i \in V_R; m,n \in C_i; s,t \in V_R. \tag{6.6}$$

$$v_{i_m i_n} \in \{0,1\} \quad \textit{for all } i \in V_R; m,n \in C_i; s,t \in V_R. \tag{6.7}$$

The FRCSTP helps illustrate an important difference between typical network design problems in the literature and the same network design problem with reload costs. We can model the MSTP by using a single-source model. We cannot do that for the FRCSTP as the single-source model would not allow us to model reload costs between some pairs of edges diverging from the same node. Consider a (directed) solution composed of the arcs $\{(1,2),(2,3),(2,4)\}$. Assume that the colors of the three arcs are different, and node 1 is the source. The given model permits us to model reload transitions associated with the pairs of edges $\{\{1,2\},\{2,3\}\}$ and $\{\{1,2\},\{2,4\}\}$. However, the reload transition associated with the pair of edges $\{\{2,3\},\{2,4\}\}$ is not considered in a single-source model with node 1 as the source.

However, one could consider a single-source version of the FRCSTP. The transformation in Section 2 holds in this case as well and so this single-source (all destinations) version of the FRCSTP is NP-complete. A similar modeling change as in (6.5), (6.6), (6.7) applied to the directed model of Section 6.1 provides a directed model for the single-source variant of the FRCSTP.

The variables $v_{i_m i_n}$ allow us to model variants of the problem (both in the fixed and variable reload cost setting) with a limit on the total number of allowed reloads, or a limit on the number of allowed reloads per node. The first constraint (6.8) shown below imposes a limit of $K$ on the number of reloads in the given solution, while the subsequent constraint

(6.9) imposes a limit of $K_i$ on the number of reloads at node $i$ in the given solution.

$$\sum_{i\in V_R}\sum_{m,n\in C_i} v_{i_m i_n} \le K \tag{6.8}$$

$$\sum_{m,n\in C_i} v_{i_m i_n} \le K_i \quad \textit{for all } i \in V_R. \tag{6.9}$$

### 6.3. Reload Hop and Diameter Constraints

Hop constraints [5] bounding the number of reloads permitted along a path can be modeled by adding the constraints

$$\sum_{j\in V_R}\sum_{m,n\in C_j} h^{st}_{j_n j_m} \le H \quad \textit{for all } s,t \in V_R, s < t.$$

A generalization of hop constraints is obtained by using "path-length" constraints (see [12]) as follows

$$\sum_{j\in V_R}\sum_{m,n\in C_j} R^j_{nm} h^{st}_{j_n j_m} \le H \quad \textit{for all } s,t \in V_R, s < t.$$

In this general version, we are restricting the "length" in terms of reloads between any pair of nodes $s$ and $t$ of the graph. We note that all of the problems described so far in this subsection remain NP-complete, as the problem without the reload limit, hop or path-length constraints is NP-complete.

We can use these concepts in a slightly different way. Instead of adding path-length constraints, we can seek spanning trees where we want to minimize the length of the longest reload cost path (i.e., with a minimax objective). This is precisely the first reload cost problem introduced by Wirth and Steffan [17], called the minimum diameter spanning tree problem with reload costs, which is NP-complete. To model this, we add the variable $v$, the constraints

$$\sum_{j\in V_R}\sum_{m,n\in C_j} R^j_{nm} h^{st}_{j_n j_m} \le v \quad \textit{for all } s,t \in V_R, s < t$$

and replace the objective function with the simpler objective min $v$.

We also note that these hop and path-length constraints and the minmax objective variants can also be considered in terms of the single-source variants discussed in the previous section, and remain NP-complete.

### 6.4. Modeling Routing and Fixed Costs

As noted in the introduction models using reload costs may also involve the more conventional (edge) fixed and variable (flow) costs. Let $A_{\{i,j\}}$ be the installation cost of edge $\{i,j\} \in E_R$ and let $B_{ij}$ be the per unit of flow cost associated with arc $(i,j) \in A_R$. Then, we can model these costs by using the COLORED-GRAPH model with the more complex objective function

$$\min \sum_{s,t\in V_R: s<t} d_{st}\left(\sum_{j\in V_R}\sum_{m,n\in C_j} R^j_{nm} h^{st}_{j_n j_m} + \sum_{k\in C}\sum_{(i,j)\in A_k} B_{ij} g^{st}_{i_k j_k}\right) + \sum_{\{i,j\}\in E_R} A_{\{i,j\}} u_{\{i,j\}}.$$

### 6.5. Steiner Tree Variants

In this section, we will discuss a Steiner variant of the single-source RCSTP discussed in Section 6.1 and a Steiner version of the RCSTP. Consider a set $R$ of nodes, $R \subseteq V_R$, of so-called required nodes. For simplicity, we assume that node 1 is in $R$. In the single-source version of the problem, we define the reload cost of the tree as the sum of the reloads for all pairs $(1,p)$ with $p$ in $R$. We first note that the Steiner single-source RCTSP remains NP-complete. The transformation of Section 2 can be used again, with the set of $R$ nodes as node $r$ together with $C_1, C_2, \ldots, C_k$. This Steiner version of the single-source RCSTP is easy to model. We simply use the directed model presented in Section 6.1 with the following modifications: (i) in the constraints, the index $p$ ranges only in $R\backslash\{1\}$ and for all $j \in V_R\backslash(\{1\} \cup R)$, (ii) the constraints $\sum_{i\in V_R} x_{ij} = 1$ are replaced by $\sum_{i\in V_R} x_{ij} \le 1$.

In the multi-source Steiner version of the RCSTP, the reload cost of the tree is defined as the sum of the reloads for all pairs $(p,q)$ with $p$,$q$ in $R$. In this case, it is not as easy to adapt the COLORED-GRAPH model of Section 4, because the cardinality constraint giving the number of edges in the solution is necessary to guarantee that the solution is a tree. However, we can use well-known ideas from the literature on Steiner tree formulations and introduce binary node variables $z_i$ indicating whether node $i$ is in the solution together with the following constraints (see [9], for instance)

$$\sum_{e\in E_R} u_e = \sum_{i\in V_R} z_i - 1,$$

$$\sum_{e\in E(S)} u_e \le |S\cap R| + \sum_{i\in S\backslash R} z_i - 1 \ \textit{for all } S \subset V : S\cap R \neq \emptyset,$$

$$\sum_{e\in E(S)} u_e \le \sum_{i\in S} z_i - z_j \quad \textit{for all } S \subset V, j \in S : S\cap R = \emptyset,$$

$$z_i = 1 \quad \textit{for all } i \in R.$$

Then, by restricting the range of the commodity indices $(s,t)$ to $s,t$ in $R$, we obtain a valid formulation for the problem.

Our discussion of Steiner variants of the RCSTP has considered variable reload costs. Note that, similar Steiner variants can be formulated and modeled for fixed reload costs. It is easy to see that the Steiner variant of the single-source FRCSTP is NP-complete, using the identical transformation as for the Steiner RCSTP. Furthermore, Steiner versions of (both fixed and variable cost) reload cost tree problems with additional reload limit, hop or path-length constraints can be formulated and modeled, and remain NP-complete.

TABLE 1. Comparing the LINE-GRAPH and the COLORED-GRAPH models.

| Problem set | L-G LP time (s) | L-G IP time (s) | C-G LP time (s) | C-G IP time (s) |
|---|---|---|---|---|
| N10E25C3 | 0.359 | 3.403 | 0.066 | 1.266 |
| N10E25C5 | 0.272 | 0.850 | 0.075 | 0.353 |
| N10E25C7 | 0.400 | 2.368 | 0.094 | 0.844 |
| N15E50C3 | 18.328 | 9008.250 | 2.913 | 3589.244 |
| N15E50C5 | 4.956 | 334.456 | 1.256 | 88.644 |
| N15E50C7 | 8.353 | 241.575 | 2.194 | 135.725 |
| N15E50C9 | 9.647 | 381.928 | 2.841 | 180.794 |
| N20E100C3 | 1054.191 | 11524.241 | 86.372 | 728.672 |
| N20E100C5 | 389.734 | 14403.853 | 68.056 | 14400.316 |
| N20E100C7 | 101.209 | 12461.706 | 28.956 | 7508.584 |
| N20E100C9 | 105.375 | 8694.272 | 32.222 | 8695.159 |

### 6.6. *Uncapacitated Network Design*

Another interesting problem that appears in the telecommunications and transportation industries is the Uncapacitated Network Design Problem (UNDP). In this problem, there are costs associated with the routing of flow on the edges of the network constructed, but there are also fixed costs associated with the selection of the edges as described in the previous subsection. Quoting Balakrishnan et al. [3], the problem is "deceptively simple" and contains other well-known problems as special cases, like the Steiner tree problem, the uncapacitated facility location and the traveling salesman problem. In this section, we briefly discuss the Reload Cost Uncapacitated Network Design Problem (RCUNDP).

The RCUNDP can be formulated with the COLORED-GRAPH model by removing the constraint that restricts the number of edges and by considering the more complex objective function with routing reload costs as well as fixed reload costs

$$\min \sum_{s,t\in V_R} d_{st} \sum_{j\in V_R} \sum_{m,n\in C_j} R1^j_{nm} h^{st}_{j_n j_m} + \sum_{j\in V_R} \sum_{m,n\in C_j} R2^j_{nm} v_{j_n j_m}.$$

In the objective above, we have both variable reload costs denoted by $R1$ and fixed reload costs denoted by $R2$. As there is no constraint in this model that restricts the number of edges to be selected, the network to be designed is not restricted to the set of spanning trees. Naturally, this model can be strengthened with the same constraints we used to improve the original reload cost minimum spanning tree problem presented and discussed in Section 5.

It is important to point out an important distinction between fixed reload costs in the tree and network design setting. Suppose we have demands between all pairs of nodes. In the tree setting, if there are any two edges of different colors (say $n$ and $m$) in the tree that are adjacent at a node $i$ the fixed reload cost $R^i_{nm}$ is incurred. In the network design setting, because we no longer require a tree, it is possible for two edges of different colors to be adjacent at a node without any flow being sent from one of them directly to the other. Thus, two possible interpretations may be given to fixed reload costs in the network design setting. In the first interpretation, the fixed reload cost $R^i_{nm}$ for changing from color $n$ to color $m$ at node $i$ is incurred only if there is a commodity that goes from color $n$ to color $m$ at node $i$. In the second interpretation, the fixed reload cost $R^i_{nm}$ is incurred regardless of whether there is a commodity that changes from color $n$ to color $m$ at node $i$. We believe the first interpretation of fixed reload costs is more reasonable in a practical setting, and our COLORED-GRAPH model is for the RCUNDP under the first interpretation of fixed reload costs (the complexity of this problem is open). Under the second interpretation of fixed reload costs (which is somewhat contrived) for the RCUNDP the problem is NP-complete (again using the transformation of Section 2), but the problem is not so straightforward to model.

We note that a more general version of the RCUNDP with both reload (fixed and variable) costs and normal (routing and fixed edge) costs can be modeled by using a slightly more general objective obtained by combining the objective above with the objective given in Subsection 6.4. Furthermore, a single-source variant could be formulated as in Section 6.1, and hop, diameter, and path-length constraints can also be considered as in Section 6.3. To conclude our discussion on different variants of reload cost network design problems, we remind the reader that by combining the specifications discussed in this section many additional variants of reload cost network design problems may be formulated.

## 7. COMPUTATIONAL EXPERIMENTS

In this section, we present computational experiments to evaluate the quality of the proposed models (specifically the colored graph formulation) to solve instances of the RCSTP as well as its single-source multidestination version. Our experiments will consist of three main parts. Our first experiment (see Table 1) focused on comparing the linear programming relaxations of the non-enhanced version of the two models, for the RCSTP, to emphasize the advantages of using the COLORED-GRAPH model instead of the LINE-GRAPH model. Our second set of experiments (see Tables 2–5) was

TABLE 2. Comparing the gaps for the enhanced versions of the C-G model.

| Problem set | Set 1 LP-IP gap (%) | Set 2 LP-IP gap (%) | Set 3 LP-IP gap (%) |
|---|---|---|---|
| N10E25C3 | 19.11 | 5.42 | 4.13 |
| N10E25C5 | 9.25 | 1.33 | 0.00 |
| N10E25C7 | 7.44 | 3.80 | 1.81 |
| N15E50C3 | 40.06 | 19.89 | 12.09 |
| N15E50C5 | 11.42 | 3.91 | 2.59 |
| N15E50C7 | 14.03 | 6.45 | 2.59 |
| N15E50C9 | 10.22 | 5.98 | 3.24 |
| N20E100C3 | 0.00 | 0.00 | |
| N20E100C5 | 72.56 | 25.47 | |
| N20E100C7 | 35.65 | 7.55 | |
| N20E100C9 | 18.12 | 8.76 | |

TABLE 3. Comparing the CPU times for the enhanced versions of the C-G model.

| | Set 1 | | Set 2 | | Set 3 | |
|---|---|---|---|---|---|---|
| Problem set | LP time (s) | IP time (s) | LP time (s) | IP time (s) | LP time (s) | IP time (s) |
| N10E25C3 | 0.066 | 1.266 | 0.122 | 0.753 | 1.334 | 59.219 |
| N10E25C5 | 0.075 | 0.353 | 0.103 | 0.350 | 1.366 | 1.794 |
| N10E25C7 | 0.094 | 0.844 | 0.159 | 0.922 | 2.447 | 8.078 |
| N15E50C3 | 2.913 | 3589.244 | 7.563 | 131.081 | 2830.966 | 8964.506 |
| N15E50C5 | 1.256 | 88.644 | 3.006 | 62.544 | 219.972 | 5773.903 |
| N15E50C7 | 2.194 | 135.725 | 5.319 | 65.834 | 3761.725 | 9872.688 |
| N15E50C9 | 2.841 | 180.794 | 6.141 | 101.938 | 2860.509 | 9223.506 |
| N20E100C3 | 86.372 | 728.672 | 247.656 | 519.734 | | |
| N20E100C5 | 68.056 | 14400.316 | 208.463 | 6150.091 | | |
| N20E100C7 | 28.956 | 7508.584 | 113.556 | 2521.200 | | |
| N20E100C9 | 32.222 | 8695.159 | 99.041 | 7170.775 | | |

to assess the effectiveness of the three sets of forcing constraints (Set 1, Set 2, and Set 3) for the COLORED-GRAPH model. Recall that Set 3 contains Set 2, and Set 2 contains Set 1. Our third set of experiments (see Tables 6 and 7) focus on the single-source version of the RCSTP and the directed model given in Section 6.1.

Our computational study was conducted on a set of randomly generated problem instances with varying characteristics. All graphs were generated on a $100 \times 100$ grid. The endpoints for the edges were randomly picked among the nodes in the graphs and the color for each edge was drawn from a discrete uniform distribution. In the tables that follow, we identify each set as "N$x$E$y$C$z$" where $x$ denotes the number of nodes, $y$ the number of edges, and $z$ the number of colors. Each set consists of five problem instances. All computational results were conducted on a Dell Optiplex 740 with an AMD Athlon 64 X2 5000+ dual core processor with 3 GB of RAM running Microsoft Windows XP. The formulations were solved using ILOG CPLEX 11.0. All instances were limited to a maximum of 4 hours of running time.

In our first set of experiments, we compared the LINE-GRAPH (L-G) model with the COLORED-GRAPH (C-G) model. We considered unit reload costs and unit demands. In other words, all reload costs between all combinations of colors were set equal to 1; and the demand between all pairs of nodes was also set equal to 1. For these RCSTP instances, we generated problems where the number of nodes and edges in the graph varied between 10 and 20 and between 25 and 100, respectively. Also, we increased the number of different colors in the graph from 3 to 9.

Table 1 presents the results. The first column identifies the instance set; the second and the third give the CPU times (in s) for solving the LP relaxation and for obtaining the optimal integer solution with the L-G model. The fourth and fifth columns give similar information with respect to the C-G model. The results indicate and confirm that the C-G model is, in general, much faster than the L-G model, both in terms of solving the corresponding LP relaxations as well as for obtaining the IP optimal solutions. Thus, for our subsequent

TABLE 4. Nonunit reload cost instances. Comparing the gaps for the enhanced versions of the C-G model with Set 1 and Set 2.

| Problem set | Set 1 LP-IP gap (%) | Set 2 LP-IP gap (%) |
|---|---|---|
| N20E100C3 | 0.00 | 0.00 |
| N20E100C5 | 30.58 | 4.15 |
| N20E100C7 | 31.93 | 3.40 |
| N20E100C9 | 21.59 | 4.88 |

TABLE 5. Nonunit reload cost instances. Comparing the CPU times for the enhanced versions of the C-G model with Set 1 and Set 2.

| | Set 1 | | Set 2 | |
|---|---|---|---|---|
| Problem set | LP time (s) | IP time (s) | LP time (s) | IP time (s) |
| N20E100C3 | 89.322 | 993.284 | 272.003 | 645.272 |
| N20E100C5 | 51.875 | 5824.913 | 128.438 | 498.338 |
| N20E100C7 | 38.884 | 3110.506 | 98.981 | 460.764 |
| N20E100C9 | 33.331 | 5547.013 | 90.263 | 960.209 |

TABLE 6. Evaluating the directed model for solving the single-source version of the RCSTP.

| Problem set | LP-IP gap (%) | LP time (s) | IP time (s) |
|---|---|---|---|
| N10E25C3 | 0.00 | 0.016 | 0.034 |
| N10E25C5 | 0.00 | 0.006 | 0.028 |
| N10E25C7 | 0.00 | 0.019 | 0.028 |
| N15E50C3 | 0.00 | 0.034 | 0.056 |
| N15E50C5 | 0.00 | 0.038 | 0.075 |
| N15E50C7 | 0.00 | 0.047 | 0.078 |
| N15E50C9 | 0.00 | 0.056 | 0.075 |
| N20E100C3 | 0.00 | 0.141 | 0.175 |
| N20E100C5 | 0.00 | 0.134 | 0.194 |
| N20E100C7 | 0.00 | 0.175 | 0.338 |
| N20E100C9 | 0.00 | 0.169 | 0.263 |
| N50E300C3 | 0.00 | 98.963 | 142.228 |
| N50E300C5 | 0.00 | 11.447 | 23.103 |
| N50E300C7 | 10.00 | 19.494 | 405.003 |
| N50E300C9 | 10.00 | 5.041 | 18.881 |

TABLE 7. Evaluating the directed model for solving the single-source version of the RCSTP with nonunit reload costs.

| Problem set | LP-IP gap (%) | LP time (s) | IP time (s) |
|---|---|---|---|
| N20E100C3 | 0.00 | 0.125 | 0.141 |
| N20E100C5 | 0.00 | 0.134 | 0.263 |
| N20E100C7 | 4.00 | 0.181 | 0.347 |
| N20E100C9 | 0.00 | 0.197 | 0.386 |
| N50E300C3 | 0.00 | 183.763 | 213.250 |
| N50E300C5 | 0.00 | 16.650 | 90.156 |
| N50E300C7 | 10.00 | 15.047 | 197.503 |
| N50E300C9 | 6.40 | 8.603 | 22.456 |

experiments, we used the C-G model. We should note that in the 4 hour time limit, it was not possible to solve the IP models for problems with more than 20 nodes. This provides an inkling that the RCSTP is a very computationally challenging problem.

Our second set of experiments assessed the effectiveness of the three sets of forcing constraints. Tables 2 and 3 present the results for unit reload instances. These instances are identical to those in our first set of experiments (i.e., Table 1). As before, the first column identifies the instance set. The second, third, and fourth columns present the average percentage LP-IP gap calculated as the difference between the values of the optimal integer solution and the LP bound divided by the value of the optimal integer solution[2] for the three models, the C-G model, the C-G model with Set 2, and the C-G model with Set 3. Table 3 gives similar information with respect to the corresponding CPU times.

The results taken with Set 1 indicate that with exception of a few classes of instances, the gaps are quite large. In general, the gaps get better when the number of colors increase. Exceptions to this are obtained in some of the instances with the smallest number of colors where the reported gap was equal to zero. The reason for this is that these instances were easy to solve because it was possible to find a solution with only one color for the edge set in the spanning tree.

From the information presented in these tables, it is also clear that the forcing constraints associated with Sets and 2 and 3 can provide substantial improvements over the original model. It is interesting to see that Set 2 alone seems to be responsible for the major improvements. However, notice that these improvements come with a penalty in terms of the computation time required which becomes more noticeable for the larger sized instances. These computational penalties are a direct consequence of the increased number of constraints in the various formulations. For example, consider one of the problem instances with 15 nodes and 50 edges. For this problem, the total number of constraints was equal to 9,283. With Set 2, the total number of constraints is 10,259. With Set 3, we consider triplets of nodes, and as a result the total number of constraints goes up to 124,398. From this example, it should be clear that the significant increase in the size of the formulation with Set 3 forcing constraints is causing the very large computation times observed. Based on the improvement percentages and computation times presented in Tables 2 and 3 for problems with up to 15 nodes, one can argue that the extra running time associated with Set 3 outweighs the benefits introduced. Consequently, we have not tested constraint Set 3 on problem instances with more than 15 nodes. (Especially, because even the LP relaxation of the models did not solve to optimality within 4 h of CPU time for 20 node instances with Set 3.) Although the CPU times associated with solving the linear programming relaxation of the model with Set 2 are substantially greater than the CPU times needed by the linear programming relaxation with Set 1, the results also show that for solving the instances tested, the improvement in the linear programming bounds produced by the model with Set 2 more than compensates for the extra time. Finally, we observe that although the LP relaxations of the C-G model with Set 2 take greater time to solve than the LP relaxations of the C-G model with Set 1, the reverse is true with the IP models. The running time for the IP of the C-G model with Set 2 is usually less than the running time of the IP of the C-G model with Set 1. In fact within the 4 hour time limit, the IP of the C-G model with Set 1 only solved 45 out of the 55 test instances to optimality, whereas the IP of the C-G model with Set 2 solved 54 out of the 55 test instances to optimality. This also suggests that the C-G model with Set 2 as most appropriate for solving the IP.

We also tested instances with non-unit reload costs. The non-unit reload problems are generated in exactly the same way as the unit reload cost problem with respect to the creation of the node coordinates and the edges in the graph. In fact, the graphs are identical. The only difference is that for each pair of colors in the graph, we generate an integer reload cost whose value is drawn from a discrete uniform distribution in the range $[1, 10]$. We have restricted the presentation of results to 20 node instances with the C-G model using Set 1 and Set 2 forcing constraints. We ignore Set 3 for the reasons explained above.

The non-unit reload results are interesting because, in a certain way, they confirm the behavior of the unit reload cost experiments. For these instances, all instances have been solved to optimality by the C-G model using Set 2. However, only 14 out of the 20 test instances have been solved by using Set 1. Tables 4 and 5 provide a summary of our computational experiments and are similar to Tables 2 and 3.

Our third set of experiments focus on the single-source version of the RCSTP. For the single-source version of the problem, we consider the same set of instances as before, except we select node 1 as the source. We also generate larger instances with 50 nodes and 300 edges. In Tables 6 and 7, we present results of our computational experience with the directed model given in Section 6.1 for the single-source version of the problem. Table 6 refers to instances

[2] As we compare the values of different linear programming relaxations, we keep the denominator fixed as the value of the optimal integer solution. In other words, the LP-IP Gap is simply 1 minus the ratio of the optimal LP solution to the optimal integer solution. If the optimal integer solution is not available (this occurred in only one instance over all the problems tested), we use the best upper bound.

with unit reload costs, and Table 7 refers to instances with non-unit reload costs.

The results show that, as expected, the single-source version of the problem is much easier to solve. This is explained by the fact that for a given number of nodes, edges, and colors, the single-source multidestination model has many fewer commodities (and thus, many fewer variables and constraints) than the multi-source multidestination model of the RCSTP. Another explanation for these results, namely the 0% gaps for most of the cases tested, is the fact that the directed model implies all the forcing constraints of the corresponding undirected model.

The results are similar for the cases with non-unit reloads. In fact, the main conclusion appears to be that using non-unit reload costs instead of unit reload costs does not appear to have an effect on the performance of the methods. The results also show that the evaluation of the model starts to become interesting only for the 50 node instances (in the sense that these problems are harder to solve to integer optimality for the LP relaxations). On the other hand, some experiments performed with larger instances show that the behavior of the proposed models starts to deteriorate for even larger instances.

## 8. SUMMARY AND CONCLUSIONS

In this article, we motivated the notion of reload costs that occurs in telecommunications, transportation, and energy networks. Despite their wide applicability reload costs have only recently been treated in the literature, first in [16, 17], and more recently in [1, 6–8, 11]. In this article, we looked at a wide spectrum of reload cost tree problems, discussed their complexity, and developed strong models for these problems. We developed models using two types of extended graphs—a directed line graph and a colored graph—that allow us to assign reload costs to specific edges rather than pairs of edges.

Our computational experiment focused on the strength of the linear programming relaxations of the RCSTP that motivated this work. Given that this is the first article to consider MIP models for the reload cost problems, we focused on problems solely with reload costs. In the all pairs model, our strongest model results in a linear programming relaxation that is on average only 3.78% from optimality (over 35 test instances). This model grows rapidly in size and is not computationally viable for problems with more than 15 nodes. Our 2nd strongest model results in a linear programming relaxation that is on average 11.51% from optimality (over 75 test instances) and is the most successful one in solving the test instances to optimality. In the single-source model over 115 instances, the average LP-IP gap of our directed formulation was only 1.76% from optimality, and we were able to solve problems with up to 50 nodes to optimality.

We hope this comprehensive introduction and discussion of reload cost tree and network design problems will spark an interest in the network design community to further study reload cost network design problems. In this regard, we have the following two suggestions for future work. First, given that the models on the colored graph rapidly become large, a natural thought might be to think about a path-based model (and the use of branch-and-price) for reload cost problems in which the pricing problem is solved on the colored graph. The advantage of this approach is that it might possibly result in a faster solution procedure, as the large colored graph is implicitly considered in the model. Second, we believe it is necessary to assess the application of heuristics for "normal cost" variants of spanning tree problems to "reload cost" variants, and to develop high-quality heuristics for reload costs problems.

## REFERENCES


[1] E. Amaldi, G. Galbiati, and F. Maffioli, On minimum reload cost paths, tours and flows, Networks 57 (2011), 254–260.

[2] A. Assad and W. Xu, The quadratic minimum spanning tree problem, Nav Res Logist 39 (1992), 339–417.

[3] A. Balakrishnan, T.L. Magnanti, and R.T. Wong, A dual-ascent procedure for large scale uncapacitated network design, Oper Res 37 (1989), 716–740.

[4] W. Cook, W. Cunningham, W. Pulleyblank, and A. Schrijver, Combinatorial optimization, Wiley, New York, 1998.

[5] G. Dahl, L. Gouveia, and C. Requejo, "On formulations and methods for the hop-constrained minimum spanning tree problem," Handbook of optimization in telecommunications, M.G.C. Resende and P.M. Pardalos (Editors), Springer, New York, 2006, pp. 493–516, Chapter 19.

[6] G. Galbiati, The complexity of a minimum reload cost diameter problem, Discrete Appl Math 156 (2008), 3494–3497.

[7] I. Gamvros, Satellite network design, optimization, and management, Ph.D. Thesis, University of Maryland, College Park, 2006.

[8] I. Gamvros, L. Gouveia, and S. Raghavan, "Reload cost trees and network design," Proceedings of the International Network Optimization Conference, Spa, Belgium, 2007.

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

[10] M. Gondran and M. Minoux, Graphs and algorithms, Wiley, New York, 1984.

[11] L. Gourvès, A. Lyra, C. Martinhon, and J. Monnot, "The minimum reload s-t path/trail/walk/ problems," Proceedings of the 35th Conference on Current Trends in Theory and Practice of Computer Science, Lecture Notes in Computer Science, Vol. 5404, Springer, 2009, pp. 621–632.

[12] L. Gouveia, A. Paias, and D. Sharma, Modeling and solving the rooted distance-constrained minimum spanning tree problem, Comput Oper Res 35 (2008), 600–613.

[13] G.Y. Handler, Minimax location of a facility in an undirected tree graph, Transport Sci 7 (1973), 287–293.

[14] T.C. Hu, Optimum communication spanning trees, SIAM J Comput 3 (1979), 188–195.

[15] T.L. Magnanti and L.A. Wolsey, "Optimal trees," Handbooks in operations research and management science, M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser (Editors), Vol. 7, Elsevier, Amsterdam, 1995, pp. 503–615, Chapter 9.

[16] H.C. Wirth, Multicriteria approximation of network design and network upgrade problems, Ph.D. Thesis, Universität Würzburg, 2001.

[17] H.C. Wirth and J. Steffan, Reload cost problems: Minimum diameter spanning tree, Discrete Appl Math 113 (2001), 73–85.