The Two Level Network Design Problem with Transition Facilities

Stefan Gollowitzer[a], Luis Gouveia[b], Ivana Ljubić[a,1,*]

[a] *Department of Statistics and Operations Research, University of Vienna, Brünnerstraße 72, A-1210 Vienna, Austria*

[b] *DEIO/CIO, Faculdade de Ciências da Universidade de Lisboa, Bloco C2, Campo Grande, 1749-016 Lisboa, Portugal*

Abstract

We consider a new combinatorial optimization problem that combines network design and facility location aspects. Given a graph with two types of customers and two types of technologies that can be installed on the edges, the objective is to find a minimum cost subtree connecting all customers while the primary customers are served by a primary subtree that is embedded into the secondary subtree. In addition, besides fixed link installation costs, facility opening costs, associated to each node where primary and secondary subtree connect, have to be paid. The problem is called the *Two Level Network Design Problem with Transition Facilities* (TLNDF).

We first model the problem on an extended graph where an additional set of arcs corresponds to the installation of node facilities and propose a cut set based model for the TLNDF that is defined on this extended graph. We present several theoretical results relating families of cut set inequalities on the extended graph with subfamilies of cut set inequalities on the original graph. We then show how a standard multi-commodity flow model defined on the original graph can be strengthened using disaggregation "by technology". We prove that the disaggregated compact formulation on the original graph provides the same lower bound as the cut set formulation on the extended graph.

To solve the TLNDF to optimality, a branch-and-cut algorithm has been developed. The performance of this algorithm is improved by separating subfamilies of cut set inequalities on the original graph. Our computational study confirms the efficiency and applicability of the new approach.

Keywords: OR in telecommunications, Integer programming, Linear programming relaxations, Hierarchical network design, Connected facility location, Steiner trees

*Corresponding author. Tel: +43-1-4277-38661 Fax: +43-1-4277-38699

Email addresses: stefan.gollowitzer@univie.ac.at (Stefan Gollowitzer), legouveia@fc.ul.pt (Luis Gouveia), ivana.ljubic@univie.ac.at (Ivana Ljubić)

[1]Supported by the APART Fellowship of the Austrian Academy of Sciences (OEAW).

1. Introduction

The *Multi-Level Network Design Problem* (MLND) has been originally defined by Balakrishnan et al. [1]: We are given an undirected graph with a set of nodes partitioned into L levels and a set of edges such that along each edge one of L different technologies can be installed, with higher grade technologies inducing higher fixed costs. The goal is to find a (spanning) subtree and decide which technology to install along each edge, so that all customers at level ℓ can communicate with each other along a path using technology of grade ℓ or higher. We extend the definition of the MLND by introducing the fixed costs for *transition nodes*, i.e., the nodes where a change of technology takes place, and by considering so-called Steiner nodes which are nodes that need not be included in the solution. In this problem, which we denote by *Multi-Level Network Design Problem with Transition Facilities*, the overall goal is to find a MLND subtree that minimizes the sum of fixed edge and facility installation costs. In this paper we study the case with $L = 2$, which we will denote by the *Two Level Network Design Problem with Transition Facilities* (TLNDF).

TLNDF arises in the topological design of hierarchical communication, transportation, and electric power distribution networks. One of the most important applications of TLNDF is in the context of telecommunication networks, where networks with two cable technologies, fiber optic and copper, are built. Telecommunication companies distinguish between *primary* and *secondary customers*. The switching centers, important infrastructure nodes and small businesses are considered as primary customers (i.e., those to be served by fiber optic connections). Single households are not considered as being consumers of a high potential and hence they only need to be supplied using copper cables. The secondary technology is much cheaper, but the guaranteed quality of the connections and bandwidth is significantly below the quality provided by the primary technology. The goal is to build a network (with tree topology) such that there is a fiber optic connection between each primary customer and a designated root node (e.g., a central office), and each secondary customer is connected to the root along a path using either of the two technologies. Typically, at transition nodes, expensive switching devices need to be installed to transmit the electrical into optical signal, and the respective purchasing and equipment operating costs are not negligible. This particular application involves two new features that have not been considered in the previous literature (see, e.g., Balakrishnan et al. [2], Duin and Volgenant [9]). First, the application considers additional *transition costs* due to the presence of two technologies on the network. Second, in practical applications nodes like street intersections need to be considered as well, i.e., we need to allow that the set of primary and secondary customers is a real subset of the set of nodes, and a subset of remaining nodes may be a part of the solution, if it helps in establishing a cheaper connection. Those remaining nodes will be referred to as *Steiner nodes*.

More formally, the problem can be defined as follows:

Definition 1 (TLNDF)**.** We are given an undirected graph $G = (V, E)$ with a set of customers $R \subseteq V$. To each edge $e \in E$ we associate two installation costs, $c_e^1 \geq c_e^2 \geq 0$. These correspond to the *primary* and *secondary technology*, respectively. The set of customers, R, is partitioned into the sets of *primary* and *secondary customers* P and S, respectively

($P \cap S = \emptyset$, $P \cup S = R$). We are also given a root node $r \in V$, otherwise we choose one of the primary customers as such. To each node $i \in V$ we associate *facility opening cost* $d_i \geq 0$.

Our goal is to determine a *subtree* T (built of a set of primary and secondary edges, T_1 and T_2, respectively) with the set F of transition nodes (i.e., nodes that are adjacent to edges from both T_1 and T_2), satisfying the following properties:

(P) Each *primary node* in P is connected to the root node by a path that consists of T_1 edges only,

(S) each *secondary node* in S is connected to the root by a path consisting of edges from $T_1 \cup T_2$,

(F) facilities need to be open at each transition node $i \in F$ and

(M) the sum of fixed edge and facility installation costs

$$\sum_{e \in T_1} c_e^1 + \sum_{e \in T_2} c_e^2 + \sum_{i \in F} d_i$$

is minimized.

The following observation can be made about optimal solutions of this problem: i) Since $c_e^1 \geq c_e^2$ for all $e \in E$, there always exist an optimal solution in which the subgraph induced by T_1 is a rooted subtree of T (*primary subtree*) and the subgraph induced by T_2 is a forest (a union of *secondary subtrees*) attached to it. ii) If facility opening costs are uniform for all nodes, leaves of the primary subtree are nodes from $P \cup S$. In addition, any leaf of the primary subtree that has a secondary subtree attached to it will be a primary node. iii) Otherwise, if facility opening costs are location-dependent, placing facilities at locations of Steiner nodes or secondary customers may provide cheaper solutions, i.e., a secondary subtree can be attached to any node from V, and henceforth, a leaf of the primary subtree can be any node from V.

Notice also that our general definition covers the case in which potential facility locations are a true subset of V (which can be modeled by setting $d_i := \infty$ for the non-facility locations).

This important problem generalizes problems with tree-star and star-tree topologies including connected facility location, hierarchical network design, Steiner trees and uncapacitated facility location.

Overview of the paper. In Section 2 we model the TLNDF in an extended graph, where an additional set of arcs corresponds to the installation of node facilities. We present several theoretical results relating families of cut set inequalities on the extended graph with special families of cut set inequalities on the original graph. Cut set inequalities on the extended graph can be separated in polynomial time using maximum flow algorithms. We also study special classes of cut set inequalities that are obtained by projecting subsets of constraints of the extended graph formulation on the original graph. In Section 4 we show how these can

be separated efficiently on the original graph extended by a single node. Our computational study, reported in Section 5, confirms the efficiency and applicability of these separation procedures.

It is quite straightforward to model the TLNDF on the original graph with a standard multi-commodity flow model. The linear programming (LP) relaxation of the layered graph formulation is easily shown to dominate the linear programming relaxation of this formulation. This dominance result is also known from similar problems (see, e.g., [4]). In this paper (cf. Section 3), we show that by disaggregating the previous multi-commodity flow formulation *by technology* we obtain a formulation on the original graph that provides the same lower bound as the cut set formulation on the extended graph. Our result also extends to the two level network design problem *without* transition nodes and, as far as we know, this is the first time a compact formulation on the original graph is given that provides the same LP bound as the extended graph cut formulation.
Preliminary results of this work appeared in Gollowitzer et al. [10].

1.1. Literature Review

The concept of two level network design problems (more precisely, *two-level spanning trees*) has been developed in the 80's and early 90's.

Hierarchical Network Design (HNDP). The *hierarchical network design* problem, in which $R = V$ and $|P| = 2$, was the "initial" variant of the TLND introduced by Current et al. [7]. The authors propose an integer programming model based on subtour elimination constraints and a heuristic for the problem. Later, Duin and Volgenant [8] prove structural properties of HNDP that enable reductions of the problem graph and elimination of variables from an integer programming model. Pirkul et al. [21] derive a heuristic based upon a Lagrangian relaxation of a flow-based formulation for the problem. A dynamic programming procedure that finds suboptimal solutions has been proposed by Sancho [23]. Recently, the problem has been studied by Obreque et al. [20], who propose a branch-and-cut algorithm for this problem.

Two Level Network Design (TLND). This problem, a generalization of HNDP in which $|P| \geq 2$ and $R = V$, has been introduced by Duin and Volgenant [9]. Balakrishnan et al. [2] proposed several network flow based models for this latter problem setting and have compared the linear programming bounds of the proposed formulations. The same authors also propose a composite heuristic that provides an approximation ratio of 4/3 if the embedded Steiner tree is solved to optimality and $c_e^1/c_e^2 = q > 1$ for all $e \in E$, and a ratio of $\frac{4}{4-\rho}$ if the Steiner tree problem is solved with an approximation ratio of $\rho < 2$. For non-proportional edge costs, this ratio becomes $\rho + 1$. Balakrishnan et al. [1] tested a dual ascent method derived on the strongest formulation proposed in [2] (which is a directed multi-commodity flow formulation on G, cf. Section 3). Gouveia and Telhada [14] propose another formulation in which the primary subtree is modeled as a directed arborescence embedded into the secondary spanning arborescence. The authors propose to solve the problem using a Lagrangian relaxation based method. In a later paper, Gouveia and Telhada [15] improve this formulation by using a

"reformulation by intersection" concept to derive a new compact formulation whose lower bounds are at least as strong as the strongest ones proposed in [2]. In a recent work Chopra and Tsai [4] develop a branch-and-cut approach for a generalization of the TLND with more than 2 levels.

Hierarchical Network Design with Transshipment Facilities (HNDF). The HNDP was introduced by [5]. In this problem additional transshipment costs need to be paid for each node of the primary path whenever a change of technology takes place. The main difference between the HNDF and the TLNDF, besides the restriction $|P| = 2$ and $R = V$, is that secondary nodes included in the primary path are not considered as "served", and therefore they also need to be connected by a (possibly empty) path to a transshipment facility. In addition, the union of primary and secondary edge sets may form a cycle, i.e., the optimal solution is not necessarily a tree.
Current [5] proposes a heuristic approach in which, for a given root r and terminal node t, K shortest paths are calculated. For each of these paths an auxiliary problem is constructed, in which the nodes of the path are connected to a dummy root node by edges whose weights are set to their facility opening costs. The edges of the path are deleted and in the graph obtained by this procedure a minimum spanning tree using secondary edge costs is calculated. Current and Pirkul [6] propose a new formulation of the problem based on the introduction of the dummy root node (as above) and provide computational results for two Lagrangian based heuristics derived from this model.

Other Related Problems. The TLNDF problem belongs to a class of problems with a tree-tree topology. The reader is referred to Gourdin et al. [12], who describe several variants of related problems such as star-tree, tree-star and star-star problems as well as other variants of tree-tree problems. In particular, when secondary subtrees are stars, the TLNDF becomes the *connected facility location problem* (see, e.g., [11]).

1.2. Notation

It is known that for rooted spanning or Steiner tree problems, modeling the problem on a directed graph provides models with stronger linear programming bounds than their undirected counterparts (see, e.g., [18]). Henceforth we will consider a directed graph $G = (V, A)$ that is obtained from the original undirected graph $G = (V, E)$ as follows: Instead of each edge $e = \{i, j\} \in E$ we use two arcs ij and ji in A, both of which are assigned the cost of the original edge. Since a solution on the undirected graph corresponds to an arborescence directed away from the root node, edges $\{r, j\}$ are replaced by a single arc rj.

In our models we will use the following binary variables:

$$x^1_{ij} = \begin{cases} 1, & \text{if the primary technology is installed on arc } ij \\ 0, & \text{otherwise} \end{cases} \qquad \forall ij \in A$$

$$x^2_{ij} = \begin{cases} 1, & \text{if the secondary technology is installed on arc } ij \\ 0, & \text{otherwise} \end{cases} \qquad \forall ij \in A, j \notin P$$

$$z_i = \begin{cases} 1, & \text{if a facility is installed on node } i \\ 0, & \text{otherwise} \end{cases} \qquad \forall i \in V$$

Observe that no feasible solution will contain secondary arcs pointing to a primary node (i.e., $x^2_{ij} = 0$ for $j \in P$). We will ignore the variables corresponding to these arcs in our models but, to simplify the notation, we will allow them in the indexation of the summation terms.

For a set $W \subseteq V$, we will write $z(W) = \sum_{i \in W} z_i$. For any $W \subset V$ we denote its complement set by $W^c = V \setminus W$. For any $M, N \subset V$, $M \cap N = \emptyset$, we denote the induced cut set of arcs by $(M, N) = \{ij \in A \mid i \in M, j \in N\}$. In particular, let $\delta^-(W) = (W^c, W)$ and $\delta^-(i) = (V \setminus \{i\}, \{i\})$. For a set of arcs $\hat{A} \subseteq A$, we will write $x^\ell(\hat{A}) = \sum_{ij \in \hat{A}} x^\ell_{ij}$, for $\ell = 1, 2$, and $(x^1 + x^2)(\hat{A}) = \sum_{ij \in \hat{A}} (x^1_{ij} + x^2_{ij})$.

We will describe models based on these variables, but in the next sections several models using other variables will be described as well. In order to relate all of these models, for a mixed integer programming model M let $\mathcal{P}_{\mathbf{a}^1,\ldots,\mathbf{a}^n}(M)$ denote the orthogonal projection of the convex hull of LP solutions of M onto the space defined by variables $\mathbf{a^1}$, ..., $\mathbf{a^n}$.

The illustrations in the next sections use the following symbols: $\boxed{\text{r}}$ represents the root node, ◯ represents a Steiner node. □ represents a primary customer, △ represents a secondary customer. Whenever we solve a problem as the Steiner tree problem, terminals are denoted by ◊.

2. Cut set-based formulations

In Gollowitzer et al. [10] we show that the TLNDF can be modeled as a Steiner arborescence problem on an extended graph with additional degree constraints. In this section we will review the extended graph definition, state the most important results taken from [10] and, in addition, present some new results.

Consider the graph $G_{NS} = (V_{NS}, A_{NS})$, with the root r' and the set of terminals R_{NS}, defined as follows:

$V_{NS} := V' \cup V'' \cup S$ where
$\quad V' := \{i' \mid i \in V\}$,
$\quad V'' := \{i'' \mid i \in V\}$,
$\quad S$ is the set of secondary nodes;

$R_{NS} := P' \cup S$ where
$\quad P' = \{i' \mid i' \in V', i \in P\}$;

$A_{NS} := A' \cup A'' \cup A_z \cup A_S$ where
$\quad A' := \{i'j' \mid ij \in A\}$,
$\quad A'' := \{i''j'' \mid ij \in A\}$,
$\quad A_z := \{i'i'' \mid i \in V\}$,
$\quad A_S := \{i'i \mid i' \in V', i \in S\}$
$\qquad \cup \{i''i \mid i'' \in V'', i \in S\}$.

The graph G_{NS} consists of several components:

i) A subgraph $G' = (V', A')$ corresponds to the primary network. It contains nodes and arcs that may be included in the primary subtree.

ii) A subgraph $G'' = (V'', A'')$ corresponds to the secondary network. It contains nodes and arcs that may be contained in the secondary subtrees.

iii) Arcs linking nodes in G' to the corresponding copy in G'' represent potential facilities.

iv) An additional copy of the secondary nodes (with arcs pointing from their representatives in graphs G' and G'') represents *terminals* that will make sure that each secondary node is either a part of the primary or the secondary network.

Arc costs C_{uv} for $uv \in A_{NS}$ are defined as follows:

$$C_{uv} = \begin{cases} c^1_{ij}, & u = i', v = j', ij \in A, \\ c^2_{ij}, & u = i'', v = j'', ij \in A, \\ d_i, & u = i', v = i'', i \in V, \\ 0, & \text{otherwise}, \end{cases} \qquad uv \in A_{NS}$$

We observe that if, for a primary node $i \in P$, its copy $i'' \in V''$ belongs to the optimal solution on the extended graph, then no ingoing arc of i'', except for the facility arc $i'i''$, will be used. Thus, we reduce the size of G_{NS} by removing all the arcs, except $i'i''$, leading into primary customer nodes in V''. Notice that a third copy of secondary nodes in G_{NS}, namely the set S, is needed, since the secondary customers can either be part of the primary subtree or be part of one of the secondary subtrees. The copies of secondary customers in G' and G'' are considered as Steiner nodes, with their third copy being a terminal. To ensure the tree topology, we will impose a restriction that for each node $i \in V$ at most one of the copies i' and i'' is allowed to have its $\mathbf{x}^1$- and $\mathbf{x}^2$-in-degree equal to one. Figure 1b) illustrates G_{NS} corresponding to the original graph shown in Figure 1a). We have the following result:

Lemma 1 ([10])**.** *The TLNDF problem can be modeled as the Steiner arborescence problem with additional node in-degree constraints on some node pairs on the graph G_{NS} with the root r' and terminal set R_{NS}.*

To obtain an integer programming (IP) model, we assign binary variables X_{ij} to all arcs $ij \in A_{NS}$. Let $X(\delta^-(\tilde{W}))$ denote the sum of X variables that are in the directed cut set $(\tilde{W}^c, \tilde{W})$ in G_{NS}. Based on the classical cut set model for Steiner trees (cf. [3]) we derive the

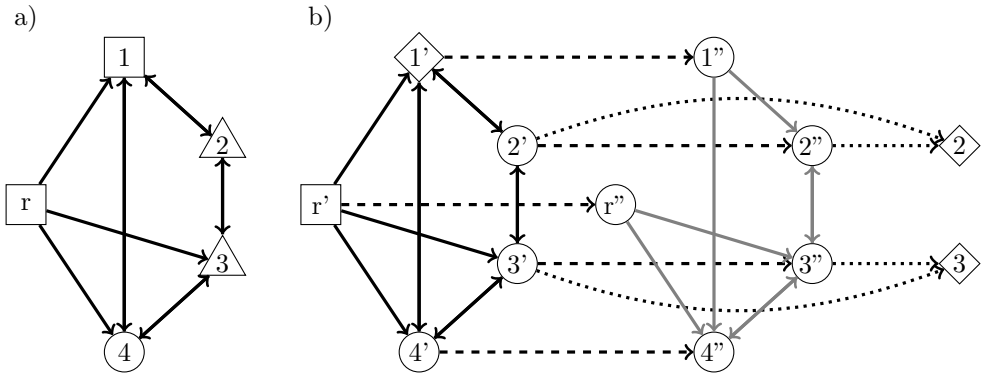


Figure 1: a) Instance of *TLNDF*; b) Transformed Steiner arborescence instance on G_{NS}.

following IP formulation:

$$(SA) \quad \min \sum_{ij \in A_{NS}} C_{ij} X_{ij}$$

$$\text{s.t.} \quad X(\delta^-(\tilde{W})) \geq 1 \qquad \forall \tilde{W} \subseteq V_{NS} \setminus \{r'\}, \tilde{W} \cap R_{NS} \neq \emptyset \tag{1a}$$

$$\sum_{ij \in A} (X_{i'j'} + X_{i''j''}) \leq 1 \qquad \forall j \in V \setminus \{r\} \tag{1b}$$

$$X_{ij} \in \{0,1\} \quad \forall ij \in A_{NS} \tag{1c}$$

Constraints (1a) are connectivity cuts between the root node and each terminal. Inequalities (1b) ensure that any solution of *SA* does not contain the copies i' and i'' of $i \in V$ at the same time, unless the facility arc $i'i''$ is part of the solution.

Lemma 2 ([10])**.** *Cut set inequalities* (1a) *such that* $\delta^-(\tilde{W}) \cap A_S \neq \emptyset$ *are redundant in the model SA.*

A straightforward algorithmic approach based on the separation of inequalities (1a) might prove to be computationally expensive, since the separation of inequalities (1a) requires the solution of maximum flow problems on the graph G_{NS} with up to to $3|V|$ nodes and $2|A|+3|V|$ arcs. This has motivated us to investigate and implement a two-phase separation method where we start by separating cut set inequalities on the original graph and only then move to separation on the extended graph. The reason for this two-phase approach is that the corresponding maximum flow algorithm for the cut set constraints on the original graph is applied to a much smaller graph.

To find these sets of inequalities on the original graph we add the following constraints to the model *SA*. They link the variables on the extended graph and the variables on the

original graph.

$$x^1_{ij} = X_{i'j'} \qquad \forall ij \in A, i'j' \in A', \tag{2a}$$

$$x^2_{ij} = \begin{cases} X_{i''j''} \\ 0 \end{cases} \qquad \forall ij \in A, \begin{cases} i''j'' \in A'', \\ \text{otherwise} \end{cases} \tag{2b}$$

$$z_i = X_{i'i''} \qquad \forall i \in V, i'i'' \in A_z. \tag{2c}$$

Adding these equalities to the model SA will not alter its LP value but will allow us to characterize $\mathcal{P}_{\mathbf{x}^1,\mathbf{x}^2,\mathbf{z}}(SA)$.

Lemma 3. *$\mathcal{P}_{\mathbf{x}^1,\mathbf{x}^2,\mathbf{z}}(SA) = \mathcal{P}(CUT)$ where $\mathcal{P}(CUT)$ is given by the set of vectors $(\mathbf{x}^1, \mathbf{x}^2, \mathbf{z})$ satisfying*

$$(x^1 + x^2)(\delta^-(i)) \leq 1 \qquad \forall i \in V \setminus \{r\} \tag{3}$$

and

$$x^1(\delta^-(W')) + x^2(\delta^-(W'')) + z(W'' \setminus W') \geq 1 \qquad \begin{array}{l} r \notin W', W' \cap W'' \cap S \neq \emptyset \\ \text{or } W' \cap P \neq \emptyset. \end{array} \tag{x12-z}$$

where $W' = \{i \in V \mid i' \in \tilde{W}\}$ and $W'' = \{i \in V \mid i'' \in \tilde{W}\}$ for an arbitrary cut set $\tilde{W} \subseteq V_{NS} \setminus \{r'\}$ such that $\tilde{W} \cap R_{NS} \neq \emptyset$ and $\delta^-(\tilde{W}) \cap A_S = \emptyset$.

Constraints (x12-z) and (3) correspond to non redundant cut sets (1a) and constraints (1b) on the extended graph, respectively.

The following sets of inequalities are special cases of constraints (x12-z) (see Figure 2).

i) If $W' = W$ and $W'' = \emptyset$, we obtain *primary connectivity* constraints:

$$x^1(\delta^-(W)) \geq 1 \qquad \forall W \subseteq V \setminus \{r\},\ W \cap P \neq \emptyset \tag{x1}$$

ii) For $W'' = V$ and $W' = W$ we obtain constraints of the form

$$z(W^c) + x^1(\delta^-(W)) \geq 1 \qquad \forall W \subseteq V \setminus \{r\}, W \cap S \neq \emptyset \tag{x1-z}$$

iii) For $W' = W'' = W$, we obtain *secondary connectivity* cuts:

$$(x^1 + x^2)(\delta^-(W)) \geq 1 \qquad \forall W \subseteq V \setminus \{r\},\ W \cap S \neq \emptyset \tag{x12}$$

iv) For $W' = \{k\}$, $k \in S$, and $W'' = W \cup \{k\}$ we obtain constraints of the form

$$z(W) + x^1(\delta^-(k)) + x^2(\delta^-(W_k)) \geq 1 \qquad \forall k \in S, W \subseteq V \setminus \{k\}, W_k = W \cup \{k\} \tag{x2-z}$$

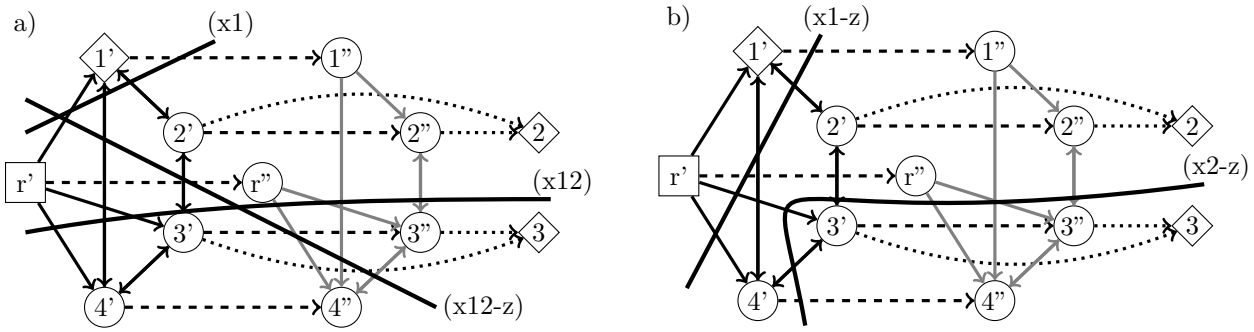


Figure 2: a) Illustration of inequalities (x1) for $W = \{1\}$, (x12) for $W = \{3, 4\}$, and (x12-z) for $W' = \{1, 2\}$, $W'' = \{r, 1, 2, 3\}$. b) Illustration of inequalities (x1-z) for $W = \{2, 3, 4\}$ and (x2-z) for $W = \{4\}$ and $k = 3$.

Constraints (x1) ((x12)) are connectivity cuts for primary (secondary) customers and ensure a path that consists of primary (primary and secondary) edges between the root node and each primary (secondary) customer. They have already been stated in [1] for the related problem without transition nodes. The remaining constraints involving $\mathbf{z}$ variables are new. Constraints (x1-z) state that for any subset of nodes that contains a secondary customer, there must either be an ingoing primary arc or an installed node facility in its complement. The interpretation of (x2-z) is not straightforward. However, these constraints were "found" in an indirect way. By subtracting $x^1(\delta^-(k))$ and $x^2(W^c, k)$ on both sides we obtain

$$z(W) + x^2(W_k^c, W) \geq x^2(W, \{k\}) \quad \forall k \in S,\ W \subseteq V \setminus \{k\}, W_k = W \cup \{k\}. \qquad \text{(x2-z')}$$

Constraints (x2-z') are a generalization of inequalities suggested in [13] for models with node transition variables. These inequalities state that if there is a secondary arc leading into node k from a node in a given node set W, then either there is a facility installed in a node in W or there is a secondary arc leading into W from the complement of $W \cup \{k\}$.

The four sets of constraints just described are the constraints that will be separated in the first phase of our branch-and-cut algorithm. Separation algorithms for these will be described in Section 4. Clearly, there are other constraints included in the general description given by (x12-z) that do not correspond to any constraint of these four sets. For these remaining constraints, it is not clear how to separate them in the original space. This is precisely the set of constraints that will be separated in the larger extended graph.

It is obvious that the model defined by (3) and (x12-z) gives a valid model for the TLNDF defined only on variables x^1, x^2 and z. Notice that another valid model is obtained by only considering the inequalities (x1), (x12) and the family (x2-z) for singleton sets W instead of the whole set (x12-z). In the next section we will present a multi-commodity flow formulation whose LP relaxation is equivalent to the LP relaxation of this latter model. However, our computational experiments (cf. Section 5) will show that this model provides much weaker bounds than the model with all general cut set inequalities (x12-z).

3. Flow-based formulations

In this section we will present a compact multi-commodity flow formulation that extends the strongest model proposed in [2], by introducing node transition variables. As we shall show later, the LP relaxation bound of this model is not as good as the one provided by the cut set model on the extended graph (presented in Lemma 3). A similar dominance result is known for the TLNDF problem without node transition costs and is given in Chopra and Tsai [4]. Therefore, we propose a new model based on multi-commodity flows that is obtained from the previous one by disaggregating the variables and constraints "by technology". We show that the LP relaxation of the new disaggregated model is equally strong as the LP relaxation of the cut set model on the extended graph.

3.1. Multi-commodity flow formulation

Let us define the following flow variables: $f_{ij}^k \geq 0$ corresponds to the flow from r to the commodity $k \in R$, using arc $ij \in A$. A multi-commodity formulation for the classical two level network design problem (without transition nodes) is the following (cf., e.g., [1]):

$$
\begin{aligned}
(MCF) \quad & \min \sum_{ij \in A} c_{ij}^1 x_{ij}^1 + \sum_{ij \in A} c_{ij}^2 x_{ij}^2 \\
& \sum_{ji \in A} f_{ji}^k - \sum_{ij \in A} f_{ij}^k = \begin{cases} 1 & i = k \\ -1 & i = r \\ 0 & i \neq k, r \end{cases} \qquad \forall i \in V,\ \forall k \in R \\
& 0 \leq f_{ij}^k \leq x_{ij}^1 \qquad \forall ij \in A, k \in P \\
& 0 \leq f_{ij}^k \leq x_{ij}^1 + x_{ij}^2 \qquad \forall ij \in A, k \in S \\
& x_{ij}^1, x_{ij}^2 \in \{0,1\} \qquad \forall ij \in A
\end{aligned}
$$

An extension to model transition node costs is obtained by changing the objective function to

$$\min \sum_{ij \in A} c_{ij}^1 x_{ij}^1 + \sum_{ij \in A} c_{ij}^2 x_{ij}^2 + \sum_{i \in V} d_i z_i$$

and adding the following compact sets of constraints to *MCF*:

$$
\begin{aligned}
z_i + \sum_{ji \in A, j \neq k} x_{ji}^2 &\geq x_{ik}^2 \qquad \forall ik \in A, i \neq r && \text{(4a)} \\
z_r &\geq x_{rk}^2 \qquad \forall rk \in A && \text{(4b)} \\
(x^1 + x^2)(\delta^-(i)) &\leq 1 \qquad \forall i \in V \setminus \{r\} && \text{(4c)} \\
z_i &\in \{0,1\} \qquad \forall i \in V
\end{aligned}
$$

Inequalities (4a) and (4b) are *coupling constraints* and link facility variables to secondary edges. Constraints (4c) prevent secondary cycles for which no facility is installed and ensure that the obtained solution has a tree topology.

For simplicity we maintain the same designation *MCF* for this model.

3.2. Disaggregated multi-commodity flow formulation

We now show how to strengthen the MCF model by disaggregating the variables f^k by technology. Consider the following two types of flow variables: $f_{ij}^{1k} \geq 0$ ($f_{ij}^{2k} \geq 0$) correspond to the primary (secondary) flow from r to the commodity $k \in R$, using arc $ij \in A$.

Consider then the following model, which extends a model described in Gouveia and Janssen [13] by node transition variables.

$$(dMCF) \quad \min \sum_{ij \in A} c_{ij}^1 x_{ij}^1 + \sum_{ij \in A} c_{ij}^2 x_{ij}^2 + \sum_{i \in V} d_i z_i$$

$$\text{s.t.} \quad \sum_{ji \in A} f_{ji}^{1k} - \sum_{ij \in A} f_{ij}^{1k} = \begin{cases} 1 & i = k \\ -1 & i = r \\ 0 & i \neq k, r \end{cases} \qquad \forall i \in V,\ \forall k \in P \tag{5a}$$

$$\sum_{ji \in A} (f_{ji}^{1k} + f_{ji}^{2k}) - \sum_{ij \in A} (f_{ij}^{1k} + f_{ij}^{2k}) = \begin{cases} 1 & i = k \\ -1 & i = r \\ 0 & i \neq k, r \end{cases} \qquad \forall i \in V,\ \forall k \in S \tag{5b}$$

$$z_i + \sum_{ji \in A} f_{ji}^{2k} \geq \sum_{ij \in A} f_{ij}^{2k} \qquad \forall k \in S, \forall i \in V, i \neq k \tag{5c}$$

$$(x^1 + x^2)(\delta^-(i)) \leq 1 \qquad \forall i \in V \tag{5d}$$

$$0 \leq f_{ij}^{1k} \leq x_{ij}^1 \qquad \forall ij \in A, k \in R \tag{5e}$$

$$0 \leq f_{ij}^{2k} \leq x_{ij}^2 \qquad \forall ij \in A, k \in S \tag{5f}$$

$$z_i, x_{ij}^1, x_{ij}^2 \in \{0, 1\} \qquad \forall i \in V, ij \in A \tag{5g}$$

In the context of this model the flow variables f^{1k} and f^{2k} can be reinterpreted as indicating whether arc ij has technology 1 or 2 installed and whether it is in the path to node k. The new constraints (5c) state that a facility needs to be installed when the technology used on the arcs changes on the path to node k.

Equations (5a) and (5b) ensure one unit of primary (primary or secondary) flow to primary (secondary) customer nodes. Constraints (5d) limit the number of ingoing arcs for each node to one and inequalities (5e) and (5f) link the flow variables and design variables.

3.3. Polyhedral comparison

Next we will show that formulation *dMCF* on the original graph G provides the same LP bound as the cut set model on the extended graph. To prove this result we will introduce an auxiliary model and prove that the two models, the cut set model on the extended graph and the flow model *dMCF* provide the same LP bound as the auxiliary model.

The auxiliary model is a straightforward multi-commodity flow reformulation of the cut set model on the extended graph. To define this model we consider the following sets of variables. Flow variables f_{ij}^{1k} and f_{ij}^{2k}, that correspond to the flow from r to the commodity $k \in R$, using arcs $i'j' \in A'$ or $i''j'' \in A''$, respectively. Variables y_i^k correspond to the flow of commodity k sent through $i'i'' \in A_z$. Finally, for all $k \in S$, we also define f_k^{1k} and f_k^{2k} to

be the flow values on arcs $k'k$ and $k''k$ in A_S, respectively. Obviously, $f_k^{1\ell} = f_k^{2\ell} = 0$ for all $k \neq \ell$.

$$(MCF_{NS}) \quad \min \sum_{ij\in A} c_{ij}^1 x_{ij}^1 + \sum_{ij\in A} c_{ij}^2 x_{ij}^2 + \sum_{i\in V} d_i z_i$$

$$\text{s.t. (5a), (5d), (5e), (5f), (5g),}$$

$$y_r^k + \sum_{rj\in A} f_{rj}^{1k} = 1 \qquad \forall k \in S \tag{6a}$$

$$y_r^k - \sum_{rj\in A} f_{rj}^{2k} = 0 \qquad \forall k \in S \tag{6b}$$

$$\sum_{ji\in A} f_{ji}^{1k} - \sum_{ij\in A} f_{ij}^{1k} - y_i^k = 0 \qquad \forall k \in S, i \in V \setminus \{r,k\} \tag{6c}$$

$$y_i^k + \sum_{ji\in A} f_{ji}^{2k} - \sum_{ij\in A} f_{ij}^{2k} = 0 \qquad \forall k \in S, i \in V \setminus \{r,k\} \tag{6d}$$

$$\sum_{jk\in A} (f_{jk}^{1k} + f_{jk}^{2k}) = 1 \qquad \forall k \in S \tag{6e}$$

$$0 \leq y_i^k \leq z_i \qquad \forall i \in V, k \in R \tag{6f}$$

Using the linking constraints (2) and $X_{i'i} = x^1(\delta^-(i))$ and $X_{i''i} = 1 - X_{i'i}$ for all $i \in S$, the max-flow min-cut theorem implies the following

Lemma 4. $\mathcal{P}(SA) = \mathcal{P}_X(MCF_{NS})$.

We can also show the following

Lemma 5. $\mathcal{P}_{\mathbf{x}^1,\mathbf{x}^2,\mathbf{z},\mathbf{f}^1,\mathbf{f}^2}(MCF_{NS}) = \mathcal{P}(dMCF)$.

Proof. We show the claim by mutual inclusion:

Let $(\mathbf{x}^1, \mathbf{x}^2, \mathbf{z}, \mathbf{f}^1, \mathbf{f}^2, \mathbf{y}) \in \mathcal{P}(MCF_{NS})$. Then $(\mathbf{x}^1, \mathbf{x}^2, \mathbf{z}, \mathbf{f}^1, \mathbf{f}^2) \in \mathcal{P}(dMCF)$: By eliminating y_i^k from (6a) and (6b) ((6c) and (6d)) we obtain constraints (5b) for the case $i = r$ ($i \neq k, r$). Constraints (5b) for $i = k$ are equivalent to (6e). Limiting y_i^k from above in (6b) and (6d) using (6f) we obtain (5c).

Let now $(\mathbf{x}^1, \mathbf{x}^2, \mathbf{z}, \mathbf{f}^1, \mathbf{f}^2) \in \mathcal{P}(dMCF)$ and variables y_i^k be defined as follows.

$$y_i^k := \sum_{ij\in A} f_{ij}^{2k} - \sum_{ji\in A} f_{ji}^{2k} \quad \forall k \in S, i \in V \setminus \{r,k\} \text{ and}$$

$$y_r^k := \sum_{j\in A} f_{rj}^{2k} \quad \forall k \in S.$$

Then one can easily verify that $(\mathbf{x}^1, \mathbf{x}^2, \mathbf{z}, \mathbf{f}^1, \mathbf{f}^2, \mathbf{y}) \in \mathcal{P}(MCF_{NS})$. □

The two preceding lemmata imply the following

Theorem 1. $\mathcal{P}_{\mathbf{x}^1,\mathbf{x}^2,\mathbf{z}}(dMCF) = \mathcal{P}_{\mathbf{x}^1,\mathbf{x}^2,\mathbf{z}}(SA)$.

4. Branch-and-cut framework

Since our models comprise an exponential number of constraints, we solve them using the cutting plane technique embedded into a branch-and-bound framework, commonly known as the *branch-and-cut* approach. Non-standard ingredients of our approach are described below. The primal heuristic that we use is described in [10].

4.1. Initialization

To reduce the number of separated cut set constraints and improve the general performance, in our computational experiments we initialize all models with degree and coupling constraints (4) and the following sets of inequalities:

$$\sum_{ij \in A} (x^1_{ij} + x^2_{ij}) \leq \sum_{jk \in A} (x^1_{jk} + x^2_{jk}) \qquad \forall j \in V \setminus R \tag{7a}$$

$$\sum_{ij \in A, i \neq k} x^1_{ij} \geq x^1_{jk} \qquad \forall jk \in A, j \in V \setminus \{r\} \tag{7b}$$

Inequalities (7a) are strengthening *degree balance constraints* for Steiner nodes. Inequalities (7b) guarantee that for each outgoing primary arc of a node $j \in V \setminus \{r\}$ there is at least one primary arc entering this node.

4.2. Cut separation

In Section 2 we have proposed several classes of valid cut set inequalities for the TLNDF. In this section we will show that the separation of some of them can be done on the original graph G, extended by an extra node, while only the more general (x12-z) inequalities need to be separated on the extended graph.

Separation of constraints (x1) and (x12) is performed on the original graph G: we solve a maximum flow problem between the root and each $i \in P$ and $i \in S$, respectively, using the values of $\mathbf{x}^1$ and $\mathbf{x}^1 + \mathbf{x}^2$ as arc capacities. To separate the more general constraints (x12-z), we build the extended graph G_{NS}, set the arc capacities on G' and G'' to $\mathbf{x}^1$ and $\mathbf{x}^2$, respectively, and capacities of arcs $k'k''$ to z_k, for all $k \in F$. To make sure that only non-redundant cuts between the root and a secondary node $i \in S$ are separated, we set the capacities of arcs $i'i$ and $i''i$ to $M > 1$, for each $i \in S$. We will now describe how constraints (x1-z) and (x2-z) can be separated on graphs that are much smaller than the extended graph G_{NS}.

Lemma 6. *Inequalities* (x1-z) *can be separated by solving the maximum flow problem on a graph with* $|V| + 1$ *nodes and* $|A| + |V|$ *arcs.*

Proof. For each $k \in S$ we generate a graph $G_t = (V' \cup \{t\}, A' \cup A_t)$ with weights w_{uv} as follows:

1. $A_t = \{(i', t) \mid i' \in V'\}$
2. $w_{i'j'} := x^1_{ij}$, $i'j' \in A'$ and $w_{i't} := z_i$, $i \in V \setminus \{k\}$ and $w_{k't} := 1$.

Then each (r', t)-cut in G_t with a weight of less than 1 corresponds to a violated (x1-z) inequality for $k \in W \cap S$. □

Lemma 7. *Inequalities* (x2-z) *can be separated by solving the maximum flow problem on a graph with* $|V| + 1$ *nodes and* $|A| + |V|$ *arcs.*

Proof. For each $k \in S$ we generate a graph $G_s = (V'' \cup \{s\}, A'' \cup A_s)$ with weights w_{uv} as follows:

1. $A_s = \{(s, i'') \mid i'' \in V''\}$
2. $w_{i''j''} := x^2_{ij}$, $i''j'' \in A''$, $w_{si''} := z_i$, $i \in V \setminus \{k\}$ and $w_{sk''} := x^1(\delta^-(k))$.

Then each (s, k'')-cut in G_s with a weight of less than 1 corresponds to a violated (x2-z) inequality for $k \in W \cap S$. □

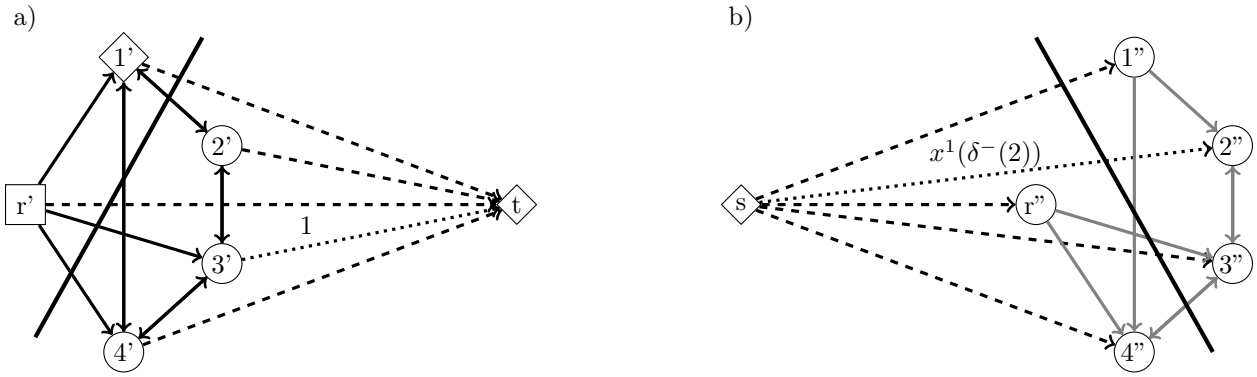


Figure 3: a) Illustration of G_t for $k = 3$ and inequality (x1-z) for $W = \{2, 3, 4\}$. b) Illustration of G_s for $k = 2$ and inequality (x2-z) for $W = \{1, 3\}$.

We separate violated cut set inequalities in every node of the the branch-and-bound tree (B&B). To improve the computational efficiency of the separation, for each family of cuts we search for nested minimum cardinality cuts. To do so, all capacities in the respective separation graph are increased by some $\epsilon > 0$. Thus, every detected violated cut contains the least possible number of arcs. We resolve the linear program after adding at most 50 violated inequalities of any class. Finally, we randomly choose the target nodes to search for violated cuts. To ensure comparability, we fix the seed value for the computational results reported.

As mentioned in Section 2, we propose a two-phase approach to separate the cut set inequalities. In the first phase we add violated inequalities that can be separated on G or on G extended by a single node. Only when no more violated inequalities of these types can be found, we resort to the separation on the extended graph.

The constraint sets (x1-z), (x12) and (x2-z) can be ordered with respect to pairwise inclusion of the corresponding sets W' and W'': Consider an inequality (x1-z). By removing

the set $V \setminus W'$ from W'' we obtain an inequality of the form (x12). By removing all but a single node $k \in S$ from W' we obtain an inequality of the form (x2-z). Thus we will always detect violated inequalities in the following order: (x1) - (x1-z) - (x12) - (x2-z) - (x12-z).

In our computational study (see Section 5) we choose different subsets of the mentioned four families for the separation in phase one. The four different strategies we experiment with are given in Table 1.

	Phase 1				Phase 2
Model	(x1)	(x1-z)	(x12)	(x2-z)	(x12-z)
EG^+	✓	✓	✓	✓	✓
EG	✓				✓
OG^+	✓	✓	✓	✓	
OG	✓		✓		

Table 1: Constraint subsets in Phase 1 and 2

Strategy EG^+ has the advantage of providing the strong lower bounds of the extended graph model but performs the computationally demanding separation of general cut sets (x12-z) only when it is needed. Strategy EG provides the same lower bounds as EG^+, but it is a more naive implementation of the previous strategy. The whole separation procedure is performed on the extended graph. For the last two strategies, denoted by OG and OG^+, separation is performed on the graph G or G extended by an extra node, respectively. Strategy OG^+ derives slightly weaker lower bounds than the previous two because we refrain from the separation of the general family of cut sets (x12-z) and insert only four special subfamilies, namely (x1), (x1-z), (x12) and (x2-z). Finally, the weakest lower bounds are obtained using the strategy OG that separates only connectivity cuts for primary and secondary nodes, and uses a compact constraint set to model the transition nodes. Inequalities (4a) and (4b) used in the initialization phase of the branch-and-cut procedure guarantee the feasibility of this model.

5. Computational Study

In this section we report on our computational experience with the four MIP strategies described above. All experiments were performed on a desktop machine with an 8-core Intel Core i7 CPU at 2.80 GHz and 8 GB RAM. Each run was performed on a single processor. We used the CPLEX [16] branch-and-cut framework, version 12.2. All cutting plane and heuristic routines provided by CPLEX are turned off, the other parameters are set to their default values. We set the optimal solution value as global cutoff value in the first and second part of our computations. The primal heuristic was not used in that case.

5.1. Instances

For our computational study we transform instances of the Steiner tree problem (STP) using the following procedure: 30% of STP terminals are chosen as primary customers, the

remaining 70% are selected as secondary customers. The primary customer with the lowest index is chosen as root node. The Steiner nodes in the STP instance are Steiner nodes in the TLNDF instance. We allow installation of a facility in every node of the graph. Primary edge costs equal edge costs of the STP instance. For each secondary edge e, the cost c_e^2 is defined as qc_e^1, where q is uniformly randomly chosen from $[0.25, 0.5]$. Facility opening costs are uniform and equal 0.5 times the average primary edge costs.

The parameters for generating instances have been chosen such that trivial solutions (e.g., optimal solutions that do not contain secondary subtrees) are avoided. In our computational study we also tested the effect of alterations of the above given parameters. We use sets `B`, `C` and `D` of the Steinlib library [17] with 50-100, 500 and 1000 nodes and up to 200, 12500 and 25000 edges, and the sets of random graphs named `K` and `P` proposed by Minkoff and Karger [19], with a street-like structure and up to 400 nodes and 1576 edges. The latter instances are available online at [22].

5.2. Results

Preliminary tests showed that all instances from groups `B`, `K100` and `P100` can be solved in less than two seconds by all of the four tested approaches. Of the instances in these three groups only `b11` and `b15` were not solved to optimality at the root node of the branch-and-bound tree, but required to examine two branch-and-bound nodes each.

To avoid possibly misleading conclusions from large relative but small absolute deviations in the runtime, we do not consider these three instance groups in the following.

Comparing lower bounds and running times. We perform the first part of our computational study on the set of 18 instances with 200 and 400 nodes from test sets `K` and `P`. Our goal was to test whether the theoretical results presented in Section 3 are supported by computational experience. Since among the four MIP approaches presented above, EG and EG^+ provide the same LP bounds v_{LP} and differ only in the separation strategy, we performed this test using only three out of the four approaches, namely, EG, OG^+ and OG. We set the default time limit to 10 minutes. For each of these approaches, and for each of the 18 instances, Table 2 reports the following values: the optimal integer solution value (OPT), the running time (in seconds) needed to solve the LP relaxation (t_{LP} [s]), the LP gap (calculated as Gap [%]=$(OPT - v_{LP})/OPT$), the running time (in seconds) of the integer program (t_{IP} [s]), and the number of enumerated branch-and-bound nodes ($\#BnB$). A dash denotes that the respective model could not solve the instance within the given time limit. In that case, column $\#BnB$ indicates the number of nodes enumerated until then.

The values in Table 2 confirm the relations stated in Section 3. While the strongest model, EG, solves to optimality a majority of the instances already at the root node of the branch-and-bound tree, the LP relaxations of the two weaker models provide weaker lower bounds (see column Gap [%]) and thus require to enumerate a significantly larger number of nodes before possibly reaching optimality. The running times of the LP relaxations show that separating violated inequalities from the "small" constraint sets (x1) and (x12) (model OG) can indeed be accomplished faster than separating inequalities from a larger subset including (x1-z) and (x2-z) (model OG^+) or even the complete set (x12-z) (model EG).

		t_{LP} [s]			*Gap* [%]			t_{IP} [s]			#*BnB*		
Instance	*OPT*	*EG*	OG^+	*OG*	*EG*	OG^+	*OG*	*EG*	OG^+	*OG*	*EG*	OG^+	*OG*
K200	385.9	4.3	5.3	**2.9**	0.00	0.05	2.63	**4.6**	5.6	355.9	0	1	6832
K400	383.5	36.7	38.9	**22.5**	0.00	0.37	4.07	**38.3**	47.4	-	0	7	1033
K400-1	474.2	50.1	48.2	**33.5**	0.02	0.34	2.46	**51.8**	58.9	-	0	8	1151
K400-2	456.5	46.3	41.0	**26.8**	0.00	0.04	2.91	48.2	**45.1**	-	0	2	1007
K400-3	431.5	43.5	42.2	**28.8**	0.00	0.05	2.62	45.3	**43.2**	-	0	1	1075
K400-4	394.4	45.3	40.9	**21.8**	0.03	0.05	2.20	47.0	**42.9**	-	1	2	1070
K400-5	539.0	147.3	112.7	**98.5**	0.00	0.07	2.57	149.0	**123.2**	-	0	5	495
K400-6	449.5	84.9	82.1	**56.2**	0.13	0.16	2.25	97.0	**89.9**	-	5	5	755
K400-7	468.1	57.5	64.4	**29.7**	0.00	0.49	3.68	**59.5**	206.9	-	0	172	998
K400-8	459.4	**70.1**	75.5	73.0	0.00	0.00	1.61	**71.5**	76.6	-	0	0	879
K400-9	480.5	104.6	95.0	**75.2**	0.23	0.33	3.29	**128.4**	153.7	-	13	43	780
K400-10	330.7	20.3	21.2	**12.8**	0.00	0.12	2.29	**21.3**	22.9	-	0	3	1888
P200	1051.7	5.5	3.9	**2.8**	0.00	0.00	0.04	5.9	4.2	**2.9**	0	0	1
P400	2085.7	35.8	34.8	**11.7**	0.11	0.11	0.63	42.6	38.4	**30.3**	3	3	8
P400-1	2183.8	32.5	26.4	**24.3**	0.00	0.00	0.34	35.2	**28.1**	118.1	0	0	38
P400-2	2239.2	20.2	18.6	**9.3**	0.00	0.00	0.20	22.5	20.1	**9.7**	0	0	0
P400-3	2636.9	40.1	35.6	**22.2**	0.05	0.05	0.19	42.6	**37.3**	37.6	1	1	6
P400-4	2104.8	24.6	19.4	**18.2**	0.00	0.00	0.10	26.4	20.6	**18.5**	0	0	1

Table 2: Runtimes of IP and LP, LP gaps and number of enumerated branch-and-bound nodes for three approaches of different strength.

When comparing the running times for the complete integer programs we see that there is a tradeoff between faster separation and stronger bounds. Whenever there is a significant difference in the enumerated number of branch-and-bound nodes, the stronger model *EG* using the slower separation is faster overall.

In the second part of our computational study we assess whether the cut separation on graph G in the first phase (EG^+) speeds up the overall performance of the model containing all constraints in (x12-z) (*EG*). Since the approach *OG* has shown to be computationally inferior (cf. Table 2), this study compares only strategies EG^+, *EG* and OG^+. We group the instances in sets `C` and `D` according to the number of customers (first two blocks) and the number of edges in the graph (last two blocks). For each of these groups we calculate the geometric mean of the running time (t_{IP} [s]) and the number of cuts added (#*Cuts*). To take into account the values equal to 0 we resort to the shifted geometric mean for presenting the number of enumerated branch-and-bound nodes (#*BnB*) and LP gaps (*Gap* [%]). We use the arithmetic mean as shift.[2]

The running times indicate that for most groups the separation on the smaller graph G is beneficial. While EG^+ is faster than *EG* on 15 groups, it's the other way around for only 3 groups. The performance of approach OG^+ is surprisingly good. Even though the gaps are slightly larger, omitting the costly separation on G_{NS} leads to better overall running times,

[2] For non-negative values $v_i, i \in \{1, \ldots, k\}$ the *shifted geometric mean* for shift $s > 0$ is defined as $\mu_s(v_1, \ldots, v_k) = (\prod_{i=1}^{k}(v_i + s)^{1/k}) - s$.

Inst.	t_{IP} [s]			#BnB			#Cuts			Gap [%]	
	EG	EG^+	OG^+	EG	EG^+	OG^+	EG	EG^+	OG^+	EG	OG^+
c{01,06,11,16}	3.7	2.8	**2.5**	**1.0**	**1.0**	1.5	133	125	118	0.73	0.84
c{02,07,12,17}	7.0	5.5	**4.6**	**0.0**	**0.0**	**0.0**	294	262	245	0.00	0.00
c{03,08,13,18}	63.3	62.8	**51.8**	4.1	3.7	**3.5**	2239	2041	1683	0.09	0.11
c{04,09,14,19}	65.4	**49.4**	50.4	0.4	**0.1**	**0.1**	2309	2160	1835	0.00	0.00
c{05,10,15,20}	**83.3**	99.0	102.6	**1.5**	1.8	2.3	2708	2816	2579	0.04	0.04
d{01,06,11,16}	7.6	7.2	**6.6**	**0.1**	**0.1**	1.0	167	163	159	0.00	2.25
d{02,07,12,17}	17.9	**13.0**	13.9	0.4	**0.0**	0.6	343	264	287	2.28	4.21
d{03,08,13,18}	381.0	**349.1**	362.4	5.1	**3.7**	6.9	4800	4880	4036	0.09	0.11
d{04,09,14,19}	**311.5**	332.7	312.1	**0.4**	0.5	**0.4**	4948	4977	4053	0.01	0.01
d{05,10,15,20}	879.3	**842.5**	925.3	14.0	**11.9**	13.5	8231	7443	7282	0.09	0.10
c{01-05}	4.8	4.5	**3.8**	**0.1**	**0.1**	**0.1**	737	738	632	0.04	0.07
c{06-10}	8.4	7.6	**6.8**	**0.4**	**0.4**	**0.4**	786	742	666	0.00	0.00
c{11-15}	36.8	30.0	**28.6**	1.6	1.6	**1.1**	935	780	730	0.04	0.04
c{16-20}	243.4	214.0	**211.5**	3.3	**2.8**	4.5	1136	1140	1004	0.51	0.54
d{01-05}	21.4	**17.3**	18.0	**0.5**	**0.5**	**0.5**	1487	1318	1243	0.01	0.01
d{06-10}	49.2	**45.5**	45.9	1.1	**0.7**	**0.7**	1863	1697	1569	1.62	1.69
d{11-15}	122.5	124.1	**99.9**	2.5	**1.3**	1.5	1363	1387	1168	0.02	0.02
d{16-20}	1032.0	**957.6**	1179.7	10.7	**8.9**	17.1	1830	1666	1700	0.10	3.23

Table 3: Comparison of the 3 selected approaches. The best running times and least number of enumerated branch-and-bound nodes are shown in bold.

especially on the instances of set C with only few customers (c{1,6,11,16}, c{2,7,12,17}). However, for the instance group with the largest sets of edges and customers (d{16-20}) the significantly larger gap requires the enumeration of a lot more branch-and-bound nodes and leads to a performance worse than the one of the stronger models EG and EG^+. We believe that approach OG^+ is suitable for small to medium instances (with a small number of customers), but does not scale well to dense graphs with large customer sets.

The average values for the number of enumerated branch-and-bound nodes are unexpectedly uncorrelated with the strength of the underlying lower bounds. For some instance groups the "weaker" approach OG^+ enumerated less nodes before reaching optimality, even though it provides the weaker LP bounds. This can be explained by the fact that we use CPLEX' default branching strategy. Thus the variables selected for branching and the order in which the nodes are explored might differ.

An interesting aspect of the results in Table 3 are the number of constraints detected by the separation procedures. In approach OG^+ the least number of cuts is added to the LP and optimality is enforced by extensive branching. This is as expected, as in OG^+ inequalities of the general set (x12-z) are not separated. The number of cuts added by EG and EG^+ indicate another advantage of the latter approach. The inequalities that can be separated on the smaller graphs are more likely to be binding in the optimal LP solution of each branch-and-bound node than some of those detected on the extended graph. Thus, the linear programs in the branch-and-bound tree need less memory, allowing EG^+ better scalability when system memory becomes the limiting factor.

Influence of instance parameters on algorithmic performance. In the last part of our study we assess the effect of changes of the parameters used to generate the test instances. We compare the performance of EG^+ on instance sets generated using the default parameters to the performance when these parameters are altered.

The following choices of parameters were tested:

- Higher and lower facility opening costs (0.75 and 0.25 times the average primary edge costs), indicated by $d \uparrow$ and $d \downarrow$, respectively.
- Higher and lower secondary edge costs ($q \in [0.5, 1)$ and $q \in [0.125, 0.25]$), indicated by $c^2 \uparrow$ and $c^2 \downarrow$, respectively.
- Higher/lower facility opening costs combined with lower/higher secondary edge costs.
- Restricting potential facilities to the customer nodes $R = P \cup S$ (cf. [10]).

In Table 4 we report results aggregated over the 18 instances listed in Table 2. We report first, second and third quartile (Q_1, Q_2 and Q_3) of the running times (t_{IP} $[s]$) and number of cuts added ($\#Cuts$), the number of instances solved to optimality at the root node of the branch-and-bound tree ($\#Opt_{LP}$) and the shifted geometric mean of the LP gaps (Gap [%], where we once again use the arithmetic mean as shift). For the default setting indicated by EG^+ we report absolute numbers. For all other parameter settings we report absolute numbers for $\#Opt_{LP}$ and Gap [%]. For the runtime t_{IP} $[s]$ and the number of added cuts $\#Cuts$ we report the increase or decrease compared to the respective value for EG^+ in per cent.

	t_{IP} $[s]$			$\#Cuts$				
Setting	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3	$\#Opt_{LP}$	Gap [%]
EG^+	23.8	43.4	63.0	1817	2215	2897	13	0.04
$d \uparrow$	0.0	34.2	29.2	3.0	16.1	12.0	11	0.06
$d \downarrow$	-3.4	-31.0	-28.7	-14.0	-12.9	-12.0	16	0.01
$c^2 \uparrow$	31.9	-2.5	6.3	11.0	6.7	13.0	11	0.14
$c^2 \downarrow$	2.9	17.5	27.6	-8.0	10.9	1.0	7	0.08
$d \uparrow c^2 \downarrow$	24.8	71.1	70.3	-5.0	35.3	14.0	5	0.12
$d \downarrow c^2 \uparrow$	33.2	-11.2	-12.9	-2.0	-3.6	-5.0	13	0.02
$F = R$	-55.5	-34.8	-44.1	-58.0	-38.2	-42.0	12	0.05

Table 4: The influence of variations from the initial parameter settings for generating TLNDF instances.

We observe that increasing facility opening costs and lowering secondary edge costs leads to a significant increase of the running times. The effects add up when the two deviations are combined. Reducing the facility opening costs has the opposite effect of increasing facility opening costs and reduces the running times. Higher secondary edge costs do not show a similar opposite effect of lowering secondary edge costs.

Reducing the number of possible facility locations reduces the problem complexity and leads to shorter running times. This is not surprising as the variable space and the separation graphs for subsets of constraints involving variables $\mathbf{z}$ are much smaller in this case.

For all parameter settings (except for combined high facility opening and low secondary edge costs), the first, second and third quartile of the running time never increases by more than 35% compared to our default setting. We conclude that the cost structure of the instance has only little influence on the overall performance of our model.

The key values other than the running times reported in Table 4 confirm that increased solution time comes along with more detected violated inequalities, a larger LP gap and less instances for which the LP relaxation of our model provides the optimal integer solution.

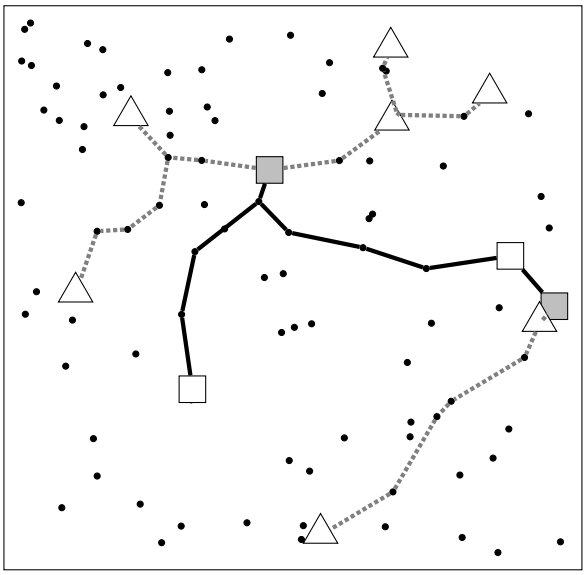
To see the effect of different cost parameters on the solution structure consider Figure 4. Figure 4(a) shows the optimal solution for the default setting (which is the same as the optimal solution for $d \uparrow$, $c^2 \downarrow$ and $F = R$). Figure 4(b) shows the optimal solution for $d \uparrow$ $c^2 \downarrow$. Even though the facility on the right hand side becomes more expensive in $d \uparrow$ the solution does not change. Lowering secondary edge costs makes it profitable to change the path to the secondary customer at the bottom and to open one facility less. Figure 4(c) shows the optimal solution for $c^2 \uparrow$ and $d \downarrow$ $c^2 \uparrow$. It illustrates that less difference between primary and secondary edge costs will reduce the size and number of secondary customers in the secondary subforest. Finally, Figure 4(d) shows the optimal solution for $d \downarrow$. Lowering facility opening costs leads to an additional open facility and increases the number of subtrees in the secondary forest.

6. Conclusions

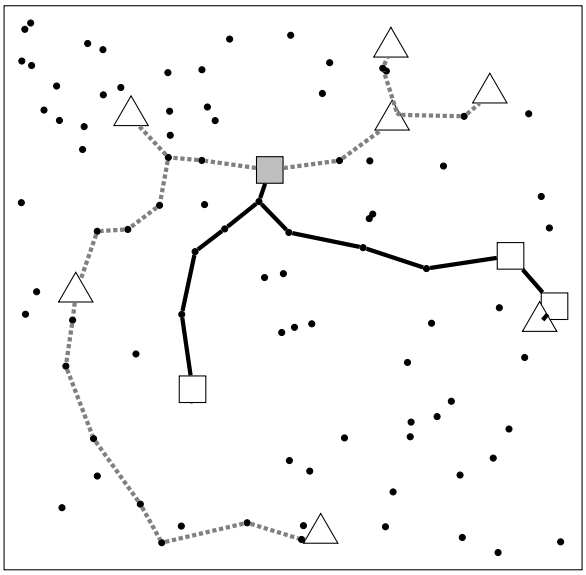
We introduced a new combinatorial optimization problem combining facility location and network design decisions. We considered several mixed integer programming formulations for the problem. Besides formulations derived on the space of original design variables, we also provided three extended formulations: two of them use a flow and a disaggregated flow concept, respectively, and the third one uses a reformulation of the problem on an extended graph in which facility nodes are modeled as arcs.

We provided a theoretical comparison of those models, with respect to the strength of their LP bounds, and in particular showed that the LP bound of the new model based on flows "disaggregated by technology" equals the LP bound of the cut set model on the extended graph. The extensive computational study compares and shows the applicability of the cutting-plane-based counterparts of these models.

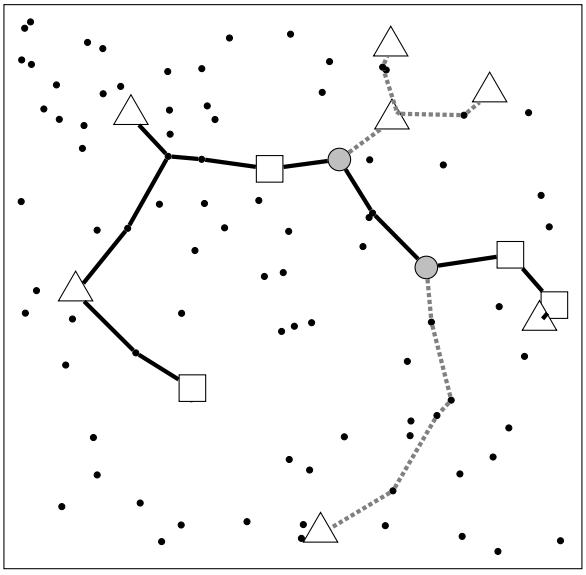
Further interesting topics on TLNDF that have not been covered by this paper include characterizations of facets of the TLNDF polytope, development of approximation algorithms and/or efficient (meta)heuristics. Furthermore, TLNDF can be extended for modeling networks in several stages. Multi-period or two-stage stochastic or recoverable robust approaches are natural extensions of this problem of great relevance in practice.



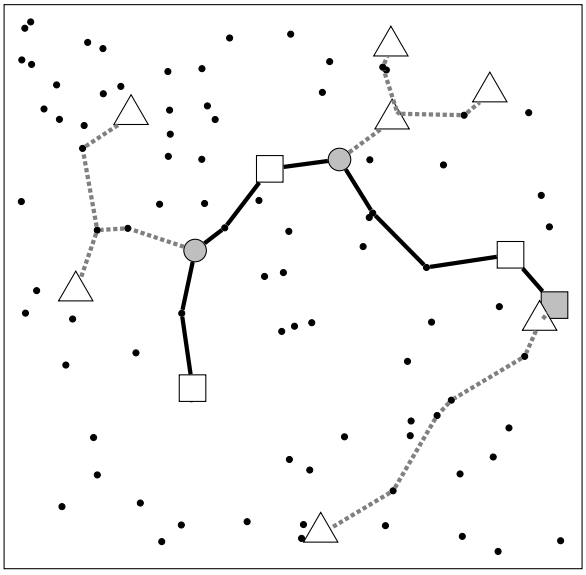
(a) Solution for the default parameter setting, and for $d \uparrow$, $c^2 \downarrow$ and $F = R$.



(b) Solution for parameter setting $d \uparrow c^2 \downarrow$.



(c) Solution for parameter settings $c^2 \uparrow$ and $d \downarrow c^2 \uparrow$.



(d) Solution for parameter setting $d \downarrow$.

Figure 4: Optimal solution of instance `K100-1` for different parameter settings. Solid (dotted grey) lines indicate primary (secondary) edges, a grey fill indicates nodes where a facility is installed.

Bibliography

[1] Balakrishnan, A., Magnanti, T. L., and Mirchandani, P. (1994a). A dual-based algorithm for multi-level network design. *Management Science*, 40(5):567–581.

[2] Balakrishnan, A., Magnanti, T. L., and Mirchandani, P. (1994b). Modeling and heuristic worst-case performance analysis of the two-level network design problem. *Management Science*, 40(7):846–867.

[3] Chopra, S., Gorres, E. R., and Rao, M. R. (1992). Solving the Steiner Tree Problem on a Graph Using Branch and Cut. *INFORMS Journal on Computing*, 4(3):320–335.

[4] Chopra, S. and Tsai, C.-Y. (2002). A branch-and-cut approach for minimum cost multi-level network design. *Discrete Mathematics*, 242(1–3):65–92.

[5] Current, J. R. (1988). Design of a hierarchical transportation network with transshipment facilities. *Transportation Science*, 22(4):270–277.

[6] Current, J. R. and Pirkul, H. (1991). The hierarchical network design problem with transshipment facilities. *European Journal of Operational Research*, 52(3):338–347.

[7] Current, J. R., ReVelle, C. S., and Cohon, J. L. (1986). The hierarchical network design problem. *European Journal of Operational Research*, 27(1):57–66.

[8] Duin, C. and Volgenant, A. (1989). Reducing the hierarchical network design problem. *European Journal of Operational Research*, 39(3):332–344.

[9] Duin, C. and Volgenant, A. (1991). The multi-weighted Steiner tree problem. *Annals of Operations Research*, 33(6):451–469.

[10] Gollowitzer, S., Gouveia, L., and Ljubić, I. (2011). A node splitting technique for two level network design problems with transition nodes. In Pahl, J., Reiners, T., and Voß, S., editors, *Network Optimization - 5th International Conference, INOC 2011, Hamburg, Germany, June 13-16, 2011. Proceedings*, volume 6701 of *Lecture Notes in Computer Science*, pages 57–70. Springer.

[11] Gollowitzer, S. and Ljubić, I. (2011). MIP models for connected facility location: A theoretical and computational study. *Computers & Operations Research*, 38(2):435–449.

[12] Gourdin, E., Labbé, M., and Yaman, H. (2001). Telecommunication and location - a survey. In Drezner, Z. and Hamacher, H., editors, *Facility Location: Applications and Theory*. Springer.

[13] Gouveia, L. and Janssen, E. (1998). Designing reliable tree networks with two cable technologies. *European Journal of Operational Research*, 105(3):552–568.

[14] Gouveia, L. and Telhada, J. (2001). An Augmented Arborescence Formulation for the Two-Level Network Design Problem. *Annals of Operations Research*, 106:47–61.

[15] Gouveia, L. and Telhada, J. (2008). The multi-weighted Steiner tree problem: A reformulation by intersection. *Computers & Operations Research*, 35(11):3599–3611.

[16] IBM (November 22nd 2011). CPLEX. `http://www.ilog.com/products/cplex/`;.

[17] Koch, T., Martin, A., and Voß, S. (2000). SteinLib: An updated library on steiner tree problems in graphs. Technical report, Konrad-Zuse-Zentrum für Informationstechnik Berlin.

[18] Magnanti, T. and Wolsey, L. (1995). Optimal trees. *Handbook in Operations Research and Management Science*, pages 503–615.

[19] Minkoff, M. and Karger, D. R. (1998). The prize collecting steiner tree problem. In *In Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 760–769.

[20] Obreque, C., Donoso, M., Gutiérrez, G., and Marianov, V. (2010). A branch and cut algorithm for the hierarchical network design problem. *European Journal of Operational Research*, 200(1):28–35.

[21] Pirkul, H., Current, J., and Nagarajan, V. (1991). The hierarchical network design problem: A new formulation and solution procedures. *Transportation Science*, 25(3):175–182.

[22] Resende, M. G. C. (November 22nd 2011). Personal website. `http://www2.research.att.com/~mgcr/data/index.html`;.

[23] Sancho, N. (1995). A suboptimal solution to a hierarchial network design problem using dynamic programming. *European Journal of Operational Research*, 83(1):237–244.