MIP Models for Connected Facility Location: A Theoretical and Computational Study

Stefan Gollowitzer   Ivana Ljubić*

Department of Statistics and Decision Support Systems

Faculty of Business, Economics, and Statistics, University of Vienna, Austria

stefan.gollowitzer@gmail.com,   ivana.ljubic@univie.ac.at

September 24, 2009

Abstract

This article comprises the first theoretical and computational study on mixed integer programming (MIP) models for the connected facility location problem (ConFL). ConFL combines facility location and Steiner trees: given a set of customers, a set of potential facility locations and some inter-connection nodes, ConFL searches for the minimum-cost way of assigning each customer to exactly one open facility, and connecting the open facilities via a Steiner tree. The costs needed for building the Steiner tree, facility opening costs and the assignment costs need to be minimized.

We model ConFL using eight compact and two exponential mixed integer programming formulations. We also show how to transform ConFL into the Steiner arborescence problem. A full hierarchy between the models is provided. For the two exponential size models we develop a branch-and-cut algorithm. An extensive computational study is based on two benchmark sets of randomly generated instances with up to 1,300 nodes and 115,000 edges. We empirically compare the presented models with respect to the quality of obtained bounds and the corresponding running time. We report optimal values for all but 16 instances for which the obtained gaps are below 0.6%.

Keywords: Facility Location, Steiner Trees, Mixed Integer Programming Models, LP-relaxations.

*Supported by the Hertha-Firnberg Fellowship of the Austrian Science Foundation (FWF)

1 Preliminary Discussion

Improving the quality of broadband connections is nowadays one of the highest priorities of telecommunication companies. Solutions are sought that search for the optimal way of "pushing" rapid and high-capacity fiber-optic networks closer to the customers. Developing respective models and answering questions related to the design of "last-mile" networks defines a new challenging area of computer science and operations research. The *Connected Facility Location Problem* (ConFL) models the following telecommunication network design problem: Traditional wired local area networks require copper cable connections between end users. To reduce the signal loss, these lines are limited by a maximum distance. To increase the quality of internet communications, telecommunication companies decide to partially or completely replace the existing copper connection by fiber-optic cables. In order to do so, different strategies, known as *fiber-to-the-home* (FTTH), *fiber-to-the-node* (FTTN), *fiber-to-the-curb* (FTTC) or *fiber-to-the-building* (FTTB), are applied.

ConFL models the FTTN / FTTC strategy: Fiber optic cables run to a cabinet serving a neighborhood. End users connect to this cabinet using the existing copper connections. Expensive switching devices are installed in these cabinets. The problem is to minimize the costs by determining positions of cabinets, deciding which customers to connect to them, and how to reconnect cabinets among each other and to the backbone.

1.1 What is Connected Facility Location? - Problem Definition

Gupta et al. [16] define the Connected Facility Location problem as follows: We are given a graph $G = (V, E)$ with a set of customers ($R \subseteq V$), a set of facilities ($F \subseteq V$) and a set of Steiner nodes ($\tilde{S} \subseteq V$) such that $\tilde{S} \cap F = \emptyset$. For all $e \in E$ we are given an edge cost $c_e \geq 0$ and for all $i \in F$ we are given facility opening costs $f_i \geq 0$. Then ConFL consists of finding an assignment of each customer to exactly one facility and connecting these facilities via a Steiner tree. Thereby, assignment costs $c_{ij}, i \in F, j \in R$ are given as the shortest path distance between i and j in G.

The overall costs in this problem are defined as $\sum_{j\ \in R} d_j c_{i(j)j} + \sum_{i \in \mathcal{F}} f_i + \sum_{e \in T} M c_e$, where $d_j \geq 1$ is demand of customer j, $i(j)$ denotes the facility serving j, $\mathcal{F}$ is the set of open facilities, T is the Steiner tree connecting open facilities and $M \geq 1$ is a constant.

Let $S = \tilde{S} \cup F$ denote the set of *core* nodes. Then we can make the following

Observation 1. *Consider a ConFL instance as defined above, where $S \cap R \neq \emptyset$. Without loss of generality, we can transform this instance into an equivalent one in which: a) $\{S, R\}$ is a non-trivial partition of V and b) all customer demands are equal to one.*

The first transformation can easily be done by replacing all the nodes $u \in S \cap R$, with a pair of nodes, $u_1 \in S$ and $u_2 \in R$, connecting all $i \in S$, core neighbors of u, to u_1, and all $i \in F$, facility neighbors of u to u_2, without changing the edge/assignment costs. Finally, if $u \in F \cap R$, we need to connect customer neighbors to u_1 and add the service link $\{u_1, u_2\}$ into E, set its costs to zero and define $f_{u_1} = f_u$.

Demands different from 1 can be set to 1 by adapting the respective assignment costs. We set $c_{ij} := d_j c_{ij} \quad \forall j \in R, \forall i \in F$ and reflect the demand in the cost structure implicitly [26]. Alternatively, we can make d_j copies of customer j, each with demand equal to one (see, e.g., [11]).

For the development of approximation algorithms there are two usual assumptions: The parameter M is used to distinguish between "cheap" assignment and "expensive" core network edges, and c is assumed to be a metric. As we will see later, both these assumptions are not necessary in our approaches. Therefore, we concentrate on a general cost structure.

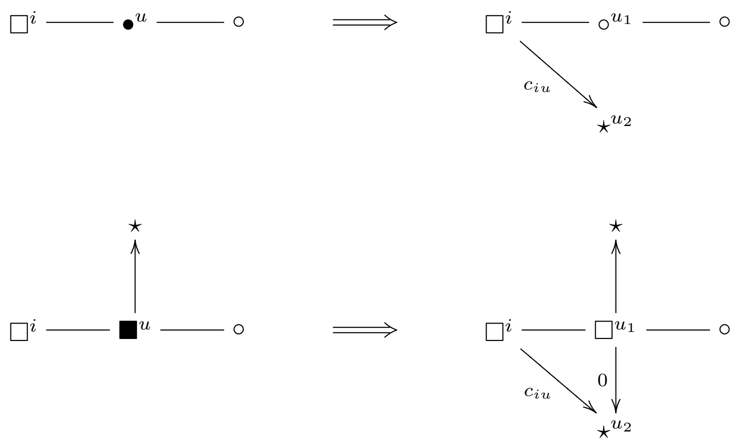


Figure 1: Transformations of nodes a) $u \in \tilde{S} \cap R$ and b) $u \in F \cap R$ where $\star \in R$, $\square \in F$, $\circ \in S$, $\blacksquare \in F \cap R$ and $\bullet \in \tilde{S} \cap R$

Definition 1 (ConFL)**.** For a given undirected graph (V, E) with edge costs $c_e \geq 0, e \in E$, facility opening costs $f_i \geq 0, i \in F$, a disjoint partition $\{S, R\}$ of V with $R \subset V$ being the set of customers, $S \subset V$ the set of possible Steiner nodes and $F \subseteq S$ the set of facilities, in the *Connected Facility Location* problem we search for a subset of open facilities such that:

- each customer is assigned to the closest open facility,
- a Steiner tree connects all open facilities, and
- the sum of assignment, facility opening and Steiner tree costs is minimized.

Optionally, a root $r \in F$ may be considered as an open facility always included in the network. In that case, we speak of the *rooted ConFL*. Obviously, every optimal ConFL solution will be a tree where customers (and possibly the root r) are leaves. In the telecommunications field a "central office" connecting to the backbone network is often predefined and may be considered as a root node active in any feasible solution. Therefore, in the following we assume that the root is given in advance. In Section 3 we show how to solve unrooted instances.

The remainder of this paper is organized as follows: The following section will provide an exhaustive literature review on the topic. In Section 3 we propose ten mixed integer programming models for ConFL and we show a transformation of ConFL into the Steiner Arborescence (SA) problem. In Section 4 we provide a full hierarchy of the models based on the theoretical comparison of the quality of their lower bounds. Section 5 describes a branch-and-cut (B&C) framework that has been used to solve two exponential size formulations. The computational results provided in Section 6 are conducted on two sets of benchmark instances introduced earlier in the literature.

2 Literature Review

The Connected Facility Location Problem has lately started to attract stronger interest in the scientific community. Compared to some closely related problem classes, there is just a small number of papers on the topic. A large share of publications about ConFL comes from the computer science community who present approximation algorithms of different kinds and qualities. The operations research community has developed a small number of heuristic methods. Preliminary results of one of our exact approaches have been published in [26].

Approximation Algorithms A majority of the publications about ConFL concentrate on approximation algorithms. However, not a single one contains computational results. Thus, no conclusion can be drawn to the practical applicability of the described algorithms.
Karger and Minkoff [18] describe an adapted version of the Steiner tree problem. They consider the distribution of single data items from a root to a set of clients. It is not known beforehand which clients demand the data item in question. For each client, there is a known probability to become active and request data. Consider caching nodes at a certain cost, i.e. nodes storing the demanded data for resending it to clients becoming active later-on. The problem of finding a tree with minimal expected cost is equal to the Connected Facility Location Problem. The authors gather the clients into clusters connected to a common facility. Second, they connect these facilities by a Steiner tree. They present a bicriterion approximation algorithm producing a solution of at most 41 times the optimum cost.
Krick et al. [23] present a similar problem as the one in [18], although in an other context. They consider a computer network where clients (corresponding to customers) issue read and write requests. The data for the requests is stored in memory modules (facilities) at a certain cost. Read and write requests are served by the nearest installed memory module for the respective client. To keep data consistent throughout the network, all other memory modules are updated with the latest version. This requires connectivity between the memory modules. Krick et al. give a constant approximation algorithm with a larger constant than the one given by Karger and Minkoff [18].
In the context of reserving bandwidth for virtual private networks, Gupta et al. [16] introduce the term Connected Facility Location. They give a proof for ConFL to be NP-hard. They present a first cut-based integer programming formulation. Their formulation will be described and discussed in detail in Section 3.2. Their approximation algorithm for ConFL has a constant factor of 10.66. For the closely related *rent-or-buy problem* (RoB), in which all nodes are potential facilities with opening costs equal to 0, the algorithm gives an approximation factor of 9.002.
Swamy and Kumar [35] develop a primal-dual approximation algorithm for ConFL, RoB and k-ConFL. The latter comprises the additional restriction that in an optimum solution at most k facilities can be opened. The integer programming formulation used is the same as in Gupta et al. [16]. As results the authors give approximation ratios of 8.55, 4.55 and 15.55 for ConFL, RoB and k-ConFL, respectively.
The approximation factors have been successively improved in Jung et al. [17] and Williamson and van Zuylen [37]. Finally, Eisenbrand et al. [11] combine approximation algorithms for the basic facility location problem and the connectivity problem of the opened facilities by running a what they call *core detouring scheme*. The randomised version of the approximation algorithm gives new best expected approximation ratios for ConFL (4.00), RoB (2.92) and k-ConFL (6.85). The ratios for the de-randomised version are 4.23, 3.28 and 6.98 respectively.

Heuristics and Exact Methods Ljubić [26] describes a hybrid heuristic combining Variable Neighborhood Search with a reactive tabu search method. The author compares it with an exact branch-and-cut approach. The corresponding integer programming model for the branch-and-cut approach will be explained in detail and compared

to other formulations in Section 3. Ljubić [26] also presents two classes of test instances as a result of combining Steiner tree and uncapacitated facility location instances. Results for these instances with up to 1300 nodes are presented.
Tomazic and Ljubić [36] present a Greedy Randomized Adaptive Search Procedure (GRASP) for the ConFL problem. Results for a new set of test instances with up to 120 nodes (facilities plus customers) are presented.

2.1 Related Problems

The Connected Facility Location problem is a combination of two other well-known problems in graph theory. These are the Steiner tree problem (STP) and the Uncapacitated Facility Location problem (UFL). ConFL contains them both as special cases. For a set of possible facility locations connected to a root via a star, we have UFL. In case each customer can only be served by one predefined facility, we know the set of facilities that needs to be opened in advance. Thus, we then have an STP to solve.

Rent-or-buy Problem (RoB) The rent-or-buy problem is often viewed as a special case of the ConFL problem. In the RoB problem facility opening costs are 0 and facilities can be opened anywhere. Thus, also customer nodes can act as facilities and have other customers assigned to them. The cost for each edge in a solution to the RoB depends on its adjacent nodes. If an edge is used to assign a customer to a facility, only assignment costs are incurred. If an edge connects two facilities, a comparatively higher cost, i.e. M times the assignment cost, has to be paid for.

The (general) Steiner tree-star problem ((G)STS) The Steiner tree-star problem was introduced by Lee et al. [24]. It arises in the design of some specific telecommunication networks, where bridging occurs. The Steiner tree-star problem is the following: Given a graph with disjoint sets of possible facility nodes and customers, we want to find a minimum cost tree such that each customer is assigned to a facility and that all open facilities are connected by a Steiner tree. Facility opening costs are incurred for any facility in the solution tree, regardless of whether any customers are assigned to it or not.
Exact methods to solve the STS problem have been described by Lee et al. [24, 25], a tabu search based heuristic was developed by Xu et al. [39]. Khuller and Zhu [19] introduced the *general* Steiner tree-star problem. There, the sets of possible facilities and customers must not be disjoint. Nodes can act in both ways and an open facility can serve the customer in its own place at no additional cost. Khuller and Zhu [19] derive two approximation algorithms for the general STS with approximation factors of 5.16 and 5 respectively.

General Connected Facility Location (GConFL) Bardossy and Raghavan [4] develop a dual-based local search (DLS) heuristic for a family of problems combining facility location decisions with connectivity requirements, namely the (general) Steiner tree-star, ConFL and RoB. They introduce the general ConFL problem, into which any of the aforementioned 4 problem classes can be transformed. The presented DLS heuristic works in two phases. After applying dual-ascent in order to get a lower and upper bound in the first phase, in the second phase a local search procedure is carried out on the facilities and Steiner nodes selected before. Computational results for instances with up to 100 nodes are presented. Running time and the quality of solutions of Ljubić' VNS heuristic and DLS are compared for the set of instances introduced in [26].

3 (M)ILP Formulations for ConFL

It is well known that the MIP formulations for Steiner trees and related problems provide stronger lower bounds when defined on directed graphs (see, e.g., [8, 14]). In this section we will first describe how to transform undirected instances for ConFL into directed ones. A range of (M)ILP formulations for the ConFL will be presented afterwards. As the exponential size formulations are hard to implement by means of a modeling language, various compact MIP formulations will be described in this section as well. They are either flow formulations or based on sub-tour elimination constraints.

3.1 Transformation Into Directed Graphs

Throughout this paper, an arc from i towards j will be denoted by ij, and the corresponding undirected edge by $\{i, j\}$. Let (V, E) be a given instance of ConFL with $\{S, R\}$ being a partition of V and $F \subseteq S$. This instance can be transformed into a bidirected instance (V, A) as follows (cf. [36]):

- Replace core edges $e \in E$ with $e = \{i, j\}, i, j \in S$ by two directed arcs $ij \in A$ and $ji \in A$ with cost $c_{ij} = c_{ji} = c_e$.
- Replace assignment edges $e \in E$ with $e = \{j, k\}, j \in F, k \in R$ by an arc $jk \in A$ with cost $c_{jk} = c_e$ respectively.

Rooting Unrooted Instances To obtain an optimal solution for a directed, unrooted instance (V, A) by solving a model for rooted instances we adapt the input instance and the corresponding model as follows:

- Expand the set of facilities F by adding an artificial root r to $V' = V \cup \{r\}$ with cost $f_r = 0$.
- Expand the set of arcs by adding an arc rj for all core nodes $j \in F$ with $c_{rj} = 0$.
- Limit the number of arcs emanating from the root r to 1, e.g. add the additional constraint $\sum_{j \in F} x_{rj} \leq 1$.

In the remainder of this paper we will refer to the Connected Facility Location problem on directed graphs as the following:

Definition 2 (ConFL on directed graphs)**.** We are given a directed graph (V, A) with edge costs $c_{ij}, ij \in A$, facility opening costs $f_i, i \in F$ and a disjoint partition $\{S, R\}$ of V with $R \subset V$ being the set of customers, $S \subset V$ the set of possible Steiner tree nodes, $F \subset S$ the set of facilities, and the root node $r \in F$. Find a subset of open facilities such that

- each customer is assigned to exactly one open facility,
- a Steiner arborescence rooted in r connects all open facilities, and
- the cost defined as the sum of assignment, facility opening and Steiner arborescence cost, is minimized.

To model the problem, we will use the following binary variables:

$$x_{ij} = \begin{cases} 1, & \text{if } ij \text{ belongs to the solution} \\ 0, & \text{otherwise} \end{cases} \quad \forall ij \in A \qquad z_i = \begin{cases} 1, & \text{if } i \text{ is open} \\ 0, & \text{otherwise} \end{cases} \quad \forall i \in F$$

We will use the following notation: $A_R = \{ij \in A \mid i \in F, j \in R\}$, $A_S = \{ij \in A \mid i, j \in S\}$. Furthermore, for any $W \subset V$ we denote by $\delta^-(W) = \{ij \in A \mid i \notin W, j \in W\}$ and $\delta^+(W) = \{ij \in A \mid i \in W, j \notin W\}$.

3.2 Cut-Based Formulations

In the literature there are two different exponential size formulations for ConFL. They are both based on cuts and differ in strength.

Cut Set Formulation of Gupta et al. [16] Gupta et al. [16] first introduced an undirected ILP formulation for ConFL. To ensure comparability, a directed version will be presented here. One might think of any ConFL solution as a Steiner arborescence rooted at r with customers as leaves and with node weights that need to be payed for any node that is adjacent to a customer. Therefore, instead of requiring connectivity among open facilities and assignment of customers to open facilities, we are going to ask for the solution that ensures a directed path between r and any customer $j \in R$, using the arcs from A.
The cut-based model reads then as follows:

$$(CUT_R) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

$$\text{s.t.} \quad \sum_{uv \in \delta^-(U)} x_{uv} \geq \sum_{j \in U : jk \in A_R} x_{jk} \quad \forall U \subseteq S \setminus \{r\}, U \cap F \neq \emptyset, \ \forall k \in R \tag{1}$$

$$\sum_{jk \in A_R} x_{jk} = 1 \quad \forall k \in R \tag{2}$$

$$x_{jk} \leq z_j \quad \forall jk \in A_R \tag{3}$$

$$z_r = 1 \tag{4}$$

$$x_{ij} \in \{0,1\} \quad \forall ij \in A \tag{5}$$

$$z_i \in \{0,1\} \quad \forall i \in F \tag{6}$$

The objective comprises the cost for the Steiner arborescence ($\sum_{ij \in A_S} x_{ij} c_{ij}$), the cost to connect customers to facilities (that we also refer to as *assignment cost*, i.e. $\sum_{ij \in A_R} x_{ij} c_{ij}$) and the facility opening cost ($\sum_{i \in F} z_i f_i$). Constraints (2) ensure that every customer is connected to at least one facility, constraints (3) ensure that each facility is opened if customers are assigned to it, equation (4) defines the root node. Inequalities (1) represent the set of cuts. For every subset $U \subseteq S \setminus \{r\}$ and for each customer $k \in R$, an open arc from a facility in U toward j, necessitates a directed path from r towards U. Constraints (2) can be replaced by inequality in case that $c_{ij} > 0$, for all $ij \in A_R$. Furthermore, the same optimization problem with continuous assignment variables x_{ij}, for all $ij \in A_R$, returns an optimal ConFL solution. This is because the underlying assignment matrix is totally unimodular, whenever z_i values are fixed to zero or one.

Observation 2. *Using equations* (2)*, we can re-write constraints* (1) *as follows:*

$$\sum_{uv \in \delta^-(U)} x_{uv} + \sum_{jk \in A_R : j \notin U} x_{jk} \geq 1, \quad \forall U \subseteq S \setminus \{r\}, U \cap F \neq \emptyset, \ \forall k \in R. \tag{7}$$

Denote by $W = S \setminus U$, and let $A_S^W := \delta^+(W) \cap A_S$ and $A_R^W = \delta^+(W) \cap A_R$. Now, we can interpret these constraints as follows: every cut separating customer k from r (involving all arcs from $A_S \cup A_R$) has to be greater than or equal to one, i.e.:

$$\sum_{uv \in A_S^W} x_{uv} + \sum_{jk \in A_R^W} x_{jk} \geq 1, \quad \forall W \subseteq S, \ r \in W, \ W \cap F \neq F, \ \forall k \in R.$$

Figure 2 illustrates an example of these cut set inequalities.

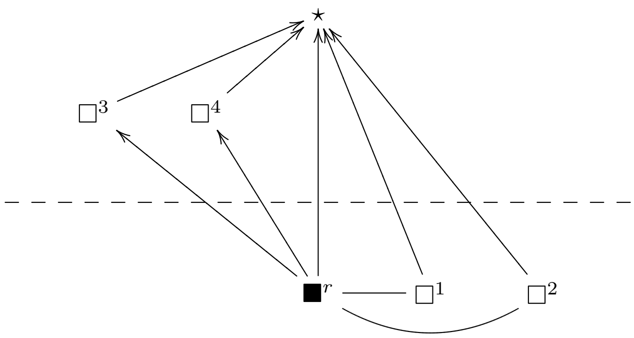


Figure 2: Graphic illustration for cut inequalities (2). $W = \{r, 1, 2\}$, $U = \{3, 4\}$

According to the result of Swamy and Kumar [35], the integrality gap of the LP-relaxation of (CUT_R) is not greater than 8.55, if c is a metric, and core costs are M times more expensive than the assignment costs ($M \geq 1$).

Ljubić' Cut Set Formulation Ljubić [26] presents a slightly different formulation where the cuts are defined according to the open facilities:

$$(CUT_F) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

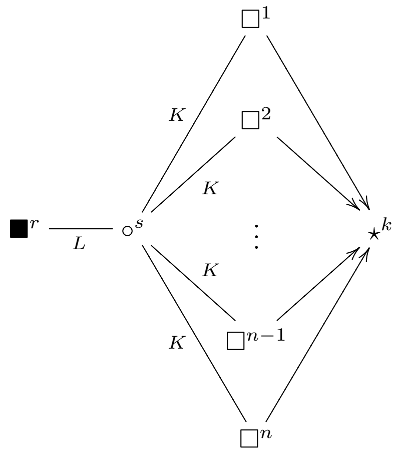
$$\text{s.t.} \quad \sum_{uv \in \delta^-(W)} x_{uv} \geq z_i \qquad \forall W \subseteq S \setminus \{r\},\ \forall i \in W \cap F \neq \emptyset \tag{8}$$

$$(2) \text{ - } (6)$$

Lemma 1. *There are instances for which the values of the LP-relaxation of the CUT_F model can be as bad as $\frac{1}{|F|-1} OPT$, where OPT denotes the integer optimal solution.*

Proof. Example 1 illustrates such a situation. In this example $n := |F| - 1$. The optimal solution value for the LP relaxation of CUT_L is $v_{LP}(CUT_L) = \frac{L}{n} + K + 3$ and the optimal integer solution value is $OPT = L + K + 3$. For $K >> L$, we get $\frac{vlpCUT_L}{OPT} \approx \frac{1}{n}$. □

Example 1. The cost structure is as follows: all facility opening and assignment costs are 1. $c_{rs} = L$ and $c_{si} = K$, for all $i \in \{1, \ldots, n\}$.



3.3 Flow-based Formulations

Extending flow formulations for the (prize-collecting) Steiner tree problem (see, e.g., [27, 34]), several ways to model ConFL as a flow problem are possible. One option is to have a flow from the root to each customer. Alternatively, flow can be allowed from the root node to open facilities only, with additional constraints ensuring customers to be assigned to an open facility. Further it is possible to consider just one single commodity or separate commodities for each customer or facility respectively.

In the following we propose six different flow formulations for ConFL. The strength of the different formulations is discussed later in Section 4.

Single-Commodity Flow Between Root and Facilities This single commodity-flow formulation with flow between root node and facilities is an extension of the single-commodity flow formulation for the prize-collecting Steiner tree problem (see, e.g., Ljubić [27]). The amount of flow terminating in a facility is linked to the variable indicating whether the facility is open or not. For all $ij \in A_S$, continuous variable g_{ij} denotes the amount of flow that is simultaneously routed from r toward all open facilities over arc ij.

$$(SCF_F) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

$$\text{s.t.} \quad \sum_{ji \in A_S} g_{ji} - \sum_{ij \in A_S} g_{ij} = \begin{cases} z_k & i = k,\, k \in F \\ -\sum_{k \in F} z_k & i = r \\ 0 & i \in S \setminus \{F\} \end{cases} \qquad \forall i \in S \tag{9}$$

$$0 \leq g_{ij} \leq (|F| - 1) \cdot x_{ij} \quad \forall ij \in A_S \tag{10}$$

$$(2)\text{ - }(6)$$

Constraints (9) ensure that each facility $j \in F$ receives z_j units of flow from the root. The coupling constraints (10) ensure that on every arc ij, there is enough capacity to simultaneously route that flow. They also force an arc ij to be installed if there is a flow sent through it. Model SCF_F comprises $O(|A|)$ constraints and $O(|A|)$ binary and continuous variables.

The following result is due to the usage of "big-M" constraints in (10):

Lemma 2. *There are instances for which*

a) the values of the LP-relaxation of the SCF_F model can be as bad as $\frac{1}{|F|-1} OPT$, and

b) the ratio $\frac{\upsilon_{LP}(SCF_F)}{\upsilon_{LP}(CUT_F)} \approx \frac{1}{|F|}$.

Proof. a) The same example given in Figure 1 provides $\upsilon_{LP}(SCF_F) = \frac{L}{n} + \frac{K}{n} + 3$ which gives ratio $\frac{\upsilon_{LP}(SCF_F)}{OPT} \approx \frac{1}{|F|-1}$.

b) If $K >> L$ in the same example, we obtain $\frac{\upsilon_{LP}(SCF_F)}{\upsilon_{LP}(CUT_F)} = \frac{\frac{L}{n} + \frac{K}{n} + 3}{\frac{L}{n} + K + 3} = \frac{1}{|F|-1} \approx \frac{1}{|F|}$.

□

Single-Commodity Flow between Root and Customers We now consider single commodity-flow from the root node to each of the customers. At the expense of more flow variables this allows us to drop constraints (2) used in SCF_F:

$$(SCF_R) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

$$\text{s.t.} \quad \sum_{ji \in A_S} f_{ji} - \sum_{ij \in A} f_{ij} = \begin{cases} 1 & i \in R \\ -|R| & i = r \\ 0 & i \in S \setminus \{r\} \end{cases} \qquad \forall i \in V \tag{11}$$

$$0 \leq f_{ij} \leq |R| \cdot x_{ij} \quad \forall ij \in A \tag{12}$$

$$(3) \text{ - } (6)$$

Constraints (11) ensure that each customer receives one unit of flow from the root node and constraints (12) are similar to (10). However, one easily observes that, although redundant for the MIP formulation, assignment constraints (2) can strengthen the quality of lower bounds. We denote by SCF_R^+ the formulation SCF_R extended by (2). Models SCF_R and SCF_R^+ comprise $O(|A|)$ constraints and $O(|A|)$ binary variables.

Lemma 3. *There are instances for which*

a) the values of the LP-relaxation of the SCF_R (SCF_R^+) model can be as bad as $\frac{1}{|R|} OPT$, and

b) the ratio $\frac{v_{LP}(SCF_R)}{v_{LP}(CUT_R)} \approx \frac{1}{|R|}$.

Multi-Commodity Flow with One Commodity per Facility The two flow formulations presented above can be improved by disaggregation of commodities.

Choosing one commodity per facility, each variable indicating an open facility is linked to a distinct commodity. A multi-commodity flow formulation with one commodity per facility is given by:

$$(MCF_F) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

$$\text{s.t.} \quad \sum_{ji \in A_S} g_{ji}^k - \sum_{ij \in A_S} g_{ij}^k = \begin{cases} z_k & i = k \\ -z_k & i = r \\ 0 & i \neq k, r \end{cases} \qquad \forall i \in S \quad \forall k \in F \tag{13}$$

$$0 \leq g_{ij}^k \leq x_{ij} \qquad \forall ij \in A_S, \ \forall k \in F \tag{14}$$

$$(2) \text{ - } (6)$$

Equations (13) are the flow preservation constraints defining the flow from the root node to each facility. These constraints ensure the existence of a connected path from r to every open facility. The stronger coupling constraints ensure that the arc is open if a flow is sent through it. Formulation MCF_F comprises $O(|A_S||F| + |A_R|)$ constraints, $O(|A_S||F|)$ continuous and $O(|A|)$ binary variables.

Multi-Commodity Flow with One Commodity per Customer Another choice for the commodities we use, is the set of customers. Assigning a commodity of size 1 to each customer allows to remove the $\mathbf{z}$ variables from the flow preservation constraints. Using one commodity per customer, ConFL can be stated as:

$$(MCF_R) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

$$\text{s.t.} \quad \sum_{ji \in A} f_{ji}^k - \sum_{ij \in A} f_{ij}^k = \begin{cases} 1 & i = k \\ -1 & i = r \\ 0 & i \neq k, r \end{cases} \qquad \forall i \in V \quad \forall k \in R \tag{15}$$

$$0 \leq f_{ij}^k \leq x_{ij} \qquad \forall ij \in A, \ \forall k \in R \tag{16}$$

$$(3) - (6)$$

Formulation MCF_R comprises $O(|A||R|)$ constraints, $O(|A||R|)$ continuous and $O(|A|)$ binary variables.

Observation 3. *Variables x_{ij}, $ij \in A_R$, are redundant in this formulation, as every LP-optimal solution of MCF_R also satisfies:*

$$f_{jk}^l = \begin{cases} x_{jk}, & \text{if } l = k \\ 0, & \text{otherwise} \end{cases} \qquad \forall l \in R, \quad \forall jk \in A_R.$$

Therefore, constraints (2) are redundant, for both, the MCF_R model and its LP-relaxation. However, we keep variables $x_{ij}, ij \in A_R$ in this model for better readability.

3.3.1 Strong Formulations Comprising Common Flow Variables

Polzin and Daneshmand [34] have developed a formulation which they call *Common Flow* formulation for the Steiner arborescence problem. It is based on a disaggregation of multi commodity-flow formulation with additional 4-index variables. These variables indicate the common flow from the root towards any pair of terminals. For ConFL this gives two choices on the common flows considered, towards facilities or towards customers. The variant, where common flows towards facilities are considered, is an extension of MCF_F, the other one is an augmentation of MCF_R and it is the strongest one among all formulations presented in this paper (see Section 4).

Common Flow Between Root and Facilities Let $\bar{g}_{ij}^{kl}$ denote the common flow towards facilities k and l, $k, l \in F, k \neq l$, over an arc ij. Then a MIP formulation of ConFL using common flows from the root to facilities is given by:

$$(CF_F) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

$$\text{s.t.} \quad \sum_{ji \in A_S} g_{ji}^k - \sum_{ij \in A_S} g_{ij}^k = \begin{cases} z_k & i = k \\ -z_k & i = r \\ 0 & i \neq k, r \end{cases} \qquad \forall i \in S \quad \forall k \in F \tag{17}$$

$$\sum_{ij \in A_S} \bar{g}_{ij}^{kl} - \sum_{ji \in A_S} \bar{g}_{ji}^{kl} \leq \begin{cases} \min(z_k, z_l) & i = r \\ 0 & \forall i \in S \setminus \{r\} \end{cases} \qquad \forall i \in S \quad \forall k, l \in F \tag{18}$$

$$0 \leq \bar{g}_{ij}^{kl} \leq \min(g_{ij}^k, g_{ij}^l) \qquad \forall ij \in A_S, \quad \forall k, l \in F \tag{19}$$

$$0 \leq g_{ij}^k + g_{ij}^l - \bar{g}_{ij}^{kl} \leq x_{ij} \qquad \forall ij \in A_S, \quad \forall k, l \in F \tag{20}$$

$$(2) - (6)$$

Constraints (17) are flow preservation constraints as in MCF_F. Constraints (18) ensure that the common flow from the root toward facilities k and l is non-increasing. Inequalities (19) define the relation between common flow and commodity flow variables. The coupling constraints (20) ensure that the arc is installed whenever there is a flow sent through it.

Formulation CF_F comprises $O(|A_S||F|^2)$ constraints, $O(|A_S||F|^2)$ continuous and $O(|A|)$ binary variables.

Common Flow Between Root and Customers Starting from the MCF_R model, we can now derive the other common flow formulation. Let $\bar{f}_{ij}^{kl}$ denote the common flow towards customers k and l, $k \neq l$. Then the common flow formulation with flows from the root to customers is given by:

$$(CF_R) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

$$\text{s.t.} \quad \sum_{ji \in A} f_{ji}^k - \sum_{ij \in A} f_{ij}^k = \begin{cases} 1 & i = k \\ -1 & i = r \\ 0 & i \neq k, r \end{cases} \qquad \forall k \in R \tag{21}$$

$$\sum_{ij \in A_S} \bar{f}_{ij}^{kl} - \sum_{ji \in A_S} \bar{f}_{ji}^{kl} \leq \begin{cases} 1 & i = r \\ 0 & \forall i \in S \setminus \{r\} \end{cases} \qquad \forall i \in V \quad \forall k, l \in R \tag{22}$$

$$0 \leq \bar{f}_{ij}^{kl} \leq \min(f_{ij}^k, f_{ij}^l) \quad \forall ij \in A, \quad \forall k, l \in R \tag{23}$$

$$0 \leq f_{ij}^k + f_{ij}^l - \bar{f}_{ij}^{kl} \leq x_{ij} \qquad \forall ij \in A, \quad \forall k, l \in R \tag{24}$$

$$(3) - (6)$$

Constraints (21) are flow preservation constraints as in MCF_R. Inequalities (22) ensure that the common flow from the root to customers k and l is non-increasing. Constraints (23)-(24) are equivalents of (19) - (20). Formulation CF_R comprises $O(|A||R|^2)$ constraints, $O(|A||R|^2)$ continuous and $O(|A|)$ binary variables.

3.4 Formulations Based on Sub-tour Elimination Constraints

Another well-studied group of formulations for problems on graphs are based on sub-tour elimination. We present here one compact and one exponential size model.

Miller-Tucker-Zemlin Formulation One very simple strategy for sub-tour elimination was proposed by Miller, Tucker and Zemlin [32] and has been applied to a number of problems, including (Asymmetric) Traveling Salesman, Vehicle Routing, Minimum Spanning Tree and Steiner Tree Problem [9, 10, 15, 33]. In addition to x and z variables, we now introduce *level variables* $u_i \geq 0$, for all $i \in S$, determining the level of node i in the tree solution. The root node is assigned to the level zero.

Using the lifted Miller-Tucker-Zemlin (MTZ) constraints (see, e.g., [9]), ConFL can be stated as:

$$(MTZ) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

$$\sum_{i \in S \setminus \{k\}} x_{ij} \geq x_{jk} \qquad \forall j \in S \setminus \{r\}, k \in V \tag{25}$$

$$(|S|-2) \cdot x_{ji} + |S| \cdot x_{ij} + u_i \leq u_j + |S| - 1 \quad \forall ij \in A_S \tag{26}$$

$$u_r = 0 \tag{27}$$

$$u_i \geq 0 \qquad \forall i \in S \setminus \{r\} \tag{28}$$

$$(2) \text{ - } (6)$$

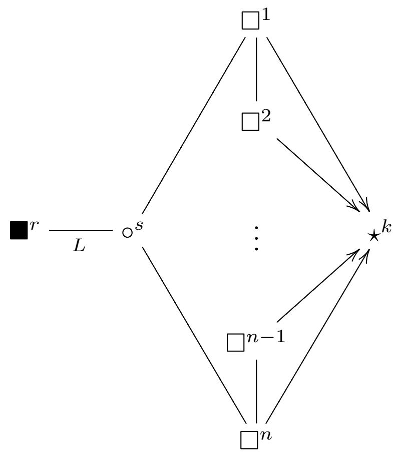
Constraints (25) limit the out-degree of a node by its in-degree. Constraints (26) are Miller-Tucker-Zemlin sub-tour elimination constraints, setting the difference $u_j - u_i$ for an open arc ij to exactly 1, thereby eliminating cycles in the Steiner tree connecting the facilities. Constraint (27) sets the level of the root node to zero.

Formulation MTZ comprises $O(|A|)$ constraints, $O(|S|)$ continuous and $O(|A|)$ binary variables. The formulation is small in the number of constraints and variables, compared to the aforementioned formulations based on flows or cut sets. The quality of the lower bounds, i.e. the strength of the formulations will be analyzed in the subsequent section.

Lemma 4. *The values of the LP-relaxation of the MTZ model can be arbitrarily bad.*

Proof. Consider Example 2: The LP-solution opens each facility with $1/n$, and builds one directed cycle of $\{s\} \cup \{1, \ldots, n\}$ where for each arc ij in the cycle $x_{ij} = 1/n$. It assigns $v_{LP}(MTZ) = 4 + \frac{1}{n}$ and $OPT = L + 4$, which gives ratio $\frac{v_{LP}(MTZ)}{OPT} \approx \frac{1}{L}$. □

Example 2. In this example $n := |F| - 1$. The cost structure is as follows: all facility opening, arc opening and assignment costs are 1, except for $c_{rs} = L$, where $L \gg 0$ is an arbitrarily large number.



Formulation Based on Generalized Sub-tour Elimination Constraints To model the Steiner tree in the core network, one might consider another formulation extended by the following node variables:

$$w_i = \begin{cases} 1, & \text{if } i \text{ belongs to the solution,} \\ 0, & \text{otherwise} \end{cases}, \qquad \forall i \in S$$

Such model has been used for the node-weighted Steiner tree problems (see, e.g., [13, 29, 30]).

$$(GSEC) \quad \min \sum_{ij \in A} x_{ij} c_{ij} + \sum_{i \in F} z_i f_i$$

$$\sum_{uv \in A: u,v, \in U} x_{uv} \leq \sum_{i \in U \setminus \{k\}} w_i \quad \forall U \subset S, \forall k \in U \tag{29}$$

$$\sum_{uv \in A} x_{uv} = \sum_{i \in S \setminus \{r\}} w_i \tag{30}$$

$$w_i \geq z_i \quad \forall i \in F \tag{31}$$

$$0 \leq w_i \leq 1 \quad \forall i \in S \tag{32}$$

(2) - (6)

Equality (30) ensures that the set of edges is equal to the number of selected nodes minus one. In order to ensure the tree structure, sub-tours are eliminated by deploying constraints (29). Since facility nodes can also be used only as Steiner nodes, in which case $w_i = 1$ and $z_i = 0$, inequalities (31) must hold.

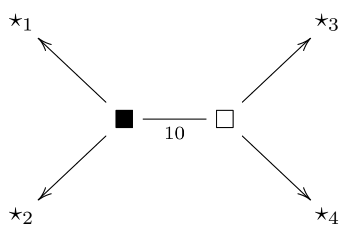
We will see in the following section that the results known for Steiner trees with respect to *GSEC*, directly apply to ConFL.

4 Polyhedral Comparison

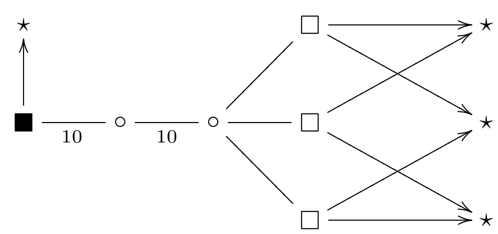
In this section we provide a theoretical comparison of the MIP models described above with respect to optimal values of their LP-relaxations. The examples given below are used in the proofs of this section. These examples employ the following notation:

$\blacksquare$ represents the root node, $\circ$ represents a Steiner node. $\square^l$ represents a facility with label l. $\star$ represents a customer. Arc costs different from 1 are displayed next to the respective arc. Facility opening, assignment and core costs are all 1 in all examples, unless stated differently. All the values od facility node variables stated in the descriptions below refer to optimal LP solutions. The core network is presented as undirected graph, except in Example 5.

Example 3. The underlying network is given in the figure below. The facility node variable is 1/4 for SCF_R and 1 for all other models.



Example 4. This example is a small variant of Example 1. It will show the weakness of models where the flows are only defined on the core subgraph A_S. Facility node variables are 1/8 for SCF_R and 1/2 for all other models.



Example 5. The core network is directed and there is exactly one customer that can be assigned to each facility. Thus, every facility needs to be open in a feasible solution. The underlying graph is shown in Figure 3. Facility node variables are 1/5 for SCF_R and 1 for all other models. A version of this example was described by Polzin and Daneshmand [34].

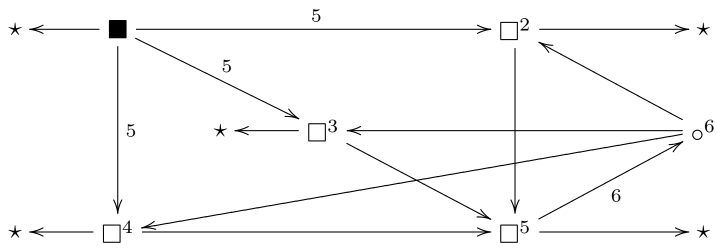
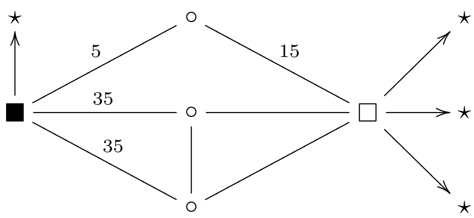
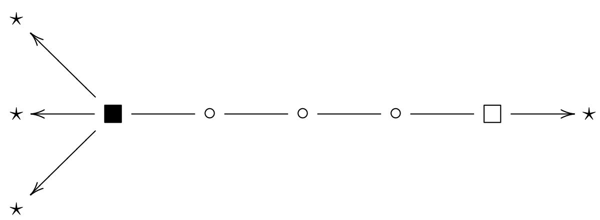


Figure 3: Example 5

Example 6. The example shown below will demonstrate the weakness of Miller-Tucker-Zemlin constraints. The facility node variable is 1/4 for SCF_R and 1 for all other models. In the LP solution for model MTZ there is a cycle consisting of the arcs of weight 1. The open facility is not connected to the root.



Example 7. The example shown below will demonstrate the weakness of "big-M" constraints in the models comprising single commodity flow. The facility node variable is 1/4 for SCF_R and 1 for all other models.



	Ex. 3	Ex. 4	Ex. 5	Ex. 6	Ex. 7
MTZ	16	18	20	9	10
SCF_F	11	$14\frac{3}{8}$	$14\frac{1}{5}$	16	8
SCF_R	$7\frac{1}{4}$	$18\frac{1}{8}$	7	$17\frac{1}{4}$	$3\frac{1}{4}$
SCF_R^+	11	$22\frac{1}{4}$	$14\frac{1}{5}$	21	7
MCF_F	16	18	22	26	10
MCF_R	16	28	22	26	10
CF_F	16	18	24	26	10
CF_R	16	28	24	26	10

Table 1: Optimal LP solutions for Examples 3 - 7

Let $\upsilon_{LP}(.)$ denote the optimal solution value of the LP relaxation of a given model. By comparing the optimal LP solution values for the aforementioned examples, provided by the models in Section 3, we can state the following

Lemma 5. *The following pairs of formulations are incomparable with respect to the quality of lower bounds:*

a) MTZ and SCF_F,

b) MTZ and SCF_R (SCF_R^+),

c) SCF_F and SCF_R (SCF_R^+),

d) SCF_R (SCF_R^+) and MCF_F,

e) SCF_R (SCF_R^+) and CF_F,

f) MCF_R and CF_F.

Proof. a) In Example 3 we have $\upsilon_{LP}(SCF_F) = 11 < 16 = \upsilon_{LP}(MTZ)$ and in Example 6 we have $\upsilon_{LP}(MTZ) = 9 < 10 = \upsilon_{LP}(SCF_F)$.

b) In Example 3 we have $\upsilon_{LP}(SCF_R) = 7.25 < \upsilon_{LP}(SCF^+_R) = 11 < \upsilon_{LP}(MTZ) = 16$ and in Example 6 we have $\upsilon_{LP}(MTZ) = 9 < 17.25 = \upsilon_{LP}(SCF_R) < \upsilon_{LP}(SCF^+_R) = 21$.

c) In Example 4 we have $\upsilon_{LP}(SCF_F) = 14.325 < 18.125 = \upsilon_{LP}(SCF_R)$ and in Example 7 we have $\upsilon_{LP}(SCF_R) = 3.25 < \upsilon_{LP}(SCF^+_R) = 7 < \upsilon_{LP}(SCF_F) = 8$.

d) For Example 4 we have $\upsilon_{LP}(SCF_R) = 18.125 > 18 = \upsilon_{LP}(MCF_F)$. For Example 3 we have $\upsilon_{LP}(SCF_R) = 7.25 < \upsilon_{LP}(SCF_R) = 11 < \upsilon_{LP}(MCF_F) = 16$.

e) For Example 3 we have $\upsilon_{LP}(SCF_R) = 7.25 < \upsilon_{LP}(SCF^+_R) = 11 < \upsilon_{LP}(CF_F) = 16$, for Example 4 we have $\upsilon_{LP}(CF_F) = 18 < \upsilon_{LP}(SCF_R) = 18.125 < \upsilon_{LP}(SCF^+_R) = 22.25$.

f) Consider Examples 4 and 5. For Example 4 we have $\upsilon_{LP}(CF_F) = 18 < 28 = \upsilon_{LP}(MCF_R)$, for Example 5 we have $\upsilon_{LP}(MCF_R) = 22 < 24 = \upsilon_{LP}(CF_F)$.

□

Denote by $\mathcal{P}_.$ the polytope of the LP-relaxation of any of the MIP models described above, and with $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_.)$ the natural projection of that polytope onto the space of variables $\mathbf{x}$ and $\mathbf{z}$.

Lemma 6. *The following results hold:*

a) $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{CF_F}) \subset Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MCF_F}) \subset Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{SCF_F})$*, and*

b) $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{CF_R}) \subset Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MCF_R}) \subset Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{SCF^+_R}) \subset Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{SCF_R})$.

Proof. The results follow immediately from the corresponding results for Steiner trees, see e.g., [34]. Instances that prove the strict inclusion can be found in Table 1. □

Lemma 7. *The following results hold:*

a) $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MCF_F}) = \mathcal{P}_{CUT_F} = Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{GSEC})$*, and*

b) $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MCF_R}) = \mathcal{P}_{CUT_R}$.

Proof.

a) The first equality follows from the min-cut max-flow theorem, the second one follows from the related result for node-weighted Steiner trees, see e.g. [30].

b) This result follows from the min-cut max-flow theorem.

□

Lemma 8. *The following results hold:*

a) $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MCF_R}) \subset Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MCF_F})$ *and*

b) $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{CF_R}) \subset Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{CF_F})$.

Proof.

a) According to Lemma 7, it is enough to show this relationship by comparing $\mathcal{P}_{CUT_R}$ and $\mathcal{P}_{CUT_F}$. Then it is easy to see that every solution $(\mathbf{x}', \mathbf{z}') \in \mathcal{P}_{CUT_R}$ also belongs to $\mathcal{P}_{CUT_F}$. Example 4, with $\upsilon_{LP}(CUT_R) = 28 > 18 = \upsilon_{LP}(CUT_F)$, proves that the opposite is not true.

b) $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{CF_R}) \subseteq Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{CF_F})$**:** Let $(\mathbf{f}', \bar{\mathbf{f}}', \mathbf{x}', \mathbf{z}')$ be in $\mathcal{P}_{CF_R}$. We define the capacities on the subgraph $G_S = (S, A_S)$ as x_{ij}, for all $ij \in A_S$. Since $x_{ij} = max_{k \in R} f_{ij}^k$, and $z_i = max_{ij \in A_R} x_{ij}$, there will be enough capacity to independently route z_i units of flow, for all $i \in F$, such that $z_i > 0$. Now, we are going to construct $(\mathbf{g}, \bar{\mathbf{g}}, \mathbf{x}, \mathbf{z}) \in \mathcal{P}_{CF_F}$ as follows: We fix the ordering of the outgoing arcs of every node $i \in S$ and then apply an adapted Ford-Fulkerson maximum flow algorithm. To define $\mathbf{g}$, we send z_i units of flow from r towards $i \in F$, for all $i \in F$ such that $z_i > 0$. When searching for augmenting paths, we always follow the fixed ordering. Therefore, the outgoing arcs of a node always get saturated in the same order, independently on the commodity under consideration. It follows directly from construction that the common flow $\bar{\mathbf{g}}$ for any pair of facilities k and l, once it splits up, will never meet again, i.e., ineqalities (18) will be satisfied.

$Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{CF_F}) \nsubseteq Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{CF_R})$**:** Consider Example 4, where $\upsilon_{LP}(CF_R) = 28 > 18 = \upsilon_{LP}(CF_F)$.

□

Lemma 9. *Formulation MCF_F (i.e., CUT_F, GSEC) is strictly stronger than formulation MTZ, i.e. $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MCF_F}) \subset Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MTZ})$.*

Proof. To show that $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{GSEC}) \subseteq Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MTZ})$ we assume that $(\mathbf{x}, \mathbf{z}) \in Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{GSEC})$ does not satisfy constraints (26). But then there must exist a cycle $K \subset S$ such that by summing up inequalities (26) over all arcs in K we obtain

$$(|S| - 2) \sum_{ji:ij \in K} x_{ji} + |S| \sum_{ij:ij \in K} x_{ij} > |K|(|S| - 1). \tag{33}$$

After dividing this inequality by $|S|$, the left hand side becomes:

$$\sum_{ji:ij \in K} x_{ji} + \sum_{ij:ij \in K} x_{ij} - \frac{2}{|S|} \sum_{ji:ij \in K} x_{ji} \leq \sum_{ij \in A_S: i,j \in K} x_{ij} - \frac{2}{|S|} \sum_{ji:ij \in K} x_{ji} \leq$$

$$\leq \sum_{ij \in A_S: i,j \in K} x_{ij} \overset{(29)}{\leq} \min_{l \in K} \sum_{i \in K} w_i - w_l \leq |K| - 1 \leq |K| - \frac{|K|}{|S|},$$

which is a contradiction to (33).

Let us finally suppose that inequalities (25) are not satisfied, i.e., that there is an arc $jk \in A_S$ such that $\sum_{ij \in A_S: i \neq k} x_{ij} < x_{jk}$. After adding x_{kj} to both sides, we obtain $x_{jk} + x_{kj} > \sum_{ij \in A_S} x_{ij} = w_j$ which is a direct contradiction to generalized sub-tour elimination constraints applied to $U = \{j, k\}$.

To show that $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{MTZ}) \nsubseteq Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{GSEC})$ consider Example 6, where $\upsilon_{LP}(MTZ) = 9 < 26 = \upsilon_{LP}(GSEC)$.

□

4.1 Reformulation as the Steiner Arborescence Problem

As we already observed in [36], the ConFLP can be transformed into the Steiner Arborescence Problem. This transformation is done by using the well-known *node splitting* technique that has proven useful for different network design problems, see e.g., [3, 6].

To solve an instance of ConFL as SA, we use the following procedure:

- Generate a directed graph $\tilde{G} = (\tilde{V}, \tilde{A})$ with costs $\tilde{\mathbf{c}} : \tilde{A} \mapsto \mathrm{R}_0^+$, as follows:
  - Initialize $\tilde{V} = V$, $\tilde{A} = A$ and $\tilde{\mathbf{c}} = \mathbf{c}$.
  - For any facility node i, add a node i' to the graph, connect i to i', and set $\tilde{c}_{ii'} = f_i$.
  - Replace arcs $ik \in A_R$ by $i'k$.
- Solve the Steiner arborescence problem on the transformed graph $\tilde{G}$ with customers as terminals.

Recall that, given a directed graph $\tilde{G} = (\tilde{V}, \tilde{A})$, with arc weights $\tilde{\mathbf{c}} : \tilde{A} \mapsto \mathbb{R}$, a root $r \in \tilde{V}$, and a set of terminal nodes $R \subset \tilde{V}$, the Steiner arborescence problem searches for the cheapest subtree rooted at r that connects all terminals. Figure 4 shows a simple example that illustrates the transformation of ConFL into the SA problem, according to the procedure described above:

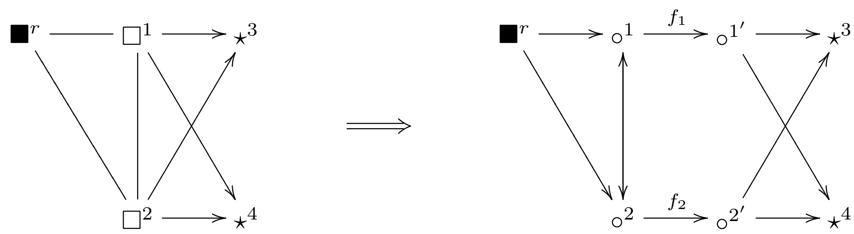


Figure 4: Initial undirected ConFL instance and transformed SA instance

For each facility $i \in F$, i corresponds to node's function as Steiner node, while i' corresponds to its function as open facility. With this transformation we ensure that the arc ii' belongs to a solution if and only if facility i is open. Similarly, facility i is used as Steiner node if and only if i belongs to the solution, but arc ii' does not. A similar, but undirected transformation has been used by Bardossy and Raghavan to transform (G)STS, ConFL and RoB into the GConFL [4].

To solve the SA problem as a MIP, let us define binary variables v_{ij} as follows:

$$v_{ij} = \begin{cases} 1, & \text{if } ij \text{ belongs to the solution} \\ 0, & \text{otherwise} \end{cases}, \quad \forall ij \in \tilde{A}.$$

We extend the directed cut-based formulation for Steiner trees (originally proposed by Chopra and Rao [8]) by the root out-degree constraint as follows:

$$(SA) \qquad \min \sum_{ij \in \tilde{A}} \tilde{c}_{ij} v_{ij} \tag{34}$$

$$\sum_{ij \in \delta^-(W)} v_{ij} \geq 1, \qquad \forall W \subseteq \tilde{V} \setminus \{r\}, W \cap R \neq \emptyset \tag{35}$$

$$v_{rr'} = 1 \tag{36}$$

$$v_{ij} \in \{0,1\} \quad \forall ij \in \tilde{A} \tag{37}$$

Let us denote by

$$Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{SA}) = \{(\mathbf{x},\mathbf{z}) \in [0,1]^{|A|} \times [0,1]^{|F|} \mid \mathbf{v} \in \mathcal{P}_{SA} \text{ and}$$

$$x_{kl} = v_{kl}\ \forall kl \in A_S;\ x_{ij} = v_{i'j}\ \forall ij \in A_R;\ z_i = v_{ii'}\ \forall i \in F\},$$

the projection of the $\mathcal{P}_{SA}$ polytope onto the space of variables $(\mathbf{x}, \mathbf{z})$.

We show the following result:

Lemma 10. *The LP-relaxation of the Steiner arborescence formulation is equally strong as the LP-relaxation of CUT_R, i.e.:*

$$Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{SA}) = \mathcal{P}_{CUT_R}.$$

Proof. We prove equality by showing mutual inclusion:

$Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{SA}) \subseteq \mathcal{P}_{CUT_R}$**:** Let $\mathbf{v}'$ be an optimal fractional solution of the LP-relaxation of SA, and $(\mathbf{x}', \mathbf{z}')$ its projection into $Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{SA})$. Obviously, (1), (2) and (4) are satisfied by $(\mathbf{x}', \mathbf{z}')$. It only remains to show that $x'_{ij} \leq z'_i, \forall ij \in A_R$. Let us assume that $\exists i \in F$, $\exists ij \in A_R$ such that $x'_{ij} > z'_i$. Without loss of generality assume also that $c_{ij} > 0$. In $\tilde{G}$, $x'_{ij} > z'_i$ implies that $v'_{i'j} > v'_{ii'}$. Given graph $\tilde{G}$ with capacities v'_{ij} on the arcs, the only possibility to send flow from r to j over i' is through the arc ii'. But given the capacity of $v'_{ii'} < v'_{i'j}$, and given the objective function (34), it follows that we can find another LP-solution $\mathbf{v}''$ whose objective value is strictly less than $\tilde{\mathbf{c}}^{\mathbf{t}}\mathbf{v}'$, without violating connectivity constraints (35), by simply setting $v''_{ij} := v'_{ii'}$ and keeping the rest of values unchanged. This however contradicts the assumption that $\mathbf{v}'$ is an optimal LP-solution.

$\mathcal{P}_{CUT_R} \subseteq Proj_{\mathbf{x},\mathbf{z}}(\mathcal{P}_{SA})$**:** Let $(\mathbf{x}', \mathbf{z}')$ be a fractional solution satisfying (1)-(4), and let us assume that the corresponding solution $\mathbf{v}'$ from $\mathcal{P}_{SA}$ is not feasible. In other words, assume that there exists a cut-set $\tilde{W} \subseteq \tilde{V} \setminus \{r\}, \tilde{W} \cap R \neq \emptyset$, such that $\sum_{ij \in \delta^-(\tilde{W})} v_{ij} < 1$. Obviously, there must exist at least one $i \in F \setminus \{r\}$, such that $ii' \in \delta^-(\tilde{W})$. We now construct a new cut-set $\tilde{W}_n$ such that $\delta^-(\tilde{W}_n) = \delta^-(\tilde{W}) \cup \{i'j \mid j \in \tilde{W}\} \setminus \{ii'\}$. Obviously, if $\sum_{ij \in \delta^-(\tilde{W})} v_{ij} < 1$, then also $\delta^-(\tilde{W}_n) < 1$. By repeating this procedure for all $i \in F$ such that $ii' \in \delta^-(\tilde{W})$, we end up with a cut-set containing only arcs from $A_R \cup A_S$, that violates inequality (35), which is a contradiction.

□

4.2 Full Hierarchy of Formulations

The hierarchical scheme given in Figure 4.2 summarizes the relationships between the LP relaxations of the MIP models considered throughout this paper. A filled arrow specifies that the target formulation is strictly stronger than the source formulation. A dashed connection specifies that the formulations are not comparable to each other. Note that we do not display formulation SCF_R^+ separately, because it has the same relations as the formulation SCF_R.

Note that all three models SCF_F, MCF_F and CF_F may have lower bounds as bad as $OPT/|F|$. Model CF_R is the strongest one among all considered throughout this paper. Observe that there are several other tree models known for Steiner trees, that can directly be interpreted in ConFL context. Therefore we do not mention them here, but refer the interested reader to Magnanti and Wolsey [30] and Polzin and Daneshmand [34].

5 Branch-and-Cut Framework

We are going to calculate lower bounds and provably optimal solutions of CUT_F and CUT_R models using the same branch-and-cut framework described below. The only difference is in the separation of cut set inequalities. The main ingredients of our implementation are provided in this section.

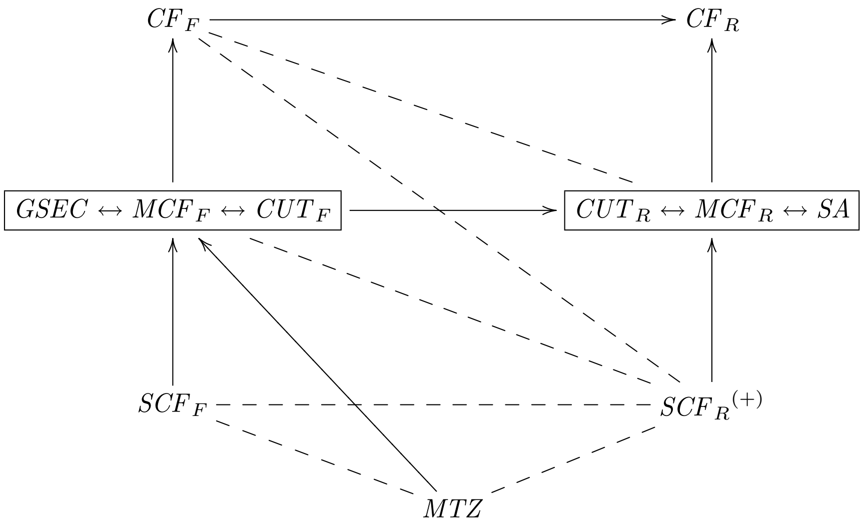


Figure 5: Relations between LP-relaxations of MIP models for ConFL

Initialization: We initialize the LP with assignment, capacity- and root-inequalities (2)-(4). The following flow-balance constraints introduced by Koch and Martin [20] are also introduced in the initialization phase. These constraints ensure that the in-degree of each Steiner node is less or equal than its out-degree:

$$\sum_{kl \in A} x_{kl} \leq \sum_{lk \in A} x_{lk}, \quad \forall l \in S \setminus F. \tag{38}$$

These constraints are not induced by any of the MIP formulations presented above, i.e., they can further strengthen the quality of lower bounds (see, e.g., [28, 34]).
Finally, we insert the following in-degree inequalities:

$$\sum_{kl \in A} x_{kl} \leq 1, \quad \forall l \in S \setminus \{r\} \quad \text{and} \sum_{ir \in A_S} x_{ir} = 0,$$

and the *sub-tour elimination constraints* of size two:

$$x_{kl} + x_{lk} \leq 1, \quad \forall \{k,l\} \in E, \ k,l \in S \ k \neq r.$$

The latter two groups of constraints are not necessarily binding, but they can speed up the cutting plane phase at the root node of the branch-and-bound (B&B) tree.

Branching: Branching on single arc variables produces a huge disbalance in the branch-and-bound tree. Whereas discarding an edge from the solution (setting x_{ij} to zero) doesn't bring much, setting the facility variable to one significantly reduces the size of the search subspace. Therefore we set the highest branching priorities to variables z_i, $i \in F$.

5.1 Separation

Separation of cut set inequalities (8)**:** In each node of the branch-and-bound tree we separate the cut-inequalities (8). For a given LP-solution $(\hat{\mathbf{x}}, \hat{\mathbf{z}})$, we construct a support graph $G_S = (S, A_S, \hat{\mathbf{x}})$ with arc capacities set to $\hat{x}_{ij}$, for all $ij \in A_S$. Then we calculate the maximum flow from the root node r to each potential facility node $i \in F$ such that $\hat{z}_i > 0$. If this maximum flow value is less than z_i, we have found a violated inequality (8), induced by the corresponding min-cut in the graph G_S, and we insert it into the LP. For the calculation of the maximum flow we used an adaptation of Cherkassky and Goldberg's maximum flow algorithm [7].

Separation of Cut Set Inequalities (1)**:** In order to separate cut set inequalities (1), we build a support graph by copying $G = (V, A)$. For a given fractional solution $(\hat{\mathbf{x}}, \hat{\mathbf{z}})$, we set the capacities to $\hat{x}_{ij}$, for all $ij \in A$. We then calculate the maximum flow that can be sent from r to each of the customers $j \in R$. If there exists customer j such that the value of the maximum flow is less than one, we obtain a cut set, say $W \subset V$, $r \in W$, such that capacity of $\delta^+(W)$ is less than one. Obviously, $W \cap F \neq F$, since all the cuts involving only arcs from A_R are satisfied by (2). According to Observation 2, the violated cut set inequality (1) induced by W can then be written as: $\sum_{ij \in A_S^W} x_{ij} + \sum_{ij \in A_R^W} x_{ij} \geq 1$.

Enhancing Separation To improve computational efficiency, we search for *nested, back* and *minimum-cardinality cuts* and insert at most 100 violated inequalities in each separation phase. For more details, see our implementation of the B&C algorithm for the prize-collecting Steiner tree problem, where the same separation procedure has been used [27, 28]. It is important to mention that the performance of the branch-and-cut algorithm can further be improved if we permute the order in which the minimum cuts between r and $i \in F$, $z_i > 0$, in CUT_F case, and between r and j, $j \in R$, in CUT_R case, are calculated. Since this permutation is done randomly, we fix the seed value for the results reported in Section 6.

5.2 Primal Heuristic

The primal heuristic works as follows: First, we initialize the set of open facilities according to fractional values z_i: if $z_i > \pi$, we label the facility as *selected*. Default value of π is set to 0.1. Denote by $\mathcal{F} = \{i \in F \mid z_i = 1\}$, the set of initially selected facilities. Starting with $\mathcal{F}$, we then calculate a feasible ConFL solution according to the pseudo-code provided in Algorithm 1. We use the following notation:

- vector $\mathbf{x^S}$ refers to the core tree structure, i.e., $x_{ij}^S = 1$ if $ij \in A_S$ belongs to the solution, and it is zero otherwise.
- vector $\mathbf{x^A}$ refers to assignment values, i.e., $x_{ij}^A = 1$ if customer j is assigned to facility i and $x_{ij}^A = 0$, otherwise, for all $ij \in A_R$.
- vector $\hat{\mathbf{z}}$ is set to one if facility i is open, and to zero otherwise.
- T^S denotes the core Steiner tree (the set of nodes and edges) that is uniquely defined by $\mathbf{x^S}$.

Outline The algorithm works in three phases: In the *assignment phase* (*Assign*), the cheapest assignment of customers to facilities from $\mathcal{F}$ is found. If there are non-assigned customers, solution is discarded. The set $\mathcal{F}$ is updated to contain only *open* facilities, i.e., those that serve at least one customer. In the *Steiner tree phase*, the set of *open facilities* is connected by a Steiner tree. For that purpose, we use the minimum spanning tree heuristic (*MSTHeuristic*) described below. Finally, we apply a *local improvement* procedure (*Peeling*) that tries to remove leaves of the Steiner tree in the core network and to re-assign customers to already open facilities, by decreasing the overall costs.

Data: Binary vector $\hat{\mathbf{z}}$: a facility i is *selected* if $\hat{z}_i = 1$.

Result: Locally improved solution $(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}})$.

if *Hash*$(\hat{\mathbf{z}})$ *defined* **then**
  $(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}}) = Hash(\hat{\mathbf{z}})$;
else
  if *Assignment exists?* **then**
    $(\mathbf{x^A}, \hat{\mathbf{z}}) := Assign(\hat{\mathbf{z}})$;
    $(\mathbf{x^S}, \hat{\mathbf{z}}) := MSTHeuristic(\hat{\mathbf{z}})$;
    $(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}}) := Peeling(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}})$;
    Insert $(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}})$ into *Hash*;
  else
    return infeasible;
  end
end
return $(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}})$;

Algorithm 1: The primal heuristic: calculation of the objective function for a given vector $\hat{\mathbf{z}}$.

Hashing Given a vector of selected facilities, $\hat{\mathbf{z}}$, we first check if the objective value for this configuration has been already calculated before (see, e.g., [22]). If so, we get the corresponding solution $(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}})$ from the hash-table *Hash*. Otherwise, we run a three-step procedure whose steps are described below.

Detailed Description

Step 1: $(\mathbf{x^A}, \hat{\mathbf{z}}) := Assign(\hat{\mathbf{z}})$**:** For each customer $j \in R$, we find the cheapest possible assignment to a facility from $\hat{\mathbf{z}}$. The assignment values are stored in vector $\mathbf{x^A}$. We close those facilities i from $\mathcal{F}$ that do not serve any customer, i.e., we set $\hat{z}_i := 0$. If such assignment is not possible (e.g., the subgraph induced by A_R is not a complete bipartite graph), we discard the solution.

This operation is calculated from scratch. Thus, the total computational complexity for finding the cheapest assignment in the worst case is $O(|\mathcal{F}||R|)$.

Step 2: $(\mathbf{x^S}, \hat{\mathbf{z}}) := MSTHeuristic(\hat{\mathbf{z}})$**:** We consider the graph $G' = (S, E_S)$ – a subgraph of G induced by the set of facilities and Steiner nodes with the edge costs $\mathbf{c}$. For G', we generate the so-called *distance network*[1] - a complete graph whose nodes correspond to facilities $i \in F$, and whose edge-lengths l_{ij} are defined as shortest paths in G', for all $i, j \in F$.

We use the minimum spanning tree (MST) heuristic [31] to find a spanning tree T^S that connects all open facilities ($\hat{z}_i = 1$).

1. Let G'' be the subgraph of G' induced by $\mathcal{F}$.
2. Calculate the minimum spanning tree MST''_G of the distance sub-network G''.

[1]Calculation of the distance network is done only once, during the initialization of the branch-and-cut algorithm.

3. On the subgraph of (S, E_S) obtained by back-mapping the edges from MST''_G, re-calculate the minimum spanning tree (T^S) to obtain vector $\mathbf{x^S}$.

Step 3: $(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}}) := \mathit{Peeling}(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}})$**:** We finally want to get rid of some of those facilities that are still part of the Steiner tree, but that are not used at all. We do this by applying the so-called *peeling procedure*. Our peeling heuristic tries to recursively remove all redundant leaf nodes (including corresponding tree-paths) from the tree-solution defined by $\mathbf{x^S}$. Let k denote a leaf node of T^S, and let P_k be a path that connects k to the next open facility from $\mathcal{F}$, or to the next branch, towards the root r.

1. If the leaf node is not an (open) facility, i.e. if $\hat{z}_k = 0$, we simply delete P_k.
2. Otherwise, we try to re-assign customers (originally assigned to k) to already open facilities (if possible). If such obtained solution is better, we delete P_k and continue processing other leaves.

The main steps of this procedure are given in Algorithm 2.

If, for each customer, the set of facilities is sorted in increasing order with respect to its assignment costs[2], this procedure can be implemented very efficiently. Indeed, in order to find an open facility from $\mathcal{F}$, nearest to j and different from k (denoted by $i^k(j)$), we only need to proceed this ordered list starting from k until we encounter a facility i such that $\hat{z}_i = 1$.

The algorithm stops when only one node is left, or when all the leaves from the tree have been proceeded. Thus, the worst-case running time of the whole peeling method is $O(|\mathcal{F}||R|)$.

6 Computational Results

In our computational study, two groups of instances were considered:

Randomly Generated Graphs From [36] For this set of instances the parameters for the generation were set as follows: $|S| \in \{20, 50, 100\}$, $|R| \in \{20, 50, 100\}$. Edges of the core network are generated with probability $p(S) \in \{0.1, 0.5, 1\}$, while the connections between facilities and customers are established with probability $p(R) \in \{0.18, 0.55, 1\}$. Edge weights were uniformly randomly set to an integer value between 50 and 100. Finally, the facility opening costs were uniformly randomly assigned to values between 150 and 200. Increasing only the core costs did not significantly change the behavior of the GRASP algorithm for this set of instances. The core network was generated by MAPLE, using the parameters given above. Finally, customers are randomly linked to the existing nodes using the density values $p(R)$.
As the original instances are unrooted we selected the facility with the highest index for the root node respectively.

Graphs Derived From OR-library [5] and UflLib [1] We consider another class of benchmark instances, obtained by merging data from two public sources. In general, we combine an UFLP instance with an STP instance, to generate ConFL input graphs in the following way: first $|F|$ nodes of the STP instance are selected as potential facility locations, and the node with index 1 is selected as the root. The number of facilities, the number of customers, opening costs and assignment costs are provided in UFLP files. STP files provide edge-costs and additional Steiner nodes.

[2] Also sorting of these lists is done once, in the initialization phase of the branch-and-cut algorithm.

Data: Assignment $\mathbf{x^A}$, open facilities $\hat{\mathbf{z}}$ and a Steiner tree T^S corresponding to $\mathbf{x^S}$.

Result: Locally improved solution $(\mathbf{x^S}, \mathbf{x^A}, \hat{\mathbf{z}})$.

for *all leaves k in T^S* **do**
  Determine path P_k and its costs $c(P_k) := \sum_{e \in P_k} c_e$;
  if $\hat{z}_k = 0$ **then**
    $T^S := T^S - P_k$;
  else
    $R_k := \{j \mid j \in R,\, x^A_{kj} = 1\}$;
    $i^k(j) = \arg\min\{c_{ij} \mid i \in F,\ \hat{z}_i = 1,\ i \neq k\}, \forall j \in R_k$;
    if $\exists j \in R_k : i^k(j) = \emptyset$ **then**
      continue;
    end
    if $\sum_{j \in R_k} c_{i^k(j)j} < f_k + c(P_k) + \sum_{j \in R_k} c_{kj}$ **then**
      $\hat{z}_k := 0$;
      $T^S := T^S - P_k$;
      $x^A_{kj} := 0,\, x^A_{i^k(j)j} := 1, \forall j \in R_k$;
    end
  end
end

Algorithm 2: Peeling procedure.

- We consider two sets of non-trivial UFLP instances from UflLib [1]:
  - `mp-{1,2}` and `mq-{1,2}` instances have been proposed by Kratica et al. [22]. They are designed to be similar to UFLP real-world problems and have a large number of near-optimal solutions. There are 6 classes of problems, and for each problem $|F| = |R|$. We took 2 representatives of the 2 classes MP and MQ of sizes 200×200 and 300×300, respectively.
  - The `gs-{250,500}a-{1,2}` benchmark instances were initially proposed by Koerkel [21] (see also Ghosh [12]). Here we chose two representatives of the 250×250 and 500×500 classes, respectively. The authors drew uniformly at random connection costs from [1000, 2000], and the facility opening costs from [100, 200].
- STP instances: Instances `{c,d}n`, for $n \in \{5, 10, 15, 20\}$ were chosen randomly from the OR-library [5] as representatives of medium size instances for the STP. These instances define the core networks with between 500 and 1000 nodes and with up to 25,000 edges.

Combined with assignment graphs, the largest instances of this data set contain 1,300 nodes and 115,000 edges.
All experiments were performed on a Intel Core2 Quad 2.33 GHz machine with 3.25 GB RAM, where each run was performed on a single processor. For solving the linear programming relaxations and for a generic implementation of the branch-and-cut approach, we used the commercial packages IBM CPLEX (version 11.2) [2] and ILOG Concert Technology (version 2.7).

6.1 Testing Randomly Generated Instances

For the following tests we turn the primal heuristics off, in order to compare lower bounds of all presented MIP formulations. Furthermore, our preliminary results have shown that turning all CPLEX general purpose cuts speeds up the performance. Therefore, and in order to avoid biased results, all the results reported in this paper are obtained without usage of these cuts.

LP-gaps We first test the performance and the quality of lower bounds for proposed formulations. For that purpose, we run the models as linear programs. Table 3 provides the average gaps calculated as $(OPT - v_{LP}(.))/OPT$, where optimal values are obtained by running the branch-and-cut approach (see below). The set of 81 instances is divided into 3 groups according to the size of the core- and the assignment-subgraph.
Not surprisingly, the worst gaps are obtained by running SCF_R model in which "big-M" constraints affect all the arcs in G. Comparing gap values of SCF_F model on these three groups, we observe that the gap increases with the size of the nodes of the core network. This is also not surprising, since "big-M" constraints of the SCF_F model affect only the core network. We observe that there is a correlation between the size of the two subgraphs and the quality of obtained lower bounds for the other models as well. The gaps obtained by MTZ model are surprisingly good, and very close to those obtained by MCF_F. The best LP-gaps are obtained by MCF_R model. Interestingly, the most difficult instances for the latter three models appear to be those with the equal number of facilities and customers.
Finally, we tried to make the same experiment with CF_F and CF_R models, but apparently in almost all cases the execution has been erminated because of memory overconsumption.

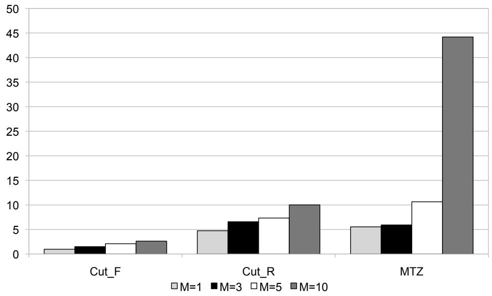
Solving MIPs Table 2 shows the running times in seconds ($t[s]$) and the number of branch-and-bound nodes ($B\&B$) needed to solve this set of instances. Each row corresponds to three instances generated according to the same probabilities $p(R)$ and $p(S)$. We provide values for $t[s]$ and $B\&B$ averaged over the respective group. We set the time limit to 1000 seconds. If at least one of the three instances per group is not solved to optimality, we denote this by "-".
As expected, due to the weak lower bounds of the SCF_R^+, most of the instances could not be solved to optimality within the given time limit. The exceptions are graphs with complete bipartite structure of the assignment subgraph A_R that appear to be easy for SCF_R^+. The second worse performance was shown by the MCF_R model, which is easily explained by its huge number of variables.
This test gives two surprising results:

1. Despite the fact, that the integrality gap of model CUT_F can be as bad as $\frac{1}{|F|}$ it outperforms even the strongest cut set based model CUT_R with respect to the running time. On average, the number of $B\&B$ nodes needed by CUT_F is 2.3 times larger than for CUT_R. However, averaged over all 81 instances, CUT_F is about 4.6 times faster than CUT_R.

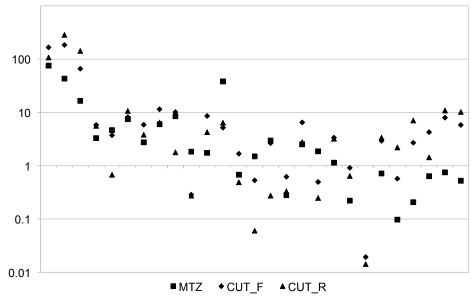
2. The compact MTZ model with arbitrarily bad lower bounds performs comparatively well. It outperforms CUT_R: the average running time over all instances for MTZ is 1.06 times less than the corresponding time for CUT_R.

					MTZ		SCF_R^+		MCF_R		CUT_F		CUT_R	
$\|S\|$	$\|R\|$	$p(S)$	$p(R)$	Opt	$t[s]$	$B\&B$	$t[s]$	$B\&B$	$t[s]$	$B\&B$	$t[s]$	$B\&B$	$t[s]$	$B\&B$
20	100	0.1	0.18	9,768	**0.10**	1	-	-	2.00	0	**0.10**	0	0.47	0
20	100	0.5	0.18	9,577	0.29	10	-	-	2.77	0	**0.09**	0	0.48	0
20	100	1.0	0.18	9,554	0.56	48	-	-	9.53	2	**0.12**	0	0.57	0
20	100	0.1	0.55	7,428	2.52	103	-	-	26.92	36	**1.57**	70	5.09	43
20	100	0.5	0.55	7,289	1.46	52	-	-	301.27	31	**1.26**	55	8.18	38
20	100	1.0	0.55	7,316	1.97	57	-	-	-	-	**1.41**	67	9.24	49
20	100	0.1	1.00	6,675	2.39	48	1.59	29	10.54	4	**1.21**	28	4.04	4
20	100	0.5	1.00	6,683	2.02	25	1.41	22	110.56	10	**1.40**	37	6.50	11
20	100	1.0	1.00	6,632	1.97	25	1.20	27	258.67	9	**1.05**	19	9.92	11
50	50	0.1	0.18	5,295	4.47	171	-	-	-	-	**2.50**	81	20.39	45
50	50	0.5	0.18	5,019	10.61	242	-	-	-	-	**2.09**	22	22.04	28
50	50	1.0	0.18	4,987	4.43	42	-	-	-	-	**3.38**	67	16.31	37
50	50	0.1	0.55	4,045	5.24	123	-	-	217.10	12	**3.97**	94	9.05	14
50	50	0.5	0.55	4,011	**6.67**	55	-	-	-	-	7.09	118	38.06	31
50	50	1.0	0.55	3,896	7.52	47	-	-	-	-	**4.13**	74	28.82	21
50	50	0.1	1.00	3,615	4.91	51	**1.10**	4	25.53	3	1.81	29	2.77	4
50	50	0.5	1.00	3,596	5.44	26	2.18	16	284.24	5	**1.64**	21	4.87	8
50	50	1.0	1.00	3,596	7.30	16	**2.17**	21	-	-	2.74	21	10.28	13
100	20	0.1	0.18	2,489	1.84	16	251.27	171,598	122.51	6	**1.22**	14	2.72	3
100	20	0.5	0.18	2,463	10.43	35	-	-	-	-	**5.30**	33	23.57	16
100	20	1.0	0.18	2,487	144.35	378	-	-	-	-	**7.79**	44	42.62	39
100	20	0.1	0.55	1,921	4.44	51	118.32	27,557	43.20	7	**1.75**	30	2.68	9
100	20	0.5	0.55	1,876	8.66	31	-	-	-	-	4.08	29	**3.26**	5
100	20	1.0	0.55	1,873	14.16	13	-	-	-	-	**2.12**	13	4.95	4
100	20	0.1	1.00	1,638	1.16	4	0.82	4	3.07	0	**0.29**	3	1.02	0
100	20	0.5	1.00	1,638	2.70	1	1.26	1	8.98	0	**0.47**	2	1.59	0
100	20	1.0	1.00	1,633	7.26	2	1.31	1	21.08	0	**0.91**	2	2.59	0

Table 2: Running times (in seconds) and the number of Branch-and-Bound nodes for selected MIP formulations with CPLEX cuts turned off.



(a) Average slow-down factors for three MIP models and for $M \in \{1, 3, 5, 10\}$.



(b) Speed-up factors obtained by using branching priorities for facility nodes against default branching times.

Figure 6: Results for randomly generated instances from [36].

Testing the influence of the factor M In the following test, we multiply the costs of the core network by a factor $M \in \{3, 5, 10\}$. Our goal is to test the influence of the cost structure of the core network on the overall performance of proposed MIP models. For that purpose, we select the best performing models according to the results obtained above, namely: MTZ, CUT_F and CUT_R. As a reference value, we take the average running time the model CUT_L needed to solve the problems with $M = 1$ to optimality. For each of the three MIP models, and for each of possible M values, we divide the corresponding average running time with the reference time to calculate the so-called *slow down factor* shown in Figure 6(a).

The obtained slow down factors indicate that the MTZ model is the most affected by increasing the costs of the core network: MTZ needs about 7 times more time to solve the instances to optimality, if the costs of the core network are multiplied by factor $M = 10$. This result is due to decreasing quality of lower bounds of the MTZ model with increasing M values. On the other hand, models CUT_F and CUT_R are not so much affected by that effect: in the worst case, when $M = 10$, the average running time increases by roughly a factor of 2.6 and 2.1 for CUT_F and CUT_R, respectively. We also observe that CUT_F outperforms MTZ by a factor of 5 for $M = 1$, and by a factor of 16 for $M = 10$.

Branching We also tested our branching strategy described in Section 5 against CPLEX default branching strategy. For each of 27 density settings, Figure 6(b) shows the speed up factor obtained by dividing two running times: one needed to solve the instance with default CPLEX setting to optimality and the other one obtained with our branching strategy. The values are averaged over three instances per setting. In most of the cases our branching strategy significantly reduces the overall running time. On average over all 81 instances, our branching strategy outperforms CPLEX default branching by a factor of 1.4, 3.3 and 2.9, when models MTZ, CUT_F and CUT_R are solved, respectively.

6.2 Testing Larger Graphs

The set of instances is divided into three groups according to the underlying instance for the assignment graph. We refer to them as `mp`, `mq` and `qs` group. Tables 4 and 5 report on the results obtained trough this experiment. Note

$\lvert S\rvert$	$\lvert R\rvert$	MTZ	SCF_F	SCF_R	MCF_F	MCF_R
20	100	1.36 %	5.44 %	96.24 %	1.33 %	0.73 %
50	50	2.57 %	7.33 %	93.28 %	2.51 %	1.36 %
100	20	2.48 %	8.33 %	85.19 %	2.43 %	1.22 %

Table 3: Average Integrality Gaps for selected MIP formulations

that the optimal values, as well as lower bounds reported in this paper differ from those reported in [26]. This is due to in-degree inequalities used in [26], that turned out to model the Steiner tree star problem, instead of ConFL.

Comparing Two Branch-and-Cut Approaches: First, we compare the two branch-and-cut approaches by running them with the proposed primal heuristic. Regarding 32 instances obtained by combining `stein` and `mp/q` instances, CUT_F solves all 32 instances to provable optimality within 213 seconds on average. The gaps we report for each model were calculated as

$$gap[\%] = \frac{UB - LB}{UB},$$

where UB and LB are the upper and lower bound obtained by the respective model. In addition, we report on the running time in seconds (t $[s]$), the model CUT_F needs to solve the instances of the `mp/q` group to optimality. Note that CUT_R solves only 7 out of 32 `mp/q` instances to optimality. For the majority of instances CUT_R does not branch at all, as it has not finished the cutting plane phase at the root node of the branch-and-bound tree. This is because the assignment graphs for these instances are complete bipartite, which means that many dense cuts of the CUT_R model need to be separated.

Comparing MIP Models Initialized with Best Upper Bound: Second, we run all three models, MTZ, CUT_F and CUT_R, but we deactivate the primal heuristic. Instead, we initialize the models with the best upper bound found in the previous setting. For the `gs` group of instances, the best lower and upper bounds obtained with this setting can be found in the right hand half of Table 5. Each of the models MTZ and CUT_R solves only 8 instances to optimality. For the `mp` subgroup, MTZ gives much smaller gaps though, on average 0.17% compared to 1.42% for CUT_R. For the group of `mq` instances MTZ also outperformes CUT_R with an average gap of 1.86% vs. 3.18% for the latter.

In the last group of large scale instances derived from the `gs` group, the performance of MTZ is comparatively better. CUT_F obtains the smallest gap in 11 cases, but MTZ performs best on 7 instances. Not a single instance of `gs` group has been solved to optimality. Note that for this last group of instances the cost structure is special. The factor M, describing the scale between core and assignment costs is about 0.001.

			PH on, no UB given					PH off, best UB given						
			CUT_R		CUT_F			MTZ		CUT_R		CUT_F		
Stein	UFL	OPT	gap[%]	$B\&B$	gap[%]	$B\&B$	t [s]	gap[%]	$B\&B$	gap[%]	$B\&B$	gap[%]	$B\&B$	t [s]
c05	mp1	2,691.5	**0.00**	13	**0.00**	27	73	0.34	605	**0.00**	23	**0.00**	33	50
c10	mp1	2,661.7	**0.00**	17	**0.00**	17	67	**0.00**	86	**0.00**	23	**0.00**	25	47
c15	mp1	2,634.7	1.45	1	**0.00**	15	100	0.15	1084	1.39	3	**0.00**	17	73
c20	mp1	2,618.7	1.91	3	**0.00**	33	185	**0.00**	58	1.50	1	**0.00**	11	104
d05	mp1	2,677.9	**0.00**	9	**0.00**	27	62	**0.00**	19	**0.00**	9	**0.00**	37	40
d10	mp1	2,676.5	2.39	0	**0.00**	21	92	0.24	542	2.39	1	**0.00**	21	66
d15	mp1	2,635.7	1.05	5	**0.00**	13	67	**0.00**	43	**0.00**	15	**0.00**	11	41
d20	mp1	2,619.7	1.59	0	**0.00**	27	229	0.06	49	1.59	1	**0.00**	15	82
c05	mp2	2,692.5	**0.00**	11	**0.00**	15	37	**0.00**	58	**0.00**	17	**0.00**	13	26
c10	mp2	2,661.5	**0.00**	9	**0.00**	5	27	**0.00**	97	**0.00**	7	**0.00**	11	23
c15	mp2	2,640.5	0.61	3	**0.00**	10	47	0.13	1772	0.89	0	**0.00**	5	28
c20	mp2	2,626.5	**0.00**	11	**0.00**	11	55	0.06	300	**0.00**	11	**0.00**	11	43
d05	mp2	2,710.6	**0.00**	25	**0.00**	19	41	**0.00**	1048	**0.00**	31	**0.00**	17	31
d10	mp2	2,682.5	1.14	0	**0.00**	29	50	0.26	574	0.94	3	**0.00**	27	50
d15	mp2	2,647.5	0.53	7	**0.00**	7	43	**0.00**	14	0.53	7	**0.00**	7	31
d20	mp2	2,628.5	2.14	0	**0.00**	11	222	0.09	70	2.14	0	**0.00**	11	142
c05	mq1	3,907.0	3.08	1	**0.00**	53	261	1.56	11	3.08	1	**0.00**	41	193
c10	mq1	3,866.5	4.12	0	**0.00**	35	214	1.49	20	4.12	0	**0.00**	37	146
c15	mq1	3,842.5	3.09	0	**0.00**	41	183	1.61	12	3.09	0	**0.00**	35	142
c20	mq1	3,826.5	3.08	0	**0.00**	33	289	1.43	7	3.08	0	**0.00**	35	173
d05	mq1	3,879.0	2.56	1	**0.00**	31	210	**0.00**	25	2.12	3	**0.00**	51	127
d10	mq1	3,869.1	2.99	0	**0.00**	43	242	1.72	15	2.92	0	**0.00**	29	156
d15	mq1	3,843.5	2.68	3	**0.00**	61	173	1.07	28	2.02	5	**0.00**	37	134
d20	mq1	3,828.5	2.80	0	**0.00**	45	483	1.87	5	2.80	0	**0.00**	39	387
c05	mq2	3,768.6	2.89	0	**0.00**	73	561	2.99	10	2.88	0	**0.00**	71	283
c10	mq2	3,732.6	5.14	0	**0.00**	63	320	2.99	9	5.14	1	**0.00**	50	190
c15	mq2	3,689.6	2.31	0	**0.00**	41	259	1.23	6	2.31	0	**0.00**	69	231
c20	mq2	3,686.5	4.58	0	**0.00**	45	620	2.33	3	4.03	0	**0.00**	27	317
d05	mq2	3,741.5	2.60	0	**0.00**	47	276	1.34	8	2.59	0	**0.00**	73	236
d10	mq2	3,720.9	**4.24**	0	**0.00**	31	285	4.07	6	2.52	0	**0.00**	43	396
d15	mq2	3,696.5	3.96	0	**0.00**	41	328	1.49	5	2.44	0	**0.00**	33	198
d20	mq2	3,685.5	5.73	0	**0.00**	27	727	2.60	2	5.73	0	**0.00**	33	402

Table 4: Results for large scale instances I: The best obtained gaps per setting and instance are shown in bold.

		PH on, no UB given						PH off, best UB given							
				CUT_R		CUT_F				*MTZ*		CUT_R		CUT_F	
Stein	UFL	best UB	best LB	*gap*[%]	*B&B*	*gap*[%]	*B&B*	best UB	best LB	*gap*[%]	*B&B*	*gap*[%]	*B&B*	*gap*[%]	*B&B*
c5	gs250a-1	258,568.0	258,088.8	0.27	2	**0.19**	162	258,540.0	258,112.9	0.20	180	0.27	5	**0.17**	289
c10	gs250a-1	258,480.0	257,955.7	0.25	1	**0.20**	147	258,464.0	257,986.5	0.20	201	0.20	7	**0.18**	227
c15	gs250a-1	258,387.0	257,823.3	**0.22**	0	-	-	258,387.0	257,858.5	**0.20**	280	0.23	3	-	-
c20	gs250a-1	258,250.0	257,786.4	0.50	0	**0.18**	15	258,250.0	257,798.6	**0.18**	28	0.52	0	0.49	28
c5	gs250a-2	258,287.0	257,724.9	**0.22**	0	0.31	68	258,077.0	257,744.4	0.23	125	0.42	2	**0.13**	192
c10	gs250a-2	257,990.0	257,600.0	0.24	0	**0.15**	92	257,990.0	257,625.1	**0.14**	120	0.22	3	0.19	175
c15	gs250a-2	257,911.0	257,564.4	0.45	0	**0.13**	17	257,911.0	257,536.4	**0.15**	109	0.27	1	-	-
c20	gs250a-2	258,193.0	257,462.5	0.53	0	**0.28**	6	258,054.0	257,471.5	0.28	11	0.53	0	**0.23**	15
c5	gs500a-1	513,476.0	510,860.9	0.53	0	**0.51**	0	513,364.0	510,866.9	0.51	0	**0.49**	0	0.55	0
c10	gs500a-1	513,148.0	510,733.5	0.48	0	**0.47**	0	513,091.0	510,734.9	0.47	0	0.52	0	**0.46**	2
c15	gs500a-1	512,919.0	510,637.7	0.47	0	**0.45**	0	512,919.0	510,635.8	**0.45**	0	0.47	0	**0.45**	0
c20	gs500a-1	513,158.0	510,568.0	0.51	0	**0.50**	0	513,131.0	510,568.0	-	-	0.52	0	**0.50**	0
c5	gs500a-2	513,663.0	510,844.5	0.61	0	**0.55**	0	513,544.0	510,846.2	0.55	0	0.61	0	**0.53**	0
c10	gs500a-2	513,357.0	510,717.7	0.57	0	**0.51**	0	513,357.0	510,719.7	**0.52**	0	0.55	0	**0.52**	0
c15	gs500a-2	513,127.0	510,616.9	**0.49**	0	**0.49**	0	513,127.0	510,617.4	**0.49**	0	**0.49**	0	**0.49**	0
c20	gs500a-2	513,511.0	510,545.7	**0.58**	0	0.59	0	513,254.0	510,545.7	-	-	**0.53**	0	0.58	0

Table 5: Results for large scale instances III: The best obtained gaps per setting and instance are shown in bold.

7 Conclusion

We provide a first theoretical comparison of MIP models for ConFL. We show that there are basically two groups of models, derived from the way the connectivity requirements in the whole graph are defined. Our “F” models require connectivity among open facilities and the root node, and in addition a proper assignment of customers. We derive the stronger “R” models by requiring connectivity between customers and the root node. There is also the weak Miller-Tucker-Zemlin formulation which follows a sub-tour elimination concept, instead of a connectivity-based one. In contrast to known results for the traveling salesman problem [38], we show that *MTZ* is not dominated by the two single commodity flow models. The second interesting result is that the integrality gap of all “F” models is not a constant value.

In our computational study we also obtain two surprising results. First, the branch-and-cut algorithm for the correspondingly weaker “F” cut-based model, significantly outperforms all other models in practice. Second, the weak but small *MTZ* formulation performs comparatively well, and in most cases outperforms even the branch-and-cut derived for the stronger “R” model.

Acknowledgements

The authors thank to Markus Chimani for the proof of Lemma 8b.

References

[1] Ufllib. URL `http://www.mpi-inf.mpg.de/departments/d1/projects/benchmarks/UflLib/`.

[2] IBM CPLEX. URL `http://www.ilog.com/products/cplex/`.

[3] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows.* Prentice Hall, 1993.

[4] M. G. Bardossy and S. Raghavan. Dual-based local search for the connected facility location and related problems. Technical report, Smith School of Business and Institute for Systems Research, University of Maryland, 2009.

[5] J. E. Beasley. OR-library. URL `http://people.brunel.ac.uk/~mastjjb/jeb/orlib/steininfo.html`.

[6] S. Chen, I. Ljubić, and S. Raghavan. The regenerator location problem. *Networks*, 2009. to appear.

[7] B. V. Cherkassky and A. V. Goldberg. On implementing push-relabel method for the maximum flow problem. *Algorithmica*, 19:390–410, 1994.

[8] S. Chopra and M. R. Rao. The Steiner tree problem I: Formulations, compositions and extension of facets. *Mathematical Programming*, 64:209–229, 1994.

[9] A. M. Costa, J.-F. Cordeau, and G. Laporte. Models and branch-and-cut algorithms for the Steiner tree problem with revenues, budget and hop constraints. *Networks*, 53(2):141–159, 2009.

[10] M. Desrochers and G. Laporte. Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters*, 10(1):27 – 36, 1991.

[11] F. Eisenbrand, F. Grandoni, T. Rothvoß, and G. Schäfer. Approximating connected facility location problems via random facility sampling and core detouring. In *SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1174–1183. SIAM, 2008.

[12] D. Ghosh. Neighborhood search heuristics for the uncapacitated facility location problem. *European Journal of Operations Research*, 150:150–162, 2003.

[13] M. X. Goemans. The Steiner tree polytope and related polyhedra. *Mathematical Programming*, 63:157–182, 1994.

[14] M. X. Goemans and Y. soo Myung. A catalog of Steiner tree formulations. *Networks*, 23, 1993.

[15] L. Gouveia. Using the Miller-Tucker-Zemlin constraints to formulate a minimal spanning tree problem with hop constraints. *Comput. Oper. Res.*, 22(9):959–970, 1995.

[16] A. Gupta, J. Kleinberg, A. Kumar, R. Rastogi, and B. Yener. Provisioning a virtual private network: A network design problem for multicommodity flow. In *In Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 389–398, 2001.

[17] H. Jung, M. Hasan, and K.-Y. Chwa. Improved primal-dual approximation algorithm for the connected facility location problem. *Combinatorial Optimization and Applications*, pages 265–277, 2008.

[18] D. R. Karger and M. Minkoff. Building Steiner trees with incomplete global knowledge. In *FOCS '00: Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pages 613–623, 2000.

[19] S. Khuller and A. Zhu. The general Steiner tree-star problem. *Information Processing Letters*, 84(4):215–220, 2002.

[20] T. Koch and A. Martin. Solving Steiner tree problems in graphs to optimality. *Networks*, 32:207–232, 1998.

[21] M. Koerkel. On the exact solution of large-scale simple plant location problems. *European Journal of Operations Research*, 39:157–173, 1989.

[22] J. Kratica, D. Tošić, V. Filipović, and I. Ljubić. Solving the simple plant location problem by genetic algorithms. *RAIRO - Operations Research*, 35(1):127–142, 2001.

[23] C. Krick, H. Räcke, and M. Westermann. Approximation algorithms for data management in networks. In *SPAA '01: Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures*, pages 237–246. ACM, 2001.

[24] Y. Lee, L. Lu, Y. Qiu, and F. Glover. Strong formulations and cutting planes for designing digital data service networks. *Telecommunication Systems*, 2(1):261–274, 1993.

[25] Y. Lee, Y. Chiu, and J. Ryan. A branch and cut algorithm for a Steiner tree-star problem. *INFORMS Journal on Computing*, 8(3):194–201, 1996.

[26] I. Ljubić. A hybrid VNS for connected facility location. In T. Bartz-Beielstein, M. J. B. Aguilera, C. Blum, B. Naujoks, A. Roli, G. Rudolph, and M. Sampels, editors, *Hybrid Metaheuristics*, volume 4771 of *Lecture Notes in Computer Science*, pages 157–169. Springer, 2007.

[27] I. Ljubić. *Exact and Memetic Algorithms for Two Network Design Problems*. PhD thesis, Faculty of Computer Science, Vienna University of Technology, November 2004.

[28] I. Ljubić, R. Weiskircher, U. Pferschy, G. Klau, P. Mutzel, and M. Fischetti. An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem. *Mathematical Progamming, Series B*, 105(2-3): 427–449, 2006.

[29] A. Lucena and M. G. C. Resende. Strong lower bounds for the prize-collecting Steiner problem in graphs. *Discrete Applied Mathematics*, 141:277–294, 2004.

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

[31] K. Mehlhorn. A faster approximation for the Steiner problem in graphs. *Information Processing Letters*, 27: 125–128, 1988.

[32] C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4):326–329, 1960.

[33] T. Öncan, I. K. Altınel, and G. Laporte. Invited review: A comparative analysis of several asymmetric traveling salesman problem formulations. *Comput. Oper. Res.*, 36(3):637–654, 2009.

[34] T. Polzin and S. V. Daneshmand. A comparison of Steiner tree relaxations. *Discrete Applied Mathematics*, 112 (1-3):241–261, 2001.

[35] C. Swamy and A. Kumar. Primal-dual algorithms for connected facility location problems. *Algorithmica*, 40: 245–269, 2004.

[36] A. Tomazic and I. Ljubić. A GRASP algorithm for the connected facility location problem. In *SAINT*, pages 257–260. IEEE Computer Society, 2008.

[37] D. P. Williamson and A. van Zuylen. A simpler and better derandomization of an approximation algorithm for single source rent-or-buy. *Operations Research Letters*, 35(6):707–712, 2007.

[38] R. T. Wong. Integer programming formulations of the traveling salesman problem. *Proceedings of the IEEE international conference of circuits and computers*, pages 149–52, 1980.

[39] J. Xu, S. Y. Chiu, and F. Glover. Tabu search for dynamic routing communications network design. *Telecommunication Systems*, 8(1):55–77, 1997.