TECHNISCHE UNIVERSITÄT WIEN
Institut für Computergraphik und Algorithmen

Branch-and-Cut-and-Price for Capacitated Connected Facility Location

Markus Leitner and Günther R. Raidl

Forschungsbericht / Technical Report

TR–186–1–10–01

May 27, 2010

Favoritenstraße 9-11 / E186, A-1040 Wien, Austria
Tel. +43 (1) 58801-18601, Fax +43 (1) 58801-18699
www.cg.tuwien.ac.at

Noname manuscript No.
(will be inserted by the editor)

Branch-and-Cut-and-Price for Capacitated Connected Facility Location

Markus Leitner · Günther R. Raidl

Received: dd.mm.yyyy / Accepted: dd.mm.yyyy

Abstract We consider a generalization of the Connected Facility Location problem (ConFL), suitable to model real world network extension scenarios such as fiber-to-the-curb. In addition to choosing a set of facilities and connecting them by a Steiner tree as in ConFL, we aim to maximize the resulting profit by potentially supplying only a subset of all customers. Furthermore, capacity constraints on potential facilities need to be considered. We present two mixed integer programming based approaches which are solved using branch-and-cut and branch-and-cut-and-price, respectively. By studying the corresponding polyhedra we analyze both approaches theoretically and show their advantages over previously presented models. Furthermore, using a computational study we are able to additionally show significant advantages of our models over previously presented ones from a practical point of view.

Keywords connected facility location · network design · branch-and-cut · branch-and-cut-and-price · mixed integer programming

Mathematics Subject Classification (2000) 90C10 · 90C11 · 90C57 · 90C90

M. Leitner · G. R. Raidl
Institute of Computer Graphics and Algorithms
Vienna University of Technology, Vienna, Austria
E-mail: leitner@ads.tuwien.ac.at, raidl@ads.tuwien.ac.at

1 Introduction

We consider a real-world network design problem with additional location aspects which occurs when extending existing fiber-optic networks. Nowadays, telecommunication companies are often confronted with rising bandwidth requirements of customers while especially in smaller cities and rural areas realizing connections entirely with fiber-optic routes (i.e. fiber-to-the-home) is often too expensive and does not pay off economically. In such situations, providers need to make a compromise between the bandwidth offered to individual customers and the resulting construction costs.

Frequently, these companies deal with such situations by extending the fiber-optic infrastructure by new routes to so-called *mediation points* that bridge the high-bandwidth network with an older lower-bandwidth network. While the original network is still used between a customer and its assigned mediation point, the newly installed high-bandwidth routes are used in the remaining network. Ensuring that the maximum distance between a customer and its mediation point is not too high, the bandwidth available for each customer can be significantly increased while avoiding too high construction costs. Depending on the network used between these mediation points and the customers, these scenarios are typically referred to as *fiber-to-the-curb* in case of a traditional copper network or *powerline* in case of using electric power transmission lines.

From an optimization point of view these scenarios can be modeled as variants of the *Connected Facility Location Problem (ConFL)* [26], where new facilities, which correspond to the above mentioned mediation points, need to be installed and connected with each other and customer nodes need to be assigned to them. However, the classical ConFL often cannot be used to model and solve real-world scenarios since it neglects real-world constraints such as those imposed by individual client bandwidth demands and corresponding maximum assignable demands to individual facilities. Furthermore, telecommunication providers are usually interested in upgrading not necessarily all but only the most profitable subset of potential customers by additionally considering the expected return on investment for individual customers.

To overcome these shortages, our model to which we refer as the *Rooted Prize Collecting Capacitated Connected Facility Location Problem (CConFL)* resembles a prize collecting variant of ConFL and additionally considers capacity constraints on potential facility locations.

After formally introducing CConFL in Section 2, we review previous and related work in Section 3. Afterwards, we present a branch-and-cut approach based on directed connectivity cuts in Section 4 and a branch-and-cut-and-price approach involving an exponential number of so-called pattern variables in Section 5. Theoretical comparisons of the corresponding polyhedra of these two formulations as well as to previously proposed formulations are given in Section 6. After describing the used test instances in Section 7, computational results are given in Section 8, before we finally draw conclusions and outline potential future work in Section 9.

2 Problem Definition

Formally, an instance of CConFL is given by an undirected connected graph $G^{\mathrm{o}} = (V^{\mathrm{o}}, E^{\mathrm{o}})$ with a connected subgraph $G_{\mathrm{I}} = (V_{\mathrm{I}}, E_{\mathrm{I}})$, $V_{\mathrm{I}} \subsetneq V^{\mathrm{o}}$, $E_{\mathrm{I}} \subsetneq E^{\mathrm{o}}$ representing the existing fiber-optic infrastructure, see Figure 1.

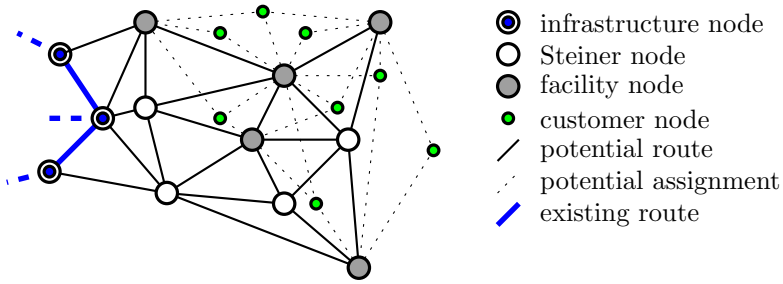

Fig. 1 Original problem instance.

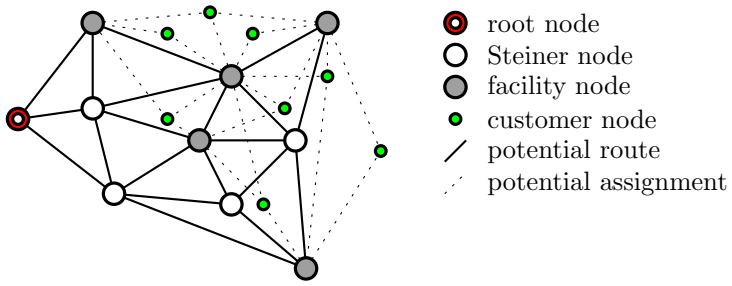

Fig. 2 Rooted problem instance.

Each edge $e = (u, v) \in E^{\mathrm{o}}$ has associated costs $c^{\mathrm{o}}_e \geq 0$ corresponding to the costs of installing a new route between u and v. Potential facility locations (mediation points) $F^{\mathrm{o}} \subseteq V^{\mathrm{o}} \setminus V_{\mathrm{I}}$ are given with associated costs $f_i \geq 0$ for installing them (*opening costs*) and maximum assignable demands $D_i \in \mathbb{N}_0$, $\forall i \in F^{\mathrm{o}}$. All remaining nodes $v \in V^{\mathrm{o}} \setminus (V_{\mathrm{I}} \cup F^{\mathrm{o}})$ are Steiner nodes that may be used in a solution. Note that each facility node might also be used as a Steiner node when no customer is assigned to it, in which case its opening costs need not to be paid. Furthermore, we are given a set of potential customers C^{o} with individual demands $d_k \in \mathbb{N}_0$ and prizes $p_k \geq 0$, $\forall k \in C^{\mathrm{o}}$, the latter corresponding to the expected profit when supplying customer k. Finally, costs $a_{i,k} \geq 0$ for assigning the complete demand of customer $k \in C^{\mathrm{o}}$ to a potential facility location $i \in F^{\mathrm{o}}$ are given (*assignment costs*). If a client k cannot be assigned to facility i we assume here for simplicity $a_{i,k} = \infty$.

During preprocessing we shrink the existing fiber-optic infrastructure $G_{\mathrm{I}} = (V_{\mathrm{I}}, E_{\mathrm{I}})$ into a single root node r, yielding a reduced graph $G = (V, E)$ with node set $V = (V^{\mathrm{o}} \cup \{r\}) \setminus V_{\mathrm{I}}$ and edge set $E = \{(u, v) \in E^{\mathrm{o}} \mid u, v \notin E_{\mathrm{I}}\} \cup \{(r, v) \mid \exists (u, v) \in E^{\mathrm{o}} : u \in V_{\mathrm{I}} \wedge v \notin V_{\mathrm{I}}\}$; see Figure 2 for such a rooted problem instance. Edge costs $c_e \geq 0$ are defined as

$$c_e = \begin{cases} c^{\mathrm{o}}_e & \text{if } u, v \in V^{\mathrm{o}} \setminus V_{\mathrm{I}} \\ \min_{f=(w,v) \in E^{\mathrm{o}} \mid w \in V_{\mathrm{I}}} c^{\mathrm{o}}_f & \text{otherwise} \end{cases} \qquad \forall e = (u, v) \in E.$$

Furthermore, we remove all possibly existing assignment possibilities between customers $k \in C^{\mathrm{o}}$ and facilities $i \in F^{\mathrm{o}}$ where $a_{i,k} \geq p_k$ by setting $a_{i,k} = \infty$. In case strict inequality holds – i.e. $a_{i,k} > p_k$ – such an a assignment cannot be part of an optimal

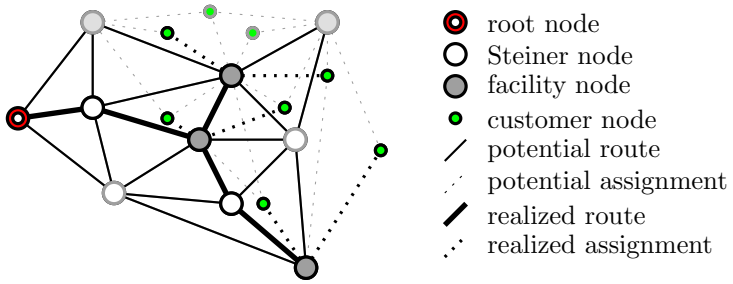

Fig. 3 An exemplary solution to CConFL.

solution as it does not pay off, while at least one optimal solution not including the assignment between i and k exists if $a_{i,k} = p_k$.

Customers with no remaining assignment possibilities are entirely removed. Similarly, some potential facilities $i \in F^{\mathrm{o}}$ that cannot be profitable can be identified by solving a 0–1 knapsack problem for each facility with knapsack size D_i, and an item with weight d_k and profit $p_k - a_{i,k}$ for each assignable customer. A facility can be removed if the profit of the optimal solution to this knapsack problem does not exceed the facility's opening costs f_i. If solving these knapsack problems for all the facilities is too time-consuming, an option is to only solve the corresponding linear programming relaxations and to use the hereby obtained upper bounds to the optimal solutions' profits.

We denote by $C \subseteq C^{\mathrm{o}}$ and $F \subseteq F^{\mathrm{o}}$ ($F \subseteq V$) the resulting, possibly reduced sets of potential customers and facility locations, respectively. Furthermore, $C_i = \{k \in C \mid a_{i,k} < p_k\}$ denotes the set of customers that may be assigned to facility $i \in F$ and $F_k = \{i \in F \mid k \in C_i\}$ the set of potential facilities a customer $k \in C$ may be assigned to.

As depicted in Figure 3, a solution to CConFL $S = (R_S, T_S, F_S, C_S, \alpha_S)$ consists of a set of opened facilities $F_S \subseteq F$ connected to each other as well as to the root node r by a Steiner tree (R_S, T_S), $R_S \subseteq V$, $T_S \subseteq E$. $C_S \subseteq C$ is the set of customers feasibly (i.e. respecting the capacity constraints) assigned to facilities F_S, whereas the actual mapping between customers and facilities is described by $\alpha_S : C_S \to F_S$. Each customer may be assigned to at most one facility. The objective function of CConFL can be stated as

$$c(S) = \sum_{e \in T_S} c_e + \sum_{i \in F_S} f_i + \sum_{k \in C_S} a_{\alpha_S(k),k} + \sum_{k \in C} p_k - \sum_{k \in C_S} p_k \tag{1}$$

$$= \sum_{e \in T_S} c_e + \sum_{i \in F_S} f_i + \sum_{k \in C_S} a_{\alpha_S(k),k} + \sum_{k \in C \setminus C_S} p_k \tag{2}$$

An optimal solution S^* (i.e. a most profitable one) has minimal objective value, i.e. $c(S^*) \leq c(S)$ for all feasible solutions S. Note that we add the profits lost – i.e. the profits of uncovered customers – instead of subtracting the collected profits in equation (2), ensuring a nonnegative objective value for any feasible solution. Since CConFL combines the (prize collecting) Steiner tree problem (STP) on a graph with the single source capacitated facility location problem (SSCFLP), which are both strongly $\mathcal{NP}$-hard [18,7], CConFL is strongly $\mathcal{NP}$-hard as well.

3 Related Work

Karger and Minkoff [17] considered the so-called maybecast problem which can be modeled as a connected facility location problem and described a constant factor approximation for their problem. The name connected facility location has been introduced by Gupta et al. [14] in their work on virtual private networks.

Since then several authors proposed approximation algorithms for diverse variants of ConFL. Swamy and Kumar [32] presented a primal-dual algorithm with an approximation ratio of 8.55 which is also a factor 4.55 approximation for the so-called rent-or-buy problem, a variant of ConFL where no opening costs are given and facilities may be opened at all nodes. By considering a linear programming (LP) rounding technique, Hasan et al. [15] improved their method to a factor 8.29 approximation algorithm for the case of edge costs obeying the triangle inequality and a factor seven approximation in case all opening costs are equal. Recently, a randomized approximation algorithm with an expected approximation ratio of four, which can be derandomized with a resulting approximation factor of 4.23, has been presented by Eisenbrand et al. [9].

Ljubić [26] described a branch-and-cut approach based on directed connection cuts as well as a hybrid metaheuristic combining variable neighborhood search (VNS) with reactive tabu search for the rooted variant of ConFL. Tomazic and Ljubić [33] considered the unrooted version of ConFL and presented a greedy randomized adaptive search procedure. Furthermore, they transformed the problem into the minimum Steiner arborescence problem and solved it by an exact branch-and-cut method. Ten different integer programming formulations for ConFL have been presented by Gollowitzer and Ljubić [13]. Next to computational results on their models, they further ranked them by comparing the various polyhedra. The same authors subsequently discussed a large number of models for a hop constrained variant of ConFL [27, 28]. Bardossy and Raghavan [31, 3] combined dual ascent with local search to derive lower and upper bounds for a more general variant of ConFL.

The current authors presented two VNS variants for a version of CConFL without assignment and opening costs in [23]. Subsequently, we proposed a Lagrangian relaxation based approach which has been hybridized with local search and very large scale neighborhood search as well as two mixed integer programming models based on multi-commodity flows [24, 25].

A closely related problem is the Steiner tree star (STS) problem, where opening costs for facilities included in the Steiner tree must be paid even if no customers are assigned to them. Exact methods for the STS problem have been described by Lee et al. [22, 21], while Xu et al. [35] presented a tabu search metaheuristic. A generalized variant of the STS problem, where customer nodes and potential facilities are not necessarily disjoint, has been described by Khuller and Zhu [19].

Furthermore, literature on the (prize collecting) Steiner tree problem on a graph (STP), as well as on the (single source) capacitated facility location problem (SSCFLP) can be considered as relevant, since CConFL is composed of these two problems; see e.g. [34] for a survey on the STP and [2] for a recent work on the SSCFLP including a comprehensive list of further references on that topic.

4 Branch-and-Cut for CConFL

In this section we present an exact approach for CConFL. The underlying integer programming model dCut involves an exponential number of constraints and can be solved by dynamically including them on demand at each node of the branch-and-bound search tree, i.e. by branch-and-cut. dCut is based on so-called directed connection cuts. It is well known that such models often outperform multi-commodity flow based models like the ones presented in our previous work [24, 25] from a computational point of view. As will be shown in Section 6, our directed cut model dCut is also theoretically stronger than both previously presented flow models.

For model dCut we define a directed extended graph (V', A') combining G with all potential customers, i.e. $V' = V \cup C$. Its arc set A' consists of one arc going out of the root node for each edge in G adjacent to r, while all other edges of G are replaced by two oppositely directed arcs. Furthermore, A' contains one assignment arc (i, k) for each potential assignment between a facility $i \in F$ and a customer $k \in C_i$. Arc costs $c'_{u,v}$, $\forall (u, v) \in A'$, are defined as

$$c'_{u,v} = \begin{cases} c_e & \text{if } e = (u, v) \in E \\ a_{u,v} & \text{otherwise.} \end{cases} \tag{3}$$

Model dCut uses variables $z_i \in \{0, 1\}$, $\forall i \in F$, indicating whether or not a facility is opened, variables $y_k \in \{0, 1\}$, $\forall k \in C$, denoting if a customer is supplied or not, and variables $x_{i,j} \in \{0, 1\}$, $\forall (i, j) \in A'$, specifying whether or not an arc is used.

$$\text{(dCut)} \quad \min \sum_{(u,v)\in A'} c'_{u,v} x_{u,v} + \sum_{i\in F} f_i z_i + \sum_{k\in C} p_k (1 - y_k) \tag{4}$$

$$\text{s.t.} \quad x_{u,v} + x_{v,u} \leq 1 \qquad \forall e = (u, v) \in E \mid u, v \neq r \tag{5}$$

$$x_{i,k} \leq z_i \qquad \forall i \in F,\ \forall k \in C_i \tag{6}$$

$$\sum_{i\in F_k} x_{i,k} \geq y_k \qquad \forall k \in C \tag{7}$$

$$\sum_{k\in C_i} d_k x_{i,k} \leq D_i z_i \qquad \forall i \in F \tag{8}$$

$$\sum_{(u,v)\in\delta^+(W)} x_{u,v} \geq z_i \qquad \forall i \in F,\ \forall W \subsetneq V \mid r \in W \wedge i \notin W \tag{9}$$

$$\sum_{(u,v)\in\delta^+(W)} x_{u,v} + \sum_{i\in F_k \cap W} x_{i,k} \geq y_k \qquad \forall k \in C,\ \forall W \subsetneq V \mid r \in W \tag{10}$$

$$x_{u,v} \in \{0, 1\} \qquad \forall (u, v) \in A' \tag{11}$$

$$z_i \in \{0, 1\} \qquad \forall i \in F \tag{12}$$

$$y_k \in \{0, 1\} \qquad \forall k \in C \tag{13}$$

Due to using (V', A'), assignment costs are represented as arc costs in the objective function (4). Constraints (5) ensure that no more than one out of each pair of oppositely directed arcs between two nodes is chosen, linking constraints (6) guarantee that an assignment arc may only be used if the corresponding facility is opened, while inequalities (7) ensure that a customer's prize can only be earned

if it is assigned to a facility by an assignment arc. Constraints (8) are the capacity constraints for each facility. In inequalities (9) and (10) which resemble the directed connection inequalities for facilities and customers, respectively, we denote by $\delta^+(W) = \{(u,v) \in A' \mid u,v \in V \wedge u \in W \wedge v \notin W\}$ the set of arcs going out of node set W, i.e. the cutset of W. Since customer nodes have only incoming arcs, we need not consider other customer nodes than k for the directed connection constraints to $k \in C$ in (10). Note that the directed connection inequalities for customers (10) only strengthen the LP relaxation of model dCut, but omitting them would also yield a valid model for CConFL.

Since the number of connectivity constraints (9) and (10) is exponentially large, we dynamically identify inequalities violated by a current solution to the LP relaxation as cutting planes during runtime. Computing a cut of minimum capacity between two nodes u and v is equivalent to determining a maximum flow between these nodes. Thus, we use an implementation of the push-relabel method for the maximum flow problem by Cherkassky and Goldberg [6] for identifying violated connectivity inequalities.

5 Branch-and-Cut-and-Price for CConFL

Model dBCP presented in this section considers whole profitable assignment patterns between customers and facilities instead of taking into account each potential assignment individually. We consider the set of all feasible and profitable assignment patterns Ω_i for facility $i \in F$ and denote by $\Omega = \bigcup_{i \in F} \Omega_i$ the total set of such assignment patterns. By $\Omega(k) \subseteq \Omega$, $k \in C$, we denote the set of patterns connecting customer k. Each pattern $\omega \in \Omega$ assigns a set of customers $\mathcal{C}(\omega) = \{k \in C \mid \omega \in \Omega(k)\}$ to a dedicated facility $\mathcal{F}(\omega) \in F$, with $\mathcal{F}(\omega) = i \in F \Leftrightarrow \omega \in \Omega_i$. Furthermore, let Ω only contain valid and profitable patterns, i.e. $\mathcal{C}(\omega) \subseteq C_{\mathcal{F}(\omega)}$, $\sum_{k \in \mathcal{C}(\omega)} d_k \leq D_{\mathcal{F}(\omega)}$, and $\sum_{k \in \mathcal{C}(\omega)} p_k - a_{\mathcal{F}(\omega),k} > f_{\mathcal{F}(\omega)}$, $\forall \omega \in \Omega$. As dCut, model dBCP uses variables $z_i \in \{0,1\}$, $\forall i \in F$, indicating opened respectively closed facilities, and variables $y_k \in \{0,1\}$, $\forall k \in C$, denoting if a customer is connected. Variables $\gamma_\omega \in \{0,1\}$, $\forall \omega \in \Omega$, denote whether a pattern is realized or not. Since these pattern variables implicitly model assignments between facilities and customers, we need not further consider corresponding assignment arcs of the graph and thus, variables $x_{u,v} \in \{0,1\}$, $\forall (u,v) \in A = \{(u,v),(v,u) \mid (u,v) \in E \wedge u,v \neq r\} \cup \{(r,v) \mid (r,v) \in E\}$ indicate whether an arc is used in the Steiner tree connecting open facilities and the root node.

$$\text{(dBCP)} \quad \min \sum_{(u,v) \in A} c'_{u,v} x_{u,v} + \sum_{i \in F} f_i z_i + \sum_{k \in C} p_k (1 - y_k) + \\ + \sum_{\omega \in \Omega} \sum_{k \in \mathcal{C}(\omega)} a_{\mathcal{F}(\omega),k} \gamma_\omega \tag{14}$$

$$\text{s.t.} \quad \sum_{\omega \in \Omega_i} \gamma_\omega \leq z_i \qquad \forall i \in F \tag{15}$$

$$\sum_{\omega \in \Omega(k)} \gamma_\omega \geq y_k \qquad \forall k \in C \tag{16}$$

$$x_{u,v} + x_{v,u} \leq 1 \qquad \forall (u,v) \in E \mid u,v \neq r \tag{17}$$

$$\sum_{(u,v) \in \delta^+(W)} x_{u,v} \geq z_i \qquad \forall i \in F,\ \forall W \subsetneq V \mid r \in W \wedge i \notin W \tag{18}$$

$$\sum_{(u,v)\in\delta^+(W)} x_{u,v} + \\ + \sum_{i\in F_k\cap W} \sum_{\omega\in\Omega_i\cap\Omega(k)} \gamma_\omega \geq y_k \qquad \forall k \in C,\ \forall W \subsetneq V \mid r \in W \qquad (19)$$

$$z_i \in \{0,1\} \qquad \forall i \in F \qquad (20)$$

$$y_k \in \{0,1\} \qquad \forall k \in C \qquad (21)$$

$$x_{u,v} \in \{0,1\} \qquad \forall (u,v) \in A \qquad (22)$$

$$\gamma_\omega \in \{0,1\} \qquad \forall \omega \in \Omega \qquad (23)$$

Constraints (15) and (16) are the coupling constraints between assignment patterns and facilities respectively customers. As for model dCut, constraints (17) ensure that no more than one arc of each pair of oppositely directed arcs can be used, while constraints (18) are the directed connection inequalities for facilities. Constraints (19) – which are again only included to strengthen the LP relaxation of model dBCP – resemble the directed connectivity inequalities for customers. They need to be partly expressed in terms of pattern variables, since no variables explicitly modeling assignments between facilities and customers are included in dBCP.

As for model dCut, connectivity cuts for facilities as well as for customers are added as cutting planes to the model on demand only. Note that variables z_i, $\forall i \in F$, as well as y_k, $\forall k \in C$, are declared as binary due to our branching strategy – see Section 5.1 – while defining them as continuous would also yield a valid model.

Since Ω contains exponentially many variables, we cannot solve dBCP directly by branch-and-cut but additionally have to apply column generation. See e.g. [4,8] for general introductions to column generation and branch-and-price. As usual in such approaches we consider the reduced master problem (RMP) containing only a small subset of variables $\tilde{\Omega} \subsetneq \Omega$ where constraints (20)–(23) are replaced by their continuous relaxations. After solving this RMP, we search for new pattern variables that price out favorably in the pricing problem. If at least one such column is found, it is added to RMP, which in turn is resolved. This process is repeated until no further columns can be added.

Let $\mu_i \leq 0$, $\forall i \in F$, be the dual variables associated to constraints (15), $\pi_k \geq 0$, $\forall k \in C$, the dual variables associated to constraints (16), and $\lambda_{k,W} \geq 0$, $\forall k \in C$, $\forall W \subsetneq V \mid r \in W$, the dual variables associated to the customers' connection inequalities (19). Let $W(i,k) = \{W \subseteq V \mid r, i \in W\}$, $\forall i \in F$, $\forall k \in C_i$, denote the set of all subsets of V including the root node and at least one facility to which a customer k can be assigned.

When solving RMP, we obtain optimal dual variable values μ_i^*, π_k^*, and $\lambda_{k,W}^*$, defining reduced costs $\bar{c}_\omega$ for variables $\omega \in \Omega \setminus \tilde{\Omega}$:

$$\bar{c}_\omega = \sum_{k\in\mathcal{C}(\omega)} a_{\mathcal{F}(\omega),k} - \mu_{\mathcal{F}(\omega)} - \sum_{k\in\mathcal{C}(\omega)} \pi_k - \sum_{k\in\mathcal{C}(\omega)} \sum_{Q\in W(\mathcal{F}(\omega),k)} \lambda_{k,Q} \qquad (24)$$

$$= -\mu_{\mathcal{F}(\omega)} - \sum_{k\in\mathcal{C}(\omega)} \left(\pi_k - a_{\mathcal{F}(\omega),k} + \sum_{Q\in W(\mathcal{F}(\omega),k)} \lambda_{k,Q} \right). \qquad (25)$$

The pricing problem is to find a pattern $\omega^* \in \Omega \setminus \tilde{\Omega}$ yielding minimum reduced costs, i.e.

$$\omega^* = \text{argmin}_{\omega \in \Omega \setminus \tilde{\Omega}} \{\bar{c}_\omega\}.$$

In other words, we need to find a feasible assignment ω between some customers $\mathcal{C}(\omega)$ and a facility $\mathcal{F}(\omega)$ yielding negative reduced costs $\bar{c}_\omega$ or prove that no such assignment exists.

Thus, we need to solve a binary knapsack problem for each facility $i \in F$, with one item for each customer $k \in C_i$ assignable to i, demand d_k, and profit $\pi_k - a_{i,k} + \sum_{Q \in W(i,k)} \lambda_{k,Q}$, where we obviously need not consider items with negative or zero profit. The total capacity of the knapsack is D_i. If $|\mu_i|$ is smaller than the total profit of the optimal solution to such a knapsack problem, the corresponding pattern variable has negative reduced costs, in which case it is added to RMP.

5.1 Branching in Branch-and-Price

Branching on the exponentially large set of variables γ_ω, $\forall \omega \in \Omega$, is not a viable option since it would lead to strong asymmetries in the partitioning of the search space. Thus, next to variables z_i, $\forall i \in F$, variables $x_{u,v}$, $\forall (u,v) \in A$, and variables y_k, $\forall k \in C$, we accomplish branching by decisions on assignments between facilities and customers. Integrality on one such assignment between a facility $i \in F$ and a customer $k \in C_i$ can be achieved by adding either branching constraint (26) or (27) to the model if $\sum_{\omega \in \tilde{\Omega}(k) \cap \tilde{\Omega}_i} \gamma_\omega$ is fractional.

$$\sum_{\omega \in \tilde{\Omega}(k) \cap \tilde{\Omega}_i} \gamma_\omega = 0 \tag{26}$$

$$\sum_{\omega \in \tilde{\Omega}(k) \cap \tilde{\Omega}_i} \gamma_\omega = 1 \tag{27}$$

For each included branching constraint, we need to consider its dual variable value in the pricing problem when solving a knapsack problem with an item corresponding to an assignment fixed due to an already included branching constraint. Adding such additional terms in the pricing problem eventually modifies an item's profit but does not affect the structure of the pricing problem, i.e. the approach is robust.

Lemma 1 proves that any solution S' to the LP relaxation of dBCP (denoted by dBCP^{LP}) for which – according to above mentioned branching rules – no further branching can be accomplished represents a feasible solution to CConFL, i.e. eventually existing pattern variables with fractional values can be replaced by pattern variables with integral values while maintaining all assignments between facilities and customers.

Lemma 1 *Consider a solution S' to dBCP^{LP} and an arbitrary facility $i \in F$. Let $\Omega' = \{\omega \in \tilde{\Omega}_i \mid \gamma_\omega^{\text{LP}} \neq 0\}$ denote the set of active patterns for i in S', and $C' = \{k \in C \mid \exists \omega \in \Omega'(k)\}$ denote the set of customers assigned to i in S'. Furthermore, assume that $\sum_{\omega \in \Omega'(k)} \gamma_\omega = 1$, $\forall k \in C'$. Then $\zeta \in \Omega_i$ exists such that $C' = \mathcal{C}(\zeta)$.*

Proof Let $\zeta \in \Omega_i$ denote the single variable replacing all variables $\omega \in \Omega'$, i.e. $\mathcal{C}(\zeta) = C'$. Due to the implicit integrality of each assignment between i and a customer $k \in C'$ we only need to prove that ζ does not violate the capacity constraints. Due to constraints (15) the following inequality holds:

$$D_\zeta = \sum_{k \in C'} d_k = \sum_{\omega \in \Omega'} \gamma_\omega^{\mathrm{LP}} \sum_{k \in \mathcal{C}(\omega)} d_k \leq \sum_{\omega \in \Omega'} \gamma_\omega^{\mathrm{LP}} D_i = D_i \sum_{\omega \in \Omega'} \gamma_\omega^{\mathrm{LP}} \leq D_i.$$

6 Polyhedral Comparison

In this section, we compare the polyhedra corresponding to the sets of feasible solutions of the LP relaxations of dCut and dBCP as well as the two previously presented directed multi-commodity flow based formulations $\mathrm{dMCF_f}$ and $\mathrm{dMCF_c}$ [25]. Models $\mathrm{dMCF_f}$ and $\mathrm{dMCF_c}$ mainly differ by means of the target nodes of their correspondingly defined flows, which are potential facility nodes for $\mathrm{dMCF_f}$ and customer nodes for $\mathrm{dMCF_c}$.

In the following, we denote by $\mathcal{P}_{\mathrm{dMCF_f}}$ the polyhedron corresponding to the set of feasible solutions to the LP relaxation of model $\mathrm{dMCF_f}$. Similarly, $\mathcal{P}_{\mathrm{dMCF_c}}$ denotes the polyhedron induced by the LP relaxation of model $\mathrm{dMCF_c}$, $\mathcal{P}_{\mathrm{dCut}}$ the one of model dCut, and $\mathcal{P}_{\mathrm{dBCP}}$ the one model dBCP. Furthermore, superscript LP denotes the linear programming relaxation of a model, e.g. $\mathrm{dMCF_f}^{\mathrm{LP}}$ denotes the LP relaxation of model $\mathrm{dMCF_f}$. By $proj_{x,y,z}(\mathcal{P})$ we refer to the projection of a polyhedron $\mathcal{P}$ into the space of x, y, z variables only. As a prerequisite, we are also reviewing the two MCF formulations from [25] in this section.

Model $\mathrm{dMCF_f}$ presented in [25] which is based on sending one unit of flow to each potential facility location uses the directed extended graph (V', A') as defined in Section 4, the undirected edge set $E' = E \cup \{(i,k) \mid i \in F \wedge k \in C_i\}$, and the corresponding undirected edge cost function

$$c''_e = \begin{cases} c_e & \text{if } e \in E \\ a_{i,k} & \text{otherwise} \end{cases} \qquad \forall e = (i,k) \in E'.$$

$A_i = A' \setminus \{(j,k) \in A' \mid j \in F \wedge k \in C_j\}$, $\forall i \in F$, is the set of arcs relevant for connecting a facility $i \in F$ to the root node r. In model $\mathrm{dMCF_f}$ decision variables $x_e \in \{0,1\}$, $\forall e \in E'$, indicating whether an edge is used in a solution (in which case $x_e = 1$) or not and variables $y_k \in \{0,1\}$, $\forall k \in C$, to specify whether a customer is feasibly assigned to an opened facility ($y_k = 1$) or not are used. Furthermore, to specify whether an arc is used in the connection to a potential facility we use flow variables $s^i_{u,v} \in \{0,1\}$, $\forall i \in F$, $\forall (u,v) \in A_i$, and design variables $z_i \in \{0,1\}$, $\forall i \in F$, to indicate if a potential facility is opened ($z_i = 1$).

$$(\mathrm{dMCF_f})\ \min \sum_{e \in E'} c''_e x_e + \sum_{i \in F} f_i z_i + \sum_{k \in C} p_k (1 - y_k) \tag{28}$$

$$\text{s.t.} \sum_{(u,v) \in A_i} s^i_{u,v} - \sum_{(v,u) \in A_i} s^i_{v,u} = \begin{cases} -z_i & \text{if } v = r \\ z_i & \text{if } v = i \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in F,\ \forall v \in V \tag{29}$$

$$s^i_{u,v} + s^i_{v,u} \leq x_e \qquad \forall i \in F,\ \forall e = (u,v) \in E \mid u, v \neq r \tag{30}$$

$$s^i_{r,v} \leq x_e \qquad \forall i \in F,\ \forall e = (r,v) \in E \tag{31}$$

$$x_e \leq z_i \qquad \forall e = (i,k) \in E' \mid k \in C \tag{32}$$

$$\sum_{e=(i,k)\in E'\setminus E} d_k x_e \leq D_i z_i \qquad \forall i \in F \quad (33)$$

$$\sum_{e=(i,k)\in E'\setminus E} x_e \geq y_k \qquad \forall k \in C \quad (34)$$

$$0 \leq s^i_{u,v} \leq 1 \qquad \forall i \in F,\ \forall (u,v) \in A_i \quad (35)$$

$$0 \leq z_i \leq 1 \qquad \forall i \in F \quad (36)$$

$$x_e \in \{0,1\} \qquad \forall e \in E' \quad (37)$$

$$y_k \in \{0,1\} \qquad \forall k \in C \quad (38)$$

Model dMCF$_\mathrm{c}$ sends one unit of flow to each potential customer, but otherwise is similar to model dMCF$_\mathrm{f}$. We define the set of relevant arcs for each customer $k \in C$ as $A_k = A' \setminus \{(i,k') \in A' \mid k' \in C \wedge k' \neq k\}$ and similar to dMCF$_\mathrm{f}$ use decision variables $x_e \in \{0,1\}$, $\forall e \in E'$, for indicating used edges, variables $y_k \in \{0,1\}$, $\forall k \in C$, to specify supplied customers, variables $z_i \in \{0,1\}$, $\forall i \in F$, to indicate if a potential facility is opened, and flow variables $s^k_{u,v}$, $\forall k \in C$, $\forall (u,v) \in A_k$, to specify if an arc is used to connect customer k.

$$(\mathrm{dMCF_c})\ \min \sum_{e\in E'} c''_e x_e + \sum_{i\in F} f_i z_i + \sum_{k\in C} p_k(1-y_k) \quad (39)$$

$$\text{s.t.} \sum_{(u,v)\in A_k} s^k_{u,v} - \sum_{(v,u)\in A_k} s^k_{v,u} = \begin{cases} -y_k & \text{if } v = r \\ y_k & \text{if } v = k \\ 0 & \text{otherwise} \end{cases} \qquad \forall k \in C,\ \forall v \in E' \quad (40)$$

$$s^k_{u,v} + s^k_{v,u} \leq x_e \qquad \forall k \in C,\ \forall e = (u,v) \in E' \mid u,v \neq r \quad (41)$$

$$s^k_{r,v} \leq x_e \qquad \forall k \in C,\ \forall e = (r,v) \in E \quad (42)$$

$$x_e \leq z_i \qquad \forall e = (i,k) \in E' \mid k \in C \quad (43)$$

$$\sum_{e=(i,k)\in E'\setminus E} d_k x_e \leq D_i z_i \qquad \forall i \in F \quad (44)$$

$$0 \leq s^k_{u,v} \leq 1 \qquad \forall k \in C,\ \forall (u,v) \in A_k \quad (45)$$

$$0 \leq z_i \leq 1 \qquad \forall i \in F \quad (46)$$

$$x_e \in \{0,1\} \qquad \forall e \in E' \quad (47)$$

$$y_k \in \{0,1\} \qquad \forall k \in C \quad (48)$$

Lemma 2 dCut *dominates* dMCF$_\mathrm{f}$*, i.e.* $proj_{x,y,z}(\mathcal{P}_{\mathrm{dCut}}) \subseteq proj_{x,y,z}(\mathcal{P}_{\mathrm{dMCF_f}})$.

Proof dMCF$_\mathrm{f}$ differs from dCut by modeling connections to facilities by multi-commodity flow constraints instead of directed connection inequalities (9), whereas dCut additionally contains directed connection inequalities for customers (10). The max-flow min-cut theorem [10] implies that for an arbitrary facility $i \in F$ with $\sum_{(u,v)\in\delta^+(W)} x^{\mathrm{LP}}_{u,v} \geq z^{\mathrm{LP}}_i$, $\forall W \subsetneq V \mid r \in W \wedge i \notin W$, a feasible flow of value z^{LP}_i from the root node to i exists; compare [29]. Thus, when projecting solutions into the domain of x, y, and z variables only, any solution to dCut$^{\mathrm{LP}}$ is also valid for dMCF$_\mathrm{f}{}^{\mathrm{LP}}$.

Lemma 3 dCut *dominates* dMCF$_\mathrm{c}$*, i.e.* $proj_{x,y,z}(\mathcal{P}_{\mathrm{dCut}}) \subseteq proj_{x,y,z}(\mathcal{P}_{\mathrm{dMCF_c}})$.

Proof dMCF_c differs from dCut by modeling connections to customers by multi-commodity flow constraints instead of directed connection inequalities (10) whereas dCut additionally contains directed connection inequalities for facilities. Thus, as for Lemma 2 the max-flow min-cut argument also holds for the flow to customers.

Theorem 1 dCut *strictly dominates* dMCF_f *and* dMCF_c*, i.e.* $proj_{x,y,z}(\mathcal{P}_{\text{dCut}}) \subsetneq proj_{x,y,z}(\mathcal{P}_{\text{dMCF}_\text{f}})$ *and* $proj_{x,y,z}(\mathcal{P}_{\text{dCut}}) \subsetneq proj_{x,y,z}(\mathcal{P}_{\text{dMCF}_\text{c}})$.

Proof Since none of the multi-commodity flow formulations dominates the other [25], i.e. $proj_{x,y,z}(\mathcal{P}_{\text{dMCF}_\text{c}}) \nsubseteq proj_{x,y,z}(\mathcal{P}_{\text{dMCF}_\text{f}})$ and $proj_{x,y,z}(\mathcal{P}_{\text{dMCF}_\text{f}}) \nsubseteq proj_{x,y,z}(\mathcal{P}_{\text{dMCF}_\text{c}})$, Theorem 1 follows from Lemmas 2 and 3.

Theorem 2 dBCP *strictly dominates* dCut*, i.e.* $proj_{x,y,z}(\mathcal{P}_{\text{dBCP}}) \subsetneq proj_{x,y,z}(\mathcal{P}_{\text{dCut}})$.

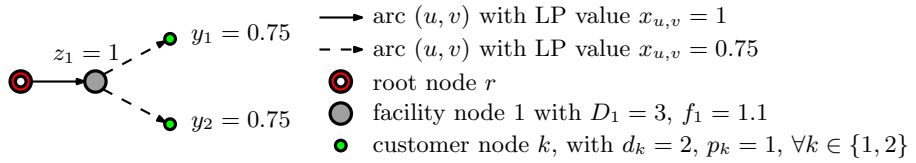

Fig. 4 Feasible LP solution of dCut which is infeasible for dBCP.

Proof Consider a fractional solution S' according to the example given in Figure 4 assuming zero costs for all included arcs. As can be easily seen S' is valid for dCut^{LP}. For describing S' in the space of dBCP, each assignment pattern ω can only contain one of the customers. However, since those patterns do not pay off – i.e. the collected profit is smaller than the facilities' opening costs f_1 – $\omega \notin \Omega$ and thus $S' \notin \text{dBCP}^{\text{LP}}$.

Now, we consider a solution $S^{\text{bcp}} \in \text{dBCP}^{\text{LP}}$ and denote by $\gamma_\omega^{\text{bcp}}$, $\forall \omega \in \Omega$, $x_{u,v}^{\text{bcp}}$, $\forall (u, v) \in A$, z_i^{bcp}, $\forall i \in F$, and y_k^{bcp}, $\forall k \in C$, the values of all variables of S^{bcp}. Using equations (49)–(52) we transform these values to the space of dCut, where superscript cut denotes a value with respect to dCut^{LP} and S^{cut} the corresponding solution to dCut^{LP}.

$$z_i^{\text{cut}} = z_i^{\text{bcp}} \qquad \forall i \in F \tag{49}$$

$$y_k^{\text{cut}} = y_k^{\text{bcp}} \qquad \forall k \in C \tag{50}$$

$$x_{u,v}^{\text{cut}} = x_{u,v}^{\text{bcp}} \qquad \forall (u, v) \in A \tag{51}$$

$$x_{i,k}^{\text{cut}} = \sum_{\omega \in \Omega_i \cap \Omega(k)} \gamma_\omega^{\text{bcp}} \qquad \forall i \in F,\ \forall k \in C_i \tag{52}$$

To show that $S^{\text{cut}} \in \text{dCut}^{\text{LP}}$ and thus $\text{dBCP}^{\text{LP}} \subseteq \text{dCut}^{\text{LP}}$ we consider each set of constraints from dCut in turn. S^{cut} obviously does not violate constraints (5), since (17) identically models them in dBCP. Validity of constraints (6) follows from above mentioned transformation rules and constraints (15):

$$x^{\text{cut}}_{i,k} = \sum_{\omega \in \Omega_i \cap \Omega(k)} \gamma^{\text{bcp}}_{\omega} \leq \sum_{\omega \in \Omega_i} \gamma^{\text{bcp}}_{\omega} \leq z^{\text{bcp}}_i = z^{\text{cut}}_i.$$

Using our transformation rules and constraints (16) the following inequality ensures that S^{cut} does not violate constraints (7):

$$y^{\text{cut}}_k = y^{\text{bcp}}_k \leq \sum_{\omega \in \Omega(k)} \gamma^{\text{bcp}}_{\omega} = \sum_{i \in F_k} \sum_{\omega \in \Omega_i \cap \Omega(k)} \gamma^{\text{bcp}}_{\omega} = \sum_{i \in F_k} x^{\text{cut}}_{i,k}.$$

Using constraints (15) and the fact that the total demand of a single pattern $\omega \in \Omega_i$ does not exceed the maximum assignable demand D_i of its facility $i \in F$, the validity of the capacity constraints (8) is ensured as follows:

$$\begin{aligned}\sum_{k \in C_i} d_k x^{\text{cut}}_{i,k} = \sum_{k \in C_i} d_k \sum_{\omega \in \Omega_i \cap \Omega(k)} \gamma^{\text{bcp}}_{\omega} = \sum_{\omega \in \Omega_i} \gamma^{\text{bcp}}_{\omega} \sum_{k \in \mathcal{C}(\omega)} d_k \leq \\ \leq \sum_{\omega \in \Omega_i} \gamma^{\text{bcp}}_{\omega} D_i \leq D_i z^{\text{bcp}}_i = D_i z^{\text{cut}}_i.\end{aligned}$$

Since directed connection cuts for facilities are identically included in both formulations and the validity of customer connection cuts (10) immediately follows by substituting $\sum_{\omega \in \Omega_i \cap \Omega(k)} \gamma_\omega$ by $x_{i,k}$ in the customer connection cuts (19) of dBCP, we conclude that $S^{\text{cut}} \in \text{dCut}^{\text{LP}}$.

7 Test Instances and Environment

For ConFL, Ljubić [26] combined benchmark instances for the STP with instances for uncapacitated facility location. Similarly, we created instances for CConFL in [24][1] by combining STP instances from the OR-library[2] with instances for the SSCFLP created with the instance generator of Kratica et al. [20][3].

The node with index one in the STP instance is selected as root node, while $|F|$ other nodes are randomly chosen as potential facility locations. Customers with associated demands, assignment costs as well the maximum assignable demands and opening costs for each facility are given by the SSCFLP instance. Next, we need to choose reasonable customer prizes, high enough to ensure that some customers will be supplied while avoiding to create relatively easy instances by setting these values too high. For each customer $k \in C$, we randomly choose its prize $p_k \in \mathbb{N}_0$ from the interval $\left[\overline{a}(k), a_{\max}(k) + \overline{f}\right]$, where $\overline{a}(k) = \frac{\sum_{i \in F_k} a_{i,k}}{|F_k|}$ denotes the average assignment costs of customer k, $a_{\max}(k) = \max_{i \in F_k}\{a_{i,k}\}$ the maximum assignment costs of customer k, and $\overline{f} = \frac{\sum_{i \in F} f_i}{|F|}$ the average facility opening costs. This ensures that each customer may be assigned to the majority of potential facilities in a profitable way. In particular it turned out that no customers or facilities are completely removed from an instance during preprocessing. Finally, degree-one and degree-two filtering [5] is applied

[1] available at http://www.ads.tuwien.ac.at/people/mleitner/cconfl/instances.tar.gz
[2] http://people.brunel.ac.uk/~mastjjb/jeb/orlib/steininfo.html
[3] http://alas.matf.bg.ac.yu/~kratica/instances/splp_gen_w32.zip

to remove some Steiner nodes and edges. A more detailed description of the individual characteristics of all resulting instances is given in Tables 1 and 2.

We performed all computational experiments on a single core of an Intel Core 2 Quad with 2.83GHz and 8GB RAM. IBM CPLEX 12.1 [16] has been used for directly solving dMCF_f, dMCF_f, and dCut as well as their LP relaxations ${\text{dMCF}_\text{c}}^{\text{LP}}$, ${\text{dMCF}_\text{f}}^{\text{LP}}$, and dCut^{LP}. SCIP 1.2.0 [1,36] with IBM CPLEX 12.1 [16] as embedded LP solver has been used for solving dBCP and its LP relaxation dBCP^{LP}.

We used the single threaded variant of CPLEX to allow for a fair comparison. A CPU-time limit of 7200 seconds has been applied in all experiments.

8 Computational Results

In the following the obtained computational results are summarized. First, results on the exact models dCut, and dBCP are given and compared to dMCF_f which has shown to outperform dMCF_c in our previous work [25]. After evaluating their LP relaxation values and corresponding runtimes, we further analyze obtained bounds and optimality gaps after 7200 CPU-seconds. Finally, the most promising model is compared to the Lagrangian relaxation / very large scale neighborhood search hybrid from [25], denoted here by LDV.

When solving dCut and dBCP, we separate directed connection cuts for customers only if no further violated connection cuts for facilities can be found. For dBCP, we initially set $\tilde{\Omega} = \emptyset$ and accomplish branching by considering variables z_i, $\forall i \in F$, $x_{u,v}$, $\forall (u,v) \in A$, and y_k, $\forall k \in C$, in this order before considering assignments between facilities and customers. For each set of variables, branching is performed on a most fractional variable; ties are broken at random. We did not implement problem specific primal heuristics to speed-up the solution of our models, but simply trust on the built-in heuristics of CPLEX and SCIP, respectively. We use the Combo algorithm[4] of Martello et al. [30] for solving the binary knapsack problems occurring in the pricing subproblems of dBCP.

Computational results for the LP relaxations of dMCF_f, dCut, and dBCP are summarized in Table 1 for instances with $|F| = |C|$ and in Table 2 for instances with $|F| \neq |C|$. These tables also detail the used test instances. For each considered instance, its number of potential facility locations $|F|$, its number of customers $|C|$, as well as its number of nodes $|V|$ and edges $|E|$ are given. All further tables will refer to an instance by its number only, which is given in the first column of Tables 1 and 2, respectively.

We conclude that in addition to their theoretical advantages and thus better LP relaxation values, the necessary CPU times to solve the linear relaxations of dCut and dBCP are significantly smaller for all tested instances than for dMCF_f. Moreover, dBCP^{LP} can be solved much faster than dCut^{LP} for almost all instances.

Results on best obtained lower and upper bounds as well as corresponding gaps and needed CPU times for dMCF_f, dCut, and dBCP are presented in Table 3 for instances with $|F| = |C|$ and in Table 4 for instances with $|F| \neq |C|$. Since dMCF_f could not solve any instance to proven optimality, we do not report its runtime which is equal to the time limit of 7200 seconds in each run. All lower and upper bounds are rounded to the first decimal place.

[4] http://www.diku.dk/~pisinger/codes.html

Table 1 Comparison of LP relaxation values and corresponding CPU-times in seconds for $dMCF_f$, dCut, and dBCP (time limit 7200s) on instances with $|F| = |C|$. Best values are marked bold.

	Instance					LP value			CPU time [s]		
Nr	Name	$\|F\|$	$\|C\|$	$\|V\|$	$\|E\|$	$dMCF_f^{LP}$	$dCut^{LP}$	$dBCP^{LP}$	$dMCF_f^{LP}$	$dCut^{LP}$	$dBCP^{LP}$
1	c10-mo75	75	75	408	908	2878.7	2912.5	**2914.8**	94	7	**5**
2	c10-mq75	75	75	405	905	7095.2	7116.2	**7119.9**	116	5	**3**
3	c10-ms75	75	75	407	907	9506.3	9533.8	**9536.7**	194	11	**3**
4	c15-mo75	75	75	500	2500	2747.5	2766.8	**2767.9**	877	94	**26**
5	c15-mq75	75	75	500	2500	7466.5	7489.0	**7493.3**	1567	30	**12**
6	c15-ms75	75	75	500	2500	9354.6	9368.5	**9371.0**	2040	16	**5**
7	d10-mo75	75	75	771	1770	2772.6	2800.5	**2802.8**	484	19	**11**
8	d10-mq75	75	75	775	1774	7295.0	7328.2	**7332.7**	167	16	**4**
9	d10-ms75	75	75	781	1780	10069.3	10112.7	**10115.1**	1103	31	**9**
10	d15-mo75	75	75	1000	5000	2641.8	2662.0	**2664.1**	2402	61	**21**
11	d15-mq75	75	75	1000	5000	-	7395.5	**7401.7**	7200	90	**27**
12	d15-ms75	75	75	1000	5000	-	9256.0	**9258.4**	7200	136	**31**
13	c10-mo100	100	100	406	906	3330.9	3363.0	**3365.3**	217	22	**15**
14	c10-mq100	100	100	406	906	9352.6	9397.7	**9403.5**	367	17	**8**
15	c10-ms100	100	100	416	916	11740.1	11781.9	**11788.7**	166	10	**8**
16	c15-mo100	100	100	500	2500	3422.6	3449.7	**3454.3**	2809	37	**15**
17	c15-mq100	100	100	500	2500	9120.5	9141.2	**9149.0**	4008	23	**8**
18	c15-ms100	100	100	500	2500	11277.0	11301.3	**11306.1**	5204	46	**13**
19	d10-mo100	100	100	788	1787	3376.7	3411.3	**3414.9**	435	26	**13**
20	d10-mq100	100	100	778	1777	9179.2	9216.7	**9223.8**	581	28	**13**
21	d10-ms100	100	100	783	1782	11049.0	11093.1	**11096.8**	603	49	**17**
22	d15-mo100	100	100	1000	5000	-	3330.7	**3335.0**	7200	80	**29**
23	d15-mq100	100	100	1000	5000	-	9183.3	**9192.4**	7200	104	**34**
24	d15-ms100	100	100	1000	5000	-	11358.2	**11362.1**	7200	102	**19**
25	c10-mo200	200	200	433	933	7116.2	7180.3	**7184.8**	353	**47**	50
26	c10-mq200	200	200	428	928	19270.3	19326.2	**19332.2**	579	**38**	47
27	c10-ms200	200	200	431	931	25190.6	25254.2	**25257.1**	1040	178	**71**
28	c15-mo200	200	200	500	2500	-	7169.8	**7173.4**	7200	137	**59**
29	c15-mq200	200	200	500	2500	-	19220.9	**19227.9**	7200	149	**61**
30	c15-ms200	200	200	500	2500	-	24717.7	**24720.2**	7200	238	**60**
31	d10-mo200	200	200	816	1815	7194.1	7249.0	**7251.6**	3273	127	**94**
32	d10-mq200	200	200	814	1813	18789.0	18866.7	**18872.6**	3791	229	**131**
33	d10-ms200	200	200	806	1805	24509.6	24567.1	**24571.2**	6624	192	**70**
34	d15-mo200	200	200	1000	5000	-	7201.6	**7206.4**	7200	795	**231**
35	d15-mq200	200	200	1000	5000	-	19528.9	**19536.4**	7200	469	**240**
36	d15-ms200	200	200	1000	5000	-	24085.2	**24088.5**	7200	429	**124**

We conclude that the lower bounds obtained by dCut and dBCP are better than those of $dMCF_f$ for all test instances. With respect to primal solution quality, we observe that dCut only found the trivial upper bound given by connecting none of the customers in 18 and dBCP in two out of 60 test instances. In these instances the upper bounds due to $dMCF_f$, which failed to find any primal solution for four instances and additionally found the trivial solution only for another eight instances, are eventually better than or equal to those of dCut and dBCP, respectively. For all other instances, the upper bounds and resulting optimality gaps of dCut and dBCP are better than those of $dMCF_f$. Thus, both dCut and dBCP significantly outperform $dMCF_f$.

Model dBCP solved 44 out of 60 test instances to proven optimality, while dCut could only solve 14 instances. For the remaining instances, the resulting optimality gap of dBCP exceeded 0.01% for only three instances. Thus we conclude that, next to

Table 2 Comparison of LP relaxation values and corresponding CPU-times in seconds for $dMCF_f$, dCut, and dBCP (time limit 7200s) on instances with $|F| \neq |C|$. Best values are marked bold.

Instance						LP value			CPU time [s]		
Nr	Name	$\|F\|$	$\|C\|$	$\|V\|$	$\|E\|$	$dMCF_f^{LP}$	$dCut^{LP}$	$dBCP^{LP}$	$dMCF_f^{LP}$	$dCut^{LP}$	$dBCP^{LP}$
37	c10-mo	75	200	404	904	8153.5	8206.0	**8209.6**	713	72	**39**
38	c10-mp	75	200	403	903	14917.4	14969.5	**14972.1**	228	**10**	17
39	c10-mq	75	200	403	903	20717.2	20786.4	**20789.4**	328	**16**	22
40	c15-mo	75	200	500	2500	-	7971.9	**7975.6**	7200	42	**25**
41	c15-mp	75	200	500	2500	14493.1	14526.4	**14529.2**	5533	45	**30**
42	c15-mq	75	200	500	2500	21570.7	21611.9	**21615.1**	3574	40	**33**
43	d10-mo	75	200	775	1775	8228.0	8293.9	**8296.9**	2166	55	**49**
44	d10-mp	75	200	775	1774	14836.9	14909.7	**14911.2**	2265	45	**40**
45	d10-mq	75	200	774	1773	20834.2	20893.6	**20896.3**	1001	31	**27**
46	d15-mo	75	200	1000	5000	-	8179.8	**8184.1**	7200	221	**93**
47	d15-mp	75	200	1000	5000	-	14771.5	**14775.3**	7200	134	**54**
48	d15-mq	75	200	1000	5000	-	21459.0	**21461.7**	7200	189	**66**
49	c10-mo	200	75	435	935	2957.0	2981.7	**2984.5**	6229	285	**111**
50	c10-mp	200	75	428	928	5444.6	5480.4	**5483.7**	3439	78	**28**
51	c10-mq	200	75	430	930	8093.5	8124.2	**8129.2**	1930	37	**10**
52	c15-mo	200	75	500	2500	-	2962.3	**2965.8**	7200	67	**26**
53	c15-mp	200	75	500	2500	-	5171.1	**5174.8**	7200	243	**37**
54	c15-mq	200	75	500	2500	-	7683.2	**7689.8**	7200	62	**11**
55	d10-mo	200	75	811	1810	-	3069.6	**3073.0**	7200	421	**276**
56	d10-mp	200	75	809	1808	5377.7	5407.7	**5410.5**	5608	39	**16**
57	d10-mq	200	75	820	1819	7698.7	7735.8	**7740.0**	3620	166	**49**
58	d15-mo	200	75	1000	5000	-	2978.9	**2982.6**	7200	727	**384**
59	d15-mp	200	75	1000	5000	-	5415.1	**5419.7**	7200	748	**383**
60	d15-mq	200	75	1000	5000	-	7590.8	**7594.3**	7200	187	**27**

its theoretical strength and tight lower bounds, dBCP allows for deriving high quality primal solutions relatively easily and significantly outperforms all other considered models. Furthermore, one can observe that the instances with $|F| = 200$ and $|C| = 50$ – i.e. instances 49–60 – seem to be particularly hard. While dBCP is able to provide reasonable results on most of them, $dMCF_f$ and dCut often fail to compute meaningful primal solutions already for those instances where the underlying STP instance is relatively small. Finally, we need to mention that due to numerical issues (differences between the used solvers) the obtained optimal solution values of dCut and dBCP slightly differ for three instances in the last shown digit (instances 30, 42, and 43). Furthermore, solving dBCP for instance 58 has been interrupted since the memory limit was reached.

In the following, the performance of dBCP and LDV – the Lagrangian relaxation / very large scale neighborhood search hybrid from [25] – which showed to outperform the other Lagrangian methods will be compared. Relative upper bounds and runtimes of dBCP and LDV are given in Tables 5 and 6, respectively. Here, instances are grouped by the size of the underlying SSCFLP instance in Table 5 and by the size of the original STP instance in Table 6.

Since dBCP successfully solved the majority of instances to proven optimality it dominates LDV with respect to obtained upper bounds. Thus, the gaps due to LDV are usually larger than those of dBCP, but exceeded 4.4% only for three instances with $|F| = 200$ and $|C| = 75$, which seem to be particularly hard and are smaller than or equal to 2% for 70% of all tested instances. When the instances get larger, dBCP often

Table 3 Comparison of solution values and corresponding CPU-times in seconds for $dMCF_f$, dCut, and dBCP (time limit 7200s) on instances with $|F| = |C|$. Best values are marked bold.

	lower bound			upper bound			gap in %			CPU time [s]	
Nr	$dMCF_f$	dCut	dBCP	$dMCF_f$	dCut	dBCP	$dMCF_f$	dCut	dBCP	dCut	dBCP
1	2880.1	2915.8	**2919.6**	2944.2	2923.7	**2919.6**	2.227	0.268	**0.000**	7200	**501**
2	7105.2	7126.0	**7128.8**	7171.3	7129.3	**7128.9**	0.930	0.047	**0.002**	7200	7200
3	9509.5	**9536.9**	**9536.9**	9578.1	**9536.9**	**9536.9**	0.721	**0.000**	**0.000**	35	**5**
4	2748.7	2767.7	**2771.2**	2833.3	2788.6	**2771.2**	3.076	0.755	**0.000**	7200	**1035**
5	7469.7	7493.8	**7495.9**	7966.2	7497.8	**7495.9**	6.646	0.053	**0.000**	7200	**195**
6	9357.7	**9373.1**	**9373.1**	10918.9	**9373.1**	**9373.1**	16.684	**0.000**	**0.000**	1627	**38**
7	2776.3	2804.9	**2807.0**	2842.2	2815.6	**2807.0**	2.374	0.382	**0.000**	7200	**107**
8	7299.7	7334.0	**7338.9**	7373.0	7351.6	**7338.9**	1.004	0.239	**0.000**	7200	**243**
9	10073.9	10115.0	**10118.0**	10233.6	10144.7	**10118.0**	1.585	0.293	**0.000**	7200	**277**
10	2645.0	2664.4	**2666.1**	3397.2	8496.0	**2666.2**	28.439	218.868	**0.002**	7200	7200
11	7380.2	**7401.7**	**7401.7**	8528.6	**7401.7**	**7401.7**	15.561	**0.000**	**0.000**	**164**	310
12	9237.4	9258.6	**9259.0**	11007.6	9355.5	**9259.0**	19.164	1.048	**0.000**	7200	**284**
13	3333.0	3364.9	**3367.0**	3380.0	3371.6	**3367.0**	1.409	0.199	**0.000**	7200	**135**
14	9359.0	**9405.0**	9404.9	9473.7	**9405.0**	**9405.0**	1.226	**0.000**	0.002	**3829**	7200
15	11746.0	**11789.3**	**11789.3**	11855.1	**11789.3**	**11789.3**	0.929	**0.000**	**0.000**	383	**13**
16	3426.5	3453.9	**3455.3**	3933.8	3467.8	**3455.3**	14.803	0.403	**0.000**	7200	**98**
17	9125.1	**9149.3**	9149.2	9739.6	**9149.4**	**9149.4**	6.735	**0.000**	0.002	**940**	7200
18	11281.4	11307.3	**11308.5**	12722.2	11308.8	**11308.5**	12.771	0.013	**0.000**	7200	**243**
19	3380.7	3415.5	**3417.3**	3483.2	3418.3	**3417.3**	3.031	0.083	**0.000**	7200	**250**
20	9185.4	9225.0	**9226.2**	9258.9	9226.4	**9226.2**	0.800	0.015	**0.000**	7200	**119**
21	11055.0	**11098.1**	**11098.1**	11197.6	**11098.1**	**11098.1**	1.290	**0.000**	**0.000**	533	**34**
22	3314.0	3333.6	**3336.0**	3862.1	3338.6	**3336.2**	16.537	0.149	**0.004**	7200	7200
23	-	9191.8	**9194.4**	23780.0	23780.0	**9194.4**	-	158.708	**0.000**	7200	**889**
24	11332.4	11362.1	**11363.9**	12715.9	11391.5	**11363.9**	12.208	0.259	**0.000**	7200	**414**
25	7123.0	7184.7	**7185.2**	7329.4	7208.3	**7185.2**	2.898	0.328	**0.000**	7200	**66**
26	19279.8	19332.4	**19335.1**	19539.8	19340.7	**19335.1**	1.349	0.043	**0.000**	7200	**854**
27	25197.3	25256.6	**25258.9**	25327.2	77024.0	**25258.9**	0.516	204.966	**0.000**	7200	**647**
28	7139.0	7173.4	**7174.3**	8383.9	7196.4	**7174.3**	17.437	0.321	**0.000**	7200	**1880**
29	19191.4	19227.6	**19229.4**	21455.8	56699.0	**19229.4**	11.799	194.884	**0.000**	7200	**378**
30	24683.6	24720.7	**24721.1**	26764.0	**24721.0**	**24721.1**	8.428	0.001	**0.000**	7200	**563**
31	7197.4	7251.5	**7252.6**	8021.9	7263.5	**7252.6**	11.455	0.165	**0.000**	7200	**941**
32	18796.9	18870.8	**18875.5**	21247.5	18916.3	**18875.5**	13.037	0.241	**0.000**	7200	**4515**
33	24517.3	**24571.6**	**24571.6**	27880.1	**24571.6**	**24571.6**	13.716	**0.000**	**0.000**	6209	**141**
34	-	7206.0	**7207.2**	-	23975.0	**7207.2**	-	232.710	**0.000**	7200	**5606**
35	-	19535.2	**19537.5**	-	19544.5	**19537.7**	-	0.048	**0.001**	7200	7200
36	-	24088.3	**24090.7**	73434.0	73434.0	**24090.7**	-	204.853	**0.000**	7200	**3742**

needed longer than LDV and completely failed to compute meaningful solutions for two out of 60 instances.

Thus, while dBCP has the potential to compute superior solutions, LDV can be regarded as the more stable and – if the instances get more difficult – also faster approach. Overall, dBCP can be recommended for small to medium sized instances when enough runtime is allowed, while LDV should be used to approximately solve even larger instances or when keeping the runtime small is more important than reducing the optimality gap by a few percent.

9 Conclusions and Outlook

In this article, we considered a generalized variant of the rooted connected facility location problem with capacity constraints and customer prizes where only the most profitable client subset shall be supplied.

Two new mixed integer programming models for solving CConFL to proven optimality have been presented. The first model involves an exponential number of so-called

Table 4 Comparison of solution values and corresponding CPU-times in seconds for $dMCF_f$, dCut, and dBCP (time limit 7200s) on instances with $|F| \neq |C|$. Best values are marked bold.

	lower bound			upper bound			gap in %			CPU time [s]	
Nr	$dMCF_f$	dCut	dBCP	$dMCF_f$	dCut	dBCP	$dMCF_f$	dCut	dBCP	dCut	dBCP
37	8158.2	8209.2	**8212.0**	9181.3	8286.8	**8212.2**	12.541	0.945	**0.002**	7200	7200
38	14924.5	**14973.3**	**14973.3**	15056.9	**14973.3**	**14973.3**	0.887	**0.000**	**0.000**	1624	**96**
39	20725.5	**20791.3**	**20791.4**	20915.4	**20791.4**	**20791.4**	0.916	**0.000**	**0.000**	4017	**277**
40	7948.0	7975.7	**7976.4**	9634.2	7989.4	**7976.8**	21.215	0.173	**0.005**	7200	7200
41	14497.5	14529.1	**14529.8**	15722.6	14557.0	**14530.1**	8.450	0.192	**0.002**	7200	7200
42	21576.2	**21615.3**	**21615.4**	22973.5	**21615.3**	**21615.4**	6.476	**0.000**	**0.000**	1536	**42**
43	8234.7	**8297.4**	**8297.5**	8511.6	**8297.4**	**8297.5**	3.362	**0.000**	**0.000**	1375	**1238**
44	14842.5	14911.1	**14912.6**	15075.2	38988.0	**14912.6**	1.568	161.470	**0.000**	7200	**324**
45	20839.4	**20896.9**	**20896.9**	21044.1	**20896.9**	**20896.9**	0.982	**0.000**	**0.000**	1811	**82**
46	-	8183.5	**8185.2**	20610.0	20610.0	**8185.2**	-	151.847	**0.000**	7200	**2838**
47	14731.9	14774.9	**14776.2**	15760.3	41720.0	**14776.4**	6.981	182.371	**0.001**	7200	7200
48	-	21461.4	**21462.1**	57923.0	57923.0	**21462.3**	-	169.894	**0.001**	7200	7200
49	2957.0	2982.6	**2989.7**	7209.0	7163.0	**3016.6**	143.794	140.156	**0.898**	7200	7200
50	5448.9	5482.5	**5486.9**	5517.8	13672.0	**5486.9**	1.265	149.376	**0.000**	7200	**320**
51	8098.0	**8130.1**	**8130.1**	8203.5	**8130.1**	**8130.1**	1.304	**0.000**	**0.000**	520	**35**
52	2948.5	2965.1	**2969.6**	7189.0	3047.3	**2969.6**	143.817	2.773	**0.000**	7200	**1453**
53	5150.7	5174.2	**5179.2**	12693.0	12693.0	**5179.4**	146.433	145.311	**0.005**	7200	7200
54	7667.3	7690.1	**7693.2**	10212.3	7719.7	**7693.2**	33.193	0.386	**0.000**	7200	**443**
55	3040.5	3070.5	**3076.5**	7500.0	**7405.0**	**7405.0**	146.667	141.168	**140.699**	7200	7200
56	5381.4	5409.9	**5414.6**	5598.2	5497.5	**5414.6**	4.029	1.620	**0.000**	7200	**874**
57	7702.2	7738.8	**7743.7**	10753.2	21242.0	**7743.9**	39.612	174.489	**0.003**	7200	7200
58	-	2980.7	**2983.5**	-	7226.0	**7185.6**	-	142.423	**140.844**	7200	**6446**
59	-	5418.6	**5423.0**	13849.0	13849.0	**5423.0**	-	155.583	**0.000**	7200	**4603**
60	-	7593.9	**7598.9**	-	18640.0	**7598.9**	-	145.461	**0.000**	7200	**3795**

Table 5 Relative solution values and corresponding CPU-times for dBCP and LDV grouped by SSCFLP instance size (12 instances per set).

Instance Set			relative upper bound $\frac{LDV-dBCP}{dBCP}$ in %			relative CPU-time $\frac{LDV}{dBCP}$		
Nr.	$\|F\|$	$\|C\|$	minimum	median	maximum	minimum	median	maximum
1–12	75	75	0.12	0.53	1.25	0.01	0.65	35.65
13–24	100	100	0.38	0.65	1.28	0.02	1.30	19.29
25–36	200	200	0.19	0.52	1.69	0.69	6.77	33.32
37–48	75	200	0.22	0.54	1.57	0.14	0.97	52.12
49–60	200	75	-54.96	1.21	6.31	0.05	0.13	13.09

Table 6 Relative solution values and corresponding CPU-times for dBCP and LDV grouped by STP instance size (15 instances per set).

Instance Set			relative upper bound $\frac{LDV-dBCP}{dBCP}$ in %			relative CPU-time $\frac{LDV}{dBCP}$		
Name	$\|V^o\|$	$\|E^o\|$	minimum	median	maximum	minimum	median	maximum
c10-*	500	1000	0.18	0.58	1.97	0.01	5.31	35.65
c15-*	500	2500	0.12	0.62	6.31	0.02	0.9	52.12
d10-*	1000	2000	-48.34	0.54	3.32	0.05	1.15	33.32
d15-*	1000	5000	-54.96	0.59	1.69	0.04	0.47	1.62

connectivity constraints and can efficiently be solved by branch-and-cut. Furthermore, an alternative model incorporating an exponential number of constraints and variables has been proposed and its solution by branch-and-cut-and-price has been discussed in detail. A polyhedral comparison showed that this model dominates the former, while both new models are theoretically stronger than so far existing ones.

Computational results show that the branch-and-cut-and-price approach based on the theoretically strongest exact model significantly outperforms all other existing integer programming approaches. It could solve the majority of test instances to proven optimality relatively fast, and the resulting optimality gaps are usually extremely small in case the computation is aborted due to the given time limit. The branch-and-cut-and-price approach has further been compared to a previously proposed hybrid Lagrangian decomposition approach involving VLSN search. Here, the obtained computational results indicate clear advantages for the branch-and-cut-and-price approach for small and medium sized instances.

Interesting areas for further research include the development of pure (meta-) heuristic methods for CConFL. Such metaheuristics can be used to better tackle very large scale instances and might include the methods presented in this article for solving smaller subproblems. It might also be possible to further strengthen the proposed exact models by considering additional cutting planes, e.g. from the multiple knapsack polytope [11, 12].

References

1. Achterberg, T.: Constraint Integer Programming. Ph.D. thesis, Technische Universität Berlin (2007)
2. Ahuja, R.K., Orlin, J.B., Pallottino, S., Scaparra, M.P., Scutella, M.G.: A multi-exchange heuristic for the single-source capacitated facility location problem. Management Science **50**(6), 749–760 (2004)
3. Bardossy, M.G., Raghavan, S.: Dual-based local search for the connected facility location and related problems. Tech. rep., Smith School of Business and Institute for Systems Research, University of Maryland (2009)
4. Barnhart, C., Johnson, E.L., Nemhauser, G.L., Savelsbergh, M.W.P., Vance, P.H.: Branch-and-price: Column generation for solving huge integer programs. Operations Research **46**(3), 316–329 (1998). DOI http://dx.doi.org/10.1287/opre.46.3.316
5. Beasley, J.E.: An algorithm for the Steiner problem in graphs. Networks **14**(1), 147–159 (1984)
6. Cherkassky, B.V., Goldberg, A.V.: On implementing the push-relabel method for the maximum flow problem. Algorithmica **19**(4), 290–410 (1997)
7. Cornuejols, G., Nemhauser, G.L., Wolsey, L.A.: The uncapacitated facility location problem. In: P.B. Mirchandani, R.L. Francis (eds.) Discrete Location Theory, pp. 119–171. Wiley (1990)
8. Desaulniers, G., Desrosiers, J., Solomon, M.M. (eds.): Column Generation. Springer (2005)
9. Eisenbrand, F., Grandoni, F., Rothvoß, T., Schäfer, G.: Approximating connected facility location problems via random facility sampling and core detouring. In: ACM-SIAM Symposium on Discrete Algorithms, pp. 1174–1183 (2008)
10. Elias, P., Feinstein, A., Shannon, C.: A note on the maximum flow through a network. IRE Transactions on Information Theory **2**(4), 117–119 (1956)
11. Ferreira, C.E., Martin, A., Weismantel, R.: Facets for the multiple knapsack polytope. Tech. Rep. SC 93-04, Konrad-Zuse Zentrum für Informationstechnik (1993)
12. Ferreira, C.E., Martin, A., Weismantel, R.: Solving multiple knapsack problems by cutting planes. SIAM Journal on Optimization **6**, 858–877 (1996)
13. Gollowitzer, S., Ljubić, I.: MIP models for connected facility location: A theoretical and computational study. Tech. Rep. 2009–07, University of Vienna (2009)

14. Gupta, A., Kleinberg, J., Kumar, A., Rastogi, R., Yener, B.: Provisioning a virtual private network: a network design problem for multicommodity flow. In: Proceedings of the 33rd annual ACM symposium on theory of computing, pp. 389–398 (2001)
15. Hasan, M.K., Jung, H., Chwa, K.: Approximation algorithms for connected facility location problems. Journal of Combinatorial Optimization **16**(2), 155–172 (2008)
16. IBM: CPLEX 12.1. http://www-01.ibm.com/software/integration/optimization/cplex-optimizer
17. Karger, D.R., Minkoff, M.: Building Steiner trees with incomplete global knowledge. In: Proceedings of the 41st Annual Symposium on Foundations of Computer Science, pp. 613–623. IEEE Computer Society (2000)
18. Karp, R.M.: Reducibility among combinatorial problems. In: E. Miller, J.W. Thatcher (eds.) Complexity of Computer Computations, pp. 85–103. Plenum Press (1972)
19. Khuller, S., Zhu, A.: The general Steiner tree-star problem. Information Processing Letters **84**(4), 215–220 (2002)
20. Kratica, J., Tosic, D., Filipovic, V., Ljubić, I.: Solving the simple plant location problem by genetic algorithm. RAIRO Operations Research **35**, 127–142 (2001)
21. Lee, Y., Chiu, S.Y., Ryan, J.: A branch and cut algorithm for a Steiner tree-star problem. INFORMS Journal on Computing **8**(3), 194–201 (1996)
22. Lee, Y., Lu, L., Qiu, Y., Glover, F.: Strong formulations and cutting planes for designing digital data service networks. Telecommunication Systems **2**(1), 261–274 (1993)
23. Leitner, M., Raidl, G.R.: Variable neighborhood search for a prize collecting capacity constrained connected facility location problem. In: Proceedings of the 2008 International Symposium on Applications and the Internet, pp. 233–236. IEEE Computer Society (2008)
24. Leitner, M., Raidl, G.R.: A Lagrangian decomposition based heuristic for capacitated connected facility location. In: S. Voß, M. Caserta (eds.) Proceedings of the 8th Metaheuristic International Conference (MIC 2009). Hamburg, Germany (2009)
25. Leitner, M., Raidl, G.R.: Combining Lagrangian decomposition with very large scale neighborhoood search for capacitated connected facility location. In: Post-Conference Book of the Eight Metaheuristics International Conference – MIC 2009 (accepted 2010)
26. Ljubić, I.: A hybrid VNS for connected facility location. In: T. Bartz-Beielstein, et al. (eds.) Hybrid Metaheuristics, 4th International Workshop, HM 2007, *LNCS*, vol. 4771, pp. 157–169. Springer (2007)
27. Ljubić, I., Gollowitzer, S.: Hop constrained connected facility location. Tech. Rep. 2009–09, University of Vienna (2009)
28. Ljubić, I., Gollowitzer, S.: Modelling the hop constrained connected facility location problem on layered graphs. In: International Symposium on Combinatorial Optimization (ISCO 2010). Hammamet, Tunisia (2010). To appear
29. Magnanti, T.L., Wolsey, L.A.: Optimal trees. In: M.O. Ball, T.L. Magnanti, C.L. Monma, G.L. Nemhauser (eds.) Handbooks in Operations Research and Management Science, vol. 7, pp. 503–615. Elsevier (1995)
30. Martello, S., Pisinger, D., Toth, P.: Dynamic programming and strong bounds for the 0-1 knapsack problem. Management Science **45**(3), 414–424 (1999)
31. Raghavan, S., Bardossy, M.G.: Dual based heuristics for the connected facility location problem. In: M.G. Scutellà, et al. (eds.) Proceedings of the International Network Optimization Conference 2009 (2009)
32. Swamy, C., Kumar, A.: Primal-dual algorithms for connected facility location problems. Algorithmica **40**(4), 245–269 (2004)
33. Tomazic, A., Ljubić, I.: A GRASP algorithm for the connected facility location problem. In: Proceedings of the 2008 International Symposium on Applications and the Internet, pp. 257–260. IEEE Computer Society (2008)
34. Winter, P.: Steiner problem in networks: a survey. Networks **17**(2), 129–167 (1987)
35. Xu, J., Chiu, S.Y., Glover, F.: Tabu search for dynamic routing communications network design. Telecommunication Systems **8**(1), 55–77 (1997)
36. ZIB: SCIP 1.2.0. http://scip.zib.de