TECHNISCHE UNIVERSITÄT WIEN
Institut für Computergraphik und Algorithmen

Combining Lagrangian Decomposition with Very Large Scale Neighborhood Search for Capacitated Connected Facility Location

Markus Leitner and Günther R. Raidl

Forschungsbericht / Technical Report

TR–186–1–09–02

October 01, 2009

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

Combining Lagrangian Decomposition with Very Large Scale Neighborhood Search for Capacitated Connected Facility Location

Markus Leitner and Günther R. Raidl

Institute of Computer Graphics and Algorithms
Vienna University of Technology
Favoritenstraße 9–11
1040, Vienna, Austria
{leitner|raidl}@ads.tuwien.ac.at

Abstract. We consider a generalized version of the rooted Connected Facility Location problem (ConFL) which occurs when extending existing communication networks in order to increase the available bandwidth for customers. In addition to choosing facilities to open and connecting them by a Steiner tree as in the classic ConFL, we have to select a subset of all potential customers and assign them to open facilities respecting given capacity constraints in order to maximize profit. We present two exact mixed integer programming formulations and a Lagrangian decomposition (LD) based approach which uses the volume algorithm. Feasible solutions are derived using a Lagrangian heuristic. Furthermore, we present two hybrid variants combining LD with local search and a very large scale neighborhood search. By applying those improvement methods only to the most promising solutions, we are able to compute much better solutions without increasing the necessary runtime too much. As documented by our computational results, our hybrid approaches compute high quality solutions with tight optimality gaps in relatively short time.

Key words: Connected facility location, network design, Lagrangian decomposition, very large scale neighborhood search, mixed integer programming.

1 Introduction

We consider a real-world network design problem 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. 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 old network is still used between a customer and its correspondingly assigned mediation point the use of the newly installed high-bandwidth routes in the remaining network results in an increased bandwidth for most customers. Depending on the network used between those mediation points and the customers, those 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 those scenarios can be modeled as variants of the *Connected Facility Location Problem (ConFL)* 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 does neglect 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 supplying not necessarily all but only the most profitable subset of potential customers by additionally considering the expected return of invest for individual customers. As formally described in the following, our model to which we refer as the *rooted Price Collecting Capacitated Connected Facility Location Problem (CConFL)* overcomes those shortages of ConFL.

After formally defining CConFL in Section 2 and discussing previous and related work in Section 3 we present two mixed integer programming (MIP) formulations for solving small instances of CConFL to proven optimality in Section 4. For larger instances, Section 5 describes a new Lagrangian decomposition (LD) approach based on one of those MIP formulations. A Lagrangian heuristic to derive feasible solutions as well as methods for improving those solution in order to obtain tight optimality gaps between the lower and upper bounds within reasonable time are presented in Sections 6 and 7. Test instances and computational results are discussed in Section 8, before drawing conclusions in Section 9.

This article significantly extends our previous work [1] by proposing an additional MIP formulation in Section 4 and a new very large scale neighborhood search procedure in Section 7.3; more computational results are given, and the remaining parts are more detailed.

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. Each edge $e = (u, v) \in E^{\mathrm{o}}$ has associated costs $c_e^{\mathrm{o}} \geq 0$ 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 $D_i \in \mathbb{N}_0$, $\forall i \in F^{\mathrm{o}}$. 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 return of invest 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 0, yielding a reduced graph $G = (V, E)$ with node set $V = (V^{\mathrm{o}} \cup \{0\}) \setminus V_{\mathrm{I}}$ and edge set $E = \{(u,v) \in E^{\mathrm{o}} \mid u, v \notin E_{\mathrm{I}}\} \cup \{(0,v) \mid \exists (u,v) \in E^{\mathrm{o}} : u \in V_{\mathrm{I}} \wedge v \notin V_{\mathrm{I}}\}$; 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 eventually 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 $a_{i,k} = \infty$, since those assignments cannot be part of an optimal solution as they do not pay off. 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. Furthermore, $C_i = \{k \in C \mid a_{i,k} \leq p_k\}$ 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 0 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 concrete mapping between customers and facilities is described by $\alpha_S : C_S \to F_S$. Since we are considering a single source variant of the connected facility location problem, each customer may be assigned to at most one facility. The objective function of CConFL can be stated as

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

An optimal solution S^* (i.e. a most profitable one) is given by the minimal objective value, i.e. $c(S^*) \leq c(S)$ for all feasible solutions S. Since CConFL combines the (Price Collecting) Steiner Tree Problem (STP) on a graph with the Single Source Capacitated Facility Location Problem (SSCFLP) which are both strongly NP-hard [2, 3], CConFL is strongly NP-hard, too.

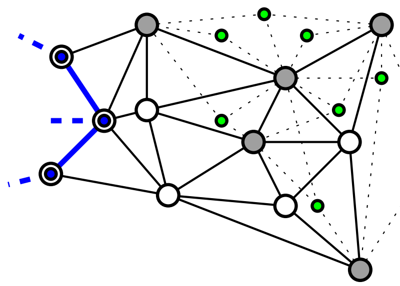


Fig. 1. Original Problem instance.

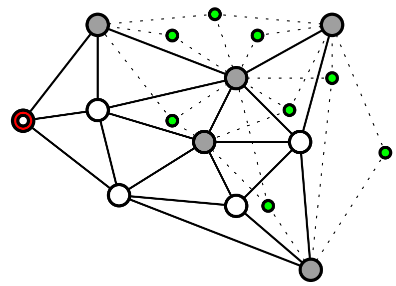


Fig. 2. Rooted Problem instance.

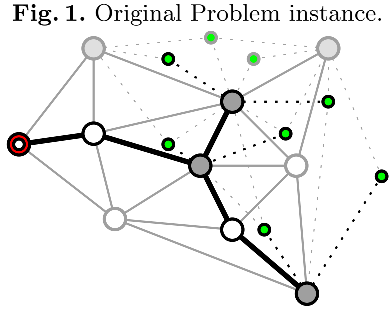
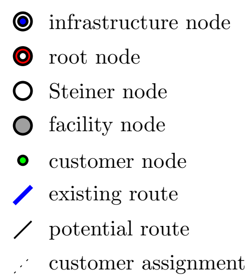


Fig. 3. Exemplary solution.



3 Related Work

Karger and Minkoff [4] considered the *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. [5] in their work on virtual private networks.

Since then several authors proposed approximation algorithms for diverse variants of ConFL. Swamy and Kumar [6] 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 the LP rounding technique, Hasan et al. [7] improved their method to a factor 8.29 approximation algorithm for the case of edge costs obeying the triangle inequality and a factor 7 approximation in case all opening costs are equal. Recently, a randomized approximation algorithm with an expected approximation ratio of 4, which can be derandomized with a resulting approximation factor of 4.23, has been presented by Eisenbrand et al. [8].

Ljubić [9] 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 [10] considered the unrooted version of ConFL and presented a greedy randomized adaptive search procedure. Furthermore, they transformed the problem to the minimum Steiner arborescence problem and solved it by an exact branch-and-cut method. Bardossy and Raghavan [11] combined dual ascent with local search to derive lower and upper bounds for ConFL. The current authors presented in [12] two VNS variants for a version of CConFL without assignment and opening costs. To the best of our knowledge our concrete variant of the connected facility location problem, which contains most of the previously discussed problem variants as special cases, has not been considered so far.

Other related problems are the Steiner tree star problem, where opening costs for facilities included in the Steiner tree must be paid even if no customers are assigned to them, as well as its generalized version [13], where customer nodes and potential facilities are not necessarily disjoint.

Furthermore, literature on the (price collecting) Steiner tree problem on graphs (STP), as well as the (single source) capacitated facility location problem (SSCFLP) can be considered as relevant, since CConFL is composed from these two problems, see e.g. [14] for a survey on the STP and [15] for a recent work on the SSCFLP with a comprehensive list of further references on that topic.

4 Multi-Commodity Flow Formulations

CConFL can be modeled as a mixed integer program (MIP) based on directed multi-commodity flows in two rather obvious ways. While our first model $dMCF_f$ presented in Section 4.1 is based on sending one unit of flow to each potential facility location, model $dMCF_c$ presented in Section 4.2 sends flow to each potential customer.

For an easier presentation we define an extended graph $G' = (V', E')$ combining G with the set of potential customers C as additional nodes and potential assignments between facilities and customers as additional edges (*assignment edges*). Formally, G' is given by its node set $V' = V \cup C$ and its edge set $E' = E \cup \{(i,j) \mid i \in F \wedge j \in C_i\}$. Edge costs $c'_e \geq 0$ are defined by

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

4.1 Facility oriented model

Let $A_0 = \{(0,v) \mid (0,v) \in E\}$ denote the set of directed edges, i.e. arcs, going out from the root node 0 and $A'_i = \{(u,v),(v,u) \mid (u,v) \in E \wedge u,v \notin \{0,i\}\}$, $\forall i \in F$, the set containing two oppositely directed arcs for each pair of nodes $u,v \in V \setminus \{0,i\}$ that are connected by an edge in G. Let $A_i^- = \{(v,i) \mid (v,i) \in E\}$ be the set of ingoing arcs for each facility $i \in F$. We can now define the set of arcs relevant for connecting a facility $i \in F$ to the root node as $A_i = A_0 \cup A'_i \cup A_i^-$. In model $dMCF_f$ (2)–(11) we use 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. 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 ($z_i = 1$).

$$
\begin{aligned}
(dMCF_f)\ \min\ & \sum_{e \in E'} c'_e x_e + \sum_{i \in F} f_i z_i + \sum_{k \in C} p_k (1 - y_k) && (2)\\
\text{s.t.}\ & \sum_{(u,v) \in A_i} s^i_{u,v} - \sum_{(v,u) \in A_i} s^i_{v,u} = \begin{cases} -z_i & \text{if } v = 0 \\ z_i & \text{if } v = i \\ 0 & \text{otherwise} \end{cases} && \forall i \in F,\ \forall v \in V \quad (3)\\
& s^i_{u,v} + s^i_{v,u} \le x_{u,v} && \forall i \in F,\ \forall (u,v) \in E \quad (4)\\
& x_{i,k} \le z_i && \forall (i,k) \in E' \mid k \in C \quad (5)\\
& \sum_{k \in C_i} d_k x_{i,k} \le D_i z_i && \forall i \in F \quad (6)\\
& \sum_{i \in F_k} x_{i,k} \ge y_k && \forall k \in C \quad (7)\\
& 0 \le s^i_{u,v} \le 1 && \forall i \in F,\ \forall (u,v) \in A_i \quad (8)\\
& 0 \le z_i \le 1 && \forall i \in F \quad (9)\\
& x_e \in \{0,1\} && \forall e \in E' \quad (10)\\
& y_k \in \{0,1\} && \forall k \in C \quad (11)
\end{aligned}
$$

The objective function (2) unifies assignment and edge costs by using the concept of the extended graph G' but otherwise corresponds to function (1). Constraints (3) are the usual flow conservation constraints, inequalities (4) link variables $s^i_{u,v}$ and x_e, and inequalities (5) ensure that a facility is opened if an incident assignment edge is used. Inequalities (6) are the capacity constraints for each facility $i \in F$, while inequalities (7) ensure that a customer's prize can only be earned if the customer is connected to a facility.

4.2 Customer oriented model

Model $dMCF_c$ (12)–(20) sends one unit of flow to each potential customer, but otherwise is similar to model $dMCF_f$. Thus we define the set of relevant arcs $A_k = A_0 \cup A' \cup A^-_k$ for each customer $k \in C$, where A_0 is the set of arcs going out from the root node as defined in Section 4.1, $A' = \{(u,v),(v,u) \mid (u,v) \in E \wedge u,v \neq 0\}$, and $A^-_k = \{(i,k) \mid (i,k) \in E'\}$.

$$(dMCF_c) \min \sum_{e \in E'} c'_e x_e + \sum_{i \in F} f_i z_i + \sum_{k \in C} p_k (1 - y_k) \tag{12}$$

$$\text{s.t.} \sum_{(u,v) \in A_k} s^k_{u,v} - \sum_{(v,u) \in A_k} s^k_{v,u} = \begin{cases} -y_k & \text{if } v = 0 \\ y_k & \text{if } v = k \\ 0 & \text{otherwise} \end{cases} \qquad \forall k \in C,\ \forall v \in E' \tag{13}$$

$$s^k_{u,v} + s^k_{v,u} \le x_{u,v} \qquad \forall (u,v) \in E' \tag{14}$$

$$x_{i,k} \le z_i \qquad \forall i \in F,\ \forall k \in C_i \tag{15}$$

$$\sum_{k \in C_i} d_k x_{i,k} \le D_i z_i \qquad \forall i \in F \tag{16}$$

$$0 \le s^k_{u,v} \le 1 \qquad \forall k \in C,\ \forall (u,v) \in A_k \tag{17}$$

$$0 \le z_i \le 1 \qquad \forall i \in F \tag{18}$$

$$x_e \in \{0,1\} \qquad \forall e \in E' \tag{19}$$

$$y_k \in \{0,1\} \qquad \forall k \in C \tag{20}$$

Here, constraints (13) resemble the flow conservation constraints for each customer $k \in C$ and similarly to $dMCF_f$ inequalities (14) and (15) link variables x with y and x with z, respectively. While the capacity constraints (16) are identical to those of formulation $dMCF_f$, we do not need explicit linking constraints between variables x and y in model $dMCF_c$ since those are implicitly included in the flow conservation constraints.

4.3 Polyhedral Analysis

In the following, we compare the set of feasible fractional solutions of the LP relaxations $dMCF_f^{\mathrm{LP}}$ and $dMCF_c^{\mathrm{LP}}$ of models $dMCF_c$ and $dMCF_f$.

Theorem 1. *None of the formulations $dMCF_c$ and $dMCF_f$ strictly dominates the other, i.e. $dMCF_c^{\mathrm{LP}} \nsubseteq dMCF_f^{\mathrm{LP}}$ and $dMCF_f^{\mathrm{LP}} \nsubseteq dMCF_c^{\mathrm{LP}}$.*

We prove each direction individually.

Lemma 1. *$dMCF_f$ does not dominate $dMCF_c$, i.e. $dMCF_c^{\mathrm{LP}} \nsubseteq dMCF_f^{\mathrm{LP}}$.*

Proof. Consider a fractional solution $S' = (R'_S, T'_S, F'_S, C'_S, \alpha'_S)$ corresponding to the example given in Figure 4. S' can be feasibly described in the LP relaxation of our facility oriented model using the variable values as indicated in the figure, i.e. $S' \in dMCF_f^{\mathrm{LP}}$. Here, the corresponding flow to each facility with value $\frac{1}{3}$ is routed over two disjoint paths. However $S' \notin dMCF_c^{\mathrm{LP}}$ since each flow to customer $k \in C'_S$ must be rooted over arcs going out from the root node 0, i.e. $\sum_{(0,u) \in A_k} s^k_{0,u} \le y_k$. Since $y_k = 1$, $\forall k \in \{1,2,3\}$, in S' but $\sum_{(0,u) \in A_k} s^k_{0,u} = \frac{1}{3}$, $S' \notin dMCF_c^{\mathrm{LP}}$.

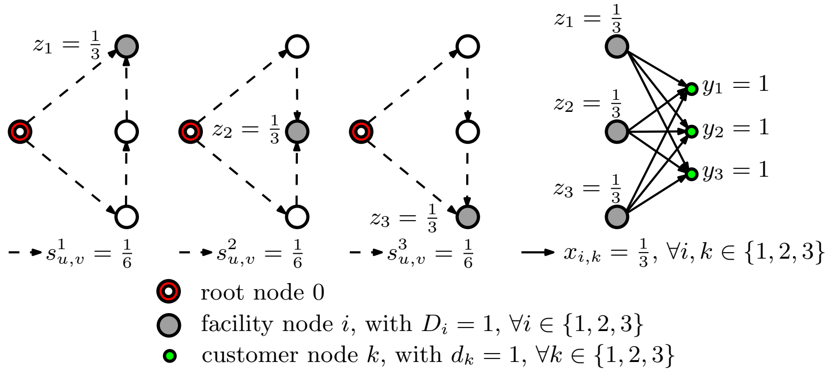


Fig. 4. Feasible LP solution of $dMCF_f$ which is infeasible for $dMCF_c$.

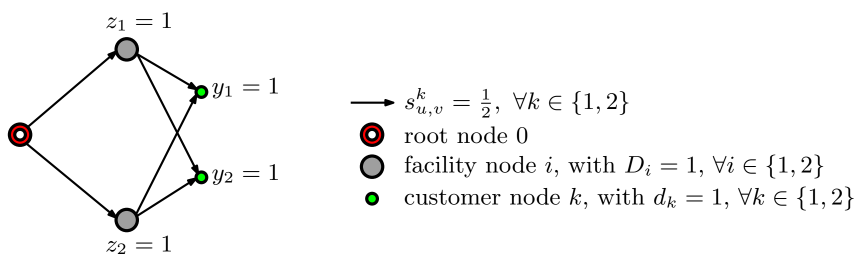


Fig. 5. Feasible LP solution of $dMCF_c$ which is infeasible for $dMCF_f$.

Lemma 2. *$dMCF_c$ does not dominate $dMCF_f$, i.e. $dMCF_f^{\mathrm{LP}} \nsubseteq dMCF_c^{\mathrm{LP}}$.*

Proof. Here, we consider a fractional solution $S'' = (R''_S, T''_S, F''_S, C''_S, \alpha''_S)$ corresponding to Figure 5. Since the capacity constraints as well as all linking constraints are met and the corresponding flow to each of the two customer is routed over two disjoint paths, where each fractional value $s^k_{u,v}$ is set to $\frac{1}{2}$, $S'' \in dMCF_c^{\mathrm{LP}}$. For feasible solutions of model $dMCF_f^{\mathrm{LP}}$, $\sum_{(u,i)\in A_i} s^i_{u,i} \leq z_i$ must hold due to the flow conservation constraints. Since $\sum_{(u,i)\in A_i} s^i_{u,i} = \frac{1}{2}$ but $z_i = 1$ we conclude that $S'' \notin dMCF_f^{\mathrm{LP}}$.

Theorem 1 immediately follows due to Lemmas 1 and 2.

5 Lagrangian Decomposition

Since Lagrangian relaxation based approaches have proven to be quite successful for the Steiner tree problem [16] as well as for the Capacitated Facility location problem [17] and CConFL is composed of these two problems it is quite natural to decompose CConFL by means of Lagrangian relaxation. Model (21)–(29) which we will relax in the following is a more abstractly written, undirected variant of model $dMCF_c$. As previously, binary variables x_e, $\forall e \in E'$, indicate if an edge e is part of the solution, variables $z_i \in [0,1]$, $\forall i \in F$, specify if a facility i is opened and variables y_k, $\forall k \in C$, if a customer k is feasibly assigned to an open facility. Similarly to the flow variables of model $dMCF_c$, we use variables $s_e^k \in \{0,1\}$, $\forall k \in C,\ \forall e \in E'$, to indicate if an edge $e \in E'$ is part of the unique path from the root node 0 to a connected customer k. Finally $P_k \in \{0,1\}^{|E'|}$ denotes the set of incidence vectors corresponding to those simple paths from 0 to $k \in C$ using exactly one assignment edge $(i,k) \in E' \setminus E$.

$$
\begin{aligned}
\min \quad & \sum_{e \in E'} c'_e x_e + \sum_{i \in F} f_i z_i + \sum_{k \in C} p_k (1 - y_k) && && (21)\\
\text{s.t.} \quad & s_e^k \le x_e && \forall k \in C,\ \forall e \in E' && (22)\\
& s^k \in P_k \text{ if } y_k = 1 && \forall k \in C && (23)\\
& x_{i,k} \le z_i && \forall i \in F,\ \forall k \in C_i && (24)\\
& \sum_{k \in C_i} d_k x_{i,k} \le D_i z_i && \forall i \in F && (25)\\
& s_e^k \in \{0,1\} && \forall k \in C,\ \forall e \in E' && (26)\\
& x_e \in \{0,1\} && \forall e \in E' && (27)\\
& z_i \in \{0,1\} && \forall i \in F && (28)\\
& y_k \in \{0,1\} && \forall k \in C && (29)
\end{aligned}
$$

We relax inequalities (22) linking variables s and x in a classical Lagrangian fashion by adding corresponding terms weighted with nonnegative Lagrangian multipliers $\pi_{k,e} \ge 0$, $\forall k \in C,\ \forall e \in E'$, to the objective function. This yields the parameterized model $LD(\pi)$. See for example [18] for a general introduction to Lagrangian relaxation.

$$
\begin{aligned}
(LD(\pi)) \min & \sum_{e \in E'} c'_e x_e + \sum_{i \in F} f_i z_i + \sum_{k \in C} p_k (1 - y_k) + \sum_{k \in C} \sum_{e \in E'} \pi_{k,e} \cdot (s_e^k - x_e) = \\
&= \sum_{k \in C} p_k + \sum_{k \in C} \left(\sum_{e \in E'} \pi_{k,e} s_e^k - p_k y_k \right) + \sum_{e \in E'} \left(c'_e - \sum_{k \in C} \pi_{k,e} \right) x_e + \sum_{i \in F} f_i z_i \\
\text{s.t.} \quad & (23)\text{–}(29)
\end{aligned}
$$

$LD(\pi)$ decomposes into independent subproblems $LD_{s,y}(\pi)$ for determining variables s_e^k, $\forall k \in C,\ \forall e \in E'$ and y_k, $\forall k \in C$, subproblem $LD_x(\pi)$ for determining variables x_e, $\forall e \in E$, and subproblem $LD_{x,z}(\pi)$ to determine variables x_e, $\forall e \in E' \setminus E$, and z_i, $\forall i \in F$. We consider these subproblems and their solving in the following in detail.

$$(LD_{s,y}(\pi)) \quad \min \quad \sum_{k \in C} p_k + \sum_{k \in C} \left(\sum_{e \in E'} \pi_{k,e} s^k_e - p_k y_k \right) \tag{30}$$

$$\text{s.t.} \quad s^k \in P_k \text{ if } y_k = 1 \qquad \forall k \in C \tag{31}$$

$$s^k_e \in \{0,1\} \qquad \forall k \in C,\ \forall e \in E' \tag{32}$$

$$y_k \in \{0,1\} \qquad \forall k \in C \tag{33}$$

$LD_{s,y}(\pi)$ consists of $|C|$ independent cheapest path problems. Thus it can be solved for customer $k \in C$ by computing the cheapest path w.r.t. edge costs $\pi_{k,e}$ from the root to customer node k which includes exactly one assignment edge $(i,k) \in E' \setminus E$, i.e. we need to determine the corresponding incidence vector $q \in P_k$. If the total costs of this path are smaller than the customers prize p_k, y_k as well as the corresponding path variables s^k_e, $\forall e \in E' \mid q_e = 1$, are set to one. Since, all edge costs $\pi_{k,e}$ are nonnegative we use $|C|$ runs of Dijkstras' algorithm [19], resulting in a total time-complexity of $O(|C|(|E| + |V|) \log |V|)$ for solving $LD_{s,y}(\pi)$ when using the binary heap implementation of Dijkstras' algorithm.

$$(LD_x(\pi)) \quad \min \quad \sum_{e \in E} \left(c_e - \sum_{k \in C} \pi_{k,e} \right) x_e \tag{34}$$

$$\text{s.t.} \quad x_e \in \{0,1\} \qquad \forall e \in E \tag{35}$$

$LD_x(\pi)$, can be trivially solved by inspection in time $O(|C||E|)$. Variables x_e, $\forall e \in E$, are set to one if $c_e < \sum_{k \in C} \pi_{k,e}$, and to zero otherwise.

$$(LD_{x,z}(\pi)) \quad \min \quad \sum_{i \in F} f_i z_i + \sum_{\substack{e=(i,k) \in E' \mid \\ i \in F \wedge k \in C_i}} \left(c'_{i,k} - \sum_{k \in C} \pi_{k,e} \right) x_{i,k} \tag{36}$$

$$\text{s.t.} \quad \sum_{k \in C_i} d_k x_{i,k} \le D_i z_i \qquad \forall i \in F \tag{37}$$

$$x_{i,k} \le z_i \qquad \forall i \in F,\ \forall k \in C_i \tag{38}$$

$$z_i \in \{0,1\} \qquad \forall i \in F \tag{39}$$

$$x_e \in \{0,1\} \qquad \forall e \in E' \setminus E \tag{40}$$

Model $(LD_{x,z}(\pi))$ resembles $|F|$ 0–1 knapsack problems, one for each facility $i \in F$. In such a knapsack problem for facility $i \in F$, we are given the total knapsack capacity D_i, and one item for each potential assignment $e = (i,k) \in E' \setminus E$, with profit $\sum_{k \in C} \pi_{k,e} - c'_e$ and weight d_k. Obviously, we can neglect all items with negative or zero profit. Let χ^*_i denote the optimal solution to the knapsack problem of facility $i \in F$, and $o(\chi^*_i)$ the according objective value (i.e.

the total profit). z_i and all variables x_e corresponding to items used in χ_i^* are set to one if $o(\chi_i^*) > f_i$. Although the knapsack problem is weakly NP-hard [20], several algorithms capable of solving large instances relatively quickly are known. In our implementation we use the Combo algorithm[1] of Martello et al. [21]. Since $LD_{x,z}(\pi)$ does not possess the integrality property, we may be able to determine better lower bounds than by a simpler LP relaxation of model (21)–(29).

In the Lagrangian dual problem, we aim at maximizing the resulting lower bound by determining optimal Lagrangian multipliers π^*. Since this maximization problem is convex and piecewise linear, we can approximately solve it using subgradient-like methods. We use the volume algorithm [22], which is an extension of the classic subgradient method [23], for solving the Lagrangian dual. Preliminary tests in our scenario indicated that it usually yields better lower bounds than the classic method, and it also has been reported to be more efficient in a number of other applications [16, 24].

6 Primal Heuristic

Applying the volume algorithm [22] to approximately solve the Lagrangian dual problem, we compute integer values for variables s_e^k, x_e, z_i, and y_k in each iteration. The solution to $LD_{s,y}(\pi)$ does connect a subset of customers with the root node, however the subgraph induced by those paths might contain redundant edges or violate capacity constraints. On the other hand, the solution to $LD_{x,z}(\pi)$ does open some facilities and assigns customers to them respecting the capacity constraints, but does not take into account whether those facilities are connected to the root node. Furthermore, customers may be assigned to multiple facilities due to $LD_{x,z}(\pi)$.

To create a feasible solution $S = (R_S, T_S, F_S, C_S, \alpha_S)$ using the solutions to $LD_{s,y}(\pi)$ and $LD_{x,z}(\pi)$ we apply the Lagrangian heuristic (LH) presented in Algorithm 1.

Algorithm 1 initially declares all facilities as open whose corresponding nodes are part of a path to some customer $k \in C$ due to the actual solution to $LD_{s,y}(\pi)$, i.e. $F_S = \{i \in F \mid \exists k \in C : s_{i,k}^k = 1\}$.

In a second phase the Steiner tree (R_S, F_S) connecting those facilities $i \in F_S$ is created. Let $W_{i,k} = \{e \in E \mid s_e^k = 1\}$, $\forall k \in C_i'$, with $C_i' = \{k \in C \mid s_{i,k}^k = 1\}$ be the set of customers connected to the root node 0 via facility i, and $W_i = \operatorname{argmin}_{W_{i,k} | k \in C_i'} \{\sum_{e \in W_{i,k}} c_e\}$ be the shortest of those subpaths for each open facility $i \in F_S$. After initializing the Steiner tree to consist of the root node only – i.e. $R_S = \{0\}$, $T_S = \emptyset$ – all facilities $i \in F_S$ are considered in increasing order w.r.t. the costs $\sum_{e \in W_i} c_e$ of the cheapest path W_i connecting them. We connect each considered facility $i \in F$ to the so far constructed Steiner tree by adding the necessary subpath $W' \subseteq W_i$ with $W' = \{(v_0 = i, v_1), (v_1, v_2), \ldots (v_l, v_m)\}$, $(v_a, v_b) \in W_i$, $0 \leq a, b \leq m$, $v_i \notin R_S$, $0 \leq i \leq l$, $v_m \in R_S$, to the Steiner tree, i.e. $R_S = R_S \cup \{v_0, v_1 \ldots, v_l\}$, and $T_S = T_S \cup W'$.

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

Algorithm 1: Primal Heuristic(Solution S', variable values s_e^k, x_e, z_i, y_k)

```
// Phase 1: open facilities
F_S = {i ∈ F | ∃k ∈ C : s^k_{i,k} = 1}
// Phase 2: construct Steiner tree (R_S, T_S) and assign initial customers
R_S = {0}
T_S = ∅
forall i ∈ F_S do
    C'_i = {k ∈ C | s^k_{i,k} = 1}
    W_{i,k} = {e ∈ E | s^k_e = 1}, ∀k ∈ C'_i
    W_i = argmin_{W_{i,k} | k ∈ C'_i} {∑_{e ∈ W_{i,k}} c_e}
forall i ∈ F_S in increasing order of ∑_{e ∈ W_i} c_e do
    if ∑_{k ∈ C'_i} d_k ≤ D_i then
        C''_i = C'_i
    else
        determine optimal assignable subset C''_i ⊆ C'_i using Combo algorithm
    C_S = C_S ∪ C''_i
    α_S(k) = i, ∀k ∈ C''_i
// Phase 3: assign additional customers
A = {(i,k) | i ∈ F_S ∧ k ∈ C \ C_S ∧ x_{i,k} = 1}
forall (i,k) ∈ A in decreasing order w.r.t. efficiency (p_k − c'_{i,k}) / d_k do
    if k ∉ C_S ∧ d_k + ∑_{k' ∈ C_S | α_S(k') = i} d_{k'} ≤ D_i then
        C_S = C_S ∪ k
        α_S(k) = i
A' = {(i,k) | i ∈ F_S ∧ k ∈ C \ C_S ∧ x_{i,k} = 0}
forall (i,k) ∈ A' in decreasing order w.r.t. efficiency (p_k − c'_{i,k}) / d_k do
    if k ∉ C_S ∧ d_k + ∑_{k' ∈ C_S | α_S(k') = i} d_{k'} ≤ D_i then
        C_S = C_S ∪ k
        α_S(k) = i
// Phase 4: primal improvement
if c(S) ≤ c(S') then
    S' = S
    Primal Improvement(S) // see Algorithm 2
```

After connecting facility $i \in F_S$ the optimal subset of customers $C''_i \subseteq C'_i$ which are connected by paths via i is assigned to facility i. If assigning all those customers C'_i would exceed the maximum demand D_i assignable to i, we use the Combo algorithm [21] again to solve the corresponding 0–1 knapsack problem, while simply all customers $k \in C'_i$ might be assigned to i if $\sum_{k \in C'_i} d_k \leq D_i$.

In the third phase of Algorithm 1 the so far created solution is further improved by assigning additional customers. Thus we first consider the set of assignments $\mathcal{A}$ between customers and open facilities $i \in F_S$ from the solution to $LD_{x,z}(\pi)$, i.e. $\mathcal{A} = \{(i,k) \mid i \in F_S \wedge k \in C \wedge x_{i,k} = 1\}$, in decreasing order w.r.t.

Algorithm 2: Primal Improvement(Solution S)

```
Key Path Improve(S) // see Algorithm 3
switch improvement mode do
    case simple:
        Customer Swap Improve(S) // see Algorithm 4
    case advanced:
        Very Large Scale Neighborhood Search(S) // see Algorithm 5
prune solution
```

their efficiency values $\frac{p_k - c'_{i,k}}{d_k}$. Each considered assignment (i, k) is added to S if the corresponding customer has not yet been assigned, i.e. $k \notin C_S$, and the facility's capacity constraint will not be exceeded, i.e. $d_k + \sum_{k' \in C_S | \alpha_S(k') = i} d_{k'} \leq D_i$. Subsequently, further assignments are added to S using an identical greedy strategy for all remaining possible assignments to facilities $i \in F_S$.

Finally, we further improve the obtained solution S using the neighborhood structures described in Section 7 in case S is better than the so far best solution S' derived by LH before applying these improvements.

7 Solution Improvement

Representing solutions by means of open facilities and computing the Steiner tree connecting them as well as assigning customers to them during the solution decoding process has been the usual approach taken in metaheuristics for variants of ConFL so far [12, 9, 10]. In our case, modifying the set of open facilities is quite expensive w.r.t. computational time, since determining the optimal connecting Steiner tree as well as assigning the optimal clients are NP-hard problems. Using some heuristic for decoding a solution after adapting the set of open facilities and subsequently trying to improve those aspects is an interesting approach for a pure metaheuristic but is likely to be also too time consuming in case of our intertwined approach in which the primal improvement procedure is repeatedly applied to solutions derived within the course of the volume algorithm.

We therefore decided to concentrate on improving a solution by means of its Steiner tree and its assigned customers, but do not modify the set of open facilities generated by our Lagrangian heuristic. Diversity by means of open facilities is ensured in our approach due to the fact that we generate one initial solution in each iteration of the volume algorithm. As shown by Algorithm 2, we use one neighborhood structure for each of the remaining solution aspects: a path exchange neighborhood – see Section 7.1 – for reducing the costs of the connecting Steiner tree and either a simple swap neighborhood – see Section 7.2 – or a very large scale neighborhood – see Section 7.3 – for improving facility customer assignments. Both neighborhoods are searched using a best improvement strategy. Finally, we remove non-profitable parts from S using strong pruning as described in [25].

It is further worth mentioning that since the improved solution aspects are independent one could easily apply the corresponding neighborhoods in parallel instead of our sequential approach to reduce the total runtime.

7.1 Key Path Improvement

For the Steiner tree problem in graphs, the concept of so called *key nodes* – also called *crucial nodes* – of a solution, which are all customer nodes as well as all Steiner nodes of degree greater than or equal to three is well known. Voß [26] was the first who considered representing a solution to STP by those key nodes – although he did not yet use the term key nodes – and trying to improve it by means of replacing the paths between those key nodes. Since then this type of neighborhood structure has been successfully used in several approaches for the STP – see e.g. [27, 28] – as well as some of its generalizations, e.g. [29].

For a solution S to CConFL the set of key nodes $\mathcal{K} = \{0\} \cup F_S \cup \{v \in R_S \mid \deg_S(v) \geq 3\}$ is given by the root node, all open facilities as well as all Steiner nodes of degree greater than or equal to three in S. A *key path* $(\mathcal{V}, \mathcal{E})$ of solution S is a non-empty path in S between two key nodes $u, v \in \mathcal{K}$ containing no other key node, i.e. $\mathcal{V} \cap \mathcal{K} = \{u, v\}$. Our *Key-Path Improvement* procedure as given in Algorithm 3 considers each such key path $(\mathcal{V}, \mathcal{E}) \in \tilde{P}(S)$ from the set of all key paths $\tilde{P}(S)$ of solution S and replaces it by the shortest connection between its end nodes using the remaining solution edges as infrastructure (i.e. zero edge costs are assumed for them); see Figure 6 for an exemplary move.

Algorithm 3: Key Path Improvement (Solution S)

```
repeat
    c'_e = { 0 if e ∈ T ; c_e else } , ∀e ∈ E
    δ = 0
    forall key paths P = (V, E) ∈ P̃(S) do
        // key (end) nodes of P are u and v
        c'_e = c_e, ∀e ∈ E
        find shortest path P' = (V', E') between u and v w.r.t. c'
        δ' = ∑_{e∈E'} c'_e − ∑_{e∈E} c_e
        if δ' < δ then
            δ = δ'
            store replacement of P by P' as best move
        c'_e = 0, ∀e ∈ E
    if δ < 0 then
        apply best move
until δ ≥ 0
```

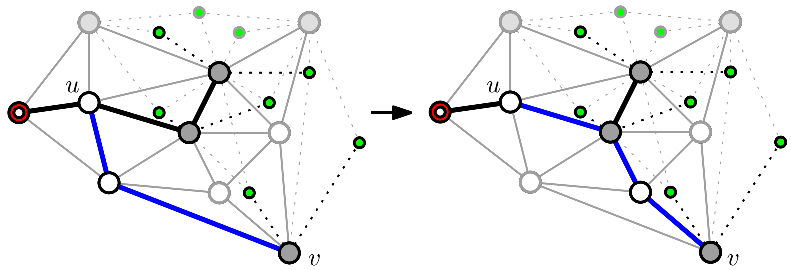


Fig. 6. An exemplary key path exchange move between key nodes u and v.

7.2 Customer Swap Neighborhood

The *Customer Swap Neighborhood* focuses on realized assignments between facilities and customers. It consists of all solutions S' differing from a solution S by swapping the assignment of exactly two customer nodes. Formally, each swap move transforms a solution S with $\alpha_S(k) = i$ and $\alpha_S(l) = j$ for customers $k, l \in C_S$ and facilities $i, j \in F_S$, into a solution S' where $\alpha_{S'}(k) = j$ and $\alpha_{S'}(l) = i$; see Figure 7 for an exemplary move. This customer swap neighborhood can be searched in $O(|C_S|^2)$ by Algorithm 4. It has already been used by Contreras et al. [30] for the SSCFLP.

Algorithm 4: Customer Swap (Solution S)

repeat
    $\delta = 0$
    $r_i = D_i - \sum_{j \in C' | \alpha_S(j) = i} d_j, \; \forall i \in F_S$
    forall $l \in C_S$ **do**
        forall $k \in C_S$ **do**
            if $\alpha_S(l) \neq \alpha_S(k)$ **then**
                if $d_l \leq r_{\alpha_S(k)} + d_k \wedge d_k \leq r_{\alpha_S(l)} + d_l$ **then**
                    $\delta' = -a_{\alpha_S(k),k} - a_{\alpha_S(l),l} + a_{\alpha_S(k),l} + a_{\alpha_S(l),k}$
                    if $\delta' < \delta$ **then**
                        $\delta = \delta'$
                        store current move as best
    if $\delta < 0$ **then**
        apply best move
until $\delta \geq 0$

7.3 Very Large Scale Neighborhood Search

Small neighborhoods as the customer swap neighborhood described above can be searched relatively fast but often yield rather poor local optima only. Recently,

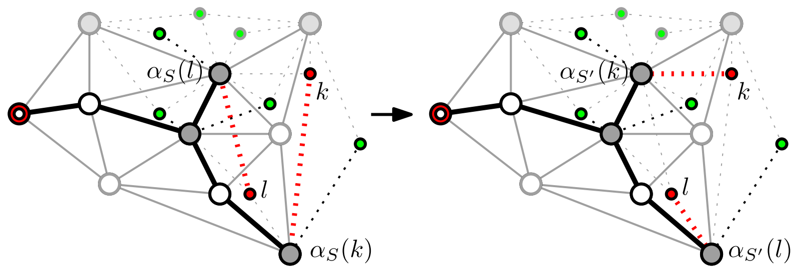


Fig. 7. An exemplary move swapping the assignments of customers k and l.

Very Large Scale Neighborhood (VLSN) search approaches have been considered for various problems to overcome limitations of simple standard neighborhood structures. If such large neighborhoods can be efficiently searched they often lead to superior solutions, since they allow for covering larger areas of a problem's search space; see e.g. [31, 32] for surveys on this topic.

Ahuja et al. [15] proposed very large scale neighborhoods for the Single Source Capacitated Facility Location Problem (SSCFLP) based on the exchange of an arbitrary number of customers and showed how to efficiently search them via shortest path calculations on a so-called improvement graph. Since CConFL contains a special variant of SSCFLP where some customers may be unassigned, in the following we generalize their work on *single-customer multi-exchanges* to be applicable to our problem variant.

To formally introduce those single-customer cyclic and path exchanges, we define the remaining capacity of each facility $i \in F$ w.r.t. a solution S as

$$r_S(i) = \begin{cases} D_i - \sum_{k \in C_S \mid \alpha_S(k)=i} d_k & \text{if } i \in F_S \\ D_i & \text{otherwise} \end{cases}, \; \forall i \in F.$$

Furthermore, by $\mathcal{F}(k) \in F_S$, $\forall k \in C_S$, we denote the facility $i \in F_S$ customer k is assigned to in S.

Analogously to Ahuja et al. [15], we define a *single-customer cyclic exchange* w.r.t. solution S as a sequence $R = (k_1, k_2, \ldots, k_q)$, $k_i \neq k_j \in C$, $1 \leq i \neq j \leq q$, such that each pair of currently assigned customers $k, t \in F_S$, $k \neq t$, from R is assigned to different facilities, i.e. $\mathcal{F}(k) \neq \mathcal{F}(t)$. Furthermore, no two consecutive customers of R may be currently unassigned, i.e. $k_i \in C_S \vee k_{i+1} \in C_S$, $i = 1, \ldots, q-1$, and $k_1 \in C_S \vee k_q \in C_S$.

Each such sequence R defines a move from an actual solution S to a solution S' by releasing each assigned customer $k_i \in C_S$ from its facility $\mathcal{F}(k_i)$, $1 \leq i \leq q$, and subsequently assigning k_i to the facility of its successor k_{i+1} in case $k_{i+1} \in C_S$, $1 \leq i \leq q-1$. Finally, k_q is assigned to $\mathcal{F}(k_1)$ if $k_1 \in C_S$. A single-customer cyclic exchange is feasible if customers may be assigned to the corresponding facilities and all capacity conditions are not exceeded.

Similarly a *single-customer path exchange* w.r.t. a solution S is a sequence $P = (k_1, k_2, \ldots, k_{q-1}, w)$ of customers $k_i \in C$, $1 \leq i \leq q-1$, and one facility

$w \in F$ as last element of the sequence with $w \neq \mathcal{F}(k_i) \neq \mathcal{F}(k_j)$, $k_i, k_j \in C_S$, $1 \leq i \neq j \leq q-1$. Thus, as for cyclic exchanges, each assigned customer $k_i \in C_S$, $i = 1, \ldots, q-1$, is released and customers k_j, $j = 1, \ldots, q-2$ are assigned to their successors' facilities $\mathcal{F}(k_{j+1})$ if $k_{j+1} \in C_S$. Finally, instead of interpreting the sequence as a cycle by eventually assigning the last customer to the first customer's original facility, k_{q-1} is simply assigned to w. As for cyclic exchanges, a path exchange is feasible, if all assignment rules as well as capacity constraints are respected.

Since applying a path exchange move might induce opening a facility and/or closing one, we also need to determine corresponding changes in the costs w.r.t. the Steiner tree in order to decide whether the corresponding move is actually improving solution S. Since computing the exact additional costs or savings would mean to re-compute a Steiner tree for each facility $k \in F$, we apply a faster shortest path heuristic that returns an upper bound for additional costs and a lower bound for savings, respectively. Thus, using those heuristic values $\zeta(i)$, $\forall i \in F$, we might miss some improving moves but can be sure that no non-improving moves are considered as improving. To determine, $\zeta(i)$, $\forall i \in F$, we compute the shortest path tree from 0 treating all solution edges as infrastructure, i.e. we use modified edge costs $c'_e = 0$, $\forall e \in T_S$ and $c'_e = c_e$, $\forall e \in E \setminus T_S$. Thus, for facilities $i \in F \setminus F_S$, $\zeta(i) = \sum_{e \in Q(i)} c'_e$, where $Q(i)$ denotes the edge set of the cheapest path from 0 to i w.r.t. edge costs c', is obviously an upper bound for the additional connection costs of facility i. Furthermore, for open facilities $i \in F_S$ we set $\zeta(i) = -\sum_{e \in Q(i) \setminus \left(\bigcup_{j \in F_S \setminus \{i\}} Q(j)\right)} c_e$, since we can obviously remove all edges $e \in Q(i) \setminus \left(\bigcup_{j \in F_S \setminus \{i\}} Q(j)\right)$ from a solution after closing facility i. For SSCFLP, Ahuja et al. [15] showed that improving path and cyclic exchanges correspond to negative subset disjoint cycles in a correspondingly defined improvement graph. Thus, in the following we show how to maintain this correlation between cycles and improving moves for our problem variant, i.e. how to define the improvement graph.

Improvement Graph: For each solution S to CConFL, we define the corresponding improvement graph $I(S) = (N(S), M(S))$. The node set $N(S) = N^{\mathrm{a}}(S) \cup N^{\mathrm{u}}(S) \cup N^{\mathrm{p}}(S) \cup \{0\}$ is the disjoint union of *assigned regular nodes* $u_k \in N^{\mathrm{a}}(S)$, $\forall k \in C_S$, *unassigned regular nodes* $v_k \in N^{\mathrm{u}}(S)$, $\forall k \in C \setminus C_S$, *pseudo nodes* $w_i \in N^{\mathrm{p}}(S)$, $\forall i \in F$, and an origin node o. The origin node o and its adjacent arcs are included to model path exchanges by means of cycles in $I(S)$, see also [15].

The set of arcs $M(S)$ is the disjoint union of
- arcs $M^{(\mathrm{a},\mathrm{a})}(S)$ between assigned regular nodes,
- arcs $M^{(\mathrm{a},\mathrm{u})}(S)$ from assigned to unassigned regular nodes,
- arcs $M^{(\mathrm{u},\mathrm{a})}(S)$ from unassigned to assigned regular nodes,
- arcs $M^{(\mathrm{a},\mathrm{p})}(S)$ from assigned regular to pseudo nodes,
- arcs $M^{(\mathrm{u},\mathrm{p})}(S)$ from unassigned regular to pseudo nodes,

- arcs $M^{(\mathrm{p,o})}(S)$ from pseudo nodes to the origin,
- arcs $M^{(\mathrm{o,a})}(S)$ from the origin to assigned regular nodes, and
- arcs $M^{(\mathrm{o,u})}(S)$ from the origin to unassigned regular nodes.

Next, we will describe these arcs as well as their costs $\gamma_{i,j}$, $\forall(i,j) \in M(S)$, corresponding to the resulting changes of the objective value formally as well as w.r.t. their interpretation.

Arcs $(u_k, u_l) \in M^{(\mathrm{a,a})}(S)$ denote releasing customer $l \in C_S$ from $i = \mathcal{F}(l)$ and in turn assigning customer $k \in C_S$ to facility i, leading to arc costs $\gamma_{u_k,u_l} = a_{i,k} - a_{i,l}$. Since, we must ensure that k can be assigned to $\mathcal{F}(l)$ as well as that capacity constraints are respected, the corresponding arc set is defined as $M^{(\mathrm{a,a})}(S) = \{(u_k, u_l) \mid u_k, u_l \in N^{\mathrm{a}}(S) : \mathcal{F}(l) \in F_k \wedge \mathcal{F}(k) \neq \mathcal{F}(l) \wedge r_S(\mathcal{F}(l)) + d_l \geq d_k\}$. Each arc $(u_k, v_l) \in M^{(\mathrm{a,u})}(S) = \{(u_k, v_l) \mid u_k \in N^{\mathrm{a}}(S), v_l \in N^{\mathrm{u}}(S)\}$, with corresponding costs $\gamma_{u_k,v_l} = p_k$ from an assigned to an unassigned regular node, models releasing customer k. Arcs $(v_k, u_l) \in M^{(\mathrm{u,a})}(S) = \{(v_k, u_l) \mid v_k \in N^{\mathrm{u}}(S), u_l \in N^{\mathrm{a}}(S) : \mathcal{F}(l) \in F_k \wedge r_S(\mathcal{F}(l)) + d_l \geq d_k\}$ with costs $\gamma_{u_k,v_l} = a_{\mathcal{F}(l),k} - a_{\mathcal{F}(l),l} - p_k$ indicate releasing l from $i = \mathcal{F}(l)$ and subsequently assigning the previously unassigned customer k to facility $i \in F_S$.

$M^{(\mathrm{a,p})}$ consists of one arc (u_k, w_i) from each each assigned regular node to each pseudo node if the corresponding customer k can be assigned to facility i, i.e. $M^{(\mathrm{a,p})}(S) = \{(u_k, w_i) \mid u_k \in N^{\mathrm{a}}(S), w_i \in N^{\mathrm{p}}(S) : i \neq \mathcal{F}(k) \wedge i \in F_k \wedge r_S(i) \geq d_k\}$. Since eventually occurring facility opening costs will be considered by arcs going out of w_i, costs $\gamma_{u_k,w_i} = a_{i,k}$ are given by the costs of assigning customer k to facility i. To allow for assigning currently unassigned customers $k \in F \setminus F_S$ to some facility $i \in F$ without previously releasing another customer from i, we include arcs $(v_k, w_i) \in M^{(\mathrm{u,p})}(S) = \{(v_k, w_i) \mid v_k \in N^{\mathrm{u}}(S), w_i \in N^{\mathrm{p}}(S) : i \in F_k \wedge r_S(i) \geq d_k\}$. As we additionally earn a customers prize here, arc $(v_k, w_i) \in M^{(\mathrm{u,p})}(S)$ has costs $\gamma_{v_k,w_i} = a_{i,k} - p_k$.

To model path exchanges as cycles in the graph, we further need to include arcs from each pseudo node to the origin and arcs from the origin to assigned as well as unassigned regular nodes. Arcs $M^{(\mathrm{p,o})}(S) = \{(w_i, 0) \mid w_i \in N^{\mathrm{p}}(S)\}$ model eventually occurring opening and connection costs of facility $i \in F$, i.e.

$$\gamma_{w_i,\mathrm{o}} = \begin{cases} 0 & \text{if } i \in F_S \\ f_i + \zeta_i & \text{otherwise} \end{cases}, \ \forall(w_i, \mathrm{o}) \in M^{(\mathrm{p,o})}.$$

Using an arc $(\mathrm{o}, u_k) \in M^{(\mathrm{o,a})}(S) = \{(\mathrm{o}, u_k) \mid u_k \in N^{\mathrm{a}}(S)\}$ from the origin node o to some assigned regular node u_k releases customer k from its facility, yielding arc costs

$$\gamma_{\mathrm{o},u_k} = \begin{cases} -a_{\mathcal{F}(k),k} & \text{if } \exists l \neq k \in C_S : \mathcal{F}(k) = \mathcal{F}(l) \\ -a_{\mathcal{F}(k),k} - f_{\mathcal{F}(k)} + \zeta_{\mathcal{F}(k)} & \text{otherwise} \end{cases}, \ \forall(\mathrm{o}, u_k) \in M^{(\mathrm{o,a})}.$$

Finally, arcs $(\mathrm{o}, v_k) \in M^{(\mathrm{o,u})}(S) = \{(\mathrm{o}, v_k) \mid v_k \in N^{\mathrm{u}}(S)\}$ from the origin to some unassigned regular node are included for allowing to assign a new customer without previously releasing another one. Consequently, those arcs have zero costs, i.e. $\gamma_{\mathrm{o},v_k} = 0$, $\forall(\mathrm{o}, v_k) \in M^{(\mathrm{o,u})}$.

Searching for improving moves: Generalizing the definition given in [15] we call a directed cycle $(u_1, \ldots, u_q)$, $u_i \in N(S)$, $i = 1, \ldots, q$, of $I(S)$ subset disjoint, if each of its assigned regular nodes and pseudo nodes are associated with different facility locations. If the total edge costs of such a cycle are negative, it is called negative cost subset disjoint. Since only feasible arcs w.r.t. assignment rules and capacity conditions are included in $I(S)$, and edge costs reflect changes in the objective value those negative cost subset disjoint cycles correspond to improving path and cyclic exchange moves. However, if such a cycle does induce opening facility $i \in F \setminus F_S$ as well as closing a facility $j \in F_S$, a cycle's cost might not be equal to the actual cost changes when applying the move since the additional costs/savings ζ due to adapting the Steiner tree have been computed independently for each facility. Since opening and connecting a new facility and assigning only one customer to it does only rarely pay off, this special case is rather unlikely to occur in practice. Therefore, we simply check whether a found cycle does simultaneously open and close two facilities and add eventually occurring additional connection costs before deciding whether this cycle is an improving one.

Thomson and Orlin [33] proved that deciding whether a graph contains a negative subset disjoint cycle is NP-hard. Subsequently, Ahuja et al. [34] proposed an effective heuristic for finding negative cost subset disjoint cycles based on the label correcting algorithm for the shortest path problem. This heuristic has already been used for the SSCFLP [15] and in practice rarely fails to find existing negative cost subset disjoint cycles if started once from each regular node. As shown in Algorithm 5, we search the neighborhood defined by the set of single customer path and cyclic exchanges using a best improvement strategy, adopting the heuristic of Ahuja et al. [15] to find negative subset disjoint cycles which is also started from every regular node.

Figure 9 depicts an exemplary improvement graph $I(S) = (N(S), M(S))$ with respect to a solution S as shown in Figure 8 assuming that each clients demant is equal to one, while each facilities maximum assignable demand is two. Figure 10 shows an exemplary feasible cyclic exchange $R = (k_1, k_4, k_5, k_2)$ with respect to solution S. Thus after applying R, customer k_2 will be assigned to facility h, k_1 to i, k_4 to j, and finally k_5 will be unassigned. Since $k_3 \notin R$ it will still be assigned to facility i. An exemplary path exchange $P = (k_2, k_1, k_4, j)$ is shown in Figure 11. Here, k_2 will be assigned to facility h, k_1 to i, and k_4 to j after applying the corresponding move, while k_3 and k_5 will remain assigned to their facilities i and j since $k_3, k_5 \notin P$. Note that the origin node o is duplicated in Figures 9, 10, and 11 to keep them simple.

8 Computational Results

For ConFL, Ljubić combined benchmark instances for the STP with instances for uncapacitated facility location. Similarly, we created instances for CConFL[2] by

[2] available at http://www.ads.tuwien.ac.at/people/mleitner/cconfl/instances.tar.gz

Algorithm 5: Very Large Scale Neighborhood Search(Solution S)

repeat
  $\delta = 0$
  construct improvement graph
  forall $k \in C$ **do**
    heuristically find negative cost subset disjoint cycle $\mathcal{C}$
    $\delta = \sum_{(u,v)\in\mathcal{C}} \gamma_{u,v}$
    if $\mathcal{C}$ induces closing facility $i \in F_S$ and opening $j \in F \setminus F_S$ **then**
      $Q = Q(i) \setminus \left(\bigcup_{l\in F_S\setminus\{i\}} Q(l)\right)$
      $\delta = \delta + \sum_{e\in Q\cap Q(j)} c_e$
    if $\delta < \delta'$ **then**
      $\delta = \delta'$
      store current cycle as best move
  if $\delta < 0$ **then**
    apply best move
until $\delta \geq 0$

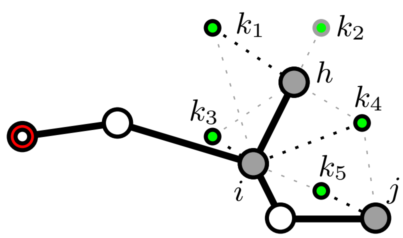


Fig. 8. An exemplary Solution S.

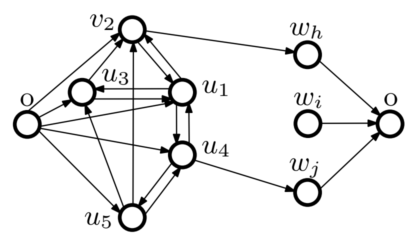


Fig. 9. Improvement graph $I(S) = (N(S), M(S))$.

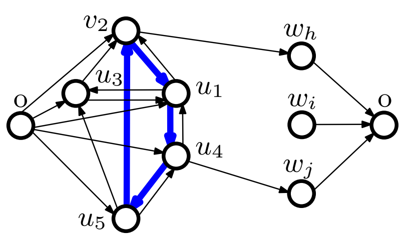


Fig. 10. An exemplary cyclic exchange $R = (k_1, k_4, k_5, k_2)$.

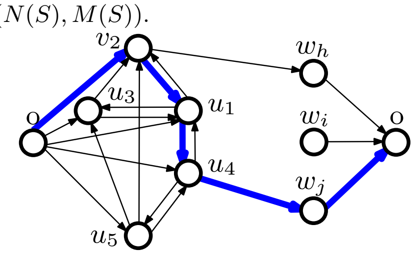


Fig. 11. An exemplary path exchange $P = (k_2, k_1, k_4, j)$.

combining STP instances from the OR-library[3] with instances for the SSCFLP created with the instance generator[4] of Kratica et al. [35].

[3] http://people.brunel.ac.uk/~mastjjb/jeb/orlib/steininfo.html
[4] http://alas.matf.bg.ac.yu/~kratica/instances/splp_gen_w32.zip

The node with index one in the STP instance is chosen 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 creating relatively easy instances by setting them too high. For each customer $k \in C$, we randomly select its prize $p_k \in \mathbb{N}_0$ from the interval $\left[\overline{a}(k), a_{\max}(k) + \overline{f}\right]$, where $\overline{a}(k) = \frac{\sum_{i \in F} a_{i,k}}{|F_k|}$ 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 [36] is applied to remove some Steiner nodes and edges.

We performed all computational experiments on a single core of an Intel Core 2 Quad with 2.83GHz and 8GB RAM. ILOG CPLEX 12.1 has been used for directly solving $dMCF_f$, $dMCF_c$ as well as their LP relaxations $dMCF_f^{\text{LP}}$ and $dMCF_c^{\text{LP}}$. To allow for a fair comparison to our Lagrangian decomposition based approaches, we used the single threaded variant of CPLEX.

Table 1 compares LP relaxation values of $dMCF_f$ and $dMCF_c$ for small test instances using a time limit of 14400 seconds. We conclude that, although none of the formulations theoretically dominates the other, $dMCF_f$ is on our instances far more efficient from a computational perspective. Thus, we only consider $dMCF_f$ for all further experiments. Further computational results for instances where $|F| = |C|$ are summarized in Table 2, and in Table 3 for instances with $|F| \neq |C|$. Here, we apply a CPU-time limit of 7200 seconds. *LD* denotes the pure Lagrangian decomposition approach applying the Lagrangian heuristic presented in Section 6 without any further primal improvement, while *LDS* corresponds to the variant applying the simpler primal improvement, i.e. considering the key path and customer swap neighborhoods, and *LDV* applies the VLSN search instead of the customer swap improvement, see also Algorithm 2. Since $dMCF_f$ could not solve any instance to proven optimality within the given time limit, we do not report its runtime in Tables 2 and 3.

We use the volume algorithm as described by Haouari and Siala [24] with the following settings for approximately solving the Lagrangian dual problem. Lagrangian multipliers are initialized by $\pi_{k,e} = c'_e$ for assignment edges $e \in E' \setminus E$ and by $\pi_{k,e} = c_e/|C|$ for edges $e \in E$. The target value T is initially set to 1.2 and multiplied by 1.1 in case $z_{\text{LB}} > 0.9 z_{\text{UB}}$ where z_{UB} and z_{LB} denote the so far best upper and lower bounds, respectively. ρ is initialized with 0.1 and multiplied by 0.67 after 20 non-improving iterations in case $\rho > 10^{-4}$ and by 1.5 in each improving iteration if $\rho < 5$ and if $\bar{v} \cdot v^t \geq 0$. Instead of computing λ_{OPT} as suggested in [24], we always use $\lambda = \lambda_{\text{MAX}}$ which we initialize with 0.01. After every 100 iterations we multiply λ_{MAX} by 0.85 in case the lower bound did

Table 1. Comparison of LP relaxation values and corresponding CPU-times (in seconds) for $dMCF_f$ and $dMCF_c$ on small instances (time limit 14400s).

Instance					${dMCF_f}^{\mathrm{LP}}$		${dMCF_c}^{\mathrm{LP}}$	
Name	$\|F\|$	$\|C\|$	$\|V\|$	$\|E\|$	obj.	time	obj.	time
c10-mo75	75	75	408	908	2878.7	94	2852.2	3272
c10-mq75	75	75	405	905	7095.2	116	7077.3	1386
c10-ms75	75	75	407	907	9506.3	194	9479.4	4487
d10-mo75	75	75	771	1770	2772.6	484	-	14400
d10-mq75	75	75	775	1774	7295.0	167	7278.9	3458
d10-ms75	75	75	781	1780	10069.3	1103	-	14400
c10-mo	75	200	404	904	8153.5	713	8118.2	6450
c10-mp	75	200	403	903	14917.4	228	-	14400
c10-mq	75	200	403	903	20717.2	328	-	14400
c10-mo	200	75	435	935	2957.0	6229	-	14400
c10-mp	200	75	428	928	5444.6	3439	5432.0	11206
c10-mq	200	75	430	930	8093.5	1931	8076.6	10748

improve less than 1% and if $\lambda_{\mathrm{MAX}} > 10^{-5}$. The volume algorithm is terminated after 250 consecutive non-improving iterations or if the time limit is reached.

Comparing Tables 1, 2, and 3 with respect to the lower bounds, we conclude that $dMCF_f$ does generate the best lower bounds if given enough time, while the lower bounds of our Lagrangian decomposition approaches are approximately equal to those of $dMCF_c^{\mathrm{LP}}$, at least for those small instances where $dMCF_c^{\mathrm{LP}}$ could be solved. However, for larger instances solving $dMCF_f^{\mathrm{LP}}$ often requires longer than applying the Lagrangian decomposition approaches which generate a slightly worse lower bound but additionally compute feasible solutions to CConFL. *LDV* clearly outperforms the other approaches with respect to the primal solution quality, i.e. the resulting upper bounds. For instances with $|F| = |C|$, see Table 2, *LDV* produced the best results for 27 out of 36 instances, while *LDS* is the winner on six, and $dMCF_f$ on only three instances. Similarly, for instances with $|F| \neq |C|$, *LDV* produced better upper bounds than the other approaches in 20 out of 22 cases, while *LDS* as well as $dMCF_f$ performed best with respect to primal solution quality on only a single instance each.

Although its lower bounds are worse than those of model $dMCF_f$, *LDV* also outperforms the other approaches with respect to the resulting relative gaps between upper and lower bounds. While *LDV* yielded the smallest gaps for 18, $dMCF_f$ for 13, and *LDS* for six out of 36 instances if $|F| = |C|$, *LDV* even performs better compared to the other approaches if $|F| \neq |C|$. Here, $dMCF_f$ won in two, *LDS* in one, and *LDV* in 10 out of 12 cases. Furthermore, while $dMCF_f$ sometimes produces enormous gaps or even completely fails, all Lagrangian approaches are relatively stable with respect to the resulting gaps. Except for three instances with $|F| = 200$ and $|C| = 75$, which seem to be particularly hard, the LDV's gaps between lower and upper bounds never exceed 4.4% and are smaller than or equal to 2% for 70% of all tested instances.

Finally, we observe from Tables 2 and 3 that, especially for larger instances, all Lagrangian approaches usually need significantly less CPU time than solving the LP relaxation of model $dMCF_f$. Due to applying primal improvement only to a relatively small, but highly promising subset of candidate solutions derived by our Lagrangian heuristic, the overhead of LDS and LDV usually is only moderate. Sometimes LDS or LDV are even faster than LD since a better upper bound eventually found in an early iteration of the volume algorithm does influence Lagrangian multipliers and the whole process of approximately solving the Lagrangian dual. Even though LDV tends to need more time than LDS for larger instances, no clear advantages with respect to runtime can be observed for one of those two approaches.

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. We presented two mixed integer programming formulations for CConFL based on multi-commodity flows and showed that neither of those dominates the other one. Furthermore, we proposed an approach based on Lagrangian relaxation decomposing CConFL, into three types of independent subproblems. Using a Lagrangian heuristic we derive feasible solutions in each iteration of the volume algorithm which we use for solving the Lagrangian dual. Furthermore, we discussed two hybrid methods combining the Lagrangian approach with local search and VLSN search. Experimental results indicated that especially the approach using VLSN is able to generate high quality solutions with tight gaps. By applying those primal improvements to highly promising solutions only, the additionally needed computational time is relatively small. It may be possible to further reduce the required time, by using alternative algorithms for solving the negative subset disjoint cycle problem [37] within our VLSN approach. We argue that our approach is feasible for solving even larger instances, since it can be easily parallelized as the various subproblems of our relaxed model are completely independent of each other. Furthermore, our primal improvement approach is naturally composed out of two independent subproblems, i.e. a Steiner tree problem and a single source capacitated facility location problem.

We are currently working on exact approaches for medium sized instances of CConFL based on branch-and-cut and branch-cut-and-price. Furthermore, we plan to develop fast metaheuristics for solving very large scale instances of CConFL within reasonable time.

Acknowledgements

This work is supported by the Austrian Research Promotion Agency (FFG) under grant 811378 and by the Austrian Science Fund (FWF) under grant P20342-N13.

Table 2. Results on instances with $|F| = |C|$.

Instance					**lower bound**					**upper bound**				**gap in %**				**CPU-time [s]**			
Name	$\|F\|$	$\|C\|$	$\|V\|$	$\|E\|$	$dMCF_f^{LP}$	$dMCF_f$	LD	LDS	LDV	$dMCF_f$	LD	LDS	LDV	$dMCF_f^{LP}$	LD	LDS	LDV	$dMCF_f^{LP}$	LD	LDS	LDV
c10-mo75	75	75	408	908	2878.7	**2880.1**	2851.9	2851.7	2852.1	2944.2	2988.2	2962.4	**2938.4**	**2.2**	4.8	3.9	3.0	94	107	106	81
c10-mq75	75	75	405	905	7095.2	**7105.2**	7079.2	7078.4	7076.9	7171.3	7239.1	7177.4	**7158.0**	**0.9**	2.3	1.4	1.1	116	130	101	73
c10-ms75	75	75	407	907	9506.3	**9509.5**	9479.9	9478.9	9479.9	9578.1	9629.8	9581.1	**9554.5**	**0.7**	1.6	1.1	0.8	194	194	106	175
c15-mo75	75	75	500	2500	2747.5	**2748.7**	2737.5	2738.2	2738.3	2833.3	2855.2	2815.3	**2793.4**	3.1	4.3	2.8	**2.0**	877	133	155	127
c15-mq75	75	75	500	2500	7466.5	**7469.7**	7457.2	7457.6	7456.8	7966.2	7576.1	7541.4	**7505.3**	6.6	1.6	1.1	**0.6**	1567	169	177	143
c15-ms75	75	75	500	2500	9354.6	**9357.7**	9341.7	9342.1	9343.0	10918.9	9487.5	9408.6	**9390.8**	16.7	1.6	0.7	**0.5**	2040	302	185	202
d10-mo75	75	75	771	1770	2772.6	**2776.3**	2741.6	2741.8	2741.5	2842.2	2921.9	2849.3	**2830.7**	**2.4**	6.6	3.9	3.3	484	224	228	244
d10-mq75	75	75	775	1774	7295.0	**7299.7**	7280.9	7281.5	7281.3	7373.0	7432.9	**7358.7**	7359.5	**1.0**	2.1	1.1	1.1	167	175	184	216
d10-ms75	75	75	781	1780	10069.3	**10073.9**	10019.0	10018.9	10018.9	10233.6	10257.3	10213.2	**10167.4**	1.6	2.4	1.9	**1.5**	1103	242	218	158
d15-mo75	75	75	1000	5000	2641.8	**2645.0**	2636.6	2636.9	2636.8	3397.2	2743.2	**2697.1**	2699.6	28.4	4.0	**2.3**	2.4	2402	251	268	306
d15-mq75	75	75	1000	5000	-	**7380.2**	7370.8	7370.1	7369.0	8528.6	7473.5	**7433.8**	7445.2	15.6	1.4	**0.9**	1.0	7200	380	239	147
d15-ms75	75	75	1000	5000	-	**9237.4**	9221.6	9222.2	9221.1	11007.6	9334.3	**9292.5**	9311.3	19.2	1.2	**0.8**	1.0	7200	298	549	237
c10-mo100	100	100	406	906	3330.9	**3333.0**	3303.2	3297.7	3302.3	**3380.0**	3486.1	3437.3	3406.3	**1.4**	5.5	4.2	3.1	217	222	470	300
c10-mq100	100	100	406	906	9352.6	**9359.0**	9322.5	9322.6	9322.5	9473.7	9610.5	9491.5	**9460.4**	**1.2**	3.1	1.8	1.5	367	250	234	228
c10-ms100	100	100	416	916	11740.1	**11746.0**	11697.9	11693.2	11696.8	**11855.1**	11979.1	11896.3	11899.4	**0.9**	2.4	1.7	1.7	166	288	207	243
c15-mo100	100	100	500	2500	3422.6	**3426.5**	3413.8	3413.6	3413.7	3933.8	3562.4	3542.6	**3493.6**	14.8	4.4	3.8	**2.3**	2809	314	295	209
c15-mq100	100	100	500	2500	9120.5	**9125.1**	9118.6	9113.3	9117.2	9739.6	9331.9	9214.8	**9192.4**	6.7	2.3	1.1	**0.8**	4008	270	205	173
c15-ms100	100	100	500	2500	11277.0	**11281.4**	11264.0	11263.6	11264.6	12722.2	11533.2	11426.2	**11379.1**	12.8	2.4	1.4	**1.0**	5204	412	329	386
d10-mo100	100	100	788	1787	3376.7	**3380.7**	3337.8	3339.6	3340.2	3483.2	3540.9	3474.6	**3461.1**	**3.0**	6.1	4.0	3.6	435	358	465	253
d10-mq100	100	100	778	1777	9179.2	**9185.4**	9129.6	9128.1	9130.3	**9258.9**	9406.5	9374.9	9261.6	**0.8**	3.0	2.7	1.4	581	400	278	486
d10-ms100	100	100	783	1782	11049.0	**11055.0**	11000.4	11001.3	11000.7	11197.6	11348.6	11234.3	**11161.8**	**1.3**	3.2	2.1	1.5	603	330	292	247
d15-mo100	100	100	1000	5000	-	**3314.0**	3297.8	3298.6	3298.8	3862.1	3454.6	3424.1	**3369.8**	16.5	4.8	3.8	**2.2**	-	704	551	404
d15-mq100	100	100	1000	5000	-	-	9149.5	**9150.4**	9149.8	23780.0	9422.2	**9232.0**	9256.4	-	3.0	**0.9**	1.2	-	457	373	492
d15-ms100	100	100	1000	5000	-	**11332.4**	11309.2	11307.7	11307.3	12715.9	11549.3	**11398.4**	11413.0	12.2	2.1	**0.8**	0.9	-	522	645	370
c10-mo200	200	200	433	933	7116.2	**7123.0**	7052.9	7053.3	7052.5	7329.4	7440.1	7325.2	**7269.6**	**2.9**	5.5	3.9	3.1	354	4302	7200	2174
c10-mq200	200	200	428	928	19270.3	**19279.8**	19211.7	19213.2	19211.8	19539.8	19673.0	19574.3	**19436.1**	1.3	2.4	1.9	**1.2**	579	3978	4409	5856
c10-ms200	200	200	431	931	25190.6	**25197.3**	25115.3	25114.6	25115.1	25327.2	25680.7	25627.1	**25306.5**	**0.5**	2.3	2.0	0.8	1040	7200	4354	4327
c15-mo200	200	200	500	2500	-	**7139.0**	7108.8	7105.5	7106.8	8383.9	7427.4	7450.5	**7252.3**	17.4	4.5	4.9	**2.0**	7200	3797	3142	4129
c15-mq200	200	200	500	2500	-	**19191.4**	19171.1	19171.0	19171.0	21455.8	19495.3	19326.5	**19290.1**	11.8	1.7	0.8	**0.6**	7200	3459	4961	4823
c15-ms200	200	200	500	2500	-	**24683.6**	24654.4	24655.8	24655.2	26764.0	25302.2	25003.3	**24854.6**	8.4	2.6	1.4	**0.8**	7200	4662	4197	4899
d10-mo200	200	200	816	1815	7194.1	**7197.4**	7107.3	7106.9	7107.9	8021.9	7599.7	7448.3	**7331.6**	11.5	6.9	4.8	**3.1**	3273	5498	4613	7200
d10-mq200	200	200	814	1813	18789.0	**18796.9**	18720.8	18720.9	18720.5	21247.5	19214.6	19025.8	**18971.6**	13.0	2.6	1.6	**1.3**	3791	5900	4770	3119
d10-ms200	200	200	806	1805	24509.6	**24517.3**	24426.7	24425.9	24426.9	27880.1	24856.3	24730.5	**24696.9**	13.7	1.8	1.2	**1.1**	6624	6681	4340	4686
d15-mo200	200	200	1000	5000	-	-	**7129.0**	7126.9	7127.8	-	7448.1	7381.0	**7329.1**	-	4.5	3.6	**2.8**	7200	5159	5822	5174
d15-mq200	200	200	1000	5000	-	-	19457.0	19455.3	**19457.1**	-	19880.4	19772.2	**19606.4**	-	2.2	1.6	**0.8**	7200	4743	5081	7200
d15-ms200	200	200	1000	5000	-	-	24007.7	24008.3	**24009.4**	73434.0	24539.0	24201.9	**24198.9**	-	2.2	**0.8**	**0.8**	7200	4666	5835	6050

Table 3. Results on instances with $|F| \neq |C|$.

Instance					**lower bound**					**upper bound**				**gap in %**				**CPU time [s]**			
Name	$\|F\|$	$\|C\|$	$\|V\|$	$\|E\|$	$dMCF_f^{\text{LP}}$	$dMCF_f$	LD	LDS	LDV	$dMCF_f$	LD	LDS	LDV	$dMCF_f^{\text{LP}}$	LD	LDS	LDV	$dMCF_f^{\text{LP}}$	LD	LDS	LDV
c10-mo	75	200	404	904	8153.5	**8158.2**	8117.6	8118.5	8116.9	9181.3	8630.1	8442.8	**8258.6**	12.5	6.3	4.0	**1.7**	713	591	736	1458
c10-mp	75	200	403	903	14917.4	**14924.5**	14882.6	14881.6	14882.2	**15056.9**	15407.4	15225.5	15059.9	**0.9**	3.5	2.3	1.2	228	707	651	1672
c10-mq	75	200	403	903	20717.2	**20725.5**	20681.3	20681.6	20681.1	20915.4	21150.6	20957.2	**20844.9**	0.9	2.3	1.3	**0.8**	328	901	1387	1468
c15-mo	75	200	500	2500	-	**7948.0**	7935.1	7935.4	7935.3	9634.2	8166.6	8068.3	**8049.9**	21.2	2.9	1.7	**1.4**	7200	870	839	1609
c15-mp	75	200	500	2500	14493.1	**14497.5**	14483.2	14483.2	14483.2	15722.6	14809.9	14643.5	**14608.0**	8.5	2.3	1.1	**0.9**	5533	919	952	977
c15-mq	75	200	500	2500	21570.7	**21576.2**	21561.4	21561.4	21561.3	22973.5	21940.2	21770.5	**21662.6**	6.5	1.8	1.0	**0.5**	3575	931	984	2183
d10-mo	75	200	775	1775	8228.0	**8234.7**	8183.6	8181.9	8182.7	8511.6	8717.3	8604.5	**8427.4**	3.4	6.5	5.2	**3.0**	2166	853	791	1421
d10-mp	75	200	775	1774	14836.9	**14842.5**	14779.3	14778.0	14779.4	15075.2	15271.8	15181.1	**14975.6**	1.6	3.3	2.7	**1.3**	2265	915	857	1532
d10-mq	75	200	774	1773	20834.2	**20839.4**	20766.9	20766.6	20767.2	21044.1	21221.4	21081.6	**21009.1**	**1.0**	2.2	1.5	1.2	1001	1052	1035	2476
d15-mo	75	200	1000	5000	-	-	8127.5	8128.2	**8128.5**	20610.0	8451.4	8401.2	**8262.5**	-	4.0	3.4	**1.6**	7200	1528	1325	2268
d15-mp	75	200	1000	5000	-	**14731.9**	14717.0	14717.6	14718.5	15760.3	15115.2	14893.5	**14839.6**	7.0	2.7	1.2	**0.8**	7200	1164	1202	1090
d15-mq	75	200	1000	5000	-	-	21407.6	21407.0	**21407.8**	57923.0	21741.0	21588.7	**21526.7**	-	1.6	0.8	**0.6**	7200	1419	1672	1329
c10-mo	200	75	435	935	**2957.0**	**2957.0**	2952.0	2951.1	2949.8	7209.0	3321.4	3179.6	**3044.6**	143.8	12.5	7.7	**3.2**	6229	484	434	509
c10-mp	200	75	428	928	5444.6	**5448.9**	5434.5	5434.9	5434.9	5517.8	5668.1	5638.7	**5506.6**	1.3	4.3	3.7	**1.3**	3439	556	502	579
c10-mq	200	75	430	930	8093.5	**8098.0**	8080.2	8080.3	8080.9	8203.5	8208.3	**8189.2**	8290.7	**1.3**	1.6	**1.3**	2.6	1931	438	494	460
c15-mo	200	75	500	2500	-	**2948.5**	2947.3	2946.0	2947.4	7189.0	3312.9	3187.0	**3156.9**	143.8	12.4	8.2	**7.1**	7200	475	437	350
c15-mp	200	75	500	2500	-	5150.7	**5153.6**	5152.6	5151.3	12693.0	5446.2	5296.4	**5262.2**	146.4	5.7	2.8	**2.2**	7200	561	627	455
c15-mq	200	75	500	2500	-	7667.3	7666.4	7668.3	**7668.4**	10212.3	7837.4	7810.8	**7789.5**	33.2	2.2	1.9	**1.6**	7200	509	533	398
d10-mo	200	75	811	1810	-	**3040.5**	3026.3	3028.4	3028.5	7500.0	3887.1	3971.5	**3825.4**	146.7	28.4	31.1	**26.3**	7200	461	454	381
d10-mp	200	75	809	1808	5377.7	**5381.4**	5361.8	5361.9	5360.9	5598.2	5777.4	5657.6	**5594.5**	**4.0**	7.8	5.5	4.4	5608	628	540	558
d10-mq	200	75	820	1819	7698.7	**7702.2**	7675.5	7676.7	7676.4	10753.2	8127.2	7890.6	**7827.8**	39.6	5.9	2.8	**2.0**	3620	449	533	546
d15-mo	200	75	1000	5000	-	-	2950.1	**2951.6**	2951.1	-	3633.4	3265.0	**3236.2**	-	23.2	10.6	**9.7**	7200	776	722	500
d15-mp	200	75	1000	5000	-	-	5387.3	**5387.8**	5387.2	13849.0	5828.8	5625.1	**5486.3**	-	8.2	4.4	**1.8**	7200	680	730	621
d15-mq	200	75	1000	5000	-	-	7564.3	7563.6	**7571.7**	-	8040.7	7805.4	**7704.4**	-	6.3	3.2	**1.8**	7200	663	599	471

References

1. Leitner, M., Raidl, G.R.: A lagrangian decomposition based heuristic for capacitated connected facility location. In Voß, S., Caserta, M., eds.: Proceedings of the 8th Metaheuristic International Conference (MIC 2009), Hamburg, Germany (July 2009)
2. Karp, R.M.: Reducibility among combinatorial problems. In Miller, E., Thatcher, J.W., eds.: Complexity of Computer Computations. Plenum Press (1972) 85–103
3. Cornuejol, G., Nemhauser, G.L., Wolsey, L.A.: The uncapacitated facility location problem. In Mirchandani, P.B., Francis, R.L., eds.: Discrete Location Theory. Wiley (1990) 119–171
4. Karger, D.R., Minkoff, M.: Building Steiner trees with incomplete global knowledge. In: Proceedings of the 41st Annual Symposium on Foundations of Computer Science, IEEE Computer Society (2000) 613–623
5. 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. (2001) 389–398
6. Swamy, C., Kumar, A.: Primal-dual algorithms for connected facility location problems. Algorithmica **40**(4) (2004) 245–269
7. Hasan, M.K., Jung, H., Chwa, K.: Approximation algorithms for connected facility location problems. Journal of Combinatorial Optimization **16**(2) (2008) 155–172
8. 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. (2008) 1174–1183
9. Ljubić, I.: A hybrid VNS for connected facility location. In Bartz-Beielstein, T., et al., eds.: Hybrid Metaheuristics, 4th International Workshop, HM 2007. Volume 4771 of LNCS., Springer (2007) 157–169
10. 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, IEEE Computer Society (2008) 257–260
11. Raghavan, S., Bardossy, M.G.: Dual based heuristics for the connected facility location problem. In Scutellà, M.G., et al., eds.: Proceedings of the International Network Optimization Conference 2009. (2009)
12. 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, IEEE Computer Society (2008) 233–236
13. Khuller, S., Zhu, A.: The general steiner tree-star problem. Information Processing Letters **84**(4) (2002) 215–220
14. Winter, P.: Steiner problem in networks: a survey. Networks **17**(2) (1987) 129–167
15. 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) (2004) 749–760
16. Bahiense, L., Barahona, F., Porto, O.: Solving steiner tree problems in graphs with lagrangian relaxation. Journal of Combinatorial Optimization **7**(3) (2003) 259–282
17. Holmberg, K., Rönnqvist, M., Yuan, D.: An exact algorithm for the capacitated facility location problems with single sourcing. European Journal of Operational Research **113** (1999) 544–559
18. Beasley, J.E.: Lagrangean relaxation. In Reeves, C.R., ed.: Modern heuristic techniques in combinatorial problems., Blackwell Scientific Publications (1993) 243–303

19. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik **1** (1959) 269–271
20. Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman & Co. (1979)
21. Martello, S., Pisinger, D., Toth, P.: Dynamic programming and strong bounds for the 0-1 knapsack problem. Management Science **45**(3) (1999) 414–424
22. Barahona, F., Anbil, R.: The volume algorithm: producing primal solutions with a subgradient method. Mathematical Programming **87**(3) (2000) 385–399
23. Fisher, M.L.: The Lagrangian relaxation method for solving integer programming problems. Management Science **27**(1) (1981) 1–18
24. Haouari, M., Siala, J.C.: A hybrid Lagrangian genetic algorithm for the prize collecting Steiner tree problem. Computers and Operations Research **33**(5) (2006) 1274–1288
25. Minkoff, M.: The prize collecting Steiner tree problem. Master's thesis, Massachusetts Institute of Technology (2000)
26. Voß, S.: Steiner's problem in graphs: heuristic methods. Discrete Applied Mathematics **40** (1992) 45–72
27. Martins, S.L., Resende, M.G.C., Ribeiro, C.C., Pardalos, P.M.: A parallel GRASP for the Steiner tree problem in graphs using a hybrid local search strategy. Journal of Global Optimization **17**(1-4) (2000) 267–283
28. Verhoeven, M.G.A., Severens, M.E.M.: Parallel local search for steiner trees in graphs. Annals of Operations Research **90** (1999) 185–202
29. Leitner, M., Raidl, G.R.: Lagrangian decomposition, metaheuristics, and hybrid approaches for the design of the last mile in fiber optic networks. In Blesa, M.J., et al., eds.: Hybrid Metaheuristics 2008. Volume 5296 of LNCS., Springer (2008) 158–174
30. Contreras, I.A., Diaz, J.A.: Scatter search for the single source capacitated facility location problem. Annals of Operations Research **157**(1) (2008) 73–89
31. Ahuja, R.K., Ergun, Ö., Orlin, J.B., Punnen, A.P.: A survey of very large-scale neighborhood search techniques. Discrete Applied Mathematics **123**(1-3) (2002) 75–102
32. Chiarandini, M., Dumitrescu, I., Stützle, T.: Very large-scale neighborhood search: Overview and case. In Blum, C., Aquilera, M.J.B., Roli, A., Sampels, M., eds.: Hybrid Metaheuristics, An Emerging Approach to Optimization. Volume 114 of Studies in Computational Intelligence. Springer (2008)
33. Thompson, P.M., Orlin, J.B.: The theory of cyclic transfers. Technical Report OR 200-89, Massachusetts Institute of Technology, Operations Research Center (1989)
34. Ahuja, R.K., Orlin, J.B., Sharma, D.: Multi-exchange neighborhood structures for the capacitated minimum spanning tree problem. Mathematical Programming **91**(1) (2001) 71–97
35. Kratica, J., Tosic, D., Filipovic, V., Ljubić, I.: Solving the simple plant location problem by genetic algorithm. RAIRO Operations Research **35** (2001) 127–142
36. Beasley, J.E.: An algorithm for the Steiner problem in graphs. Networks **14**(1) (1984) 147–159
37. Dumitrescu, I.: Constrained Path and Cycle Problems. PhD thesis, University of Melbourne (2002)