A Benders decomposition based framework for solving cable trench problems

Hatice Calik, Markus Leitner, Martin Luipersbeck

A Benders decomposition based framework for solving cable trench problems

Hatice Calik[a], Markus Leitner[b,*], Martin Luipersbeck[b]

[a]*Department of Computer Science, Université Libre de Bruxelles, Brussels, Belgium*
[b]*Department of Statistics and Operations Research, Faculty of Business, Economics and Statistics, University of Vienna, Vienna, Austria*

Abstract

In this work, we present an algorithmic framework based on Benders decomposition for the Capacitated p-Cable Trench Problem with Covering. We show that our approach can be applied to most variants of the Cable Trench Problem (CTP) that have been considered in the literature. The proposed algorithm is augmented with a stabilization procedure to accelerate the convergence of the cut loop and with a primal heuristic to derive high-quality primal solutions. Three different variants of the CTP are considered in a computational study which compares the Benders approach with two compact integer linear programming formulations that are solved with CPLEX. The obtained results show that the proposed algorithm significantly outperforms the two compact models and that it can be used to tackle significantly larger instances than previously considered algorithms based on Lagrangean relaxation.

Keywords: Location, Network Design, Benders Decomposition, Integer Linear Programming

1. Introduction

The cable trench problem which has been recently introduced by Vasko et al. [20] is a combinatorial optimization problem that combines the minimum spanning tree problem and the shortest path problem. Its objective is to connect a set of nodes of an undirected graph $G = (V, E)$ to a predefined central vertex at minimum overall costs. Thereby, fixed costs occur for each used edge in addition to edge costs that depend on the number of paths between the central vertex and any of the other nodes using an edge. While any feasible solution is a spanning tree of the input graph, including the latter costs render the problem NP-hard [20] on the one hand and also induce that a minimum spanning tree of G is usually not an optimal solution to the CTP. Vasko et al. [20] motivated the problem from an application in telecommunication network design where a set of buildings (clients) need to be connected to a central server. Fixed edge costs occur for establishing connections between the buildings (i.e., for trenching) and additional costs occur per cable that is laid in each established trench, hence the name cable trench problem.

Since then, several variants of the CTP have been considered in the literature for modeling and solving different optimization problems arising in the design of telecommunication or electricity networks, see, e.g., Marianov et al. [16, 17]. Further applications arise in medical image analysis for vascular reconstruction [12, 13]. The main additional aspects of these more general variants are (i) the consideration of multiple central servers whose locations need to be chosen as part of the optimization problem and (ii) the introduction of a covering aspect in the sense that cables do not need to be laid to each node. Instead it suffices to connect a so-called secondary server that is close to the set of clients it will supply.

*Corresponding Author.
Email addresses: hatice.calik@ulb.ac.be (Hatice Calik), markus.leitner@univie.ac.at (Markus Leitner), martin.luipersbeck@univie.ac.at (Martin Luipersbeck)

In this article, we introduce an algorithmic framework based on Benders decomposition [4] that can be used to solve most of the cable trench problems considered in the literature. To this end, we introduce and study the *Capacitated p-Cable Trench Problem with Covering* (Cp-CTPC) that generalizes previously considered problem variants. Cp-CTPC is defined on a directed graph $G = (V, A)$ where the node set V contains a set of clients J with demands $q_j \geq 0$, $\forall j \in J$, and sets of potential primary and secondary servers S and I, respectively, such that $S \subseteq I \subseteq V$. For each potential secondary server $i \in I$, parameter $Q_i \in \mathbb{N}$ indicates its maximum capacity, i.e., the maximum demand of clients from J it can supply. Distances $d_{ij} \geq 0$ are given between each potential secondary server $i \in I$ and each client $j \in J$. A client $j \in J$ can be served by secondary server $i \in I$ if the associated distance is smaller than the given maximum covering radius $r \geq 0$. Parameter $p \in \mathbb{N}$ indicates the number of primary servers that shall be installed. Each installed primary server automatically acts as a secondary server and may therefore supply clients within the coverage radius up to its maximum capacity.

Finally, trenching (installation) costs $f_{uv} \geq 0$ for using a connection from u to v and cable costs $c_{uv} \geq 0$ for installing one cable between u and v are given for each arc $(u, v) \in A$.

A feasible solution to the Cp-CTPC consists of selecting precisely p primary servers and an arbitrary number of secondary servers each of which is connected by a dedicated cable to precisely one selected primary server. Each client $j \in J$ must be supplied by an open secondary server $i \in I$ within the given covering radius, i.e., $d_{ij} \leq r$. Thus, from a topological perspective, the graph induced by such a solution is a forest consisting of p connected components. Thereby, each connected component forms an arborescence rooted at one of the chosen primary servers. For each chosen secondary server $i \in I$, let $\mathcal{P}_{si}$ denote the arc set of the directed path from the primary server s, to which secondary server i is connected. One cable dedicated to supply i needs to be placed on each arc $(u, v) \in \mathcal{P}_{si}$. Thus, the number of cables placed on each arc (u, v) corresponds to the number of such paths to secondary servers routed along (u, v). The total client demand assigned to each secondary server $i \in I$ cannot exceed its capacity Q_i. The objective is to simultaneously minimize the costs for establishing the network connections (trenching costs) and for connecting each chosen secondary server to a primary server (cable costs).

An exemplary instance and a feasible solution to this instance are given in Figure 1. In this example, each client has a demand of one and each server has a capacity of five. The primary servers (nested squares) and secondary servers (squares) are located on demand nodes and the ratios given on the figure represent the proportion of clients served within the range of each server.

Outline. In the remainder of this section, we discuss related literature and show that the Cp-CTPC generalizes most of the previously considered cable trench problems and provide more details regarding the above mentioned applications in telecommunications and medical image analysis. Two Integer Linear Programming (ILP) formulations for the Cp-CTPC based on multi- and single-commodity flows are introduced and discussed in Section 2. These are generalizations of similar formulations considered in [16, 17] for the special cases of the Cp-CTPC. The formulations by [16, 17] have been used to develop Lagrangean relaxation based optimization algorithms whereas we propose to project out the flow variables in our multi-commodity flow formulation using Benders decomposition. Our Benders decomposition approach is introduced in Section 3 where we also detail our stabilization approach in order to speed up the solution process. Section 4 contains a detailed description of our primal heuristic that will be used to derive high-quality primal solutions within the developed branch-and-cut algorithm. The results of our extensive computational study are presented and discussed in Section 5. In this section, we also give a detailed description of further implementation details of our algorithmic framework and of the considered benchmark instances. Finally, our conclusions are drawn in Section 6 where we also summarize potential future research topics.

Related works, special cases, and applications. As mentioned above, the cable trench problem has been first introduced by Vasko et al. [20] in the context of planning cost-optimal telecommunication networks. Vasko et al. assumed that there is only one primary server whose location is known and focused on the case when both the trenching and cable costs are proportional to the length of an edge. They showed that the problem is NP-hard in general and also showed that the CTP may reduce to one of the following two polynomially solvable special cases: (i) the shortest path problem when the trenching costs are equal to zero and (ii)

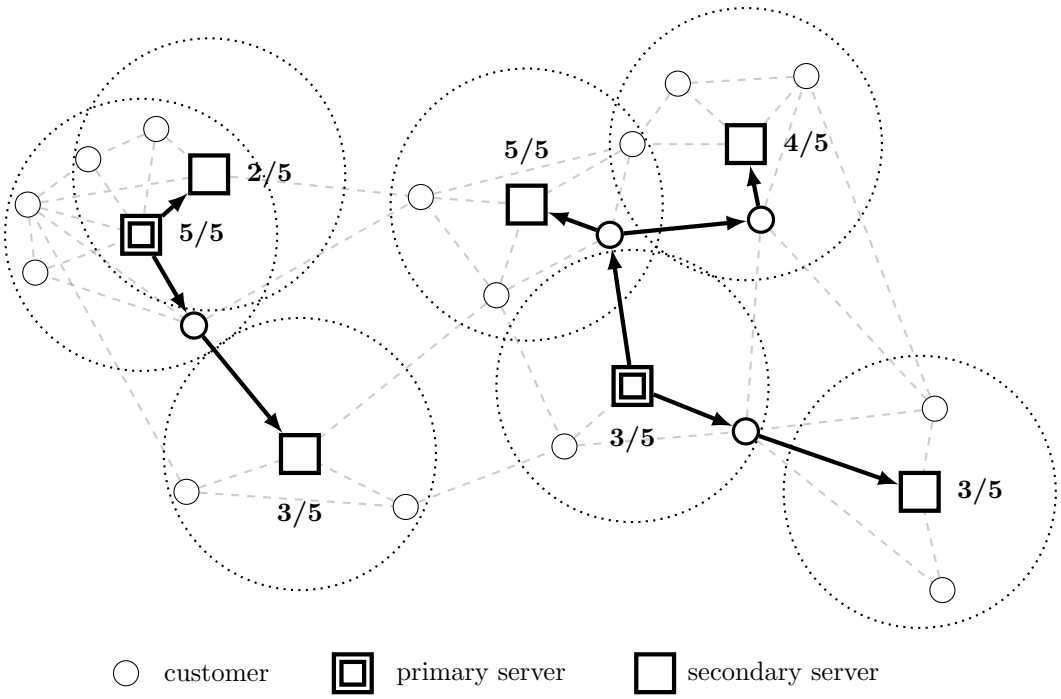

Figure 1: An example instance to the Cp-CTPC. Arcs included in a feasible solution to this instance are indicated in bold.

the minimum spanning tree problem when the cable costs are equal to zero. A compact, single-commodity flow based ILP formulation and a heuristic algorithm for solving CTP instances were proposed. Besides explaining the latter algorithm on a small numerical example, no computational study has been performed.

The CTP is obtained as the special case of the Cp-CTPC when there is only one possible primary server (i.e., $p = 1$ and $|S| = 1$), all other nodes are potential secondary servers and clients at the same time (i.e., $I = J = V \setminus S$), the covering radius is zero and all distances d_{ij} are strictly greater than zero if $i \neq j$ (i.e., a secondary server needs to be placed at each client), and all capacity constraints are non-binding (e.g., when $q_j = 1$, $\forall j \in J$, and $Q_i \geq 1$, $\forall i \in I$).

Marianov et al. [16] introduced the p-cable-trench problem (p-CTP) in which p facilities (primary servers) need to be located such that each other node of the input graph (i.e., each client) is connected to one such facility. A multi-commodity flow formulation has been introduced and subsequently used to develop two Lagrangean relaxation based approaches for the p-CTP that are based on relaxing either the flow conservation constraints or the linking inequalities between flow and arc variables. These approaches have been augmented with a two-stage Lagrangean heuristic to obtain primal solutions. A computational study has been performed on benchmark instances derived from the p-median problem sets with up to 300 nodes and for $p \in \{0.1|V|, 0.2|V|\}$ for instances obtained from the OR-library Beasley [2] and for fixed values of p (usually around $0.1|V|$) for a second instance set based on a set of discrete location problems originally proposed by Sob [1]. Optimal solutions obtained from solving the proposed ILP formulation with CPLEX have been reported for instances with at most 200 nodes. Results also show that the first Lagrangean approach based on relaxing the flow-conservation constraints seems more efficient both in terms of computing time as well as with respect to resulting optimality gaps. Marianov et al. [16] also concluded that even though the dual

bounds from this first approach are not too tight (gaps of at most 7.0%, 14.2%, and 24.1% for instances with 100, 200, and 300 nodes, respectively), the quality of the derived primal solutions seems to be quite good in general. Recently, Lalla-Ruiz et al. [14] proposed a matheuristic approach for the p-CTP and showed that the obtained upper bounds are typically better than the one reported in Marianov et al. [16]. Besides considering the concrete values of parameter p, the p-CTP is obtained as a special case of the Cp-CTPC when $S = I = J = V$ and when all other parameters are set as in the previously mentioned transformation to the CTP.

Recently, Marianov et al. [17] studied the p-cable trench problem with covering (p-CTPC). In contrast to the more general Cp-CTPC studied in this article, all nodes may be primary or secondary servers and neither customer demands nor capacity restrictions on (secondary) servers are considered. Marianov et al. [17] considered an application in network design where an antenna (secondary server) needs to be placed within the coverage radius of each client that will communicate with this antenna using a wireless communication protocol. Each antenna is then connected to a central router (primary server) using (fiber-optic) cables. The solution approach proposed is similar to the one in Marianov et al. [16]: A multi-commodity ILP formulation has been introduced which was then used to derive two Lagrangean relaxations. Primal solutions were computed using a Lagrangean heuristic and subsequent local search. In addition, an alternative heuristic based on solving a set covering formulation has been proposed. The authors applied their methods to those instances from Marianov et al. [16] that are based on the OR-library with at most 200 nodes as well as to a real world scenario. Computational results indicate that the problem variant with coverage is significantly harder to solve than the p-CTP considered in Marianov et al. [16].

The p-CTPC is the special case of the Cp-CTPC when all capacity constraints are redundant, i.e., when $Q_i \geq \sum_{j \in J: d_{ij} \leq r} q_j$, $\forall i \in I$. Consequently, in contrast to the Cp-CTPC, when modeling and solving the p-CTPC, it suffices to ensure that a secondary server is opened within the coverage range of each client, but one does not need to explicitly consider the assignments between secondary servers and their clients.

Schwarze [19] introduced the multi-commodity cable trench problem in which the trenching costs on edges are reduced if cables of multiple different utilities share the same trench. Such a cost reduction may apply if different operators coordinate their activities and hence share the trenching costs for connections that are created in joint activities. The problem was modeled as an ILP based on multi-commodity flows and several strengthening valid inequalities are proposed. A computational study has been performed to analyze the impact of the valid inequalities and the influence of parameters controlling the amount of cost sharing through jointly using trenches on instances originally proposed for the capacitated minimum spanning tree problem by Gouveia [10].

Further applications of the CTP (and its variants) can be found within the area of medical image analysis for vascular reconstruction [12, 13]. Given a set of blood vessel locations in $\mathbb{R}^3$ as well as their thickness, which both can be obtained from a computer tomography scan, the goal is to infer the structure of the associated vascular tree. Solution methods exploit physiological constraints and the assumption that the vascular system is structured such that perfusion is performed with approximately minimal effort. Jiang et al. [13] were the first to suggest modeling this problem as CTP, where trenches represent blood volume and metabolism and cables represent blood flow resistance. Cost coefficients are derived from various physical properties like length and thickness of the vessels. The availability of a fixed root location is assumed, which forms the origin of the blood flow (e.g., the heart or main artery, corresponding to the primary server). Each discrete vessel location of the resulting vascular tree is assumed to supply its surrounding tissue via capillaries (corresponding to secondary server and client). The authors applied a greedy heuristic to compute a feasible solution. Vasko et al. [21] proposed a metaheuristic approach to solve large-scale CTP instances for vascular reconstruction.

For simplification purposes, Jiang et al. [13] and Vasko et al. [21] did not consider false positives, i.e., additional blood vessel locations that appear due to measurement errors. However, both highlight their importance and recommend them to be addressed in future work. As the Cp-CTPC represents a generalization of the CTP, an interpretation in the context of the vascular reconstruction problem appears natural. Clearly, the concept of the coverage radius follows the requirement that all tissue needs to be supplied by some nearby blood vessel. Moreover, the capacity may be interpreted as the fact that each blood vessel can

effectively supply only a limited amount of its surrounding tissue. The formulation as Cp-CTPC also enables various other meaningful applications, e.g., the joint estimation of multiple, disjoint vascular systems (e.g., arterial and venous trees, as already suggested [13]) or scenarios where the root of the vascular tree is not precisely known.

In addition to the CTP variants, the Cp-CTPC studied in this article can also be considered as a generalization of the well-known p-median location problem originally defined by Hakimi [11]. The p-median problem requires the location of p facilities on a given network and allocation of customers to the selected facilities in such a way that the total distance (cable length in our context) between the customers and their facilities is minimized. The Cp-CTPC reduces to the p-median problem when capacity restrictions are not binding, locations of secondary servers are fixed, and the coverage radius as well as the construction costs are equal to zero.

2. ILP formulations

As mentioned above, Lagrangean relaxation approaches based on multi-commodity flow formulations have been proposed in [16, 17] for different variants of the cable trench problem. In this section, we introduce such a model for the Cp-CTPC. The formulation which is given by (1)–(11) considers graph $G_0 = (V_0, A_0)$ which is obtained from G by adding an artificial root node 0 and arcs from this root to all potential primary servers, i.e., $V_0 = V \cup \{0\}$, $A_0 = A \cup \{(0, s) \mid s \in S\}$. Each solution to the C$p$-CTPC will then be represented as an outgoing arborescence in G_0 with root 0 and exactly p arcs incident to 0 are chosen that will indicate the selected primary servers. Binary decision variables $x_{uv} \in \{0, 1\}$, $\forall (u, v) \in A_0$, will indicate whether arc (u, v) is included in a solution and flow variables $g^i_{uv} \in \{0, 1\}$, $\forall i \in I$, $\forall (u, v) \in A_0$, will be equal to one if and only if a secondary server is installed at node i and the path from the artificial root 0 to i contains arc (u, v). Moreover, variables $y_i \in \{0, 1\}$, $\forall i \in I$, indicate whether a secondary server is installed at node i whereas assignment variables $z_{ij} \in \{0, 1\}$, $\forall j \in J$, $\forall i \in I_j$, are equal to one if and only if client j is served by (assigned to) secondary server i. Thereby, for each client $j \in J$, notation $I_j = \{i \in I \mid d_{ij} \leq r\}$ is used to refer to all secondary servers to which j may be assigned to. In what follows, we also use notation $J_i = \{j \in J : d_{ij} \leq r\}$ for each secondary server $i \in I$, to denote the set of clients within the radius. Furthermore, for a set of nodes $W \subseteq V_0$, let $\delta^+(W) = \{(u, v) \in A_0 \mid u \in W, v \notin W\}$ and $\delta^-(W) = \{(u, v) \in A_0 \mid u \notin W, v \in W\}$ denote the outgoing and ingoing cutset, respectively. For singleton sets $W = \{u\}$ we also write $\delta^+(u)$ and $\delta^-(u)$ instead of $\delta^+(\{u\})$ and $\delta^-(\{u\})$, respectively.

$$\text{(MCF)} \qquad \min \quad \sum_{(u,v)\in A} f_{uv} x_{uv} + \sum_{i\in I} \sum_{(u,v)\in A} c_{uv} g^i_{uv} \tag{1}$$

$$\text{s.t.} \quad \sum_{(v,u)\in\delta^+(v)} g^i_{vu} - \sum_{(u,v)\in\delta^-(v)} g^i_{uv} = \begin{cases} y_i & \text{if } v = 0 \\ -y_i & \text{if } v = i \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in I,\ \forall v \in V_0 \tag{2}$$

$$g^i_{uv} \leq x_{uv} \qquad \forall i \in I,\ \forall (u, v) \in A_0 \tag{3}$$

$$\sum_{(u,v)\in A_0} x_{uv} \leq 1 \qquad \forall v \in V \tag{4}$$

$$\sum_{v\in S} x_{0v} = p \tag{5}$$

$$\sum_{i\in I_j} z_{ij} \geq 1 \qquad \forall j \in J \tag{6}$$

$$\sum_{j:i\in I_j} q_j z_{ij} \leq Q_i y_i \qquad \forall i \in I \tag{7}$$

$$g^i_{uv} \in \{0, 1\} \qquad \forall i \in I,\ \forall (u, v) \in A_0 \tag{8}$$

$$x_{uv} \in \{0,1\} \qquad \forall (u,v) \in A_0 \tag{9}$$
$$z_{ij} \in \{0,1\} \qquad \forall j \in J,\ \forall i \in I_j \tag{10}$$
$$y_i \in \{0,1\} \qquad \forall i \in I \tag{11}$$

The objective function (1) minimizes the sum of trenching and cable costs. Equations (2) are flow conservation constraints ensuring that one unit of flow is sent to each open secondary server (and therefore one cable is routed to it). Linking constraints (3) ensure that we can only install cables on arcs for which we also pay the trenching costs while indegree constraints (4) make sure that the overall solution is an arborescence in G_0 that is rooted at the artificial root 0 and therefore a forest in G. Exactly p primary servers will be selected due to equation (5). Inequalities (6) state that each client must be assigned to a secondary server while constraints (7) restrict the total demand assigned to each secondary server by its capacity limit.

It is well known that a stronger formulation is obtained by adding the strong linking constraints

$$z_{ij} \leq y_i \qquad \forall j \in J,\ \forall i \in I_j \tag{12}$$

We also add equations (13) that are obtained from the fact that every primary server is a secondary server by definition.

$$g^v_{0v} = y_v \qquad \forall v \in I \tag{13}$$

Next, we detail modifications that are necessary to apply formulation (1)–(11) to the previously mentioned special cases of the Cp-CTPC. To solve instances of the p-CTPC introduced by Marianov et al. [17], we remove assignment variables $\mathbf{z}$ as well as inequalities (6) and (7). Instead, constraints (14) are added to ensure that each client is assigned to a secondary server.

$$\sum_{i \in I_j} y_i \geq 1 \qquad \forall j \in J \tag{14}$$

No further modifications (except considering all nodes to be potential primary and secondary servers) are necessary for applying formulation (1)–(11) to the p-CTPC, the p-CTP, or the CTP, respectively.

An alternative formulation (15)–(20) with a smaller number of flow variables is obtained by using single-commodity flow variables $0 \leq g_{uv} \leq |I|$, $\forall (u,v) \in A_0$, each indicating the number of cables in the trench associated with the same arc.

$$\text{(SCF)} \qquad \min \quad \sum_{(u,v)\in A} f_{uv} x_{uv} + \sum_{(u,v)\in A} c_{uv} g_{uv} \tag{15}$$

$$\text{s.t.} \quad \sum_{(v,u)\in\delta^+(v)} g_{vu} - \sum_{(u,v)\in\delta^-(v)} g_{uv} = \begin{cases} \sum_{i\in I} y_i & \text{if } v = 0 \\ -y_v & \text{if } v \in I \\ 0 & \text{if } v \in V \setminus I \end{cases} \qquad \forall v \in V_0 \tag{16}$$

$$g_{0v} \leq (|I| - p + 1) \cdot x_{0v} \qquad \forall v \in V \tag{17}$$
$$g_{uv} \leq (|I| - p) \cdot x_{uv} \qquad \forall (u,v) \in A \tag{18}$$
$$(4) - (7), (9) - (11)$$
$$g_{uv} \in \{0, \ldots, |I| - p\} \qquad \forall (u,v) \in A_0 \tag{19}$$
$$g_{0v} \in \{0, \ldots, |I| - p + 1\} \qquad \forall v \in V \tag{20}$$

Equations (16), (17), and (18) are the flow conservation and linking constraints that have been transferred to the single-commodity case. The rest of the model corresponds to the multi-commodity flow formulation (MCF) detailed above and we therefore refrain from repeating all details.

As above, inequalities (12) can be used to obtain a stronger formulation while (14) will be used instead of (6) and (7) if the capacity constraints are not binding, i.e., to address the special cases considered earlier in the literature.

3. Benders decomposition approach (BF)

The relatively large number of flow variables in formulation (1)–(11) is problematic when attempting to solve medium or large-scale instances. Although the single-commodity flow formulation (15)–(20) uses a significantly smaller number of variables and constraints, it will typically suffer from weak linear programming (LP) relaxation bounds. Thus, too many branch-and-bound (B&B) nodes need to be enumerated for solving instances to proven optimality and this will make the formulation impractical to use on large instances. Therefore, we attempt to make use of the stronger LP bounds obtained from the multi-commodity flow formulation while avoiding the need of explicitly considering the associated flow variables. To this end, we start from formulation (1)–(11) and project out the flow variables in a Benders fashion. In contrast to the previously proposed formulations, connectivity of the solution is ensured using the arc design variables. Thus, the resulting Benders master problem is defined by (21)–(23).

$$\min \quad \sum_{(u,v)\in A} f_{uv}x_{uv} + \sum_{i\in I} w_i \tag{21}$$

$$\text{s.t.} \quad (4)-(7),(9)-(11)$$

$$\sum_{(u,v)\in\delta^-(W)} x_{uv} \geq y_i \qquad \forall W \subseteq V, i \in W \cap I \tag{22}$$

$$w_i \geq \Theta_i(y_i, \mathbf{x}) \qquad \forall i \in I \tag{23}$$

Thereby, for each secondary server $i \in I$, a new variable w_i is introduced whose value encodes the total cable costs for connecting i to its assigned primary server (in case secondary server i is installed). Cutset constraints (22) which can be interpreted as Benders feasibility cuts ensure that each secondary server is connected to the artificial root. The Benders optimality cut (23) for each secondary server $i \in I$ is obtained by solving the Benders subproblem (24)–(27) for the current values $(\bar{y}_i, \bar{\mathbf{x}})$ of variables $(y_i, \mathbf{x})$).

$$\Theta_i(\bar{y}_i, \bar{\mathbf{x}}) = \min \sum_{(u,v)\in A} c_{uv}g_{uv} \tag{24}$$

$$\text{s.t.} \sum_{(v,u)\in\delta^+(v)} g_{vu} - \sum_{(u,v)\in\delta^+(v)} g_{uv} = \begin{cases} \bar{y}_i & \text{if } v = 0 \\ -\bar{y}_i & \text{if } v = i \\ 0 & \text{if } v \in V \setminus \{i\} \end{cases} \qquad \forall v \in V_0 \tag{25}$$

$$g_{uv} \leq \bar{x}_{uv} \qquad \forall (u,v) \in A_0 \tag{26}$$

$$g_{uv} \geq 0 \qquad \forall (u,v) \in A \tag{27}$$

It is easy to observe that, the Benders subproblem is a minimum-cost flow problem with source 0 and target i where the arc costs and capacities of each arc $(u,v) \in A_0$ are set to c_{uv} and $\bar{x}_{uv}$, respectively. Our implementation solves the minimum-cost flow problems by simply solving the linear program above with CPLEX. We also note that the Benders subproblem is feasible for each $i \in I$ if the current candidate solution $(\bar{\mathbf{y}}, \bar{\mathbf{x}}, \bar{\mathbf{w}})$ satisfies all cutset constraints (22). The latter observation can be easily derived from the max-flow-min-cut theorem.

Let $\boldsymbol{\pi} \leq \mathbf{0}$ and $\boldsymbol{\rho}$ be the vectors of dual variables associated to constraints (25) and (26), respectively. Then, the dual of (24)–(27) is given by (28)–(30). Since one of the flow conservation constraints (25) is redundant, we thereby assume without loss of generality that $\rho_0 = 0$.

$$\Theta_i(\bar{y}_i, \bar{\mathbf{x}}) = \max \sum_{(u,v)\in A} \bar{x}_{uv}\pi_{uv} - \bar{y}_i\rho_i \tag{28}$$

$$\text{s.t.} \quad \rho_u - \rho_v + \pi_{uv} \leq c_{uv} \qquad \forall (u,v) \in A_0 \tag{29}$$

$$\pi_{uv} \leq 0 \qquad \forall (u,v) \in A_0 \tag{30}$$

Thus, if $\Theta_i(\bar{y}_i, \bar{\mathbf{x}}) > \bar{w}_i$, we add the Benders optimality cut

$$\sum_{(u,v)\in A} \bar{\pi}_{uv} x_{uv} - \bar{\rho}_i y_i \leq w_i$$

to the Benders master problem.

Finally, we notice that besides including the strong linking constraints (12), one can improve the LP relaxation bounds obtained by strengthening the cutset constraints (22) to

$$\sum_{(u,v)\in\delta^-(W)} x_{uv} \geq 1 \qquad \forall W \subseteq V, \exists j \in J : I_j \subseteq W \tag{31}$$

whenever W contains all possible secondary servers of at least one client.

Separation and stabilization. Earlier studies in literature reveal that in some cases it is possible to significantly improve the performance of cutting plane algorithms via careful choice of separation points [3, 8]. Such techniques can be seen as a form of stabilization akin to those applied in column generation. Our implementation includes a simple stabilization scheme similar to the in–out approach presented by Ben-Ameur and Neto [3]. Instead of generating cuts based on the optimal LP solution to the relaxed master problem, a separation point $(\mathbf{x}_{\text{sep}}, \mathbf{y}_{\text{sep}})$ in the space of $\mathbf{x}$ and $\mathbf{y}$ variables is computed as convex combination between the optimal LP solution $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ and a stabilizing interior point $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$. In our implementation, $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$ is chosen as $(1, \ldots, 1)$. The advantages of this approach are two-fold: Firstly, the generated inequalities are likely to cut off more infeasible points as the separation point is closer to the feasible region. Secondly, combination with the chosen stabilization point encourages feasibility w.r.t. constraints (22), allowing optimality cuts to be separated much earlier, potentially avoiding many time-consuming cutting-plane iterations in which only feasibility cuts can be separated.

Algorithm 1 shows a pseudocode representation of the stabilized separation procedure for the Benders formulation. Given $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{w}})$, separation is performed for multiple iterations with different separation points $(\mathbf{x}_{\text{sep}}, \mathbf{y}_{\text{sep}})$. The parameter λ_k determines the closeness of the separation point to the optimal point at each iteration. For the first iteration, we chose $\lambda_0 = 0.5$. In each subsequent iteration, λ_k is moved closer to one by midpoint bisection of the interval $[\lambda_k, 1]$. Optimality cuts are only separated if the current $(\mathbf{x}_{\text{sep}}, \mathbf{y}_{\text{sep}})$ yields no feasibility cuts. The number of iterations is limited by a cut limit $\bar{c}$ and an iteration limit $\bar{k}$. We set $\bar{c} = 0.1|I|$ and $\bar{k} = 5$. If the number of generated cuts does not exceed the cut limit $\bar{c}$ after $\bar{k}$ iterations, we perform separation for $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$, but still add $\varepsilon \times (1, \ldots, 1)$, $\varepsilon = 10^{-6}$, to the separation point. It has been observed that this strategy significantly increases the strength of the separated optimality cuts. A similar technique is applied when separating feasibility cuts, which is done by applying the maximum flow algorithm of Cherkassky and Goldberg [5]. It is well known that in this case adding a small value ε to all arc capacities yields connectivity cuts that are of minimum cardinality.

For both feasibility and optimality cuts, separation is only performed for nodes $i \in V$ where $\bar{y}_i \geq 0.1$. Note that when cuts are generated from alternative separation points during the stabilization loop (Steps 1–9), violation is still checked w.r.t. the original point $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{w}})$, as a feasible separation point might still

Data: An optimal LP solution $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{w}})$.
Result: A set of violated inequalities C.

$C \leftarrow \emptyset$
$\lambda_0 \leftarrow 0.5$
while $k < \bar{k} \wedge |C| < \bar{c}$ **do**
  $(\mathbf{x}_{\text{sep}}, \mathbf{y}_{\text{sep}}) \leftarrow \lambda_k (\bar{\mathbf{x}}, \bar{\mathbf{y}}) + (1 - \lambda_k) (\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$
  $C_f \leftarrow$ `separateFeasibilityCuts`$(\mathbf{x}_{\text{sep}}, \mathbf{y}_{\text{sep}})$
  if $C_f = \emptyset$ **then**
    $C_o \leftarrow$ `separateOptimalityCuts`$(\mathbf{x}_{\text{sep}}, \mathbf{y}_{\text{sep}})$
  end
  $C \leftarrow C \cup C_f \cup C_o$
  $\lambda_{k+1} \leftarrow (\lambda_k + 1)/2$
  $k \leftarrow k + 1$
end
if $|C| < \bar{c}$ **then**
  $(\mathbf{x}_{\text{sep}}, \mathbf{y}_{\text{sep}}) \leftarrow (\bar{\mathbf{x}}, \bar{\mathbf{y}}) + \varepsilon \cdot (1, \ldots, 1)$
  $C_f \leftarrow$ `separateFeasibilityCuts`$(\mathbf{x}_{\text{sep}}, \mathbf{y}_{\text{sep}})$
  if $C_f = \emptyset$ **then**
    $C_o \leftarrow$ `separateOptimalityCuts`$(\mathbf{x}_{\text{sep}}, \mathbf{y}_{\text{sep}})$
  end
  $C \leftarrow C \cup C_f \cup C_o$
end

Algorithm 1: Stabilized separation procedure for the Benders approach.

produce a violated cut. Finally, since in some cases the stabilization loop is very time consuming, we only execute the loop every fifth iteration, which in preliminary experiments has been found a good trade-off between decreasing the number of cutting plane iterations until convergence and the extra time spent.

The importance of avoiding naive separation with $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ as separation point has to be stressed. As an example, consider an instance of the p-CTP with $|V| = 100$, $|A| = 400$, $p = 5$ and $r = 0$ (details on the used benchmark instances are given in Section 5). If no ε is added during the separation of Benders cuts, our implementation requires 166 seconds to solve the instance to optimality. Adding ε decreases the solution time to 12 seconds, while applying the stabilization further decreases it to 4 seconds.

4. Primal heuristic

Augmenting a branch-and-bound procedure by a primal heuristic is often crucial to find good integer-feasible solutions early on during the search process and therefore effectively prune nodes of the branch-and-bound tree. This holds especially true for the Benders decomposition approach, where the simple structure of the master problem does not provide sufficient information to the otherwise powerful LP-based heuristics of a modern ILP solver such as CPLEX.

Algorithm 2 shows the pseudocode of our primal heuristic which constructs a feasible Cp-CTPC solution from an LP solution with fractional variable values. It uses only the values of the trench variables $\mathbf{x}$ from the LP solution and can therefore be applied directly to all of the formulations discussed in the previous sections.

In the heuristic, a solution is represented as triple $(S', I', A'^{|I'|})$, consisting of the sets of selected primary servers $S' \subseteq S$, selected secondary servers $I' \subseteq I$ and arcs $A'^{|I'|} \subseteq A^{|I|}$ used to connect each secondary server. The latter can also be interpreted as the set of cables connecting some possibly active secondary server $i \in I$.

The heuristic consists of two phases. In the first phase (Step 4 of Algorithm 2) p primary servers S' are selected based on the highest LP solution values of their associated artificial root arcs. Ties are broken

arbitrarily. The second phase consists of Steps 6–17 of Algorithm 2. Thereby, at each iteration, the partial solution is augmented by one secondary server from $I^* = I \setminus I'$. Besides the (partial) solution $(S', I', A'^{|I|})$, the trenching costs $\mathbf{f}'$ of the current solution and the set of uncovered clients $J^* \subseteq J$ is updated at each iteration.

Firstly, Step 7 computes combined arc costs $\mathbf{c}'$ based on the current trenching costs $\mathbf{f}'$ and cable costs $\mathbf{c}$, scaled by the given LP solution $\bar{\mathbf{x}}$. In Step 8, for each $i \in I^*$ the shortest path P_i from any $s \in S'$ to i is computed based on $\mathbf{c}'$. Note that this step can be performed efficiently by a single execution of Dijkstra's algorithm [7], using the artificial root node $0 \in V_0$ as source and with arcs $(0, s) \in A_0, s \in S \setminus S'$ temporarily hidden. For each $i \in I^*$, $c'(P_i)$ denotes the path's length based on $\mathbf{c}'$. In Step 9, for each unselected secondary server $i \in I^*$, the largest set of currently uncovered clients $J'_i \subseteq J_i \cap J^*$ that i can cover without exceeding its capacity Q_i is computed. This set is obtained by simply assigning uncovered clients to i in ascending order w.r.t. q_i.

Next, a score σ_i is computed for each $i \in I^*$, such that $\sigma_i := c'(P_i)/|J'_i|$ for $J'_i \neq \emptyset$, and $\sigma_i := \infty$ otherwise. The secondary server $i \in I^*$ with the smallest score is added to the solution. Finally, all data is updated according to this selection. The procedure is repeated until the solution covers all clients. Note that the number of iterations performed by the main loop is bounded by $|I|$. Therefore, if total coverage is not achieved after $|I|$ iterations, the instance is clearly infeasible. The worst-case time complexity of Algorithm 2 is thus $O(|I| \cdot (|A| + |V| \log |V|))$.

Data: An Instance of the Cp-CTPC and trenching-part $\bar{\mathbf{x}}$ of LP solution.
Result: A feasible solution $(S', I', A'^{|I|})$ to the Cp-CTPC.

1 $(S', I', A'^{|I|}) \leftarrow (\emptyset, \emptyset, \emptyset^{|I|})$
2 $J^* \leftarrow J,\ I^* \leftarrow I$
3 $\mathbf{f}' \leftarrow \mathbf{f}$
4 $S' \leftarrow$ `selectPrimaryServers`$(\bar{\mathbf{x}})$
5 $I^* \leftarrow I^* \setminus S'$
6 **while** $J^* \neq \emptyset$ **do**
7 $\quad c'_{uv} \leftarrow (c_{uv} + f'_{uv}) \cdot (1 - \bar{x}_{uv}) \qquad \forall (u, v) \in A$
8 $\quad P_i \leftarrow$ `computeShortestPath`$(S', i, \mathbf{c}') \qquad \forall i \in I^*$
9 $\quad J'_i \leftarrow$ `computeMaximumCoverage`$(i, J^*, J_i, Q_i) \qquad \forall i \in I^*$
10 $\quad \sigma_i \leftarrow$ `computeScore`$(i, J'_i, P_i) \qquad \forall i \in I^*$
11 $\quad i' \leftarrow$ `selectSecondaryServer`$(I^*, \boldsymbol{\sigma})$
12 $\quad J^* \leftarrow J^* \setminus J'_{i'}$
13 $\quad I^* \leftarrow I^* \setminus \{i'\}$
14 $\quad I' \leftarrow I' \cup \{i'\}$
15 $\quad A'_{i'} \leftarrow P_{i'}$
16 $\quad f'_{uv} \leftarrow 0 \qquad \forall (u, v) \in P_{i'}$
17 **end**

Algorithm 2: Primal heuristic.

5. Computational study

In this section, we first detail the benchmark instances used to test our algorithms in Section 5.1. Subsequently, we report and discuss the results that we obtain from computational experiments for solving different variants of the Cp-CTPC. All formulations and algorithms introduced in the previous sections have been implemented in C++ and compiled with GCC 4.8.3. The implementation uses the OGDF library [6] for data structures and CPLEX 12.6.3 for solving the mathematical formulations including the Benders subproblems, i.e., the minimum-cost flow problems. All experiments have been performed single-threaded

on an Intel Xeon CPU with 2.5 GHz. A time limit of two hours and memory limit of 5 GB have been set. When computing average running times, those instances that cannot be solved due to the memory limit are treated the same way as the instances that hit the time limit, that is, the running time is counted as two hours.

We performed experiments for five different values of p ($p \in \{5, 10, |V|/10, |V|/5, |V|/3\}$) for each instance in our benchmark set. In each of the tables of this section, every row reports average results over a subset of instances. By default, instances with the same number of nodes but which are based on different original graphs are grouped together. Thereby, column $|A|$ gives the corresponding average number of arcs (note that due to the random instance generation in [2], the number of arcs per graph slightly varies between the instances with the same number of nodes). Additional criteria for grouping instances are chosen based on the results reported in each table. Column *#inst.* lists the number of instances grouped together. For each of our methods MCF, BF, and SCF, column *#solv.* reports the number of instances solved, $g[\%]$ gives the average relative optimality gap (computed as $(UB - LB)/UB$, where LB and UB denote the best lower and upper bound values, respectively), and $t[s.]$ lists the average running time in seconds. If the time limit is exceeded for all instances of one group, the average running time is replaced by *TL*. Conversely, a "-" in all three columns is used to indicate that all the root LP relaxation of all instances of one group could not be solved due to the memory or time limit. The results of the best method are marked in bold for each row.

5.1. Data generation

To test the performance of our methods, we created a set of benchmark instances that are based on the uncapacitated p-median data from OR-Library [2] and which are created in a similar way as in Marianov et al. [17]. The data set contains 40 instances with $|V|$ varying between 100 and 900, and p varying between 5 and $|V|/3$. For each instance, we are given a set of arcs and their lengths. As suggested by [2], we refine the original data to eliminate multiple arcs between the same ordered pair of nodes and calculate the shortest path distance value between each node pair i, j by using the all pairs shortest path algorithm of Floyd [9]. The distance d_{ij} is obtained for each node pair i, j by multiplying this shortest path value by 0.75 and rounding it up to the nearest integer. For every arc $(u, v) \in A$, cable costs c_{uv} are generated uniformly at random from the interval $[1, \lceil 0.75 l_{uv} \rceil]$ where l_{uv} is the length of the arc. Demand and capacity values are obtained in a similar fashion as in Lorena and Senne [15]. Demand values of each node are generated uniformly at random in the interval $[1, 20]$ and the capacity values are set to $\lceil \frac{\sum_{i \in V} q_i}{M \times \alpha} \rceil$ for $\alpha \in \{0.8, 0.9\}$ and $M \in \{0.05|V|, 0.1|V|, 0.2|V|\}$. As the original instances do not distinguish different node types, we set $I = J = S = V$.

Finally, we need to identify meaningful radius values r to impose in our computational experiments. To this end, we first observe that an upper bound r_{pC} for reasonable values of r is obtained by solving an uncapacitated p-center problem on the network with primary servers being the potential centers. If we would consider r_{pC} as the radius value to solve the p-CTPC, an optimal solution would use primary servers only. Therefore, we consider smaller radius values in our experiments, more precisely, we use $r = \beta \times r_{pC}$ for $\beta \in \{0.1, 0.2, 0.3\}$. In order to solve the p-center problem, we use the classical set covering formulation (SC_r) given by (32)–(34) and follow a search methodology similar to the one proposed by Minieka [18] to select the radius value at each step. More specifically, we start solving SC_r for $r = 1$, increment the r value by one if $Z_r > p$ and repeat solving SC_r with incremented values of r until we get $Z_r \leq p$ which will be satisfied for the first time by $r = r_{pC}$.

$$(SC_r) \qquad Z_r = \min \quad \sum_{i \in I} y_i \tag{32}$$

$$\text{s.t.} \sum_{i \in I_j} y_i \geq 1 \qquad \forall j \in J \tag{33}$$

$$y_i \in \{0, 1\} \qquad \forall i \in I \tag{34}$$

Assuming a radius coverage value of r for a Cp-CTPC problem instance, we observe that a lower bound of the number of necessary secondary servers can be obtained by using any lower bound to Z_r. Let L be a

lower bound on Z_r. We conclude that at least $L - p$ secondary servers that are not primary servers need to be installed and thus, cables must be placed on at least $L - p$ arcs. Thus, the initial formulations to solve the Cp-CTPC can be tightened by introducing the corresponding constraint (35).

$$\sum_{i \in I \setminus S} y_i \geq L - p \tag{35}$$

5.2. Results on the p-CTP

We first apply our methods to the p-CTP, i.e., we consider instances with $r = 0$ and for which all capacity constraints are redundant. As mentioned in Section 2 we also remove assignment variables $\mathbf{z}$ when solving the p-CTP.

An overview on numbers of solved instances and remaining optimality gaps after reaching the time- or memory limit is given in Figure 2. We observe that our Benders decomposition algorithm BF clearly outperforms the two considered alternatives, i.e., solving the MCF or SCF formulation with CPLEX. BF manages to solve approximately 75% of the considered instances and the remaining optimality gaps are typically smaller than 20% for the unsolved instances. In contrast, MCF could only solve slightly less than 40% of the instances and fails to derive reasonable gaps for all other instances. SCF cannot solve any instances to proven optimality due to its weak LP bounds and the resulting huge number of branch-and-bound nodes that need to be considered. While it outperforms MCF for the largest and most difficult instances considered, its performance cannot compete with BF.

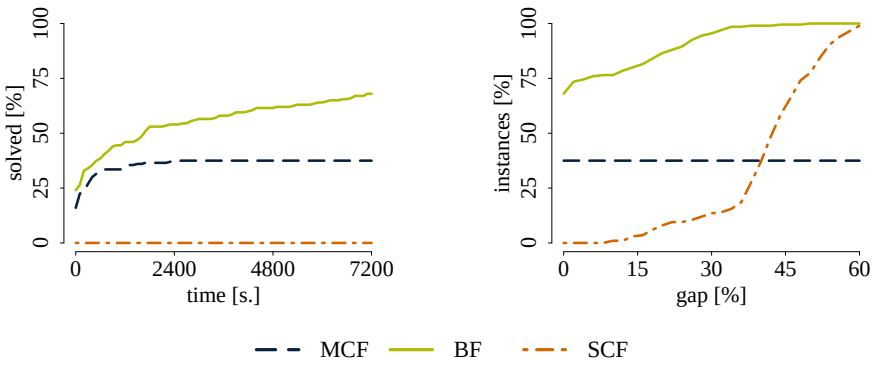

Figure 2: Graphical summary of test results on the p-CTP. Results show the relative numbers of instances solved within a certain time and the relative numbers of instances for which the final optimality gap is below a certain threshold.

Detailed results are reported in Table 1. We observe that MCF is able to solve all instances with at most 300 nodes while its root relaxation cannot be solved for any instance with at least 400 nodes (due to the given memory limit). BF converges significantly faster than MCF on instances with $|V| \leq 300$ and is able to solve several significantly larger instances (up to 700 nodes). We also observe that in general the computational difficulty decreases as p increases for all considered methods. For instance, BF has difficulties in solving problems with $p \in \{5, 10\}$ and $|V| \geq 400$ while it is able solve all the problems with up to 700 nodes optimally for $p \in \{|V|/5, |V|/3\}$.

Recall that the Lagrangean relaxation based method by Marianov et al. [16] has been tested on similar instances with at most 300 nodes and for $p \in \{|V|/10, |V|/5, |V|/3\}$. While differences in instance creation do not allow a direct comparison to our methods, the fact that the final optimality gaps obtained by their method typically exceed 10% for $|V| = 300$ and $p \in \{|V|/5, |V|/3\}$ demonstrated the high potential for solving p-CTP instances of the Benders decomposition algorithm developed in this article.

Finally, we observe that average gaps computed by SCF are significantly larger than those of BF except for the instances with $|V| = 900$ and $p = 300$.

5.3. Results on the p-CTPC

In the second part of our computational study, we conduct experiments on our methods to solve the p-CTP with coverage introduced by Marianov et al. [17]. Obtained results are summarized in Figure 3 and in Tables 2 and 3. To keep their size reasonable, we give only results for $p = \{5, |V|/5, |V|/3\}$ in these tables.

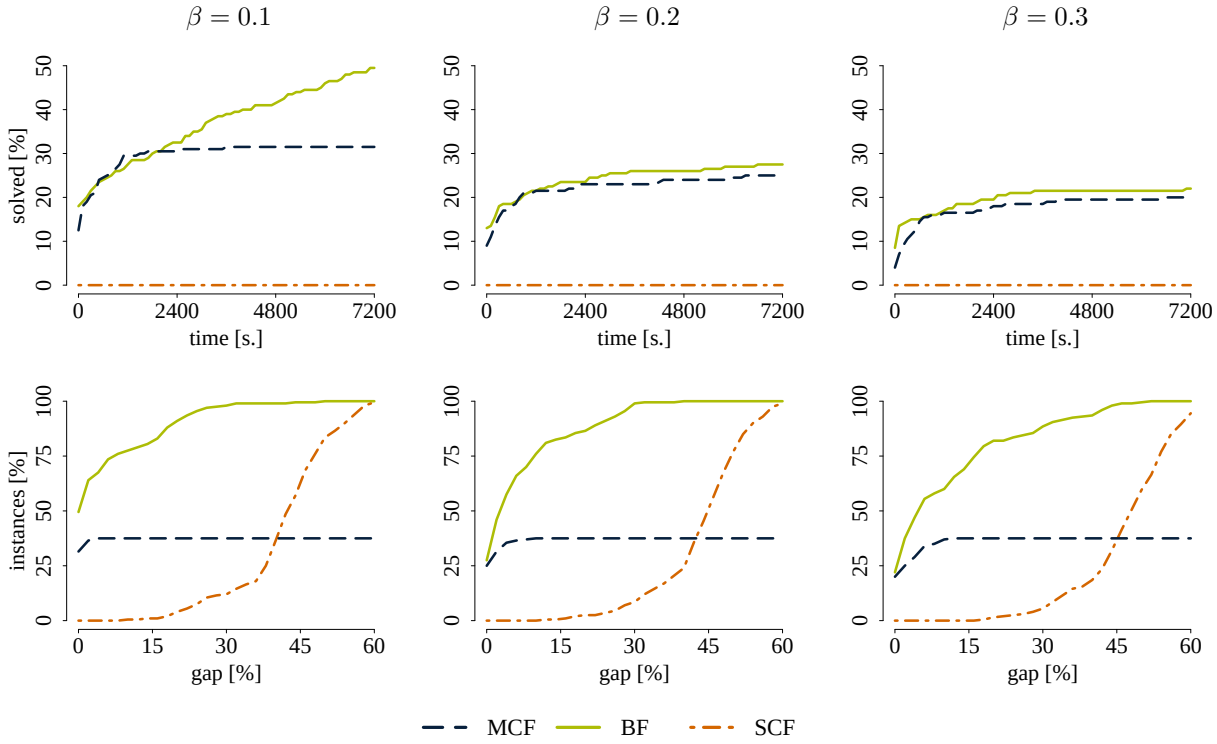

Figure 3: Graphical summary of test results on the p-CTPC for different values of β. Results show the relative numbers of instances solved within a certain time and the relative numbers of instances for which the final optimality gap is below a certain threshold.

From Figure 3, we first observe that BF outperforms both MCF and SCF for the considered p-CTPC instances. Considering the numbers of solved instances, the difference between BF and MCF does not seem to be as large as for the p-CTP, in particular for $\beta \in \{0.2, 0.3\}$. For instances that could not be solved to optimality, the typical gaps obtained from BF are, however, typically significantly smaller than of MCF. We also conclude that the computational difficulty seems to increase with increasing coverage radii, i.e., with increasing values of β. Since the fraction of instances solved within the given time and memory limits is much smaller than for the p-CTP, we also conclude that the computational difficulty increases considerably by including the coverage restrictions.

These observations are confirmed by the results given in Tables 2 and 3. The results verify that BF scales generally much better w.r.t. the radius size than MCF on problems with $|V| = 100$ to 200. For $|V| = 300$, MCF outperforms BF for large values of p ($p \in |V|/5, |V|/3\}$) while BF performs better for $p = 5$. Similar to our observations when solving the p-CTP instances, MCF hits the memory limit for the p-CTPC instances with at least 400 nodes and SCF cannot solve any of the problems optimally within the time limit. Moreover, BF outperforms SCF in all the instances except the ones with 900 nodes, $p = 300$, and $\beta = 0.1$ on the average. Finally, we observe that BF can solve all instances with $p = |V|/3$ and $\beta = 0.1$ with at most 700 nodes.

5.4. Results on the Cp-CTPC

Finally, we report the results of our computational experiments for solving Cp-CTPC instances in Figure 4 as well as Tables 4 and 5. We consider instances with different capacity levels of secondary servers represented by M and α combinations. Table 4 lists aggregated results for fixed values of p, i.e., $p = \{5, 10\}$, while Table 5 lists aggregated results for values of p that are defined as percentage of $|V|$, i.e., $p = \{|V|/10, |V|/5, |V|/3\}$. In both tables, results have been averaged over all considered radius sizes.

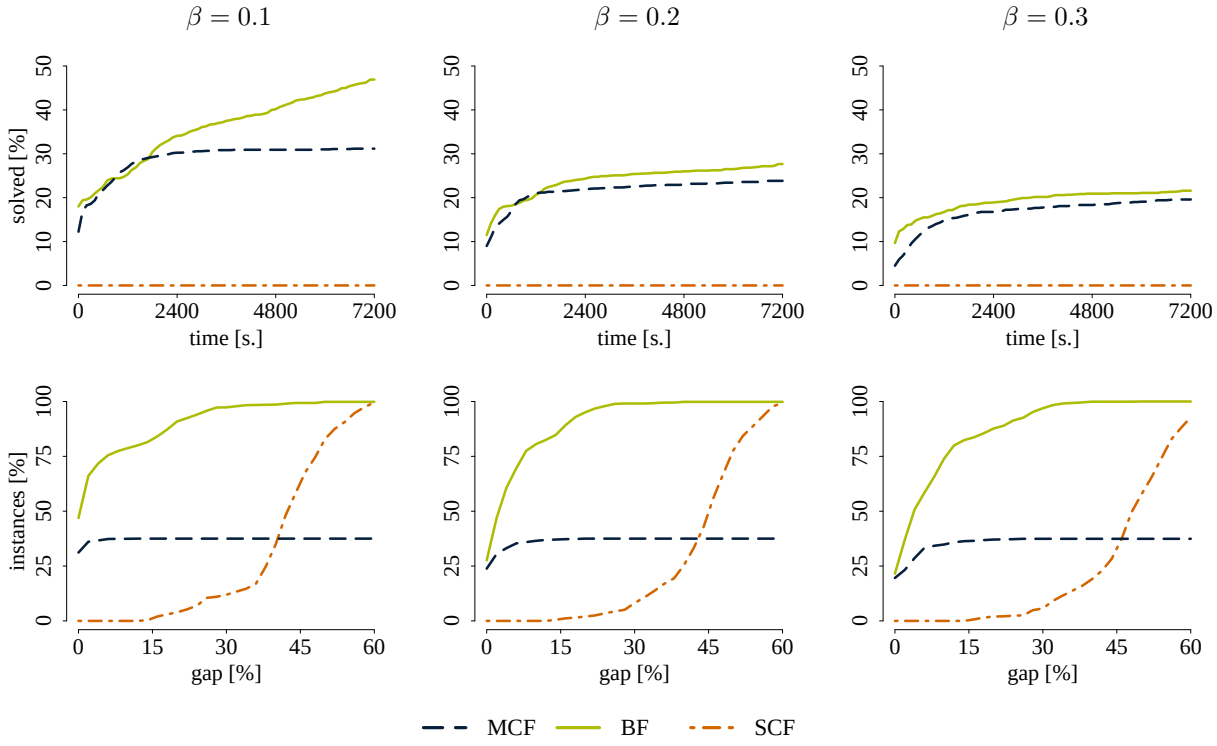

Figure 4: Graphical summary of test results on the Cp-CTPC for different values of β. Results show the relative numbers of instances solved within a certain time and the relative numbers of instances for which the final optimality gap is below a certain threshold.

Figure 4 shows similar trends as for the previously discussed case without capacity constraints, i.e., the p-CTPC. BF slightly outperforms MCF with respect to the numbers of solved instances, and clearly outperforms MCF when considering the remaining optimality gaps. Again the difficulty of instances increases with increasing size of the coverage radius.

From Tables 4 and 5, we observe that the root node relaxation of MCF cannot be solved for instances with more than 300 nodes due to the memory limit and that SCF cannot solve any instance to optimality. These conclusions hold for all three problem variants considered in our computational study. Table 4 reveals that BF solves more instances with $|V| \leq 300$ to proven optimality than MCF and almost always terminates with smaller average gap values than MCF. On the other hand, from Table 5, we observe that MCF solves a larger number of instances with $|V| = 300$ than BF when the number of primary servers is large. Nevertheless, BF still performs better with respect to the average gap values. Thus, we conclude that BF dominates the other approaches for the fixed, small values of p and also for relative, large values of p up to $|V| = 200$. For the second case, it is not clear whether MCF or BF achieves an overall better performance. BF is, however, the only considered method that can be reasonably applied to instances with more than 300 nodes. We note that even BF is not able to solve instances with $p \in \{5, 10\}$ and $|V| \geq 400$ while it could solve between 9% and 100% of the instances with up to 700 nodes for larger p values. Moreover, we also realize that instances

with smaller capacities (i.e., larger M and α) tend to be solved faster.

6. Conclusions

This work considers the Capacitated p-Cable Trench Problem with Covering (Cp-CTPC) that generalizes most variants of the Cable Trench Problem (CTP) that have been considered in the literature. An algorithmic framework based on Benders decomposition for solving the Cp-CTPC is presented. This framework contains a stabilization procedure to speed up the convergence of the cutting plane loop and a primal heuristic to compute high-quality primal solutions. Simple modifications that allow to apply the framework to most previously considered variants of the CTP are detailed. Two flow-based integer programming formulations are introduced as alternatives to the Benders framework.

An extensive computational study is performed in which the Cp-CTPC and two further variants of the CTP are considered. Results show that the proposed framework clearly outperforms the two alternatives given by directly solving the flow-based formulations. It is also shown that the framework allows to tackle and solve significantly larger instances than those previously considered.

This study also opens several directions for future research. In particular, it would be interesting to consider a formulation of the Cp-CTPC in which one variable for each possible assignment between a secondary server and its clients is considered instead of the standard way to model the capacity constraints used in this work. Such a model would clearly dominate the current formulation from the theoretical perspective and it would be utile to analyze its practical performance by developing a corresponding Branch-Price-and-Benders-Cut approach. Finally, from an application point of view it could be relevant to consider a problem variant that additionally considers capacity constraints on the arcs.

Acknowledgements

This work is supported by the Vienna Science and Technology Fund (WWTF) through project ICT15-014. Part of this research has been performed while M. Leitner was a research fellow at the Department of Computer Science, Université Libre de Bruxelles (Brussels, Belgium) where he was supported by the Interuniversity Attraction Poles P7/36 COMEX initiated by the Belgian Science Policy Office. The research of H. Calik is supported by the e4-share (Models for Ecological, Economical, Efficient, Electric Car-Sharing) project funded by Innoviris, the Brussels Institute for Research and Innovation, via JPI Urban Europe and COMEX. M. Luipersbeck is supported by the University of Vienna through the uni:docs fellowship programme. These supports are greatly acknowledged.

Table 1: Test results of solving the p-CTP. Reported values are averaged and aggregated by $|V|$.

				MCF			BF			SCF		
$\|V\|$	$\|A\|$	#inst.	p	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]
100	396	5	5	5	0.00	7	5	**0.00**	**5**	0	25.76	*TL*
100	396	5	10	5	0.00	5	5	**0.00**	**3**	0	20.30	*TL*
100	396	5	10	5	0.00	6	5	**0.00**	**3**	0	20.40	*TL*
100	396	5	20	5	0.00	5	5	**0.00**	**2**	0	15.21	*TL*
100	396	5	33	5	0.00	6	5	**0.00**	**2**	0	12.46	*TL*
200	1585	5	5	5	0.00	385	5	**0.00**	**81**	0	44.54	*TL*
200	1585	5	10	5	0.00	187	5	**0.00**	**36**	0	41.82	*TL*
200	1585	5	20	5	0.00	108	5	**0.00**	**23**	0	38.92	*TL*
200	1585	5	40	5	0.00	99	5	**0.00**	**18**	0	36.80	*TL*
200	1585	5	66	5	0.00	99	5	**0.00**	**16**	0	31.30	*TL*
300	3561	5	5	5	0.00	1631	5	**0.00**	**545**	0	43.75	*TL*
300	3561	5	10	5	0.00	1318	5	**0.00**	**410**	0	43.05	*TL*
300	3561	5	30	5	0.00	533	5	**0.00**	**227**	0	40.80	*TL*
300	3561	5	60	5	0.00	392	5	**0.00**	**192**	0	37.75	*TL*
300	3561	5	100	5	0.00	398	5	**0.00**	**253**	0	37.23	*TL*
400	6338	5	5	-	-	-	3	**0.06**	**5305**	0	50.13	*TL*
400	6338	5	10	-	-	-	4	**0.28**	**3204**	0	46.71	*TL*
400	6338	5	40	-	-	-	5	**0.00**	**1255**	0	41.77	*TL*
400	6338	5	80	-	-	-	5	**0.00**	**749**	0	39.60	*TL*
400	6338	5	133	-	-	-	5	**0.00**	**759**	0	39.29	*TL*
500	9900	5	5	-	-	-	0	**9.68**	***TL***	0	51.09	*TL*
500	9900	5	10	-	-	-	1	**1.75**	**7007**	0	50.48	*TL*
500	9900	5	50	-	-	-	5	**0.00**	**2433**	0	41.91	*TL*
500	9900	5	100	-	-	-	5	**0.00**	**1346**	0	41.62	*TL*
500	9900	5	166	-	-	-	5	**0.00**	**1909**	0	40.86	*TL*
600	14263	5	5	-	-	-	0	**19.38**	***TL***	0	52.02	*TL*
600	14263	5	10	-	-	-	0	**13.30**	***TL***	0	51.67	*TL*
600	14263	5	60	-	-	-	3	**0.04**	**6023**	0	46.74	*TL*
600	14263	5	120	-	-	-	5	**0.00**	**3102**	0	44.97	*TL*
600	14263	5	200	-	-	-	5	**0.00**	**3993**	0	39.83	*TL*
700	19394	4	5	-	-	-	0	**25.00**	***TL***	0	57.39	*TL*
700	19394	4	10	-	-	-	0	**22.89**	***TL***	0	58.01	*TL*
700	19394	4	70	-	-	-	2	**7.88**	**6800**	0	49.43	*TL*
700	19394	4	140	-	-	-	4	**0.00**	**4929**	0	44.76	*TL*
700	19394	4	233	-	-	-	4	**0.00**	**6458**	0	38.42	*TL*
800	25345	3	5	-	-	-	0	**30.12**	***TL***	0	56.33	*TL*
800	25345	3	10	-	-	-	0	**23.77**	***TL***	0	56.38	*TL*
800	25345	3	80	-	-	-	0	**16.32**	***TL***	0	50.95	*TL*
800	25345	3	160	-	-	-	0	**12.65**	***TL***	0	43.86	*TL*
800	25345	3	266	-	-	-	0	**2.40**	***TL***	0	42.10	*TL*
900	32103	3	5	-	-	-	0	**29.59**	***TL***	0	57.40	*TL*
900	32103	3	10	-	-	-	0	**27.76**	***TL***	0	53.89	*TL*
900	32103	3	90	-	-	-	0	**14.84**	***TL***	0	50.46	*TL*
900	32103	3	180	-	-	-	0	**21.53**	***TL***	0	43.80	*TL*
900	32103	3	300	-	-	-	0	43.22	*TL*	0	**34.59**	***TL***

Table 2: Test results of solving the p-CTPC ($|V| = 100 - 500$). Reported values are averaged and aggregated by $|V|$.

					MCF			BF			SCF		
$\|V\|$	$\|A\|$	#inst.	p	β	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]
100	396	5	5	0.1	5	0.00	52	5	**0.00**	**34**	0	28.71	*TL*
100	396	5	5	0.2	5	0.00	1183	5	**0.00**	**137**	0	32.91	*TL*
100	396	5	5	0.3	3	0.31	3264	5	**0.00**	**502**	0	33.73	*TL*
100	396	5	10	0.1	5	0.00	20	5	**0.00**	**18**	0	23.28	*TL*
100	396	5	10	0.2	5	0.00	120	5	**0.00**	**38**	0	29.62	*TL*
100	396	5	10	0.3	5	0.00	243	5	**0.00**	**127**	0	31.51	*TL*
100	396	5	33	0.1	5	0.00	8	5	**0.00**	**5**	0	15.50	*TL*
100	396	5	33	0.2	5	0.00	16	5	**0.00**	**15**	0	17.42	*TL*
100	396	5	33	0.3	5	0.00	26	5	**0.00**	**25**	0	21.58	*TL*
200	1585	5	5	0.1	2	0.89	4787	3	**0.29**	**4101**	0	43.24	*TL*
200	1585	5	5	0.2	1	3.37	6601	2	**1.27**	**4837**	0	46.50	*TL*
200	1585	5	5	0.3	0	6.76	*TL*	1	**3.30**	**6274**	0	48.71	*TL*
200	1585	5	20	0.1	5	0.00	519	5	**0.00**	**407**	0	39.08	*TL*
200	1585	5	20	0.2	4	0.22	2073	4	**0.16**	**1682**	0	41.40	*TL*
200	1585	5	20	0.3	2	0.79	6181	4	**0.22**	**2620**	0	44.14	*TL*
200	1585	5	66	0.1	5	0.00	122	5	**0.00**	**31**	0	30.74	*TL*
200	1585	5	66	0.2	5	**0.00**	**255**	5	0.00	334	0	32.08	*TL*
200	1585	5	66	0.3	5	**0.00**	**400**	5	0.00	609	0	34.49	*TL*
300	3561	5	5	0.1	0	1.29	*TL*	2	**0.59**	**6317**	0	43.01	*TL*
300	3561	5	5	0.2	0	3.60	*TL*	1	**1.70**	**6267**	0	45.62	*TL*
300	3561	5	5	0.3	0	**5.30**	***TL***	0	7.20	*TL*	0	47.41	*TL*
300	3561	5	30	0.1	5	**0.00**	**1087**	5	0.00	1959	0	40.71	*TL*
300	3561	5	30	0.2	1	0.47	6976	1	0.66	6480	0	42.45	*TL*
300	3561	5	30	0.3	0	2.69	*TL*	0	**1.08**	***TL***	0	45.96	*TL*
300	3561	5	100	0.1	5	**0.00**	**477**	5	0.00	1260	0	36.80	*TL*
300	3561	5	100	0.2	5	**0.00**	**746**	5	0.00	2458	0	36.50	*TL*
300	3561	5	100	0.3	5	**0.00**	**2693**	1	0.56	7195	0	38.80	*TL*
400	6338	5	5	0.1	-	-	-	0	**5.83**	***TL***	0	47.08	*TL*
400	6338	5	5	0.2	-	-	-	0	**10.23**	***TL***	0	49.34	*TL*
400	6338	5	5	0.3	-	-	-	0	**18.70**	***TL***	0	54.32	*TL*
400	6338	5	40	0.1	-	-	-	3	**0.23**	**5229**	0	42.14	*TL*
400	6338	5	40	0.2	-	-	-	0	**4.86**	***TL***	0	44.97	*TL*
400	6338	5	40	0.3	-	-	-	0	**9.55**	***TL***	0	47.16	*TL*
400	6338	5	133	0.1	-	-	-	5	**0.00**	**3864**	0	39.89	*TL*
400	6338	5	133	0.2	-	-	-	1	**0.31**	**6929**	0	41.45	*TL*
400	6338	5	133	0.3	-	-	-	0	**1.60**	***TL***	0	44.33	*TL*
500	9900	5	5	0.1	-	-	-	0	**8.44**	***TL***	0	49.87	*TL*
500	9900	5	5	0.2	-	-	-	0	**13.27**	***TL***	0	52.79	*TL*
500	9900	5	5	0.3	-	-	-	0	**27.37**	***TL***	0	55.69	*TL*
500	9900	5	50	0.1	-	-	-	2	**0.22**	**6612**	0	42.30	*TL*
500	9900	5	50	0.2	-	-	-	0	**6.64**	***TL***	0	45.44	*TL*
500	9900	5	50	0.3	-	-	-	0	**12.04**	***TL***	0	48.28	*TL*
500	9900	5	166	0.1	-	-	-	5	**0.00**	**1905**	0	40.87	*TL*
500	9900	5	166	0.2	-	-	-	0	**1.00**	***TL***	0	41.84	*TL*
500	9900	5	166	0.3	-	-	-	0	**2.31**	***TL***	0	45.73	*TL*

Table 3: Test results of solving the p-CTPC ($|V| = 600 - 900$). Reported values are averaged and aggregated by $|V|$.

					MCF			BF			SCF		
$\|V\|$	$\|A\|$	#inst.	p	β	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]
600	14263	5	5	0.1	-	-	-	0	**17.47**	***TL***	0	53.98	*TL*
600	14263	5	5	0.2	-	-	-	0	**23.21**	***TL***	0	52.86	*TL*
600	14263	5	5	0.3	-	-	-	0	**38.03**	***TL***	0	58.48	*TL*
600	14263	5	60	0.1	-	-	-	0	**1.67**	***TL***	0	45.72	*TL*
600	14263	5	60	0.2	-	-	-	0	**7.62**	***TL***	0	49.32	*TL*
600	14263	5	60	0.3	-	-	-	0	**13.90**	***TL***	0	52.94	*TL*
600	14263	5	200	0.1	-	-	-	5	**0.00**	**4016**	0	39.82	*TL*
600	14263	5	200	0.2	-	-	-	0	**1.32**	***TL***	0	40.84	*TL*
600	14263	5	200	0.3	-	-	-	0	**2.67**	***TL***	0	45.30	*TL*
700	19394	4	5	0.1	-	-	-	0	**19.61**	***TL***	0	56.90	*TL*
700	19394	4	5	0.2	-	-	-	0	**27.04**	***TL***	0	57.40	*TL*
700	19394	4	5	0.3	-	-	-	0	**41.52**	***TL***	0	61.92	*TL*
700	19394	4	70	0.1	-	-	-	0	**11.22**	***TL***	0	49.07	*TL*
700	19394	4	70	0.2	-	-	-	0	**5.05**	***TL***	0	49.93	*TL*
700	19394	4	70	0.3	-	-	-	0	**13.83**	***TL***	0	54.54	*TL*
700	19394	4	233	0.1	-	-	-	4	**0.00**	**6344**	0	39.32	*TL*
700	19394	4	233	0.2	-	-	-	0	**2.23**	***TL***	0	42.89	*TL*
700	19394	4	233	0.3	-	-	-	0	**2.28**	***TL***	0	42.96	*TL*
800	25345	3	5	0.1	-	-	-	0	**22.60**	***TL***	0	56.58	*TL*
800	25345	3	5	0.2	-	-	-	0	**27.51**	***TL***	0	58.63	*TL*
800	25345	3	5	0.3	-	-	-	0	**43.80**	***TL***	0	60.62	*TL*
800	25345	3	80	0.1	-	-	-	0	**12.70**	***TL***	0	47.53	*TL*
800	25345	3	80	0.2	-	-	-	0	**6.97**	***TL***	0	49.94	*TL*
800	25345	3	80	0.3	-	-	-	0	**16.62**	***TL***	0	54.21	*TL*
800	25345	3	266	0.1	-	-	-	0	**10.47**	***TL***	0	41.27	*TL*
800	25345	3	266	0.2	-	-	-	0	**4.30**	***TL***	0	42.74	*TL*
800	25345	3	266	0.3	-	-	-	0	**4.30**	***TL***	0	42.81	*TL*
900	32103	3	5	0.1	-	-	-	0	**22.80**	***TL***	0	56.53	*TL*
900	32103	3	5	0.2	-	-	-	0	**28.17**	***TL***	0	58.05	*TL*
900	32103	3	5	0.3	-	-	-	0	**41.21**	***TL***	0	61.21	*TL*
900	32103	3	90	0.1	-	-	-	0	**15.88**	***TL***	0	47.70	*TL*
900	32103	3	90	0.2	-	-	-	0	**18.20**	***TL***	0	47.68	*TL*
900	32103	3	90	0.3	-	-	-	0	**15.99**	***TL***	0	50.99	*TL*
900	32103	3	300	0.1	-	-	-	0	34.67	*TL*	0	**33.92**	***TL***
900	32103	3	300	0.2	-	-	-	0	**16.97**	***TL***	0	42.05	*TL*
900	32103	3	300	0.3	-	-	-	0	**5.58**	***TL***	0	42.19	*TL*

Table 4: Test results of solving the Cp-CTPC. Reported values are averaged and aggregated by $|V|$, $p = \{5, 10\}$ and $\beta = \{0.1, 0.2, 0.3\}$.

					MCF			BF			SCF		
$\|V\|$	$\|A\|$	#inst.	M	α	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]
100	396	30	5	0.8	28	0.10	994	30	**0.00**	**187**	0	30.19	*TL*
100	396	30	5	0.9	28	0.08	1006	30	**0.00**	**190**	0	30.08	*TL*
100	396	30	10	0.8	28	0.10	961	30	**0.00**	**201**	0	30.16	*TL*
100	396	30	10	0.9	28	0.09	880	30	**0.00**	**159**	0	30.05	*TL*
100	396	30	20	0.8	29	0.05	1049	30	**0.00**	**223**	0	30.53	*TL*
100	396	30	20	0.9	27	0.05	1147	30	**0.00**	**415**	0	30.04	*TL*
200	1585	30	10	0.8	7	2.95	5972	16	**1.33**	**4247**	0	45.37	*TL*
200	1585	30	10	0.9	8	2.91	5876	16	**1.23**	**4129**	0	45.44	*TL*
200	1585	30	20	0.8	7	2.86	5936	16	**1.33**	**4085**	0	45.36	*TL*
200	1585	30	20	0.9	9	2.53	5795	17	**1.41**	**4113**	0	45.42	*TL*
200	1585	30	40	0.8	7	3.40	5956	15	**1.68**	**4481**	0	45.17	*TL*
200	1585	30	40	0.9	8	3.00	5771	15	3.21	4594	0	45.24	*TL*
300	3561	30	15	0.8	1	7.02	7167	3	**2.32**	**6815**	0	45.13	*TL*
300	3561	30	15	0.9	2	5.57	7032	4	**2.23**	**6821**	0	45.25	*TL*
300	3561	30	30	0.8	1	6.82	7074	4	**2.26**	**6802**	0	44.99	*TL*
300	3561	30	30	0.9	1	3.54	7053	3	**2.23**	**6890**	0	44.95	*TL*
300	3561	30	60	0.8	2	8.63	7072	4	**2.81**	**7010**	0	45.39	*TL*
300	3561	30	60	0.9	1	5.04	7055	3	**3.01**	**6944**	0	45.48	*TL*
400	6338	30	20	0.8	-	-	-	0	**5.55**	***TL***	0	49.73	*TL*
400	6338	30	20	0.9	-	-	-	0	**5.71**	***TL***	0	49.69	*TL*
400	6338	30	40	0.8	-	-	-	0	**5.50**	***TL***	0	49.43	*TL*
400	6338	30	40	0.9	-	-	-	0	**5.65**	***TL***	0	49.57	*TL*
400	6338	30	80	0.8	-	-	-	0	**6.52**	***TL***	0	50.22	*TL*
400	6338	30	80	0.9	-	-	-	0	**8.54**	***TL***	0	49.92	*TL*
500	9900	30	25	0.8	-	-	-	0	**8.42**	***TL***	0	51.69	*TL*
500	9900	30	25	0.9	-	-	-	0	**8.16**	***TL***	0	52.15	*TL*
500	9900	30	50	0.8	-	-	-	0	**8.49**	***TL***	0	51.93	*TL*
500	9900	30	50	0.9	-	-	-	0	**8.07**	***TL***	0	51.76	*TL*
500	9900	30	100	0.8	-	-	-	0	**9.42**	***TL***	0	52.25	*TL*
500	9900	30	100	0.9	-	-	-	0	**9.64**	***TL***	0	52.28	*TL*
600	14263	30	30	0.8	-	-	-	0	**16.59**	***TL***	0	54.43	*TL*
600	14263	30	30	0.9	-	-	-	0	**15.98**	***TL***	0	54.26	*TL*
600	14263	30	60	0.8	-	-	-	0	**16.05**	***TL***	0	53.89	*TL*
600	14263	30	60	0.9	-	-	-	0	**14.97**	***TL***	0	53.66	*TL*
600	14263	30	120	0.8	-	-	-	0	**19.16**	***TL***	0	54.34	*TL*
600	14263	30	120	0.9	-	-	-	0	**17.56**	***TL***	0	54.32	*TL*
700	19394	24	35	0.8	-	-	-	0	**19.67**	***TL***	0	57.82	*TL*
700	19394	24	35	0.9	-	-	-	0	**19.58**	***TL***	0	57.37	*TL*
700	19394	24	70	0.8	-	-	-	0	**20.21**	***TL***	0	57.75	*TL*
700	19394	24	70	0.9	-	-	-	0	**19.67**	***TL***	0	58.02	*TL*
700	19394	24	140	0.8	-	-	-	0	**21.20**	***TL***	0	58.17	*TL*
700	19394	24	140	0.9	-	-	-	0	**21.14**	***TL***	0	58.34	*TL*
800	25345	18	40	0.8	-	-	-	0	**21.65**	***TL***	0	58.23	*TL*
800	25345	18	40	0.9	-	-	-	0	**21.59**	***TL***	0	57.94	*TL*
800	25345	18	80	0.8	-	-	-	0	**21.53**	***TL***	0	57.94	*TL*
800	25345	18	80	0.9	-	-	-	0	**21.65**	***TL***	0	57.69	*TL*
800	25345	18	160	0.8	-	-	-	0	**22.83**	***TL***	0	58.72	*TL*
800	25345	18	160	0.9	-	-	-	0	**22.39**	***TL***	0	58.62	*TL*
900	32103	18	45	0.8	-	-	-	0	**20.92**	***TL***	0	57.25	*TL*
900	32103	18	45	0.9	-	-	-	0	**20.81**	***TL***	0	57.11	*TL*
900	32103	18	90	0.8	-	-	-	0	**21.10**	***TL***	0	57.30	*TL*
900	32103	18	90	0.9	-	-	-	0	**20.96**	***TL***	0	57.14	*TL*
900	32103	18	180	0.8	-	-	-	0	**22.99**	***TL***	0	57.45	*TL*
900	32103	18	180	0.9	-	-	-	0	**22.42**	***TL***	0	57.40	*TL*

Table 5: Test results of solving the Cp-CTPC. Reported values are averaged and aggregated by $|V|$, $p = \{|V|/10, |V|/5, |V|/3\}$ and $\beta = \{0.1, 0.2, 0.3\}$.

					MCF			BF			SCF		
$\|V\|$	$\|A\|$	#inst.	M	α	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]	#solv.	g[%]	t[s.]
100	396	45	5	0.8	45	0.00	115	45	**0.00**	**48**	0	23.34	*TL*
100	396	45	5	0.9	45	0.00	146	45	**0.00**	**49**	0	23.33	*TL*
100	396	45	10	0.8	45	0.00	95	45	**0.00**	**47**	0	23.28	*TL*
100	396	45	10	0.9	45	0.00	94	45	**0.00**	**46**	0	23.25	*TL*
100	396	45	20	0.8	45	0.00	180	45	**0.00**	**42**	0	23.15	*TL*
100	396	45	20	0.9	45	0.00	98	45	**0.00**	**42**	0	23.43	*TL*
200	1585	45	10	0.8	42	0.12	1348	43	**0.06**	**861**	0	37.71	*TL*
200	1585	45	10	0.9	41	0.15	1444	43	**0.06**	**875**	0	37.69	*TL*
200	1585	45	20	0.8	41	0.14	1278	43	**0.06**	**919**	0	37.67	*TL*
200	1585	45	20	0.9	42	0.12	1227	43	**0.06**	**801**	0	37.70	*TL*
200	1585	45	40	0.8	41	0.12	1286	42	**0.06**	**1033**	0	37.62	*TL*
200	1585	45	40	0.9	42	0.10	1253	43	**0.06**	**970**	0	37.58	*TL*
300	3561	45	15	0.8	26	0.98	3933	19	0.70	4908	0	39.97	*TL*
300	3561	45	15	0.9	25	1.11	4166	18	0.74	4961	0	39.89	*TL*
300	3561	45	30	0.8	25	0.84	4017	19	0.70	4927	0	40.00	*TL*
300	3561	45	30	0.9	25	0.81	3912	19	0.67	4909	0	40.03	*TL*
300	3561	45	60	0.8	26	0.77	3827	21	0.67	4883	0	39.93	*TL*
300	3561	45	60	0.9	27	0.69	3724	21	0.68	4731	0	39.91	*TL*
400	6338	45	20	0.8	-	-	-	15	**1.62**	**6005**	0	42.92	*TL*
400	6338	45	20	0.9	-	-	-	15	**1.61**	**6086**	0	42.98	*TL*
400	6338	45	40	0.8	-	-	-	15	**1.71**	**6043**	0	42.94	*TL*
400	6338	45	40	0.9	-	-	-	15	**1.60**	**6019**	0	42.85	*TL*
400	6338	45	80	0.8	-	-	-	14	**1.73**	**6015**	0	42.85	*TL*
400	6338	45	80	0.9	-	-	-	14	**1.55**	**5981**	0	42.98	*TL*
500	9900	45	25	0.8	-	-	-	10	**2.26**	**6504**	0	44.32	*TL*
500	9900	45	25	0.9	-	-	-	9	**2.32**	**6557**	0	44.41	*TL*
500	9900	45	50	0.8	-	-	-	9	**2.25**	**6497**	0	44.35	*TL*
500	9900	45	50	0.9	-	-	-	10	**2.25**	**6477**	0	44.31	*TL*
500	9900	45	100	0.8	-	-	-	9	**2.31**	**6512**	0	44.25	*TL*
500	9900	45	100	0.9	-	-	-	10	**2.24**	**6448**	0	44.43	*TL*
600	14263	45	30	0.8	-	-	-	5	**3.29**	**6878**	0	47.58	*TL*
600	14263	45	30	0.9	-	-	-	4	**3.44**	**6869**	0	47.14	*TL*
600	14263	45	60	0.8	-	-	-	4	**3.33**	**6887**	0	47.54	*TL*
600	14263	45	60	0.9	-	-	-	5	**3.27**	**6878**	0	46.91	*TL*
600	14263	45	120	0.8	-	-	-	5	**3.28**	**6832**	0	46.94	*TL*
600	14263	45	120	0.9	-	-	-	5	**3.61**	**6819**	0	47.37	*TL*
700	19394	36	35	0.8	-	-	-	6	**3.65**	**7004**	0	47.86	*TL*
700	19394	36	35	0.9	-	-	-	6	**4.23**	**6929**	0	47.60	*TL*
700	19394	36	70	0.8	-	-	-	6	**4.09**	**6941**	0	47.48	*TL*
700	19394	36	70	0.9	-	-	-	8	**4.08**	**6864**	0	47.61	*TL*
700	19394	36	140	0.8	-	-	-	7	**3.40**	**6917**	0	47.31	*TL*
700	19394	36	140	0.9	-	-	-	8	**3.81**	**6890**	0	47.52	*TL*
800	25345	27	40	0.8	-	-	-	0	**9.95**	***TL***	0	47.68	*TL*
800	25345	27	40	0.9	-	-	-	0	**16.04**	***TL***	0	47.54	*TL*
800	25345	27	80	0.8	-	-	-	0	**10.96**	***TL***	0	47.20	*TL*
800	25345	27	80	0.9	-	-	-	0	**10.20**	***TL***	0	47.55	*TL*
800	25345	27	160	0.8	-	-	-	0	**13.32**	***TL***	0	47.38	*TL*
800	25345	27	160	0.9	-	-	-	0	**7.02**	***TL***	0	47.80	*TL*
900	32103	27	45	0.8	-	-	-	0	**17.41**	***TL***	0	46.32	*TL*
900	32103	27	45	0.9	-	-	-	0	**20.53**	***TL***	0	46.45	*TL*
900	32103	27	90	0.8	-	-	-	0	**18.92**	***TL***	0	45.75	*TL*
900	32103	27	90	0.9	-	-	-	0	**18.95**	***TL***	0	45.20	*TL*
900	32103	27	180	0.8	-	-	-	0	**15.49**	***TL***	0	46.54	*TL*
900	32103	27	180	0.9	-	-	-	0	**17.73**	***TL***	0	45.76	*TL*

References

[1] Discrete location problems: Benchmark library. `http://www.math.nsc.ru/AP/benchmarks/english.html`. Accessed: 2015-07-14.

[2] J.E. Beasley. OR-library: distributing test problems by electronic mail. *Journal of the Operational Research Society*, 41(11):1069–1072, 1990. `http://people.brunel.ac.uk/~mastjjb/jeb/info.html`.

[3] W. Ben-Ameur and J. Neto. Acceleration of cutting-plane and column generation algorithms: Applications to network design. *Networks*, 49(1):3–17, 2007.

[4] J.F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[5] B. Cherkassky and A. Goldberg. On implementing push-relabel method for the maximum flow problem. In E. Balas and J. Clausen, editors, *Proceedings of IPCO IV*, volume 920 of *LNCS*, pages 157–171. Springer, 1995.

[6] M. Chimani, C. Gutwenger, M. Jünger, G. Klau, K. Klein, and P. Mutzel. The open graph drawing framework (OGDF). *Handbook of Graph Drawing and Visualization*, pages 543–569, 2011.

[7] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1): 269–271, 1959.

[8] M. Fischetti, I. Ljubić, and M. Sinnl. Redesigning Benders decomposition for large scale facility location. *Management Science*, 2016. to appear.

[9] R.W. Floyd. Algorithm 97: Shortest path. *Communications of ACM*, 5(6):345–345, 1962.

[10] L. Gouveia. A comparison of directed formulations for the capacitated minimal spanning tree. *Telecommunication Systems*, 1:51–76, 1993.

[11] S.L. Hakimi. Optimum distribution of switching centers in a communication network and some related graph theoretic problems. *Operations Research*, 13(3):462–475, 1965.

[12] Y. Jiang, Z. Zhuang, A.J. Sinusas, and X. Papademetris. Vascular tree reconstruction by minimizing a physiological functional cost. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, pages 178–185. IEEE, 2010.

[13] Y. Jiang, Z.W. Zhuang, A.J. Sinusas, L.H. Staib, and X. Papademetris. Vessel connectivity using Murray's hypothesis. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 528–536. Springer, 2011.

[14] E. Lalla-Ruiz, S. Schwarze, and S. Voß. A matheuristic approach for the p-cable trench problem. In *Learning and Intelligent Optimization: 10th International Conference, LION 10*, pages 247–252, 2016.

[15] L.A.N. Lorena and E.L.F. Senne. A column generation approach to capacitated p-median problems. *Computers & Operations Research*, 31(6):863–876, 2004.

[16] V. Marianov, G. Gutiérrez-Jarpa, C. Obreque, and O. Cornejo. Lagrangean relaxation heuristics for the p-cable-trench problem. *Computers & Operations Research*, 39(3):620–628, 2012.

[17] V. Marianov, G. Gutiérrez-Jarpa, and C. Obreque. p-cable trench problem with covering. In *XXII EURO Working Group on Locational Analysis Meeting*, pages 75–76, 2015.

[18] E. Minieka. The m-center problem. *SIAM Review*, 12(1):138–139, 1970.

[19] S. Schwarze. The multi-commodity cable trench problem. In *ECIS 2015 Completed Research Papers*, 2015.

[20] F.J. Vasko, R.S. Barbieri, B.Q. Rieksts, K.L. Reitmeyer, and K.L. Stott. The cable trench problem: combining the shortest path and minimum spanning tree problems. *Computers & Operations Research*, 29(5):441–458, 2002.

[21] F.J. Vasko, E. Landquist, G. Kresge, A. Tal, Y. Jiang, and X. Papademetris. A simple and efficient strategy for solving very large-scale generalized cable-trench problems. *Networks*, 67(3):199–208, 2016.