Benders Decomposition for Very Large Scale Partial Set Covering and Maximal Covering Problems

Jean-François Cordeau[a], Fabio Furini[b,1], Ivana Ljubić[c]

[a]*HEC Montréal, Canada*
[b]*PSL, Université Paris-Dauphine, France*
[c]*ESSEC Business School of Paris, France*

Abstract

Covering problems constitute an important family of facility location problems. This paper introduces a new exact algorithm for two important members of this family: *i)* the maximal covering location problem, which requires finding a subset of facilities that maximizes the amount of demand covered while respecting a budget constraint on the cost of the facilities; and *ii)* the partial set covering location problem, which minimizes the cost of the open facilities while forcing a certain amount of demand to be covered. We study an effective decomposition approach to the two problems based on the branch-and-Benders-cut reformulation. We also report the results of a series of computational experiments demonstrating that, thanks to this decomposition techniques, optimal solutions can be found very quickly, even for benchmark instances involving up to twenty million demand points.

Keywords: Combinatorial Optimization, Location Problems, Covering, Benders Decomposition, Branch-and-Cut

1. Introduction

Covering problems constitute an important family of facility location problems with widespread applications. These problems embed a notion of proximity (or coverage radius) that specifies whether a given demand point can be served or "covered" by a potential facility location. Proximity is often defined in terms of distance or travel time between points. A demand point is then said to be covered by a facility if it lies within the coverage radius of this facility. One of the best known members of this family is the set covering location problem (SCLP) in which one must choose a minimum cost set of facilities so that every demand point is covered at least once.

The drawback of the SCLP is that it often leads to costly or unrealistic solutions because it gives the same importance to every demand point, regardless of its position and size. To overcome this

*Corresponding author
Email addresses: jean-francois.cordeau@hec.ca (Jean-François Cordeau), fabio.furini@dauphine.fr (Fabio Furini), ivana.ljubic@essec.edu (Ivana Ljubić)

weakness, two main variants have been studied in the location theory literature: *i)* the maximal covering location problem (MCLP) requires choosing a subset of facilities that maximize the demand covered while respecting a budget constraint on the cost of the facilities; *ii)* the partial set covering location problem (PSCLP) minimizes the cost of the open facilities while forcing a certain amount of demand to be covered. The MCLP is often defined as the problem of maximizing demand coverage with a fixed number of facilities, which is actually a special case of the problem obtained by assigning a cost of 1 to every facility and setting the available budget to the number of desired facilities.

Location problems with covering objectives or constraints are commonplace in the service sector (schools, hospitals, libraries, restaurants, retail outlets, bank branches) as well as in the location of emergency facilities or vehicles (fire stations, ambulances, oil spill equipments). They also find applications in many other areas such as in the location of telecommunications antennas and the installation of watchtowers or monitoring equipment. While many applications involve a relatively small number of demand points and potential facility locations, and can therefore be solved in a satisfactory way by existing heuristics or by general-purpose solvers, there are also cases where the number of demand points can run in the thousands or even millions. As noted by Murray [23], discrete covering problems are often an attempt to discretize location planning problems with discrete potential facility locations but continuously distributed customer demand. Even though there exist algorithms to handle continuous space covering location problems [25, 32], it is often more practical to approximate these problems and solve them as if they were discrete. The optimal solution obtained by this discretization process can be very sensitive to the level of discretization of the continuous demand [see, e.g., 8]. However, the finer the discretization, the harder it is to solve the resulting optimization problem.

It has often been observed that the LP relaxation value of a mixed integer programming (MIP) formulation to the SCLP provides a very good lower bound and that the continuous solution has few fractional variables [30, 26, 29]. In particular, for the case where the objective is to minimize the number of open facilities, the LP relaxation solution is often integral. This makes it relatively easy to solve the SCLP exactly or to obtain high quality heuristic solutions. Church and ReVelle [5] and Snyder [29] also report that the LP relaxation of the MCLP is often integer. Nevertheless, the integrality gap tends to grow with the size of the instances, making the solution of large-scale instances a challenge for general-purpose mixed-integer programming solvers. Surprisingly, very few exact algorithms have been developed to solve the MCLP and the PSCLP.

The primary aim of this paper is to introduce a new decomposition technique based on Benders decomposition that is capable of solving large-scale instances of the two problems. As mentioned above, discrete location problems are often used to model situations that inherently involve information and decisions associated to a continuous customer demand. Hence, if one can afford to handle large sets of demand points, the quality of the resulting solution can be significantly

improved. While many heuristics have been described for the MCLP, exact algorithms are very scarce. In addition, the PSCLP has received almost no attention in the scientific literature despite its practical relevance. Because these two problems have similar features, we propose to solve them with a unified methodology that takes advantage of their particular structure. The resulting algorithm yields optimal solutions to instances with huge sets of demand points in reasonable computing times. Another contribution of this paper is that we introduce and make available new large-scale instances that can be used as benchmark by other authors interested in covering location problems.

The remainder of the paper is organized as follows. Section 2 first reviews the relevant literature on the SCLP, MCLP and PSCLP. This is followed in Section 3 by the formal definition and mathematical formulation of the two problems under study. Sections 4 and 5 then describe our Benders decomposition approach. Computational results are presented in Section 6 and the conclusion follows in Section 7.

2. Literature Review

To the best of our knowledge, the first mention of the minimum cost covering problem in the context of location can be attributed to Hakimi [17], who gave as an example the problem of locating the minimum number of policemen so that everyone is within a given distance d from a policeman. Hakimi represents the problem with a Boolean function defined over the vertices of a graph and suggests a solution procedure based on enumeration. A few years later, Toregas et al. [30] introduced the first integer programming formulation of the problem and discussed its application to the location of emergency service facilities in a discrete space. In related research, Walker [31] formulated the problem of assigning ladder trucks to fire stations as a minimum cost covering problem and proposed a heuristic to solve it.

The MCLP was first introduced by Church and ReVelle [5] who provide an integer programming formulation of the problem. Megiddo et al. [22] explain that the problem is NP-hard by reduction from the minimum dominating set problem. Murray [23] provides a recent survey of the MCLP and lists a wide array of applications. It should be noted that most papers on this problem impose an upper bound on the number of facilities instead of the more general budget constraint considered here.

Because the LP relaxation of the standard MIP formulation of the MCLP provides good lower bounds, the problem is often solved by a branch-and-bound algorithm in which the bounds are computed by solving the continuous relaxation of the problem by the simplex algorithm. In the computational experiments performed by Snyder [29], for more than 95% of the instances solved the LP relaxation of the problem was integral and no branching was required. These experiments were performed on instances with up to 800 demand points and facility locations and a limit of 16 on the number of open facilities. The more general budget constraint that we consider here may lead to more fractional solutions than when one imposes a simple upper bound on the number of facilities.

In addition, instances with millions of demand points become intractable for general-purpose solvers and just solving their LP relaxation can be a challenge.

We are aware of only one exact algorithm for the MCLP. Downs and Camm [9] dualize the covering constraints by Lagrangian relaxation to obtain a binary knapsack problem. They then use subgradient optimization to solve the Lagrangean dual. Since the Lagrangean subproblem has the integrality property, the best bound obtained in this way is equal to the LP relaxation lower bound. This method is embedded in a branch-and-bound tree to obtain an optimal integer solution. The authors reported results on several data sets and the largest instance considered had 2241 demand points and 74 potential facility locations.

Several heuristics exist for the MCLP, many of which rely on a Lagrangean relaxation of the covering constraints. Church and ReVelle [5] describe a greedy heuristic that adds at each iteration the facility that increases the most the objective function value. They also introduce a variant of the heuristic that checks at each iteration whether swapping an open facility for a closed one can improve the solution. Galvão and ReVelle [14] describe a Lagrangean heuristic that uses the same relaxation of the covering constraints as Downs and Camm [9]. They also employ subgradient optimization to solve the Lagrangean dual. Feasible integer solutions are obtained by using a heuristic similar to that of Church and ReVelle [5]. Galvão et al. [15] compare heuristics based on Lagrangean relaxation or a surrogate relaxation. Senne et al. [28] introduce a decomposition heuristic that relies on a partial relaxation of the covering constraints, which yields stronger bounds than those obtained from the LP relaxation.

Several metaheuristics have also been described for the MCLP. In particular, ReVelle et al. [27] apply the concept of heuristic concentration to the MCLP. Heuristic concentration consists in first reducing the solution space by discarding potential facility locations that are not likely to appear in an optimal solution, before applying branch-and-bound or a local search heuristic to the reduced problem. Zarandi et al. [33] use a genetic algorithm to solve large-scale instances of the problem with up to 2500 nodes. Finally, Máximo et al. [21] describe a guided adaptive search algorithm which they apply to instances with up to 7730 nodes.

As mentioned in the previous section, the PCSLP has been the object of little research after its introduction by Daskin and Owen [7], who solved the problem with a Lagrangian heuristic similar to that of Galvão and ReVelle [14] and reported results on instances with up to 150 nodes. There are nonetheless other studies addressing similar partial set covering problems in other contexts than location. For example, Bilal et al. [4] describe an iterated tabu search heuristic for a variant of the problem arising in a mining application. Finally, Berman et al. [3] study a related problem in which customers whose distance falls between a lower and an upper bound from their nearest facility are only partially covered.

We refer to the recent book of Laporte et al. [18] for a general overview of location problems and to the survey chapters of Snyder [29] and of [16] for specific reviews on covering problems.

Farahani et al. [10] also provide a classification and survey of many variants of covering location problems.

3. Problem Definition

In this section, we formally define the two problems that are considered throughout the paper and we provide a mathematical formulation for each one.

In the covering facility location problems studied in this paper, we are given a set of potential facility locations I with opening costs $f_i \geq 0$, $i \in I$, and a set of customer locations J such that each customer $j \in J$ is associated with a demand $d_j \geq 0$. For each customer j, we are also given a subset $I(j) \subseteq I$ of facility locations that can "cover" j, i.e., that can fully serve the demand d_j. Similarly, let $J(K) \subseteq J$ for a subset of facilities $K \subseteq I$, be the subset of customers covered by K, and let $J(i) = J(\{i\})$, for $i \in I$. For a subset of customers $J' \subseteq J$, let $D(J') = \sum_{j \in J'} d_j$ be the total demand of the customers in J'. Finally, let $J_s \subset J$ be the set of customers that are covered by a single facility, i.e., $J_s = \{j \in J : |I(j)| = 1\}$. In particular, for $j \in J_s$, let $i(j)$ be the single facility that can cover this customer. Similarly, for a subset of facilities $K \subset I$, let $J_s(K)$ be the set of all customers that are covered by a single facility from K, i.e.:

$$J_s(K) = \{j \in J : |I(j) \cap K| = 1\}.$$

In Figure 1, we provide an example with 4 facilities (represented by the white vertices) and 8 customers (represented by the grey vertices). The incident edges to a customer vertex are linked to the facilities covering this customer, i.e., to the set $I(j)$; while the incident edges to a facility vertex represent the set of customers covered by the facility i, i.e., the set $J(i)$. For a subset of facilities $K = \{1, 2\}$, we graphically illustrate definitions of the terms introduced above.

3.1. The Partial Set Covering Location Problem (PSCLP)

Given a parameter $D > 0$, the PSCLP asks for a subset of facilities to open so as to make sure that the covered customer demand is at least D and the cost for opening the facilities is minimized.

The problem can be formulated with two sets of binary variables. For every potential facility location $i \in I$, let y_i take value 1 if and only if facility i is open and, for every customer $j \in J$, let z_j take value 1 if and only if customer j is covered by at least one open facility. The problem can

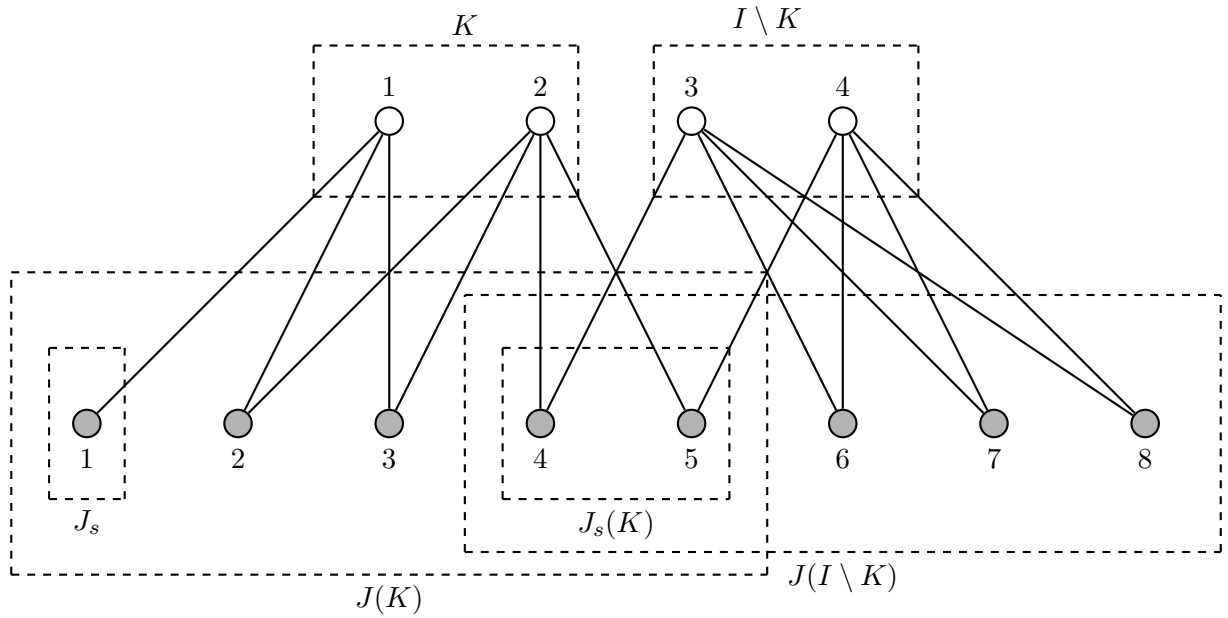

Figure 1: Example instance with 4 potential facilities and 8 customers.

then be formulated as the following integer linear programming model:

$$\min \quad \sum_{i \in I} f_i y_i \tag{1}$$

$$\sum_{i \in I(j)} y_i \geq z_j \qquad j \in J \tag{2}$$

$$\sum_{j \in J} d_j z_j \geq D \tag{3}$$

$$y_i \in \{0,1\} \qquad i \in I \tag{4}$$

$$z_j \in \{0,1\} \qquad j \in J. \tag{5}$$

In this model, the objective function (1) minimizes the cost of open facilities, whereas constraint (3) ensures that the covered customer demand is larger than or equal to D. The linking constraints (2) guarantee that whenever a customer j is covered, at least one of the facilities from its neighborhood is open. Finally, constraints (4) and (5) impose binary restrictions on decision variables y and z, respectively.

3.2. The Maximal Covering Location Problem (MCLP)

The second problem that we consider is the *Maximal Covering Location Problem* (MCLP). This problem consists of selecting a subset of facilities to open so as to maximize the covered customer demand, subject to available budget $B > 0$ on the cost of open facilities.

Using the same variables as for the PSCLP, the MCLP can be formulated as follows:

$$\max \quad \sum_{j \in J} d_j z_j \tag{6}$$

$$\sum_{i \in I} f_i y_i \leq B \tag{7}$$

$$(2), (4), (5).$$

Compared to the PSCLP, we observe that the objective function and the demand coverage constraint are swapped, so that now the objective function defined by (6) maximizes the covered demand, whereas the knapsack-like constraint (7) ensures that the available budget of B for opening the facilities is not exceeded. The remaining constraints are the same as for the PSCLP.

3.3. Property

We now make an observation that is valid for both the PSCLP and the MCLP.

Property 1. *In both ILP formulations for the PSCLP and MCLP, integrality conditions* (5) *can be relaxed to* $z_j \leq 1$, $j \in J$.

To see that this property holds for the PSCLP, assume that we relax the integrality conditions on the z variables and that we obtain an optimal solution (y^*, z^*) such that for some $j \in J' \subseteq J$, $0 < z_j^* < 1$ holds. In that case, without loss of generality, and without changing the value of the optimal solution, for each customer $j \in J'$ one can redefine $z_j = 1$. Similar arguments hold for the MCLP.

Based on the latter property, we study a Benders decomposition approach in which z variables are projected out from the model and replaced by corresponding *Benders optimality cuts* (for the PSCLP) and *Benders feasibility cuts* (for the MCLP). Our computational study will demonstrate that, thanks to this decomposition, optimal solutions for large-scale instances involving millions of demand points, can be solved within short computational time.

For the PSCLP and MCLP, observe that the value of the z variables can be computed as a function of y by setting $z_j = \max_{i \in I(j)}\{y_i\}$, for each $j \in J$. Furthermore, for any given solution vector $\tilde{y}$, let

$$\tilde{I}_j = \sum_{i \in I(j)} \tilde{y}_i$$

represent the number of facilities covering customer j. Consequently, the value of z_j for any $j \in J$ can be set as

$$\tilde{z}_j = \min\left\{1, \tilde{I}_j\right\}.$$

In the following, we propose a linear way to project out the z variables from the model, using the Benders decomposition approach.

4. Benders Decomposition for the PSCLP

In the following, we focus on the PSCLP and describe several types of Benders cuts. We start with those obtained in a standard way, i.e., by dualizing the Benders subproblem. We refer to these cuts as *LP-based Benders cuts*. Later, in Section 4.2, we propose a normalization technique that yields several closed formulas for deriving valid Benders cuts. We refer to these as *combinatorial Benders cuts*. We show that the latter can be separated in linear time, and we compare their relative strength. Finally, in Section 4.3, we apply a technique recently proposed by Conforti and Wolsey [6], to derive *facet-defining Benders cuts*.

4.1. LP-Based Benders Cuts

By projecting out the z variables from the model, we obtain the following Benders master problem:

$$\min \left\{ \sum_{i \in I} f_i y_i : \quad B_t(y) \geq 0,\ t \in P,\ \ y_i \in \{0,1\},\ i \in I \right\}, \tag{8}$$

where $B_t(y)$ refers to Benders feasibility cuts corresponding to extreme rays t of the polytope P associated to the linear programming dual of the Benders subproblem. For a given vector $\tilde{y} \in [0, 1]^{|I|}$, the Benders primal subproblem reads as follows:

$$\min \left\{ 0 : z_j \leq \tilde{I}_j,\ j \in J, \quad \sum_{j \in J} d_j z_j \geq D, \quad 0 \leq z_j \leq 1,\ j \in J \right\}. \tag{9}$$

Its dual is given as:

$$\max \left\{ D\gamma - \sum_{j \in J} \left(\tilde{I}_j \pi_j + \sigma_j \right) : (\pi, \sigma, \gamma) \in P \right\}, \tag{10}$$

where π, γ and σ are dual variables associated to the constraints of (9), respectively. These variables are constrained to belong to the polytope P which is defined as:

$$P = \{ (\pi, \gamma, \sigma) \geq 0 :\ \pi_j + \sigma_j \geq d_j \gamma,\ j \in J \}. \tag{11}$$

From the LP duality theory we know that the primal subproblem (9) is infeasible if its dual is unbounded. Let $(\tilde{\pi}, \tilde{\gamma}, \tilde{\sigma})$ be an extreme ray of the unbounded dual subproblem. Then the associated *Benders feasibility cut* can be written as:

$$\sum_{i \in I} \left(\sum_{j \in J(i)} \tilde{\pi}_j \right) y_i \geq D\tilde{\gamma} - \sum_{j \in J} \tilde{\sigma}_j. \tag{B}$$

There is an exponential number of extreme rays $t \in P$, which makes it impractical to enumerate all of them in advance. Since not all Benders feasibility cuts are necessary to find an optimal

solution, only a subset of them will be *separated* by the decomposition approach. The formulation (8) containing only a subset of Benders cuts will be referred to as the *restricted master problem*.

There are two ways of implementing a Benders decomposition approach: (1) using a cutting plane procedure, where each time a Benders cut is generated, the master problem is solved again as an ILP, or (2) using a branch-and-Benders-cut approach in which a single enumeration tree is created and Benders cuts are separated on the fly as in a classical branch-and-cut procedure. The latter approach has became more popular in the recent literature (see, e.g., [12, 13, 19]), due to the fact that it allows standard MILP techniques like enumeration, primal heuristics, variable fixing, and preprocessing to be efficiently combined with Benders decomposition. Accordingly, we implement the formulation (8) by separating constraints (B) on the fly, within a branch-and-cut framework. Note that, since z variables do not appear in the objective function (1), Benders optimality cuts are not needed.

It is well-known that the LP relaxation of (8) gives the same lower bounds as the corresponding compact formulation (1)-(5). However, the way in which the separation of these Benders cuts is implemented heavily affects the overall performance of a branch-and-Benders-cut approach and the overall number of cuts necessary to obtain this bound. A straightforward implementation requires solving the dual subproblem (10) as an LP and deriving cuts from the respective extreme rays. As already observed e.g., in [2, 11, 19], this approach has a significant drawback: it returns an arbitrarily chosen extreme ray without having any positive influence on the quality of the violated cut found. As a consequence, convergence may be slowed down due to the fact that shallow cuts are often generated. Among the successful strategies to overcome these difficulties are the *normalization* techniques. In general, normalization of Benders cuts consists of reducing optimization over an unbounded cone to solving an LP over a bounded polyhedron. There is an abundant literature on different normalization techniques for Benders feasibility cuts, see, e.g., [11, 19, 20, 24]. In our work, we consider two normalization approaches obtained by closing the unbounded dual cone.

4.2. Normalized Benders Feasibility Cuts for the PSCLP

In this section we focus on a particular normalization of Benders cuts that appears very natural in the context of the PSCLP. We namely exploit the fact that the solution $\tilde{y}$ of the restricted master problem is infeasible if and only if the demand covered by $\tilde{y}$ is strictly less than D. Hence, instead of solving a feasibility LP given by (9), we search for the maximum demand that can be covered by $\tilde{y}$. This results in the following LP:

$$\Delta(\tilde{y}) = \max \left\{ \sum_{j \in J} d_j z_j : z_j \leq \tilde{I}_j,\ j \in J, \quad 0 \leq z_j \leq 1,\ j \in J \right\}. \tag{12}$$

The latter LP is always feasible and its optimal solution can be calculated as $\tilde{z}_j = \min\{1, \tilde{I}_j\}$, $j \in J$. After associating π_j, $j \in J$ to the first, and σ_j, $j \in J$ to the second group of constraints,

respectively, the dual of this problem becomes:

$$\Delta(\tilde{y}) = \min \left\{ \sum_{j \in J} \pi_j \tilde{I}_j + \sum_{j \in J} \sigma_j : \quad \pi_j + \sigma_j \geq d_j \; j \in J, \quad (\pi, \sigma) \geq 0 \right\}, \tag{13}$$

and Benders feasibility cuts are now derived from the extreme points of the underlying polytope. Let $(\tilde{\pi}, \tilde{\sigma})$ be the optimal solution of (13). If $\Delta(\tilde{y}) < D$, the violated Benders feasibility cut can now be derived as:

$$\sum_{i \in I} \left(\sum_{j \in J(i)} \tilde{\pi}_j \right) y_i + \sum_{j \in J} \tilde{\sigma}_j \geq D. \tag{14}$$

Comparing (14) with the Benders feasibility cut (B), we observe that the extreme point $(\tilde{\pi}, \tilde{\sigma})$ corresponds to an extreme ray of (10), in which the dual cone is intersected with the hyperplane $\gamma = 1$. This is one of the possible normalization techniques for Benders feasibility cuts. Given the relatively simple structure of the dual polytope, in the following we derive combinatorial algorithms for the separation of (14).

4.2.1. Combinatorial Separation Approach

In this section, we propose several analytical ways to derive cuts of type (14) by calculating the optimal dual multipliers $(\tilde{\pi}, \tilde{\sigma})$ using a closed formula. Let $\tilde{z}$ be the optimal solution of the primal Benders subproblem (12). Given that its dual is degenerate, there may be infinitely many Benders feasibility cuts associated to a given infeasible point of the restricted master problem $\tilde{y}$. In the following, Propositions 1, 2 and 4 provide closed formulas for calculating optimal dual solutions corresponding to extreme points of the dual polytope given by (13) and, hence, they allow us to derive three particular types of Benders feasibility cuts.

Proposition 1. *An optimal solution* $(\tilde{\pi}, \tilde{\sigma})$ *of the subproblem* (13) *can be computed as:*

$$\tilde{\pi}_j = \begin{cases} d_j, & \text{if } \tilde{I}_j < 1 \\ 0, & \text{otherwise} \end{cases} \qquad \tilde{\sigma}_j = \begin{cases} 0, & \text{if } \tilde{I}_j < 1 \\ d_j, & \text{otherwise} \end{cases} \qquad j \in J. \tag{15}$$

Proof. Given $j \in J$, one of the following two situations can occur:

- $\tilde{z}_j = \sum_{i \in I(j)} \tilde{y}_i < 1$: Due to the complementary slackness conditions, it follows that $\tilde{\sigma}_j = 0$, and hence $\tilde{\pi}_j = d_j$.
- $\tilde{z}_j = 1 \leq \sum_{i \in I(j)} \tilde{y}_i$: Due to the complementary slackness conditions, it follows that $\tilde{\pi}_j = 0$ and $\tilde{\sigma}_j = d_j$.

It is not difficult to see that a dual solution constructed in this way is feasible and that its objective function value is $\sum_{j \in J} d_j \tilde{z}_j$. It is thus optimal. □

The latter result gives us an algorithm to derive Benders feasibility cuts (for both fractional and integer points). Once the values $\tilde{I}_j$ are provided, the calculation of dual multipliers (and hence the separation of the associated Benders cut) can be performed *in linear time*, which can have a significant advantage over a generic LP-based separation technique.

For a fractional point $\tilde{y}$, the cut reads as

$$\sum_{j \in J: \tilde{I}_j < 1} d_j \left(\sum_{i \in I(j)} y_i \right) \geq D - \sum_{j \in J: \tilde{I}_j \geq 1} d_j. \tag{B0f}$$

A more intuitive interpretation of these cuts can be given by considering an integer vector $\tilde{y}$. Let $\tilde{K} \subseteq I$ be the set of open facilities whose incidence vector is given by $\tilde{y}$, and let $J(\tilde{K})$ be the set of customers covered by $\tilde{K}$. Benders cut (B0f) is then:

$$\sum_{j \notin J(\tilde{K})} d_j \left(\sum_{i \in I(j)} y_i \right) \geq D - D(J(\tilde{K})). \tag{B0}$$

This cut simply states that the *residual demand* (which is expressed by the value on the right-hand-side) has to be satisfied by choosing a subset of currently uncovered customers ($J \setminus J(\tilde{K})$) and making sure that they are covered by opening facilities from their neighborhood. After some reordering of terms, the cut can be also written as:

$$\sum_{i \notin \tilde{K}} \left(\sum_{j \in J(i) \setminus J(\tilde{K})} d_j \right) y_i \geq D - D(J(\tilde{K})),$$

which means that the residual demand has to be covered by opening some of the currently closed facilities ($i \notin \tilde{K}$) and by considering only currently uncovered customers from their neighborhood ($j \in J(i) \setminus J(\tilde{K})$).

In the following, we discuss two more possibilities to construct valid Benders cuts in a combinatorial way. Looking at the structure of the dual (13), we observe that whenever $\tilde{I}_j = 1$, $j \in J$, we have the freedom of assigning d_j to either π_j or σ_j. This allows us to derive two additional families of Benders cuts, which can be seen as lifted versions of (B0f).

Proposition 2. *An optimal solution* $(\tilde{\pi}, \tilde{\sigma})$ *of the subproblem* (13) *can be computed as:*

$$\tilde{\pi}_j = \begin{cases} d_j, & \text{if } \tilde{I}_j < 1 \text{ or } j \in J_s \\ 0, & \text{otherwise} \end{cases} \quad j \in J, \qquad \tilde{\sigma}_j = \begin{cases} 0, & \text{if } \tilde{I}_j < 1 \text{ or } j \in J_s \\ d_j, & \text{otherwise} \end{cases} \quad j \in J. \tag{16}$$

Proof. Similarly to the previous result, we can distinguish between the following situations:

- $\tilde{z}_j = \sum_{i \in I(j)} \tilde{y}_i < 1$: Due to the complementary slackness conditions, it follows that $\tilde{\sigma}_j = 0$,

and hence $\tilde{\pi}_j = d_j$.

- $\tilde{z}_j = 1$ and $|I(j)| = 1$ (and hence $\sum_{i \in I(j)} \tilde{y}_i = 1$): In that case we can set $\tilde{\pi}_j = d_j$ and $\tilde{\sigma}_j = 0$.
- Otherwise (i.e., $\tilde{z}_j = 1 \leq \sum_{i \in I(j)} \tilde{y}_i$ and $|I(j)| > 1$): Due to the complementary slackness conditions, it follows that $\tilde{\pi}_j = 0$ and $\tilde{\sigma}_j = d_j$.

□

Benders cuts obtained from (16) can be stated as:

$$\sum_{j \in J : \tilde{I}_j < 1} d_j \left(\sum_{i \in I(j)} y_i \right) + \sum_{j \in J_s : \tilde{I}_j = 1} d_j \left[y_{i(j)} - 1 \right] \geq D - \sum_{j \in J : \tilde{I}_j \geq 1} d_j. \tag{B1f}$$

Similarly, given an integer point $\tilde{y}$ (with the associated set of open facilities $\tilde{K}$), the latter cut can be restated as:

$$\sum_{i \notin \tilde{K}} \left(\sum_{j \in J(i) \setminus J(\tilde{K})} d_j \right) y_i + \sum_{i \in \tilde{K}} \left(\sum_{j \in J_s \cap J(i)} d_j \right) y_i \geq D - D(J(\tilde{K}) \setminus J_s). \tag{B1}$$

We observe that separating all possible cuts of type (B0f) or (B1f) results in the same quality of lower bound, as both cuts are derived from solving the same dual Benders subproblem. However, the following proposition indicates that separating (B1f) instead of (B0f) may result in a smaller number of separating iterations, due to the fact that (B1f) may cut off a larger portion of the infeasible region compared to (B0f).

Proposition 3. *Given a solution of the restricted master problem $\tilde{y}$, the associated Benders cut* (B1f) *dominates the cut* (B0f) *unless $J_s = \emptyset$, in which case the two cuts are identical.*

Proof. Observe that for a given $\tilde{y}$, cut (B1f) is obtained by adding

$$\sum_{j \in J_s : \tilde{I}_j = 1} d_j [y_{i(j)} - 1]$$

to the left-hand-side of (B0f). So, if $J_s = \emptyset$, the two cuts are the same. However, if $J_s \neq \emptyset$, the value of this summation can sometimes be negative, which will result in a stronger cut. □

Finally, using similar arguments as in Proposition 2, we propose a third family of combinatorial Benders cuts that can be derived from another optimal dual solution.

Proposition 4. *An optimal solution* $(\tilde{\pi}, \tilde{\sigma})$ *of the subproblem* (13) *can be computed as:*

$$\tilde{\pi}_j = \begin{cases} d_j, & \text{if } \tilde{I}_j \leq 1 \\ 0, & \text{otherwise} \end{cases} \quad j \in J \qquad \tilde{\sigma}_j = \begin{cases} 0, & \text{if } \tilde{I}_j \leq 1 \\ d_j, & \text{otherwise} \end{cases} \quad j \in J. \tag{17}$$

The associated Benders cuts can be written as:

$$\sum_{j \in J: \tilde{I}_j < 1} d_j \left(\sum_{i \in I(j)} y_i \right) + \sum_{j \in J_s: \tilde{I}_j = 1} d_j [y_i(j) - 1] + \sum_{j \in J \setminus J_s: \tilde{I}_j = 1} d_j \left(\sum_{i \in I(j)} y_i - 1 \right) \geq D - \sum_{j \in J: \tilde{I}_j \geq 1} d_j. \tag{B2f}$$

Given an integer solution $\tilde{y}$, and the associated set $\tilde{K}$, the corresponding Benders cut reads as follows:

$$\sum_{i \notin \tilde{K}} \left(\sum_{j \in J(i) \setminus J(\tilde{K})} d_j + \sum_{j \in J(i) \cap J_s(\tilde{K})} d_j \right) y_i + \sum_{i \in \tilde{K}} \left(\sum_{j \in J_s(\tilde{K}) \cap J(i)} d_j \right) y_i \geq D - D(J(\tilde{K}) \setminus J_s(\tilde{K})). \tag{B2}$$

Our next result compares the quality of Benders cuts of type (B1f) and (B2f).

Proposition 5. *Given a solution* $\tilde{y}$ *of the restricted master problem, the associated Benders cuts* (B2f) *and* (B1f) *do not dominate each other.*

Proof. Comparing the left-hand-side of cuts (B1f) and (B2f), we observe that they differ in the term

$$\sum_{j \in J \setminus J_s: \tilde{I}_j = 1} d_j \left(\sum_{i \in I(j)} y_i - 1 \right),$$

which is added to the left-hand-side of the cut (B2f), whereas the right-hand-sides of both cuts are identical. The value of this summation can have an arbitrary sign (depending on the current value of the y variables). Hence, when this value is strictly negative the cut (B2f) will dominate the cut (B1f), and for the positive value it will be the other way around. □

In our computational study we assess the computational efficiency of the cuts (B1f) and (B2f), and compare them with standard LP-based separation procedures. Given that the computational effort to calculate all three cuts, (B0f)-(B2f), is the same, and the fact that (B0f) are strictly dominated by (B1f), we refrain from studying the computational relevance of (B0f). We nevertheless decide to keep cuts (B0f) and (B0), as they are the most intuitive ones and, after all, they are even sufficient to derive a valid model in the natural space of y variables.

4.3. Facet-Defining Benders Feasibility Cuts for the PSCLP

Conforti and Wolsey [6] recently proposed the following idea. Given an infeasible point $\tilde{y}$ of the Benders master problem and a *core point* y^0 (a point in the relative interior of the convex hull

of all feasible vectors y), it is possible to derive a Benders feasibility cut which is either a facet or an improper face of the LP-relaxation of (8). This procedure is particularly useful when the dual of the Benders subproblem is degenerate, which is often the case. The procedure requires finding an optimal multiplier λ that defines a new point on the segment between $\tilde{y}$ and y^0, which is supposed to be the furthest point (with respect to y^0) on this line segment which is still within the LP relaxation polyhedron. The authors derive a cut-generating LP whose optimal solution almost surely induces a facet-defining Benders feasibility cut. For PSCLP, this cut-generating LP reads as follows:

$$\min \quad \lambda \tag{18}$$

$$z_j \leq \lambda I_j^0 + (1-\lambda)\tilde{I}_j \qquad j \in J \tag{19}$$

$$\sum_{j \in J} d_j z_j \geq D \tag{20}$$

$$0 \leq z_j \leq 1 \qquad j \in J \tag{21}$$

$$0 \leq \lambda \leq 1, \tag{22}$$

where $I_j^0 = \sum_{i \in I(j)} y_i^0$, $j \in J$.

This LP always has a feasible solution. For $\lambda = 1$, we obtain the solution z^0 associated to the core point y^0 ($z_j^0 = \min\{1, I_j^0\}$). By minimizing the value of λ we are searching for a "minimal modification" of the point $\tilde{z}$ that will result in a feasible subproblem. To derive the associated Benders feasibility cut, we actually solve the associated dual:

$$\max \quad D\gamma - \sum_{j \in J} \sigma_j - \sum_{j \in J} \tilde{I}_j \pi_j \tag{23}$$

$$\pi_j + \sigma_j \geq d_j \gamma \qquad j \in J \tag{24}$$

$$\sum_{j \in J} \left(I_j^0 - \tilde{I}_j \right) \pi_j \leq 1 \tag{25}$$

$$(\pi, \sigma, \gamma) \geq 0. \tag{26}$$

If the optimal solution to this problem is equal to zero, the point $\tilde{y}$ is feasible and, hence, no Benders cut will be generated. Otherwise, the optimal solution $(\tilde{\pi}, \tilde{\gamma}, \tilde{\sigma})$ of this LP is plugged into (B).

We observe that this approach gives us an alternative normalization technique to derive Benders feasibility cuts. Indeed, the unbounded dual Benders subproblem given by (10) is simply modified by adding the normalization hyperplane $\sum_{j \in J} \left(I_j^0 - \tilde{I}_j \right) \pi_j \leq 1$ to the model.

In our implementation, we generate the core point y^0 by finding a subset of facilities $I' \subseteq I$ such that $D(J(I')) \geq D$. For all $i \in I'$ we then set $y_i^0 = 1.1$ and $y_i^0 = 0$, for $i \notin I'$ (note that y^0 is a core point due to the fact that, without loss of generality, constraints $y_i \in \{0, 1\}$ can be replaced by $y_i \in \mathbf{Z}_+$, $i \in I$). Each time our branch-and-cut finds a new incumbent solution, we redefine the

core point correspondingly.

4.4. Strengthening of Benders Cuts

We would like to highlight the property that, because all of the above families of cuts are derived from the extreme points of the same polyhedron, they yield the same LP relaxation value.

Observation 1. *The LP relaxation bound of the restricted master problem obtained by separating any of Benders cuts introduced above (i.e.,* (B0f)*,* (B1f)*,* (B2f) *or facet-defining ones) is equal to the LP relaxation bound obtained by solving the compact model* (1)-(5)*.*

In general, one could strengthen Benders cuts by applying a coefficient rounding procedure. For example, for normalized Benders cuts (14), the rounded ones are obtained as follows:

$$\sum_{i \in I} \min \left\{ \tilde{D}, \sum_{j \in J(i)} \tilde{\pi}_j \right\} y_i \geq \tilde{D}, \tag{27}$$

where $\tilde{D} = D - \sum_{j \in J} \tilde{\sigma}_j$. In particular, if we are given a set $\tilde{K} \subset I$, such that

$$\sum_{j \in J(i)} \tilde{\pi}_j = \begin{cases} \geq \tilde{D}, & \text{for all } i \in \tilde{K}, \\ 0, & \text{otherwise} \end{cases}$$

this constraint boils down to

$$\sum_{i \in \tilde{K}} y_i \geq 1,$$

which is a set-union knapsack cover inequality introduced by [1] in the context of solving the incremental connected facility location problem. The interpretation of these cuts is that opening all facilities from $I \setminus \tilde{K}$ is not sufficient to cover the whole demand D, and hence, at least one more facility from $\tilde{K}$ needs to be open.

In general, applying this rounding procedure to all Benders cuts introduced in this paper can theoretically lead to stronger lower bounds than those obtained from the LP relaxation of the compact formulation.

5. Benders Decomposition for the MCLP

For modeling and solving the MCLP, we exploit the fact that the Benders subproblem associated to a given point $\tilde{y}$ (representing a subset of open facilities) has a very similar structure to the Benders subproblem of the PSCLP. Hence, using the closed formulas given in Propositions 1-4, one can derive Benders cuts for the MCLP. Recall that with the single exact method available in the previous literature on the MCLP [9], optimal solutions were reported for instances with up to 2241 demand points and 74 potential facility locations. With our new approach, due to the fact that

the separation of Benders cuts can be performed in linear time, we are able to derive a first exact method capable of dealing with massive data sets.

We start by formulating the problem in the natural space of variables. Since the variables z (that will be projected out) appear in the objective function, *Benders optimality cuts* are introduced, and the value of the objective function is modeled using an auxiliary variable $\theta \geq 0$. The Benders master problem is now given as

$$\max \left\{ \theta : \ \sum_{i \in I} f_i y_i \leq B, \ \ B'_t(y, \theta) \geq 0, \ t \in P', \ \ y_i \in \{0,1\}, \ i \in I \right\}. \tag{28}$$

Variable θ captures the upper bound on the demand covered by the choice of y variables, P' is the polytope of the dual of the Benders subbproblem and t are extreme points of this polytope. Observe that any binary vector y that satisfies the budget constraint of (28) leads to a feasible MCLP solution. Hence, no Benders feasibility cut is required. For a given solution $\tilde{y}$ of the restricted master, one has to solve the following dual LP to derive the associated Benders optimality cut:

$$\min \left\{ \sum_{j \in J} \left(\tilde{I}_j \pi_j + \sigma_j \right) : \ (\pi, \sigma) \in P' \right\}, \tag{29}$$

where

$$P' = \{ (\pi, \sigma) : \pi_j + \sigma_j \geq d_j, \ j \in J, \quad (\pi, \sigma) \geq 0 \}.$$

We notice that the Benders subproblem (29) is exactly the same as the Benders subproblem for the PSCLP obtained after normalization with $\gamma = 1$, presented by (13). Hence, our results for obtaining optimal dual multipliers in linear time hold for the MCLP as well. So, for example, Benders cuts derived from Proposition 1 are:

$$\sum_{j \in J : \tilde{I}_j < 1} d_j \left(\sum_{i \in I(j)} y_i \right) \geq \theta - \sum_{j \in J : \tilde{I}_j \geq 1} d_j. \tag{B0f}$$

Similarly, the cuts derived from Propositions 2 and 4 are obtained from their counterparts (B1f) and (B2f) by replacing the value of the constant D on the right-hand-side, by the new variable θ.

6. Computational Study

In this computational section, we first introduce our benchmark set of instances and provide implementation details, before we discuss the obtained computational results. All the experiments have been performed on a computer equipped with a 3.40 GHz 8-core Intel Core i7-3770 processor and 16 GB of RAM, running a 64-bit Linux operating system. The source codes were compiled with `gcc 4.8.4` and using `-O3` optimization flag. We used `CPLEX 12.7.0` (called just `CPLEX` for

brevity in what follows) and the `CALLABLE LIBRARIES` framework to implement our branch-and-cut algorithms. `CPLEX` was run in single-threaded mode and all `CPLEX` parameters were set to their default values, except `Preprocessing_Linear` and `MIP_Strategy_CallbackReducedLP`, which were set to zero as recommended by the `CPLEX` user manual to use the `CPXsetusercutcallbackfunc` and `CPXsetlazyconstraintcallbackfunc` callback functions. The latter two functions allow to separate the Benders cuts described in Sections 4 and 5, for fractional and integer solutions, respectively.

6.1. Benchmarking Data Set and Massive Data Set

We have created our testbed of instances following and extending the procedure proposed in by ReVelle et al. [27], where randomly generated PSCLP instances have been introduced with up to 900 points, i.e, $|I| = |J| \leq 900$ (each point representing both a customer and a potential facility location). As previously discussed, our exact approach is specifically designed for the realistic case in which the number of customers is much larger than the number of potential facility locations (i.e., $|J| \gg |I|$). For this reason we have created our instances by setting the number of customers at different values ranging between 10,000 and up to 20 millions, and fixing the number of potential facility locations to 100. The customer demand is generated by drawing a number from the range $[1, 100]$ uniformly at random and rounding it to the nearest integer value. The (x, y) coordinates of the customer and the potential facility locations are chosen uniformly at random from $[0, 30]$. We set the maximum number of facilities to be open to 10, 15 and 20, i.e., we set $f_i = 1$ $(i \in I)$ and $B \in \{10; 15; 20\}$. For each potential facility location i, the set $J(i)$ is comprised by all customers whose Euclidean distance from i is at most $\hat{R}$ (called the *radius of coverage* of a facility). The values of $\hat{R}$ considered in our study depend on the budget level as shown in Table 1. All chosen parameters exactly reflect the instance generation procedure of [27]. In order to create PSCLP instances, the *covering demand* D is defined as a percentage of the total demand $\bar{D} = \sum_{j \in J} d_j$. We choose $D \in \{50\%\bar{D}; 60\%\bar{D}; 70\%\bar{D}\}$ (see Table 1). We keep the same values for $\hat{R}$ as for the MCLP instances. Finally, for each combination of the instance generation input parameters, we create five instances with the same characteristics, varying only the random seed generator.

We divide the testbed of instances into two groups: the first one, called *Benchmarking Data Set* (BDS), has $|J| \in \{10{,}000; 50{,}000; 100{,}000\}$. This set is composed by 210 instances and its major purpose is to serve for a comparison of the relative performance of the different Benders cuts introduced in Section 4. Moreover, BDS is also used to asses the performance of `CPLEX` directly applied to the compact formulations of Section 3 and to asses the performance of the default Benders decomposition implemented in `CPLEX` (these results are discussed in Sections 6.2 and 6.3, respectively).

The second data set, called *Massive Data Set* (MDS), considers nine different values of $|J|$, starting from half a million and going up to 20 millions (see Tables 7 and 8). The major purpose

Budget	Covering Demand	Radius of Coverage
$B = 10$	$D = 50\%\bar{D}$	$\hat{R} \in \{5.5; 5.75; 6; 6.25\}$
$B = 15$	$D = 60\%\bar{D}$	$\hat{R} \in \{4; 4.25; 4.5; 4.75; 5\}$
$B = 20$	$D = 70\%\bar{D}$	$\hat{R} \in \{3.25; 3.5; 3.75; 4; 4.25\}$

Table 1: Random-coordinate data set parameters.

of this data set, which is composed by a total of 630 instances, is to test the computational limits of our exact branch-and-Benders-cut framework (see Section 6.4).

Given the variability of input parameters and different sizes of input data, we believe that our testbed of 840 instances provides a good representative sample allowing us to determine the impact of the main features of the MCLP and PSCLP instances on the structure of optimal solutions and the overall computational performance. All the instances are available upon request from the authors.

Finally, it is worth mentioning that other MCLP instance sets were proposed in the literature, but none of them have the desired characteristics which are required by our exact approach, i.e., $|J| \gg |I|$ and $|I|$ relatively low. For example, in [21] the authors tested their metaheuristic approach on a set of instances with up to ≈ 8000 customers and facilities. In this testbed, $|J| = |I|$ and $|I|$ is relatively high. Most of these instances are challenging for general-purpose MIP solvers. We also tested our branch-and-Benders-cut algorithms on the instances from [21] and they also failed in finding optimal solutions within 10 minutes of computing time. This is not surprising, as Benders decomposition typically draws advantage over a compact formulation, when the size of the restricted master problem can be significantly reduced, compared to the size of the compact formulation. Unfortunately, and contrary to the standard facility location problems, for the MCLP and PSCLP when $|I| = |J|$ the number of decision variables is reduced only by half, which does not allow the Benders approach to draw a competitive advantage. For this reason we do not report results on this set of instances.

6.2. Comparing the Performance of the Different Families of Benders Cuts

In this section we compare the performance of the four different Benders cuts presented in Section 4 for the PSCLP. We tested the following four configurations of our branch-and-Benders-cut algorithm:

- `BEN B1`: where both fractional and integer points are separated using (B1f) and (B1) Benders cuts, respectively.

- `BEN B2`: where both fractional and integer points are separated using (B2f) and (B2) Benders cuts, respectively.
- `BEN RAYS`: where both fractional and integer points are separated using Benders cuts (B), whose coefficients are derived from extreme rays of the polyhedron associated to the dual LP of the Benders subproblem given by (9) (cf. Section 4.1).
- `BEN FACETS`: where both fractional and integer points are separated using facet-defining Benders cuts (B), whose coefficients are derived by solving the LP (23)-(26).

In Table 2, we report average computing times necessary to obtain optimal solutions for the PSCLP instances with $|J| = 10{,}000$. The instances are grouped by the three different values of increasing covering demands D and the column # reports the total number of instances per row. All four settings manage to solve all 70 instances of this set to proven optimality. However, there is an obvious discrepancy between the computational times of the two settings that rely on the separation based on solving an LP (namely, `BEN RAYS` and `BEN FACETS`) and those based on a combinatorial approach which runs in linear time (namely, `BEN B1` and `BEN B2`).

From these tests we can notice that increasing the values of D makes the instances slightly harder to solve. There is no surprise in the fact that the use of a combinatorial algorithm compares favorably to the use of the LP-based algorithm to derive Benders cuts. What is surprising is the order of magnitude of this speed-up which is already impressive for $|J| = 10{,}000$ and tends to increase more and more for bigger values of $|J|$. As far as the `BEN FACETS` cuts are concerned, we notice that on average fewer cuts are generated with respect to the `BEN RAYS` cuts; however, the LP (23) is much harder to solve than dual LP of the Benders subproblem given by (9) (using in both cases the `CPLEX` Linear Programming solver). Finally, as for the PCSLP, the LP-based Benders cut are outperformed by Benders cuts `BEN B1` and `BEN B2` also for the MCLP (see Tables 4 and 6 for detailed results on the latter two configurations).

The quality of the obtained LP bounds is the same for all four settings, which is why in the remainder of this section, we focus on the settings `BEN B1` and `BEN B2`, which outperform the LP-based ones by several orders of magnitudes.

In the following, we analyze the quality of LP relaxation bound of the formulations presented in Section 3 for the PSCLP and MCLP, and the CPU time needed to obtain it. We focus on the two best-performing settings from above, i.e., `BEN B1` and `BEN B2`. For these tests, we consider larger BDS instances with $|J| = 50{,}000$. The results obtained for the PSCLP and MCLP are shown in Tables 3 and 4, respectively; where each line reports the average values obtained for instances with the same features (same D or $\hat{R}$ values, respectively). The tables report the percentage LP gap computed for the PSCLP as $(z(ILP) - z(LP))/z(ILP)$ and for the MCLP as $(z(LP) - z(ILP))/z(LP)$; where $z(ILP)$ is the optimal solution value of an instance and $z(LP)$ is the value of its LP relaxation bound. The tables report then the computational time required by the LP

$\|J\|$	D	#	BEN B1 t[s]	BEN B2 t[s]	BEN REYS t[s]	BEN FACETS t[s]
10,000	$50\%\bar{D}$	20	0.02	0.02	7.81	25.46
	$60\%\bar{D}$	25	0.06	0.04	24.60	38.59
	$70\%\bar{D}$	25	0.17	0.14	16.33	48.22

Table 2: Computing times to solve PSCLP instances with $|J| = 10{,}000$ comparing the performances of four families of Benders Cuts.

solver of `CPLEX` to solve the LP relaxation of the ILP formulations. Finally we report the computing time to obtain the same LP bound by separating the Benders Cuts `BEN B1` and `BEN B2`. For the latter two configurations we also report the average number of generated Benders cuts. For these tests we separated all fractional Benders cuts without considering any violation tolerance, thus obtaining exactly the same bound for all the three methods.

We first observe that the ILP formulations for both problems provide relatively small LP gaps, confirming the results obtained in several other papers as discussed in the introduction of this article. The LP gaps of the PSCLP are on average below 4% and for the MCLP below 1%. The relative difference in the LP gaps between the two families of problems can be explained by the fact that optimal MCLP solution values are much bigger than optimal PSCLP ones for the considered test bed of instances (see Section 6.1 for further details on how the instances have been constructed). For the PSCLP, we observe that increasing the covering demand D results in smaller LP gaps. Similarly, increasing the budget B for the MCLP, reduces the LP gaps. As far as the CPU time necessary to compute the LP relaxation bound is concerned, the two tables clearly show that separating the `BEN B1` and `BEN B2` families of cuts is by far the best option for both problems. For the PSCLP in Table 3 we can notice that `CPLEX` can take up to an average of 30 seconds to solve PSCLP instances with low covering demand D and in all cases it takes on average more than 10 seconds. Solving the root node of our branch-and-Benders-cut algorithm requires instead a fraction of a second. For all these tested PSCLP instances it requires on average less than approximately 0.1 second. A similar comparison emerges also from Table 4 for the MCLP instances, where our branch-and-Benders-cut outperforms `CPLEX` on average by more than one order of magnitude in computing the LP relaxations. Table 4 also shows that, even if the PSCLP and the MCLP instances have been generated with similar parameters, the MCLP instances are computationally much harder. This fact can be explained by the number of cuts generated, i.e., for the MCLP instances many more cuts are necessary to compute the LP bounds with respect to the PSCLP instances.

				CPLEX	BEN B1		BEN B2	
$\|J\|$	D	#	LP gap [%]	t[s]	t[s]	# cuts	t[s]	# cuts
50,000	$50\%\bar{D}$	20	3.93	30.06	0.04	19.9	0.02	10.0
	$60\%\bar{D}$	25	2.78	13.82	0.06	42.3	0.03	21.9
	$70\%\bar{D}$	25	1.47	14.33	0.11	79.1	0.07	53.4

Table 3: LP relaxation bounds for PSC instances with $|J| = 50{,}000$.

				CPLEX	BEN B1		BEN B2	
$\|J\|$	B	#	LP gap [%]	t[s]	t[s]	# cuts	t[s]	# cuts
50,000	10	20	0.56	125.18	9.05	1044.5	8.70	1032.4
	15	25	0.40	191.78	13.08	1158.7	13.51	1156.8
	20	25	0.33	101.15	17.03	1205.6	17.45	1184.1

Table 4: LP relaxation bounds for MCLP instances with $|J| = 50{,}000$.

6.3. Comparison with the State-of-the-art MIP Solver and `CPLEX` *Automatic Benders Procedure*

To the best of our knowledge, no previous computational study on exact approaches appeared in the literature for the PSCLP and for the MCLP, despite their theoretical and practical relevance. For this reason, in this section we compare the performance of our branch-and-Benders-cut framework based on the `BEN B1` and `BEN B2` families of Benders cuts against the direct use of `CPLEX` applied as a black-box MIP solver to the compact formulations of Section 3. `CPLEX` is one of the state-of-the-art MIP solvers and is usually used as benchmark to compare the performance of newly developed exact algorithms; in addition, `CPLEX` recently made available an *Automatic Benders Procedure* which is another good candidate for comparing the performance of our new algorithms. The main difference between our branch-and-Benders-cut and the *Automatic Benders Procedure* of `CPLEX` lies in the way the Benders cuts are determined and separated (see Sections 4 and 5 for the definition of our Benders cuts `BEN B1` and `BEN B2`).

In Tables 5 and 6, we present the results of these performance comparisons for the PSCLP and for the MCLP, respectively. We considered for both problems all the BSD instances described in Section 6.1, instances with 10,000, 50,000 and 100,000 customers. Each row of the tables reports the average computing time necessary to solve instances with similar features, grouping them by increasing covering demand D and budget B (for the PSCLP and the MCLP, respectively). The tables report the results of four exact algorithms: `CPLEX` and its Automatic Bender Procedure (called `AUTO BEN` in the tables) and our two branch-and-Benders-cut algorithms, called directly `BEN B1` and `BEN B2` in the tables. For each algorithm, we also report the number of instances per group solved to proven optimality (columns #). The average CPU times are computed considering only the solved instances and setting a time limit of 600 seconds for all tests.

As far as the PSCLP results are concerned, Table 5 clearly shows that our branch-and-Benders-cut algorithms outperform both `CPLEX` and `AUTO BEN` by several orders of magnitude. Moreover, our exact algorithms are able to solve all the 210 instances of this testbed, while several instances cannot be solved neither by `CPLEX` nor `AUTO BEN`. In particular, these results show that `AUTO BEN` behaves poorly for the PSCLP, a fact which is evident by remarking that none of the instances with $|J| = 100{,}000$ can be solved by this method. `CPLEX` is able to solve all the instances up to dimension $|J| = 50{,}000$, but it slips into several "time limits" for larger instances. Comparing the relative performance of `BEN B1` and `BEN B2`, we can notice that both approaches are very effective, but `BEN B2` is slightly superior. For this reason, we test only `BEN B2` in the final experiments of Section 6.4 for the PSCLP. As expected, increasing the number of customers makes the instances harder for all methods and, as already observed in Table 2, also increasing the covering demands D generally makes the instances harder to solve. It is worth mentioning that `BEN B2` is able to solve the larger and harder instances in less that half a second on average.

As far as the MCLP results are concerned, Table 6 demonstrates a similar behavior, i.e., our branch-and-Benders-cut algorithms largely outperform also for the MCLP instances. For this prob-

			CPLEX		AUTO BEN		BEN B1		BEN B2	
$\|J\|$	D	#	t[s]	# opt	t[s]	# opt	t[s]	# opt	t[s]	# opt
10,000	$50\%\bar{D}$	20	6.53	20	12.50	20	0.02	20	0.02	20
	$60\%\bar{D}$	25	6.60	25	19.36	25	0.06	25	0.04	25
	$70\%\bar{D}$	25	5.59	25	24.27	25	0.17	25	0.14	25
50,000	$50\%\bar{D}$	20	70.23	20	346.92	16	0.06	20	0.04	20
	$60\%\bar{D}$	25	104.48	25	524.45	5	0.18	25	0.11	25
	$70\%\bar{D}$	25	157.04	25	530.70	2	0.36	25	0.32	25
100,000	$50\%\bar{D}$	20	299.79	15	t.l.	0	0.12	20	0.12	20
	$60\%\bar{D}$	25	341.38	16	t.l.	0	0.46	25	0.33	25
	$70\%\bar{D}$	25	306.10	18	t.l.	0	0.59	25	0.49	25

Table 5: Performance comparison on the PSCLP instances with $|J| \in \{10{,}000; 50{,}000; 100{,}000\}$.

lem, the relative comparison between `CPLEX` and `AUTO BEN` is reversed, i.e, `AUTO BEN` outperforms `CPLEX` and it is able to solve more instances to proven optimality. Nevertheless, our `BEN B1` and `BEN B2` approaches outperform `AUTO BEN`, in several cases by one order of magnitude, and they are able to solve all the 210 instances of this MCLP testbed. Again as expected, increasing the number of customers makes the instances harder for all approaches, while increasing the budget makes the instances harder for `AUTO BEN`, `BEN B1` and `BEN B2`, while slightly easier for `CPLEX`. Comparing the relative performance of `BEN B1` and `BEN B2`, we can notice that for the MCLP, `BEN B1` is slightly superior and, for this reason, we test only `BEN B1` in the final experiments of Section 6.4. Finally, these tests show that the MCLP instances are harder than the PSCLP instances and the CPU times for `BEN B1` can reach about half a minute.

More details on these computational tests are given in the Appendix of this article, where tables similar to Tables 5 and 6 are reported. In the appendix tables we present also the results in function of the different values of the Radius of Coverage $\hat{R}$. We did not notice a clear impact on the performance of the algorithm by increasing values of $\hat{R}$. For this reason, we do not report further details in this paper (we refer the interested reader to the Appendix).

A graphical representation of the relative performance of the different exact algorithms is given by the performance profiles of Figures 2 and 3, for the PSCLP and for the MCLP, respectively. For each instance, we compute a normalized time τ as the ratio of the computing time of the considered configuration over the minimum computing time for solving the instance to proven optimality. For

			CPLEX		AUTO BEN		BEN B1		BEN B2	
$\|J\|$	B	#	t[s]	# opt	t[s]	# opt	t[s]	# opt	t[s]	# opt
10,000	10	20	10.83	20	19.68	20	7.28	20	10.76	20
	15	25	7.13	25	36.86	25	15.33	25	27.05	25
	20	25	5.10	25	64.22	23	21.57	25	42.49	25
50,000	10	20	211.75	20	49.34	20	9.28	20	11.13	20
	15	25	162.37	25	72.20	25	16.59	25	18.00	25
	20	25	118.61	25	135.01	23	35.67	25	35.15	25
100,000	10	20	481.87	5	85.11	20	10.58	20	15.11	20
	15	25	466.83	11	132.76	25	20.11	25	27.13	25
	20	25	356.15	13	156.15	21	38.02	25	45.56	25

Table 6: Performance comparison on the MCLP instances with $|J| \in \{10{,}000; 50{,}000; 100{,}000\}$.

each value of τ in the horizontal axis, the vertical axis reports the percentage of the instances for which the corresponding configuration spent at most τ times the computing time of the fastest algorithm. The curves start from the percentage of instances in which the corresponding algorithm is the fastest and at the right end of the chart, we can read the percentage of instances solved by a specific algorithm. The best performance are graphically represented by the curves in the upper part of Figures 2 and 3. The horizontal axis is represented in logarithmic scale. The figures clearly show that our branch-and-Benders-cut algorithms `BEN B1` and `BEN B2` largely outperform `CPLEX` and `AUTO BEN` for the PSCLP and for the MCLP. For both problems, `BEN B1` and `BEN B2` reach 100% of solved instances. For the PSCLP, `CPLEX` is able to solve about 59% and `AUTO BEN` only about 29% of the instances (by allowing 1000 times more time than that taken by `BEN B1` or `BEN B2`). Figure 2 shows that `BEN B2` is the fastest method in about 75% of the instances while `BEN B1` is the fastest in only about 25% of the instances considered. For the MCLP, `CPLEX` is able to solve about 80% and `AUTO BEN` about 96% of the instance (by allowing 100 times more time than that taken by `BEN B1` or `BEN B2`). Figure 3 shows that `BEN B1` is the fastest method in about 46% of the instances while `BEN B2` is the fastest in about 31%. `CPLEX` and `AUTO BEN` are not completely dominated by `BEN B1` and `BEN B2` since `AUTO BEN` is the fastest method in 18% of the instances and `CPLEX` in about 5%.

In Figure 4, we report two optimal MCLP solutions for two instances with $|J| = 10,000$, $B = 10$ and $\hat{R} \in \{5.5; 6.25\}$. In these figures the white circles represent the open facilities and the grey

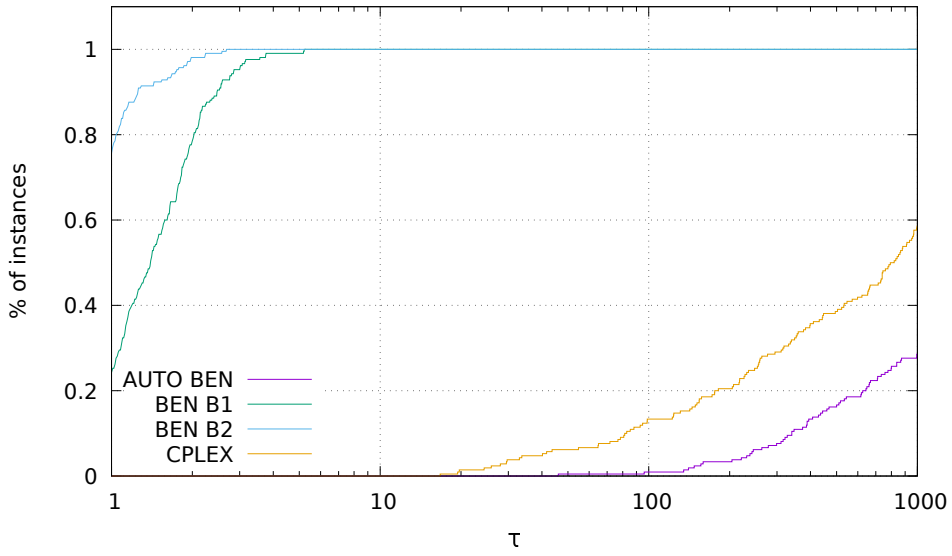

Figure 2: Performance profile for the PSCLP instances with $|J| \in \{10{,}000; 50{,}000; 100{,}000\}$.

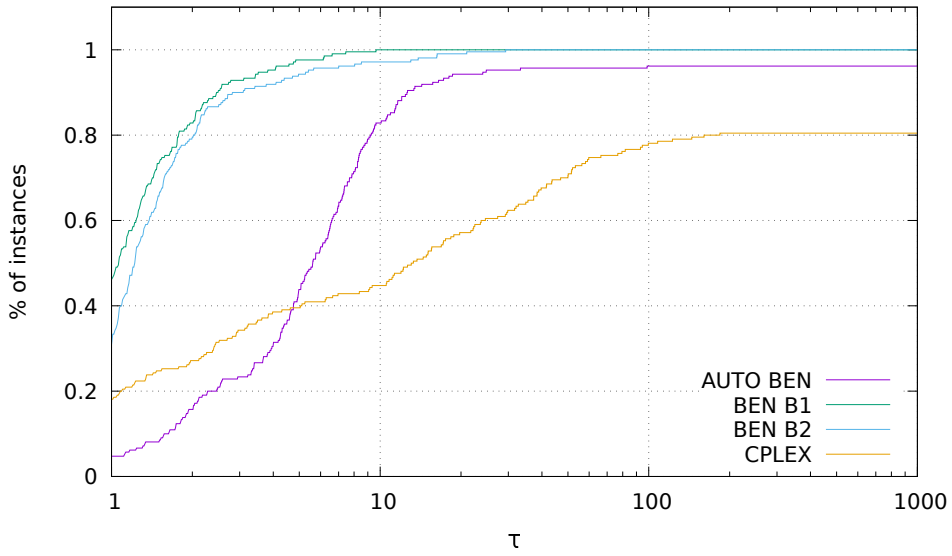

Figure 3: Performance profile for the MCLP instances with $|J| \in \{10{,}000; 50{,}000; 100{,}000\}$.

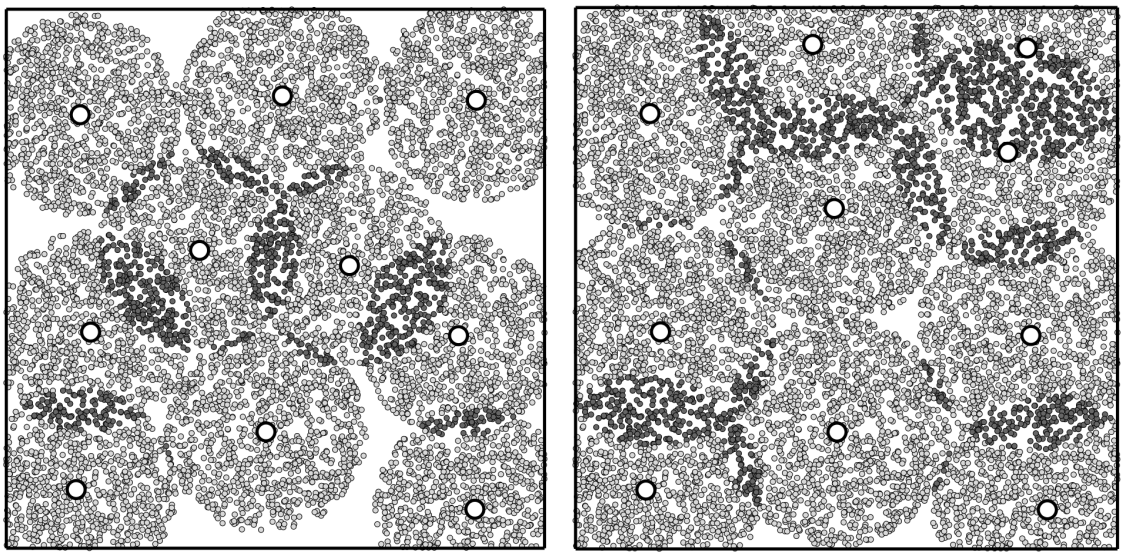

Figure 4: Two optimal MCLP solutions with $|J| = 10{,}000$, $B = 10$ and $\hat{R} \in \{5.5; 6.25\}$.

circles the served customers. Dark grey circles represents customers covered by more than one facility. Both optimal solutions, with different covering radius $\hat{R} = 5.5$ and $\hat{R} = 6.25$, use 10 facilities (fully exploiting the budget B) but not the same ones. As expected, with $\hat{R} = 6.25$, the percentage of covered customers increases, as well as the percentage of customers covered by two or more facilities. This figure gives a graphical intuition of the size of the instances which can be tackled by our branch-and-Benders-cut algorithms and it also demonstrates how our algorithm can be used as a decision support tool. For the sake of readability, we illustrate two instances with 10,000 customers, but in what follows we will demonstrate that decision makers can easily use our approach for dealing with problems involving millions of customers.

6.4. Computational Results on the Instances From the Massive Data Set

In this section we test the computational limits of our branch-and-Benders-cut algorithms, using the MDS instances presented in Section 6.1. These PSCLP and MCLP instances have up to 20 millions customers. In these tests, in order to avoid the generation of too many Benders cuts for fractional points, we stop the separation as soon as the relative cut violation goes below 1%. This relative violation is computed as the relative ratio between the difference of the covering demand D and the current value of the left hand side of the Benders cut, with respect to the value of D. Thanks to extensive preliminary tests, we determined that 1% is a good compromise between the quality of the relaxation and the number of generated Benders cuts for fractional solutions.

In Tables 7 and 8, we report the results for the PSCLP and for the MCLP, respectively. As discussed in Section 6.3, we test only the best branch-and-Benders-cut algorithm for each problem, i.e., `BEN B2` for the PSCLP and `BEN B1` for the MCLP. These two tables report the average results

of 70 instances per row, grouped by increasing number of customers $|J|$ (starting from half a million and going up to 20 millions). The tables report the following information: the number of instances solved to proven optimality (column # Opt), the average computing time (considering only solved instances), the average number of Benders cuts derived from integer and fractional points (columns # `B2` int., # `B2` fract., # `B1` int., # `B1` fract.) and finally the number of branching nodes explored (column # nodes).

As far as the PSCLP results are concerned, Table 7 shows that `BEN B2` is able to solve all the 630 instances of the MDS. The average CPU time goes from several seconds up to more than 100 seconds for the instances with 20 millions customers. The number of Benders cuts derived from integer points is relatively low, while dozens of Benders cuts are generated from fractional points. The number of explored nodes is relatively low and this figure never goes above 100 nodes, a fact which reflects the good quality of the LP relaxation bounds.

As far as the MCLP results are concerned, Table 8 shows that `BEN B2` is able to solve to proven optimality all the instances with up to 1,500,000 customers. The time limit imposed in this set of experiments is again 600 seconds. By increasing the number of customers, the number of instances that can be solved to optimality decreases, so that none of the instances with 20 millions customers can be solved within 10 minutes. The average CPU time ranges between 50 seconds for half a million of customers to approximately 100 seconds for 1.5 million customers. For the larger instances, the required CPU time for instances with 15 millions customers is about 400 seconds. The number of Benders cuts derived from integer points is relatively low (but larger compared to the PSCLP instances), while several hundreds (up to about 1500) of Benders cuts are generated at fractional points. The number of explored nodes is still relatively low and this figure never goes beyond 200 nodes, which can be also explained by the high quality of the LP relaxation bounds.

7. Conclusions

In this article we study the Partial Set Covering Location Problem and the Maximal Covering Location Problem, two location problems which require choosing a subset of facilities i) minimizing the cost of the open facilities while covering a predetermined fraction of customer demand and ii) maximizing the covered demand respecting a budget constraint for open facilities, respectively. These two important problems have not received much attention in the literature despite their theoretical and practical relevance. In this article we propose the first exact algorithm to effectively tackle realistic PSCLP and MCLP instances with millions of demand points – instances that are far beyond the reach of modern general-purpose MIP solvers. We managed to achieve these results thanks to effective branch-and-Benders-cut algorithms that exploit a combinatorial cut-separation procedure.

Our decomposition algorithms are specifically conceived for PSCLP and MCLP instances with much more customers than facilities, i.e., for $|J| \gg |I|$ and $|I|$ relatively low. Future lines of research

$\|J\|$	#	# opt	t[s]	# BEN B2 int.	# BEN B2 frac.	# nodes
500,000	70	70	1.75	4.20	40.00	64.90
1,000,000	70	70	4.14	3.77	31.16	61.03
1,500,000	70	70	7.51	4.20	31.36	58.49
2,000,000	70	70	8.51	4.33	28.94	44.76
4,000,000	70	70	23.35	3.80	31.51	64.04
6,000,000	70	70	28.20	3.99	30.17	48.51
10,000,000	70	70	55.76	3.80	34.96	60.50
15,000,000	70	70	109.61	4.39	42.44	80.47
20,000,000	70	70	117.25	5.12	36.21	58.01

Table 7: Computational performance of BEN B2 on massive PSCLP data sets.

$\|J\|$	#	# opt	t[s]	# BEN B1 int.	# BEN B1 frac.	# nodes
500,000	70	70	50.42	77.34	1559.97	175.84
1,000,000	70	70	99.24	80.07	1589.63	167.47
1,500,000	70	70	151.07	78.27	1573.73	183.53
2,000,000	70	69	184.71	80.45	1494.55	167.26
4,000,000	70	60	285.39	78.38	1243.37	143.55
6,000,000	70	45	320.80	83.76	889.67	109.07
10,000,000	70	26	377.19	76.73	594.58	72.35
15,000,000	70	10	377.24	68.90	382.80	54.40
20,000,000	70	0	-	-	-	-

Table 8: Computational performance of BEN B1 on massive MCLP data sets.

will go in the direction of developing exact algorithms for situations in which a concave utility function is used to express the value of the covered demand and a connection to the submodular cuts will be explored.

The PSCLP and MCLP are relevant problems in the field of data science, in the context of clustering and classification. Hence, as a future line of research it would be interesting to study data-driven optimization using PSCLP and MCLP. To this end, various types of data uncertainty need to be embedded into the PSCLP and MCLP models, and decomposition algorithms need to be adapted to deal with massive data sets.

References

[1] A. Arulselvan, A. Bley, and I. Ljubić. MIP Modeling of Incremental Connected Facility Location, 2018. Submitted.

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

[3] O. Berman, D. Krass, and Z. Drezner. The gradual covering decay location problem on a network. *European Journal of Operational Research*, 151(3):474–480, 2003.

[4] N. Bilal, P. Galinier, and F. Guibault. An iterated-tabu-search heuristic for a variant of the partial set covering problem. *Journal of Heuristics*, 20(2):143–164, 2014.

[5] R. Church and C. ReVelle. The maximal covering location problem. *Papers in Regional Science*, 32(1):101–118, 1974.

[6] M. Conforti and L. Wolsey. "Facet" separation with one linear program. *CORE Discussion Paper*, 21, 2016. URL `http://hdl.handle.net/2078.1/174860`.

[7] M. S. Daskin and S. H. Owen. Two new location covering problems: The partial p-center problem and the partial set covering problem. *Geographical Analysis*, 31(3):217–235, 1999.

[8] M. S. Daskin, A. E. Haghani, M. Khanal, and C. Malandraki. Aggregation effects in maximum covering models. *Annals of Operations Research*, 18(1):113–139, 1989.

[9] B. T. Downs and J. D. Camm. An exact algorithm for the maximal covering problem. *Naval Research Logistics*, 43(3):435–461, 1996.

[10] R. Z. Farahani, N. Asgari, N. Heidari, M. Hosseininia, and M. Goh. Covering problems in facility location: A review. *Computers & Industrial Engineering*, 62(1):368–407, 2012.

[11] M. Fischetti, D. Salvagnin, and A. Zanette. A note on the selection of Benders' cuts. *Mathematical Programming*, 124(1):175–182, 2010.

[12] M. Fischetti, I. Ljubić, and M. Sinnl. Benders decomposition without separability: A computational study for capacitated facility location problems. *European Journal of Operational Research*, 253(3):557–569, 2016.

[13] M. Fischetti, I. Ljubić, and M. Sinnl. Redesigning Benders decomposition for large-scale facility location. *Management Science*, 63(7):2146–2162, 2017.

[14] R. D. Galvão and C. ReVelle. A Lagrangean heuristic for the maximal covering location problem. *European Journal of Operational Research*, 88(1):114–123, 1996.

[15] R. D. Galvão, L. G. A. Espejo, and B. Boffey. A comparison of Lagrangean and surrogate relaxations for the maximal covering location problem. *European Journal of Operational Research*, 124(2):377–389, 2000.

[16] S. García and A. Marín. Covering location problems. In *Location Science*, pages 93–114. Springer, 2015.

[17] 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.

[18] G. Laporte, S. Nickel, and F. S. da Gama. *Location Science*. Springer, 2015.

[19] I. Ljubić, P. Putz, and J. J. S. González. Exact approaches to the single-source network loading problem. *Networks*, 59(1):89–106, 2012.

[20] T. L. Magnanti and R. T. Wong. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484, 1981.

[21] V. R. Máximo, M. C. Nascimento, and A. C. Carvalho. Intelligent-guided adaptive search for the maximum covering location problem. *Computers & Operations Research*, 78:129–137, 2017.

[22] N. Megiddo, E. Zemel, and S. L. Hakimi. The maximum coverage location problem. *SIAM Journal on Algebraic Discrete Methods*, 4(2):253–261, 1983.

[23] A. T. Murray. Maximal coverage location problem: impacts, significance, and evolution. *International Regional Science Review*, 39(1):5–27, 2016.

[24] N. Papadakos. Practical enhancements to the Magnanti-Wong method. *Operations Research Letters*, 36:444–449, 2008.

[25] F. Plastria. Continuous covering location problems. *Facility location: applications and theory*, 1:37–79, 2002.

[26] C. ReVelle. Facility siting and integer-friendly programming. *European Journal of Operational Research*, 65(2):147–158, 1993.

[27] C. ReVelle, M. Scholssberg, and J. Williams. Solving the maximal covering location problem with heuristic concentration. *Computers & Operations Research*, 35(2):427–435, 2008.

[28] E. L. F. Senne, M. A. Pereira, and L. A. N. Lorena. A decomposition heuristic for the maximal covering location problem. *Advances in Operations Research*, 2010, 2010.

[29] L. V. Snyder. Covering problems. In *Foundations of Location Analysis*, pages 109–135. Springer, 2011.

[30] C. Toregas, R. Swain, C. ReVelle, and L. Bergman. The location of emergency service facilities. *Operations Research*, 19(6):1363–1373, 1971.

[31] W. Walker. Using the set-covering problem to assign fire companies to fire houses. *Operations Research*, 22(2):275–277, 1974.

[32] R. Wei and A. T. Murray. Continuous space maximal coverage: insights, advances and challenges. *Computers & Operations Research*, 62:325–336, 2015.

[33] M. F. Zarandi, S. Davari, and S. H. Sisakht. The large scale maximal covering location problem. *Scientia Iranica*, 18(6):1564–1570, 2011.

Appendix

Detailed computational results

				CPLEX		AUTO BEN		BEN B1		BEN B2	
$\|J\|$	D	$\hat{R}$	#	t[s]	# opt	t[s]	# opt	t[s]	# opt	t[s]	# opt
10,000	$50\%\bar{D}$	5.5	5	8.89	5	17.66	5	0.04	5	0.04	5
		5.75	5	4.94	5	8.03	5	0.02	5	0.02	5
		6	5	6.45	5	13.75	5	0.03	5	0.02	5
		6.25	5	5.82	5	10.57	5	0.01	5	0.01	5
	$60\%\bar{D}$	4	5	4.17	5	20.33	5	0.07	5	0.05	5
		4.25	5	4.59	5	17.14	5	0.05	5	0.04	5
		4.5	5	8.96	5	21.75	5	0.07	5	0.06	5
		4.75	5	5.40	5	14.82	5	0.04	5	0.02	5
		5	5	9.91	5	22.74	5	0.06	5	0.05	5
	$70\%\bar{D}$	3.25	5	6.01	5	23.40	5	0.32	5	0.30	5
		3.5	5	3.07	5	22.85	5	0.15	5	0.14	5
		3.75	5	4.03	5	25.14	5	0.12	5	0.08	5
		4	5	5.78	5	25.45	5	0.13	5	0.09	5
		4.25	5	9.08	5	24.49	5	0.11	5	0.07	5
50,000	$50\%\bar{D}$	5.5	5	53.00	5	273.67	3	0.06	5	0.04	5
		5.75	5	40.69	5	377.79	5	0.04	5	0.02	5
		6	5	125.06	5	357.99	4	0.07	5	0.05	5
		6.25	5	62.19	5	352.19	4	0.05	5	0.04	5
	$60\%\bar{D}$	4	5	97.94	5	t.l.	0	0.29	5	0.17	5
		4.25	5	88.95	5	t.l.	0	0.14	5	0.08	5
		4.5	5	150.06	5	485.69	2	0.20	5	0.17	5
		4.75	5	78.43	5	575.47	1	0.13	5	0.10	5
		5	5	107.00	5	537.69	2	0.14	5	0.05	5
	$70\%\bar{D}$	3.25	5	166.67	5	523.83	1	0.62	5	0.47	5
		3.5	5	129.53	5	t.l.	0	0.32	5	0.28	5
		3.75	5	166.41	5	t.l.	0	0.38	5	0.37	5
		4	5	130.81	5	t.l.	0	0.27	5	0.25	5
		4.25	5	191.78	5	537.57	1	0.22	5	0.21	5
100,000	$50\%\bar{D}$	5.5	5	436.20	5	t.l.	0	0.11	5	0.12	5
		5.75	5	171.24	3	t.l.	0	0.09	5	0.14	5
		6	5	247.31	5	t.l.	0	0.10	5	0.05	5
		6.25	5	282.79	2	t.l.	0	0.20	5	0.18	5
	$60\%\bar{D}$	4	5	343.53	3	t.l.	0	0.67	5	0.40	5
		4.25	5	295.50	4	t.l.	0	0.26	5	0.15	5
		4.5	5	295.23	5	t.l.	0	0.32	5	0.15	5
		4.75	5	418.37	2	t.l.	0	0.45	5	0.49	5
		5	5	468.30	2	t.l.	0	0.62	5	0.45	5
	$70\%\bar{D}$	3.25	5	273.77	5	t.l.	0	0.55	5	0.41	5
		3.5	5	345.65	4	t.l.	0	0.62	5	0.71	5
		3.75	5	292.74	4	t.l.	0	0.65	5	0.44	5
		4	5	357.60	4	t.l.	0	0.46	5	0.39	5
		4.25	5	157.04	1	t.l.	0	0.68	5	0.49	5

Table 9: Performance comparison on the PSCLP instances with $|J| \in \{10{,}000; 50{,}000; 100{,}000\}$.

				CPLEX		AUTO BEN		BEN B1		BEN B2	
$\|J\|$	B	$\hat{R}$	#	t[s]	# opt	t[s]	# opt	t[s]	# opt	t[s]	# opt
10,000	10	5.5	5	12.54	5	25.41	5	9.07	5	9.18	5
		5.75	5	11.30	5	13.35	5	6.50	5	8.47	5
		6	5	10.90	5	16.57	5	6.79	5	13.20	5
		6.25	5	8.59	5	23.38	5	6.75	5	12.18	5
	15	4	5	4.72	5	11.83	5	6.05	5	6.60	5
		4.25	5	5.47	5	13.28	5	6.42	5	6.18	5
		4.5	5	7.31	5	43.65	5	11.87	5	19.96	5
		4.75	5	9.34	5	64.88	5	31.41	5	60.17	5
		5	5	8.80	5	50.67	5	20.89	5	42.32	5
	20	3.25	5	3.57	5	7.44	5	6.65	5	4.32	5
		3.5	5	4.05	5	39.30	5	12.21	5	10.95	5
		3.75	5	6.28	5	45.48	4	32.57	5	78.13	5
		4	5	6.83	5	190.03	4	31.47	5	73.75	5
		4.25	5	4.78	5	60.28	5	24.92	5	45.28	5
50,000	10	5.5	5	237.53	5	49.88	5	6.22	5	7.68	5
		5.75	5	229.72	5	43.48	5	8.50	5	6.54	5
		6	5	223.08	5	49.02	5	9.93	5	12.03	5
		6.25	5	156.66	5	54.97	5	12.49	5	18.26	5
	15	4	5	116.54	5	29.26	5	8.44	5	6.19	5
		4.25	5	146.39	5	48.96	5	13.11	5	10.77	5
		4.5	5	196.35	5	100.00	5	17.03	5	17.28	5
		4.75	5	180.02	5	97.77	5	21.33	5	25.16	5
		5	5	172.56	5	85.01	5	23.07	5	30.63	5
	20	3.25	5	106.60	5	25.18	5	13.89	5	7.69	5
		3.5	5	137.37	5	134.47	5	36.36	5	25.84	5
		3.75	5	130.89	5	162.52	4	54.48	5	42.21	5
		4	5	129.90	5	251.06	5	40.99	5	40.69	5
		4.25	5	88.27	5	100.41	4	32.66	5	59.31	5
100,000	10	5.5	5	t.l.	0	82.72	5	8.22	5	9.73	5
		5.75	5	561.11	1	78.92	5	11.13	5	10.64	5
		6	5	573.94	1	89.39	5	12.21	5	19.32	5
		6.25	5	424.77	3	89.41	5	10.76	5	20.75	5
	15	4	5	434.18	3	66.58	5	11.12	5	10.80	5
		4.25	5	452.15	3	80.77	5	11.15	5	13.35	5
		4.5	5	508.79	3	154.51	5	28.73	5	40.37	5
		4.75	5	506.49	1	168.54	5	23.55	5	28.14	5
		5	5	443.25	1	193.38	5	26.00	5	42.96	5
	20	3.25	5	400.79	4	49.29	5	13.35	5	10.88	5
		3.5	5	369.68	2	76.06	4	25.90	5	22.16	5
		3.75	5	203.81	1	305.23	4	49.36	5	53.20	5
		4	5	407.50	3	205.86	4	58.68	5	65.69	5
		4.25	5	287.02	3	171.00	4	42.80	5	75.87	5

Table 10: Performance comparison on the MCLP instances with $|J| \in \{10{,}000; 50{,}000; 100{,}000\}$.