Benders Decomposition for the Discrete Ordered Median Problem

Ivana Ljubić[a], Miguel A. Pozo[b,c], Justo Puerto[b,c] and Alberto Torrejón[b,c,*]

[a]*ESSEC Business School of Paris, France.*

[b]*Department of Statistics and Operational Research. Faculty of Mathematics, University of Seville, Spain.*

[c]*Institute of Mathematics of the University of Seville, Spain.*

E-mail: ljubic@essec.edu [Ivana Ljubić]; miguelpozo@us.es [Miguel A. Pozo]; puerto@us.es [Justo Puerto]; atorrejon@us.es [Alberto Torrejón]

Abstract

Ordered median optimization has been proven to be a powerful tool to generalize many well-known problems from the literature. In Location Theory, the Discrete Ordered Median Problem (DOMP) is a facility location problem where clients are first ranked according to their allocation cost to the nearest open facility, and then these costs are multiplied by a suitable weight vector λ. That way, DOMP generalizes many well-known discrete location problems including p-median, p-center or centdian. In this article, we also allow negative entries of λ, allowing us to derive models for better addressing equity and fairness in facility location, for modeling obnoxious facility location problems or for including other client preference models. We present new mixed integer programming models for DOMP along with algorithmic enhancements for solving the DOMP to optimality using mixed integer programming techniques. Specifically, starting from state-of-the-art formulations from the literature, we present several Benders decomposition reformulations applied to them. Using these approaches, new state-of-the-art results have been obtained for different ordered weighting vectors.

Keywords: Discrete location; Ordered median optimization; Benders decomposition, Branch-and-Benders-cut.

1. Introduction

The Discrete Ordered Median Problem (DOMP) is a p-facility location problem which consists of finding a set of p facilities to open so as to minimize the allocation costs of clients to facilities evaluated with an ordered median objective function. Given a vector of weights, this function sorts clients' allocation costs in non-decreasing order and then performs the scalar product with the given vector of weights. Thus, DOMP adds a sorting feature to the underlying location problem. The objective is then to minimize the weighted sum of sorted allocation costs, where the weights can be interpreted as rank dependent compensation factors applied to clients. DOMP objective function has been successfully applied in the field of discrete location (see Puerto and Tamir, 2005; Boland et al., 2006; Puerto, 2008; Marín et al., 2009; Kalcsics et al., 2010a,b; Puerto and Rodríguez-Chía, 2015; Pozo et al., 2023) and continuous location analysis (see, e.g., Blanco et al., 2013, 2014, 2016, 2023). DOMP structure has also been embedded within other well-known problems giving rise to, e.g., ordered weighted average problems (see Galand and Spanjaard, 2012; Fernández et al., 2014, 2017) or ordered hub location problems (see Puerto et al., 2011, 2013, 2016; Pozo et al., 2021), among others.

DOMP allows a straightforward and flexible generalization of facility location problems since many well-known problems in the literature can be modeled using an ordered formulation. This is the case for p-median, p-center or centdian problems, but also, undesirable or obnoxious location problems since non-monotonous and negative vectors of weights are allowed. This generalization of many location problems or client modeling preferences is addressed by means of a suitable λ-vector of weights to plug in the ordered median objective function which would be described in Section 2. As a particular application, in the context of fairness of allocation to open facilities, DOMP can be used to create a common framework for inclusion and analysis of the concept of equity in location problems. Indeed, compensation of allocation costs is based on the fact that a solution that is good for the system does not have to be acceptable for all single parties if their costs to reach their open facilities are too high in comparison to other parties. In this way, the possibility of modeling different forms of equity allows for circumventing unfair situations. For example, given the non-decreasing ordered sequence of allocation costs, by using a decreasing weight vector λ, clients with high allocation costs can be compensated (this is known as *reverse criterion*). Several other criteria such as the *range* or *absolute deviations* can also be considered (see, e.g., Mesa et al., 2003).

Since the seminal paper by Nickel (2001), several formulations and methods have been proposed to solve DOMP. In Boland et al. (2006) tailored branch-and-bound algorithms were presented which allow to solve problems of small size. More sophisticated methods, based on new formulations using radius variables, were presented in Puerto (2008); Marín et al. (2009, 2010); Labbé et al. (2017). Some decomposition approaches were also tried. In Redondo et al. (2016), a Lagrangian decomposition was applied to DOMP, in Deleplanque et al. (2020) a first branch-and-price-and-cut algorithm was specifically tailored for this problem and in Martínez-Merino et al. (2022) a row generation algorithm was also designed. More recently, a new solution approach that exploits partial monotonicity in efficient representations of the objective function has been presented in Marín et al. (2020). In spite of these many methods applied to solve DOMP, Benders decomposition has been never used to address this problem. Developing competitive Benders decompositions for DOMP and assessing their usefulness by comparing them with other methodologies is one of the goals of this paper.

Our contribution

Benders decomposition is one of the most famous solution methods for solving difficult combinatorial optimization problems (Benders, 1962). Its hallmark is the capability of decomposing mixed-integer programming formulations into smaller subproblems, each of which can be solved individually to produce local information (notably, cutting planes) to be exploited by a centralized master problem. In general terms, the continuous variables of the original problem are projected out, resulting into an equivalent model with fewer variables, but many more constraints. To find an optimal solution, a large number of these constraints will not be required, suggesting then a strategy of separation that ignores all but a few of these constraints. The technique relies on projection and solving of smaller subproblems, combined with solution strategies of dualization, outer linearization and relaxation (Minoux, 1986; Lasdon, 2002). Over the years, several improvements have been proposed in order to enhance the performance of the Benders decomposition algorithm, (see Geoffrion, 1972; Magnanti and Wong, 1981; Fischetti et al., 2010; Fortz and Poss, 2009; Naoum-Sawaya and Elhedhli, 2013; Conforti and Wolsey, 2019; Brandenberg and Stursberg, 2021; Ramírez-Pico et al., 2023). The reader could find a more detailed description about improvements in Benders decomposition in the survey of Rahmaniani et al. (2017).

The recent successful contributions of applying Benders decomposition on location problems (see Fischetti et al., 2016, 2017; Cordeau et al., 2019; Coniglio et al., 2022; Duran-Mateluna et al., 2023; Gaar and Sinnl, 2022) and the lack of any attempt to solve DOMP using this technique motivate the developments in the current work. In this article we start with two state-of-the-art MILP formulations for the DOMP (Labbé et al., 2017; Marín et al., 2020), and we demonstrate how to further boost their computational performance by applying a clever but simple Benders decomposition.

As we will show below, after applying Benders machinery to the structure of our problem, we obtain state-of-the-art results for the most difficult realizations of the DOMP in terms of the compensation vector λ. Indeed, most of the problems addressed in the previous DOMP literature concern some specific, particular classes of λ-vectors, in many cases assuming non-negative values of λ. Moreover, in those very few cases where some negative entries of λ were allowed, the structure of the λ-vectors always ensured that the resulting objective function was positive. Removing this assumption renders DOMP more difficult and brings interesting new features to the problem which we analyze in this paper. In addition, although monotonicity of the λ-vector was well-studied in several papers about DOMP, exploiting its *partial monotonicity* was discussed in detail for the first time in Marín et al. (2020). However, until now, there were no previous computational studies in which a suitable methodology for solving problems with negative or partially negative λ-vectors (resulting in a negative objective function value), was developed in depth. Also, to the best of our knowledge, this is the first time that the ordered median function is used to describe and solve obnoxious facility location problems by choosing appropriate λ-vectors.

The remainder of the paper is organized as follows. In Section 2, we formally define the DOMP introducing the notation that will follow. In Section 3 we review the radius formulation and propose a first Benders decomposition approach. In Section 4, we present the current state-of-the-art formulation and three possible Benders decomposition reformulations. We also propose some improvements for the bounds of our models and the strengthening of the Benders cuts. Section 5 shows the implementation details made in order to enhance the proposed models. The empirical performance of the resulting DOMP formulations is analyzed in Section 6, where we depict the comparison of these formulations for different cases. Finally, some conclusions are summarized in Section 7.

2. Formal problem definition

We assume that potential facilities and customers share the same set of locations indexed by the set $N = \{1, ..., n\}$. This can be done without loss of generality, as otherwise, one could introduce dummy nodes and set the facility opening and allocation costs accordingly. Assume that we are given a cost matrix $C = (c_{ij})_{i,j \in N}$, where $c_{ij} \geq 0$ represents the cost of allocating the client placed at position i to the facility opened at location j. We consider the more general situation in which we do not assume free self-service, therefore in general c_{ii} is not necessarily equal to zero. DOMP assumes single-allocation, i.e., each client node has a unique facility where it is allocated. In order to model the problem, let us consider binary variables $y_j = 1$ ~~iff~~ **if and only if** the facility at position j is open and zero otherwise; and $x_{ij} = 1$ ~~iff~~ **if and only if** client placed at position i is allocated to the facility at j and zero otherwise. Then, the p-median polytope can be described as

$$X = \{x \in [0,1]^{n \times n}, y \in [0,1]^n : \sum_{j \in N} y_j = p; \quad \sum_{j \in N} x_{ij} = 1, \ \forall i \in N; \quad x_{ij} \leq y_j, \ \forall i, j \in N; \tag{1a}$$

$$\sum_{\substack{j \in N: \\ c_{ij} > c_{im}}} x_{ij} + y_m \leq 1, \ \forall i, m \in N\}. \tag{1b}$$

Since negative vector weights will be considered, in order **to ensure that each client is allocated to a closest open facility (ties are randomly broken), the description of the p-median polytope includes the closest assignment constraints of clients to facilities (1b) reported in Espejo et al. (2012). These constraints guarantee that a client i cannot be allocated to a facility j if allocating i to j is costlier than allocating it to given facility m (that is, if $c_{ij} > c_{im}$), provided than facility m is opened.** In the following, we will also refer to the integer lattice points within X as X_I. For a given feasible point $(x, y) \in X_I$, its assignment costs are given as:

$$c(x) := \left(\sum_{j \in N} c_{1j} x_{1j}, \ldots, \sum_{j \in N} c_{nj} x_{nj} \right).$$

We can now provide a formal definition of the problem.

Definition 1 (DOMP). For a given solution $(x, y) \in X_I$, let us consider a reordering of $c(x)$ that satisfies $c_{(1)}(x) \leq c_{(2)}(x) \leq \cdots \leq c_{(n)}(x)$. Given a vector of (not necessarily non-negative) weights $(\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^n$, DOMP seeks to find a solution $(x, y) \in X_I$ such that $\sum_{k \in N} \lambda_k c_{(k)}(x)$ is minimized.

Given a vector d of n real numbers, **and $(d_{(\ell)})_{\ell=1,\ldots,n}$ are the components of d in increasing order**, let $S_k(d) = \sum_{\ell=k}^{n} d_{(\ell)}$ denote the sum of the largest $n - k + 1$ elements of d. Then the objective function of DOMP can be restated as

$$\sum_{k \in N} \lambda_k c_{(k)}(x) = \sum_{k \in N} \Delta_k S_k(c(x)), \tag{2}$$

where $\lambda_0 := 0$ and $\Delta_k := \lambda_k - \lambda_{k-1}$ for all $k \in N$. For further purposes, let us also define the sets $\Delta^- = \{k \in N | \Delta_k < 0\}$ and $\Delta^+ = \{k \in N | \Delta_k > 0\}$.

Using different definitions of the λ-vector of weights, or its Δ description given by (2), many prominent facility location problems can be defined. For instance, we obtain the median with $\lambda = (1, 1, ..., 1, 1)$, the center with $\lambda = (0, 0, ..., 0, 1)$, the α-centdian with $\lambda = (\alpha, \alpha, ..., \alpha, \alpha, 1)$, the Hurwitz criterion with $\lambda = (\alpha, 0, ..., 0, 1 - \alpha)$, the k-max criterion with $\lambda = (0, \dots, 0, \underbrace{1}_{k-th}, 0, \dots, 0)$, the sum of absolute differences with $\lambda = (2(2i - n - 1))_{i \in N}$ or the range with $\lambda = (-1, 0, ..., 0, 1)$, among many others.

DOMP is NP-hard, as it contains the p-median problem as a special case ($\lambda_i = 1, \forall i \in N$). Many different MILP formulations for DOMP have been studied in the past two decades, see Nickel and Puerto (2006); Boland et al. (2006); Kalcsics et al. (2010b); Marín et al. (2009, 2010); Puerto et al. (2011, 2013, 2016); Labbé et al. (2017); Marín et al. (2020). In the following section, we will focus on two state-of-the-art formulations and show how to efficiently apply Benders decomposition to them. The first one is a *radius formulation* for which a valid Benders reformulation can be obtained for non-increasing values of λ. The second one is a *partial monotonicity* formulation that uses three-index variables to ensure the ordering of allocation costs. We show that the second approach is more general and it allows to derive a valid Benders reformulation for any value of $\lambda \in \mathbb{R}^n$.

3. Radius formulation

3.1. Compact radius formulation

Let us consider the ordered sequence of unique allocation costs of $c_{ij} \geq 0$ for $i, j \in N$:

$$c_{(0)} := 0 < c_{(1)} < c_{(2)} < \ldots < c_{(g)} = \max_{i,j \in N} c_{ij},$$

where g is the number of different non-zero elements of the above allocation cost sequence and let $G := \{1, \dots, g\}$. This ordering can be used to perform the sorting process of the allocation costs. Let us define the following variables (namely *radius variables* in the following) for $\ell \in N, h \in G$:

$$r_{\ell h} = \begin{cases} 1 & \text{if the } \ell\text{-th allocation cost is at least } c_{(h)}, \\ 0 & \text{otherwise,} \end{cases}$$

that is to say, the ℓ-th smallest cost is equal to $c_{(m)}$ if and only if $r_{\ell m} = 1$ and $r_{\ell m+1} = 0$, $1 \leq m < g$. As observed in Puerto et al. (2011, 2013); Pozo et al. (2021, 2023), DOMP belongs to a rather difficult family of problems that combines elements from two hard combinatorial optimization problems, namely location and scheduling or ordering. Formulations using radius variables (see Elloumi et al., 2004; Nickel and Puerto, 2006; Puerto, 2008; Espejo et al., 2009; Marín et al., 2009, 2010; Puerto et al., 2011; García et al., 2011) can be used to reduce the number of variables and constraints in use. Indeed, note that radius formulation is defined over a set G whose cardinality depends on the number of different non-zero elements in the cost matrix. Thus, this formulation draws the advantage of repeated entries in the cost matrix (see Pozo et al., 2023). This happens, for instance, when our location problem assumes free self-service and symmetrical allocation costs, what implies a significant number of repetitions in the cost matrix and a diagonal of zeros. A general scheme which models DOMP using radius variables is shown below:

$$
(\mathsf{DOMP}_{r0}) \begin{cases}
\min \displaystyle\sum_{\ell\in N}\sum_{h\in G}\lambda_\ell r_{\ell h}(c_{(h)}-c_{(h-1)}) & & \text{(3a)}\\
\text{s.t.:} \displaystyle\sum_{\ell\in N} r_{\ell h} = \sum_{\substack{i,j\in N:\\ c_{ij}\geq c_{(h)}}} x_{ij} & h\in G & \text{(3b)}\\
r_{\ell h}\leq r_{\ell+1 h} & \forall h\in G, \forall \ell\in N: \ell<|N| & \text{(3c)}\\
r_{\ell h}\geq r_{\ell h+1} & \forall h\in G, \forall \ell\in N: h<|G| & \text{(3d)}\\
(x,y)\in X_I & & \text{(3e)}\\
r_{\ell h}\in\{0,1\} & \forall h\in G, \forall \ell\in N. & \text{(3f)}
\end{cases}
$$

Constraints (3b) state that the number of allocations whose cost is at least $c_{(h)}$ must be equal to the number of sites that support allocation costs greater than or equal to $c_{(h)}$. Constraints (3c) and (3d) sort the values of variables r in non-decreasing order. As indicated in Marín et al. (2009) and Labbé et al. (2017), the extra sorting group of constraints (3d) are redundant, nevertheless, they have been shown to improve the computational performance, thus, we include them in our model.

3.2. *Benders decomposition for DOMP$_{r0}$*

Radius formulation has been proven to be one of the most efficient formulation when λ is non-decreasing (see, e.g., Labbé et al., 2017 or Pozo et al., 2023). Nevertheless, when turning to a non-monotone ordered weighting vector, the performance of the formulation has been recently improved by some alternative approaches (Marín et al., 2020). **In this section, we propose a Benders decomposition approach based on projecting out r variables. The decomposition is obtained from the DOMP$_{r0}$ model after relaxing the integrality conditions on the radius variables. However, for certain configurations of the vector λ the radius variables must remain binary, which is why the proposed Benders decomposition approach only provides valid lower bounds in such cases. In spite of that, we will prove that for some interesting particular values of λ, the decomposition approach is exact and it provides a new valid reformulation for DOMP.**

We have the following result:

Proposition 1. *The following compact model*

$$
(\mathit{DOMP}_{r0B}) \begin{cases}
\min \displaystyle\sum_{h\in G}\theta_h(c_{(h)}-c_{(h-1)}) & & \text{(4a)}\\
\text{s.t.:}\quad \theta_h \geq \lambda_\ell \displaystyle\sum_{\substack{i,j\in N:\\ c_{ij}\geq c_{(h)}}} x_{ij} - \sum_{\ell<\ell'}(\lambda_\ell-\lambda_{\ell'}) & \forall h\in G, \forall \ell\in N & \text{(4b)}\\
(x,y)\in X_I. & & \text{(4c)}
\end{cases}
$$

is a reformulation of the model DOMP$_{r0}$ for any monotone non-increasing vector λ. ~~*Otherwise, the optimal value of model (4) provides a valid lower bound for DOMP.*~~

Proof. We start with the relaxation of the model DOMP$_{r0}$ in which constraints (3f) are relaxed into $0\leq r_{\ell h}\leq 1$, for all $h\in G, \ell\in N$. We first focus on monotone non-increasing values of λ and show that the model (4) can be obtained by projecting out variables r from this relaxation. To this end, terms $\sum_{\ell\in N}\lambda_\ell r_{\ell h}$ in the objective function (3a) are replaced by new variables θ_h, $h\in G$.

Given $\bar{h}\in G$, and a point $(x^*,y^*)\in X$, let us denote by $x^*_{\bar{h}} = \sum_{\substack{i,j\in N:\\ c_{ij}\geq c_{(\bar{h})}}} x_{ij}$ and consider the following subproblem:

$$
\begin{aligned}
\min \quad & \sum_{\ell \in N} \lambda_\ell r_{\ell \bar{h}} && && \text{(5a)}\\
\text{s.t.:} \quad & \sum_{\ell \in N} r_{\ell \bar{h}} = x^*_{\bar{h}} && && \text{(5b)}\\
& r_{\ell \bar{h}} \le r_{\ell+1 \bar{h}} && \forall \ell \in N : \ell < |N| && \text{(5c)}\\
& 0 \le r_{\ell \bar{h}} \le 1 && \forall \ell \in N. && \text{(5d)}
\end{aligned}
$$

When λ is monotone non-increasing, the constraints (5c) are redundant, and the problem (5) can be restated as a continuous knapsack problem in which constraints (5c) are removed. Such obtained linear program is always feasible and bounded, so we can calculate its dual, in which a variable α is associated to constraint (5b) and variables β_ℓ, $\ell \in N$, are associated to constraints (5d):

$$
\begin{aligned}
\max \quad & \alpha x^*_{\bar{h}} - \sum_{\ell \in N} \beta_\ell && && \text{(6a)}\\
\text{s.t.:} \quad & \alpha - \beta_\ell \le \lambda_\ell && \forall \ell \in N && \text{(6b)}\\
& (\alpha, \beta) \ge 0. && && \text{(6c)}
\end{aligned}
$$

Let $\mathcal{P} = \{(\alpha, \beta) \ge 0 \mid (\alpha, \beta) \text{ satisfy (6b)-(6c)}\}$ be the dual polyhedron and $\mathcal{P}^{ext}$ be the set of extreme points of $\mathcal{P}$, then the general form of Benders optimality cuts as a function of the dual and primal solution is given as:

$$
\theta_h \ge \alpha^* \sum_{\substack{i,j \in N: \\ c_{ij} \ge c_{(\bar{h})}}} x_{ij} - \sum_{\ell \in N} \beta^*_\ell \qquad (\alpha^*, \beta^*) \in \mathcal{P}^{ext},
$$

It is well known from knapsack theory (Martello and Toth, 1990) that the following Dantzig algorithm produces an optimal primal solution and optimal dual variables. Let k^* be the index of the *critical item* of the knapsack problem above, the optimal primal solution is

$$
r^*_{\ell \bar{h}} = \begin{cases} 1 & \text{if } \ell > k^*, \\ x^*_{\bar{h}} - (|N| - k^*) & \text{if } \ell = k^*, \\ 0 & \text{otherwise}, \end{cases} \qquad \ell \in N. \tag{7}
$$

Hence, the optimal value of the primal problem is

$$
\sum_{\ell > k^*} \lambda_\ell + \lambda_{k^*}(x^*_{\bar{h}} - (|N| - k^*)) = \lambda_{k^*} x^*_{\bar{h}} - \sum_{\ell > k^*} (\lambda_{k^*} - \lambda_\ell),
$$

and the dual variables can be calculated as:

$$
\begin{aligned}
\alpha^* &= \lambda_{k^*}, \\
\beta^*_\ell &= \begin{cases} \lambda_{k^*} - \lambda_\ell & \text{if } \ell > k^*, \\ 0 & \text{otherwise}, \end{cases} \qquad \ell \in N.
\end{aligned}
$$

Generalizing for any possible critical item, we get the following family of Benders optimality cuts:

$$
\theta_h \ge \lambda_\ell \sum_{\substack{i,j \in N: \\ c_{ij} \ge c_{(h)}}} x_{ij} - \sum_{\ell' > \ell} (\lambda_\ell - \lambda_{\ell'}) \quad \forall h \in G, \forall \ell \in N. \tag{8}
$$

It now remains to show that for monotone non-increasing values of λ, for the relaxation of the model DOMP_{r0} in which constraints (3f) are replaced by $0 \leq r_{\ell h} \leq 1$, $h \in G, \ell \in N$, there always exists an optimal solution in which the variables r take on integer values. Indeed, this is immediate from formula (7): whenever a feasible point $(x^*, y^*) \in X_I$ is given, the optimal solution consists of setting the last $x^*_{\bar{h}}$ values of r to one, and the remaining ones to zero.

~~In case λ is not monotone non-increasing, constraints (5c) in Problem (5) cannot be avoided, and hence the reduction to the continuous knapsack problem does not work. Moreover, solutions to the LP-relaxation of DOMP_{r0} are not always integer so that (8) only provides valid inequalities which could not ensure optimality of the master problem.~~

□

Corollary 1. *For any general not monotone non-increasing vector λ, the optimal value of model* (4) *provides a valid lower bound for* DOMP*.*

Proof. In case λ is not monotone non-increasing, constraints (5c) in Problem (5) cannot be avoided, and hence the reduction to the continuous knapsack problem does not work. Moreover, solutions to the LP-relaxation of DOMP_{r0} are not always integer so that (8) only provides valid inequalities which could not ensure optimality of the master problem. □

4. State-of-the-art formulation for general λ-vectors

As already announced, our goal is to provide improved reformulations for DOMP for general lambda weights and general costs. The Benders reformulation developed in previous section is only valid for monotone non-increasing lambda, thus in **this section,** we address the question in full generality. We will first elaborate on the state-of-the-art formulation for DOMP that can be found in Marín et al. (2020). We will then show how to project out its variables to obtain three different Benders reformulations.

4.1. Compact formulation based on partial monotonicity

Marín et al. (2020) introduced a three-index formulation in which the variables θ^k_{ij} are set to 1 if the cost of allocating client i to facility j is sorted in position k, and they are set to 0 otherwise ($i, j, k \in N$). Even though the variables are defined as binary, without loss of generality, they can be relaxed to be continuous from the interval $[0, 1]$. This fact gives rise to a rather efficient problem formulation which is presented below:

$$
(\mathsf{DOMP}_\theta)\left\{
\begin{array}{lll}
\min \displaystyle\sum_{k\in\Delta^+} \Delta_k\big((n-k+1)t_k + \sum_{i\in N} z_{ik}\big) + & & \text{(9a)}\\
\qquad + \displaystyle\sum_{k\in\Delta^-} \Delta_k \sum_{i\in N}\sum_{j\in N} c_{ij}\theta^k_{ij} & & \\
\text{s.t.:}\quad t_k + z_{ik} \geq \displaystyle\sum_{j\in N} c_{ij}x_{ij} & \forall i \in N, \forall k \in \Delta^+ & \text{(9b)}\\
\displaystyle\sum_{i\in N}\sum_{j\in N} \theta^k_{ij} = n-k+1 & \forall k \in \Delta^- & \text{(9c)}\\
0 \leq \theta^k_{ij} \leq x_{ij} & \forall i,j \in N, \forall k \in \Delta^- & \text{(9d)}\\
(x,y) \in X_I & & \text{(9e)}\\
z_{ik}, t_k \geq 0 & \forall i \in N, \forall k \in \Delta^+. & \text{(9f)}
\end{array}\right.
$$

The objective function (9a) has two parts: the first one accounts for the sum of the terms $S_k(c(x))$ for $k \in \Delta^+$, and the second one does the same for $k \in \Delta^-$. Constraints (9b) allow for a valid representation of $S_k(c(x))$ for $k \in \Delta^+$. The remaining constraints, namely (9c) and (9d), are used to get a valid representation of $S_k(c(x))$ for $k \in \Delta^-$, where (9d) define the range of the variables θ_{ij}^k. Since x_{ij} are binary and θ_{ij}^k appears in the objective function with a negative coefficient ($\Delta_k < 0$), if $x_{ij} = 1$ then $\theta_{ij}^k = 1$, thus in the optimal solution $\theta_{ij}^k \in \{0,1\}$. The remaining variables in (9f), namely z and t, are obtained from the dual of the k-sum problem as explained in Ogryczak and Tamir (2003), Kalcsics et al. (2002) and Marín et al. (2020).

4.2. Benders reformulations for DOMP_θ

Benders reformulations proposed in this section exploit the fact that the variables z and θ of the formulation DOMP_θ are continuous. Moreover, they belong to two well separated components in both the objective function and constraints, obtained by considering $k \in \Delta^+$ and $k \in \Delta^-$, respectively. Hence, we can consider Benders reformulations by projecting out (z,t) variables, θ variables, or both. For this purpose, we first need to introduce the following order relation between costs indexes, previously considered by Labbé et al. (2017):

$$ij \prec \hat{\mathrm{I}}\hat{\mathrm{J}} \equiv \begin{cases} c_{ij} < c_{\hat{\mathrm{I}}\hat{\mathrm{J}}} & \\ \text{or} & \\ c_{ij} = c_{\hat{\mathrm{I}}\hat{\mathrm{J}}} & \text{and } i < \hat{\mathrm{I}} \\ \text{or} & \\ c_{ij} = c_{\hat{\mathrm{I}}\hat{\mathrm{J}}}, \quad i = \hat{\mathrm{I}} & \text{and } j < \hat{\mathrm{J}}. \end{cases}$$

Example 1. As an illustrative example, consider the following matrix:

$$\begin{pmatrix} 0 & 2 & 7 & 4 \\ 1 & 0 & 5 & 5 \\ 3 & 6 & 0 & 2 \\ 9 & 4 & 1 & 0 \end{pmatrix},$$

then the order of couples ij by means of the above order preference is

$$11 \prec 22 \prec 33 \prec 44 \prec 21 \prec 43 \prec 12 \prec 34 \prec 31 \prec 14 \prec 42 \prec 23 \prec 24 \prec 32 \prec 13 \prec 41.$$

Let $H = \{1, \ldots, n^2\}$, then there exists an obvious bijection between costs in C and the elements of the ordered costs vector $(c_{ij(h)} | h \in H)_{i,j \in N}^{\prec}$, such that for every (i,j) pair there is a single $ij(h)$ associated to it for a certain $h \in H$. Note that while G in the previous section has a length of the number of unique costs of C, here the ordering of costs has the same number of elements as total costs.

Next, we will show that for $k \in \Delta^-$ we can obtain a valid reformulation of DOMP_θ by projecting out θ variables. The advantage of this new model (that we will denote by $\mathsf{DOMP}_{\theta 1}$) is in the reduction of the size of the underlying formulation. While the starting model DOMP_θ comprises $O(|N|^3)$ variables and $O(|N|^3)$ constraints, the new model requires only $O(|N|^2)$ variables and $O(|N|^3)$ constraints.

Proposition 2. *The following model*

$$(\mathit{DOMP}_{\theta 1}) \begin{cases} \min \displaystyle\sum_{k \in \Delta^+} \Delta_k((n-k+1)t_k + \sum_{i \in N} z_{ik}) + \sum_{k \in \Delta^-} \Delta_k \varphi_k & & \text{(10a)} \\ \textit{s.t.:} \quad t_k + z_{ik} \geq \displaystyle\sum_{j \in N} c_{ij} x_{ij} & \forall i \in N, k \in \Delta^+ & \text{(10b)} \\ 0 \leq \varphi_k \leq (n-k+1) c_{ij(h)} + \displaystyle\sum_{ij:\, c_{ij} > c_{ij(h)}} (c_{ij} - c_{ij(h)}) x_{ij} & \forall h \in H, k \in \Delta^- & \text{(10c)} \\ (x,y) \in X_I & & \text{(10d)} \\ z_{ik}, t_k \geq 0 & \forall i \in N, k \in \Delta^+. & \text{(10e)} \end{cases}$$

is a reformulation of the model DOMP_θ.

Proof. Terms $\sum_{i\in N}\sum_{j\in N} c_{ij}\theta_{ij}^k$, are replaced in the objective function (9a) by new variables $\varphi_k \geq 0$, for all $k \in \Delta^-$. Then we can consider the following relaxed master problem:

$$\begin{aligned}
\min \quad & \sum_{k\in\Delta^+} \Delta_k((n-k+1)t_k + \sum_{i\in N} z_{ik}) + \sum_{k\in\Delta^-} \Delta_k\varphi_k && && \text{(11a)}\\
\text{s.t.:} \quad & (9b), (9e), (9f) && &&\\
& 0 \leq \varphi_k \leq \mathcal{U}_k && \forall k \in \Delta^-, &&
\end{aligned}$$

where $\mathcal{U}_k$ are valid upper bounds for φ_k. Clearly, a simple, but valid upper bound for φ_k is the sum of the $(n-k+1)$ largest costs in C (see Section 4.3 for more details).

Let (x^*, y^*) be a solution of the above LP. To ensure that (x^*, y^*) corresponds to an optimal DOMP solution, we need to solve the so-called Benders subproblem which is decomposable by k. For any $\bar{k} \in \Delta^-$, since $\Delta_{\bar{k}} < 0$, the $\bar{k}$-th Benders subproblem is a *maximization* problem given as follows:

$$\begin{aligned}
\max \quad & \sum_{i\in N}\sum_{j\in N} c_{ij}\theta_{ij}^{\bar{k}} && && \text{(12a)}\\
\text{s.t.:} \quad & \sum_{i\in N}\sum_{j\in N} \theta_{ij}^{\bar{k}} \leq n-\bar{k}+1 && && \text{(12b)}\\
& 0 \leq \theta_{ij}^{\bar{k}} \leq x_{ij}^* && \forall i,j \in N. && \text{(12c)}
\end{aligned}$$

We point out that constraints (12b) are stated as inequalities instead of equations. This can be done without loss of generality, because the values $c_{ij} \geq 0$, for all $i,j \in N$. Moreover, we observe that the LP-model (12) is always feasible and bounded, which follows from the fact that $\sum_{i,j\in N} x_{ij}^* = n$. Hence, one can consider its LP-dual in which a variable α is associated to constraint (12b) and variables β_{ij}, $i,j \in N$, are associated to constraints (12c):

$$\begin{aligned}
\min \quad & (n-\bar{k}+1)\alpha + \sum_{i,j\in N} \beta_{ij}x_{ij}^* && && \text{(13a)}\\
\text{s.t.:} \quad & \beta_{ij} + \alpha \leq c_{ij} && \forall i,j \in N && \text{(13b)}\\
& (\alpha, \beta) \geq \mathbf{0}. && && \text{(13c)}
\end{aligned}$$

Let $h^* \in H$ be the index of a critical item of the continuous knapsack problem defined by (12), namely the index h^* for which the following property holds:

$$\sum_{h>h^*} x_{ij(h)}^* < n-\bar{k}+1 \leq \sum_{h\geq h^*} x_{ij(h)}^*.$$

Then, the optimal solution of the model (12) is given as follows:

$$\theta_{ij(h)}^{\bar{k}} = \begin{cases} x_{ij(h)}^* & \text{if} \quad h > h^*,\\ (n-\bar{k}+1) - \sum_{h>h^*} x_{ij(h)}^* & \text{if} \quad h = h^*,\\ 0 & \text{otherwise.}\end{cases}$$

It can be verified that the optimum value of the primal subproblem satisfies

$$\sum_{i\in N}\sum_{j\in N} c_{ij}\theta_{ij}^{\bar{k}} = \sum_{h>h^*} c_{ij(h)}x^*_{ij(h)} + c_{ij(h^*)}\Big[(n-\bar{k}+1) - \sum_{h>h^*} x^*_{ij(h)}\Big]$$
$$= c_{ij(h^*)}(n-\bar{k}+1) + \sum_{h>h^*}(c_{ij(h)} - c_{ij(h^*)})x^*_{ij(h)}$$

and, therefore, the optimal dual solution is

$$\alpha = c_{ij(h^*)},$$
$$\beta_{ij} = \begin{cases} c_{ij(h)} - c_{ij(h^*)} & \text{if } h > h^* \quad (\iff c_{ij(h)} > c_{ij(h^*)}), \\ 0 & \text{otherwise.} \end{cases}$$

Generalizing for any possible critical item $h \in H$, we get the following family of Benders optimality cuts for every $k \in \Delta^-$ and every $h \in H$:

$$\varphi_k \leq (n-k+1)c_{ij(h)} + \sum_{ij\,:\,c_{ij}>c_{ij(h)}} (c_{ij} - c_{ij(h)})x_{ij}.$$

□

For the values $k \in \Delta^+$, we now show how to project out (z,t) variables.

Proposition 3. *The following model*

$$(\textit{DOMP}_{\theta 2}) \begin{cases} \min \quad \sum_{k\in\Delta^+} \Delta_k\varphi_k + \sum_{k\in\Delta^-}\Delta_k \sum_{i\in N}\sum_{j\in N} c_{ij}\theta_{ij}^k & (14a) \\ \text{s.t.:} \quad \varphi_k \geq \sum_{i\in N_L}\sum_{j\in N} c_{ij}x_{ij} \quad \forall k\in\Delta^+, N_L\subseteq N : |N_L| = n-k+1 & (14b) \\ \sum_{i\in N}\sum_{j\in N}\theta_{ij}^k = n-k+1 \quad \forall k\in\Delta^- & (14c) \\ 0 \leq \theta_{ij}^k \leq x_{ij} \quad \forall i,j\in N, \forall k\in\Delta^- & (14d) \\ (x,y)\in X_I & (14e) \end{cases}$$

is a reformulation of the model DOMP$_\theta$.

Proof. Terms $(n-k+1)t_k + \sum_{i\in N} z_{ik}$, for $k \in \Delta^+$, are replaced in the objective function (9a) by new variables $\varphi_k \geq 0$. Let (x^*, y^*) be a solution satisfying constraints (9c) and (9e). Again, Benders subproblem is separable by k, and we are solving the following LP associated to a given $\bar{k} \in \Delta^+$:

$$\min \quad (n-\bar{k}+1)t_{\bar{k}} + \sum_{i\in N} z_{i\bar{k}} \tag{15a}$$
$$\text{s.t.:} \quad t_{\bar{k}} + z_{i\bar{k}} \geq \sum_{j\in N} c_{ij}x^*_{ij} \qquad \forall i\in N \tag{15b}$$
$$z_{i\bar{k}}, t_{\bar{k}} \geq 0 \qquad \forall i \in N. \tag{15c}$$

This subproblem is always feasible and bounded, hence we can consider its LP-dual, in which variables $\alpha_i, i \in N$, are introduced for every constraint in (15b):

$$\max \quad \sum_{i\in N}\alpha_i \sum_{j\in N} c_{ij}x^*_{ij} \tag{16a}$$

$$\text{s.t.:} \quad \sum_{i \in N} \alpha_i \leq n - \bar{k} + 1 \tag{16b}$$

$$0 \leq \alpha_i \leq 1 \qquad \forall i \in N. \tag{16c}$$

In order to compute the optimal solution of model (16), we need to sort $i \in N$ in increasing order according to $v_i^* = \sum_{j \in N} c_{ij} x_{ij}^*$. It is now enough to consider the set $N_L^* \subseteq N$ with the $(n - \bar{k} + 1)$ indexes associated with the largest v_i^* values, i.e., we have $\alpha_i = 1$ if $i \in N_L^*$, and $\alpha_i = 0$, otherwise. Since the choice of N_L^* depends on x^*, after generalizing over all possible values of x^*, we obtain the following family of Benders optimality cuts associated to subproblem $\bar{k} \in \Delta^+$:

$$\varphi_k \geq \sum_{i \in N_L} \sum_{j \in N} c_{ij} x_{ij} \qquad N_L \subseteq N, |N_L| = n - k + 1.$$

□

By merging the results of Proposition 2 and 3, we obtain the third problem reformulation.

Proposition 4. *The following model*

$$(\textit{DOMP}_{\theta 3}) \begin{cases} \min \quad \sum_{k \in N} \Delta_k \varphi_k & (17a) \\ \textit{s.t.:} \quad \varphi_k \geq \sum_{i \in N_L} \sum_{j \in N} c_{ij} x_{ij} & \forall k \in \Delta^+, N_L \subseteq N : |N_L| = n - k + 1 \quad (17b) \\ \varphi_k \leq (n - k + 1) c_{ij(h)} + \sum_{ij \,:\, c_{ij} > c_{ij(h)}} (c_{ij} - c_{ij(h)}) x_{ij} & \forall k \in \Delta^-, \forall h \in H, \quad (17c) \\ (x, y) \in X_I & (17d) \end{cases}$$

is a reformulation of the model $\textit{DOMP}_\theta$.

Finally, we point out that one can consider aggregated formulations in which a single variable ϕ^- or ϕ^+ is introduced to replace the term in the objective function associated with negative, respectively, positive multipliers Δ_k. This can be done for each of the previously introduced models. In the following we illustrate how this aggregation is derived from the model $\mathsf{DOMP}_{\theta 1}$. We introduce a new variable $\phi = \sum_{k \in \Delta^-} \varphi_k$ and obtain the following result:

Proposition 5. *The following model*

$$(\textit{DOMP}_{\theta 1a}) \begin{cases} \min \quad \sum_{k \in \Delta^+} \Delta_k ((n - k + 1) t_k + \sum_{i \in N} z_{ik}) + \phi & (18a) \\ \textit{s.t.:} \quad t_k + z_{ik} \geq \sum_{j \in N} c_{ij} x_{ij} & \forall i \in N, \forall k \in \Delta^+ \quad (18b) \\ \phi \geq \sum_{k \in \Delta^-} \Delta_k \big[(n - k + 1) c_{ij(h_k)} + \sum_{ij \,:\, c_{ij} > c_{ij(h_k)}} (c_{ij} - c_{ij(h_k)}) x_{ij} \big] & \\ & \forall (h_k)_{k \in \Delta^-} \in H \times H \times \ldots \times H \quad (18c) \\ (x, y) \in X_I & (18d) \\ z_{ik}, t_k \geq 0 & \forall i \in N, \forall k \in \Delta^+. \quad (18e) \end{cases}$$

is a reformulation of the model $\textit{DOMP}_\theta$.

While the reformulation $\mathsf{DOMP}_{\theta 1}$ contains polynomial number of constraints and variables, its aggregated counterpart $\mathsf{DOMP}_{\theta 1a}$ consists of an exponential number of Benders optimality cuts (18c).

4.3. Bounds for the values of φ_k, $k \in \Delta^-$, and ϕ

To strengthen the linear relaxations of the previous formulations, several bounds can be proposed. **In this section,** these bounds are only discussed for the models that have been tested in our computational study (cf. Section 6). To this end, let $c_{max} = \max\{c_{ij} \mid i,j \in N\}$ and let $c_{min} = \min\{c_{ij} \mid i,j \in N\}$. For the disaggregated model $\mathsf{DOMP}_{\theta 1}$, the following bounds on the value of φ_k, $k \in \Delta^-$, are valid:

$$(n-k+1)c_{min} \leq \varphi_k \leq (n-k+1)c_{max}.$$

The reader may observe that in the case of free self-service ($c_{ii} = 0,\ \forall i,\ c_{ij} > 0, i \neq j$) one can obtain more accurate lower bounds. Indeed, let $c^+_{min} = \min\{c_{ij} \mid i,j \in N, c_{ij} \neq 0\}$,

$$\begin{cases} 0 \leq \varphi_k \leq (n-k+1)c_{max} & \text{if } n-k+1 \leq p, \\ (n-k+1-p)c^+_{min} \leq \varphi_k \leq (n-k+1)c_{max} & \text{otherwise.} \end{cases}$$

We recall that φ_k represents the sum of the $n-k+1$ largest costs, i.e, $S_k = \sum_{l=k}^{n} c_{(l)}$. Therefore, under the free self-service assumptions, we have:

- If $n-k+1 \leq p$, all the addends could assume value 0 in the lower bound and therefore the lower bound in this case will be $0 \leq \varphi_k$.
- If $n-k+1 > p$, the number of addends in S_k is greater than p and therefore at most p of them could assume value zero in the lower bound and the remainder c^+_{min}, i.e, the lower bound in this case is $0 \cdot p + (n-k+1-p)c^+_{min}$.

The above upper bounds can also be enhanced. To this end, let us consider the set of maximum allocation costs for every client, $\tilde{c} = (\max\{c_{ij} \mid j \in N_{max}\})_{i \in N}$, where $N_{max} \subset N$ represents a set of open facilities obtained by solving the auxiliary problem

$$\max\{\sum_{i,j \in N} c_{ij}x_{ij} | (x,y) \in X_I\},$$

where we are asking for a subset of p facilities that maximizes the allocation costs (with additional closest facility allocation constraints). The set N_{max} corresponds to p columns in the cost matrix C, where we can consider the $(n-k+1)$ highest costs in $\tilde{c}$ for each $k \in N$ to propose a tighter bound:

$$\varphi_k \leq \sum_{h=n-k+1}^{n} \tilde{c}_{(h)},$$

where $\tilde{c}(h)$ is the h-th element of the non-decreasing sorted vector $\tilde{c}$.

For the aggregated model $\mathsf{DOMP}_{\theta 1a}$, several bounds can also be proposed for ϕ, e.g:

$$\sum_{k \in \Delta^-} \Delta_k (n-k+1)c_{max} \leq \phi \leq 0.$$

Instead we can consider, similarly to the disaggregated case, the $n-k+1$ highest single allocation costs,

$$\sum_{k \in \Delta^-} \Delta_k \Big(\sum_{h=n-k+1}^{n} \tilde{c}_{(h)} \Big) \leq \phi \leq 0.$$

Indeed, if we assume non-negative λ or simply the distribution of λ that assures that the objective function will be non-negative, since we assume non-negative costs the optimal objective function of our problem, see (18a), must be non-negative. Therefore, the sum of the contributions associated with positive $k \in \Delta^+$ (those that appear in the left of the above expression), cannot be smaller than the corresponding ones associated with negative $k \in \Delta^-$ (that appear in the right of the above expression). So, if $\Delta^+ \neq \emptyset$, we can also obtain as a lower bound for ϕ:

$$\max(-\sum_{k\in\Delta^+} \Delta_k(\sum_{h=n-k+1}^{n} \tilde{c}_{(h)}), \sum_{k\in\Delta^-} \Delta_k(\sum_{h=n-k+1}^{n} \tilde{c}_{(h)})) \leq \phi \leq 0.$$

In the general case, where λ can be positive and negative and non-negativity of the objective function cannot be assured, the bound above does not apply since it is based on the non-negativity of the objective function which is not valid in this case.

4.4. Strengthening Benders cuts

In the attempt to make our models efficient, we propose the following lifting procedure for the coefficients of the Benders optimality cuts. For strengthening the cuts (10c), let $\mathcal{U}_k$ be a valid upper bound for φ_k, $k \in \Delta^-$. Then, for a given $k \in \Delta^-$, such that

$$\delta_k := \mathcal{U}_k - (n-k+1)c_{ij(h)} > 0$$

holds, constraints

$$\varphi_k \leq (n-k+1)c_{ij(h)} + \sum_{c_{ij}>c_{ij(h)}} (c_{ij} - c_{ij(h)})x_{ij} \qquad \forall h \in H$$

can be strengthened as follows:

$$\varphi_k \leq (n-k+1)c_{ij(h)} + \sum_{c_{ij}>c_{ij(h)}} (\min\{c_{ij} - c_{ij(h)}, \delta_k\})x_{ij} \quad \forall h \in H : \mathcal{U}_k > (n-k+1)c_{ij(h)} \quad (19)$$

Indeed, this follows from the fact that x variables are binary. Hence, if all values of the x variables from the right-hand-side of (19) are zero, we have a tighter bound on φ_k. If, on the other hand, at least one of these x variables is equal to one, the bound of φ_k should not surpass $\mathcal{U}_k$, and hence the coefficient next to x_{ij} can be downlifted correspondingly. Finally, it is not difficult to see that in the compact formulation (10), the remaining constraints (10c) (i.e., those for which $\mathcal{U}_k \leq (n-k+1)c_{ij(h)}$ holds) are redundant as they are dominated by $\varphi_k \leq \mathcal{U}_k$.

In order to strengthen the cuts (18c), let now $\mathcal{L}$ be a valid lower bound for ϕ (recall $\phi \leq 0$). For a given $(h_k)_{k\in\Delta^-}$, such that

$$\delta := \sum_{k\in\Delta^-} \Delta_k(n-k+1)c_{ij(h_k)} - \mathcal{L} > 0$$

holds, constraints (18c), which can be rewritten as

$$\phi \geq \sum_{k\in\Delta^-} \Delta_k(n-k+1)c_{ij(h_k)} + \sum_{k\in\Delta^-} \Delta_k\big[\sum_{c_{ij}>c_{ij(h_k)}} (c_{ij} - c_{ij(h_k)})x_{ij}\big]$$

can be strengthened as follows:

$$\phi \geq \sum_{k\in\Delta^-} \Delta_k(n-k+1)c_{ij(h_k)} + \sum_{k\in\Delta^-} \sum_{c_{ij}>c_{ij(h_k)}} \max\{\Delta_k(c_{ij} - c_{ij(h_k)}), -\delta\}\, x_{ij}. \tag{20}$$

Indeed, as in the case of the disaggregated reformulation, this result follows from the fact that x variables are binary. If all values of the x variables from the right-hand-side of (20) are zero, we have a tighter lower bound on ϕ. If, on the other hand, at least one of these x variables is equal to one, the value of ϕ should not be smaller than $\mathcal{L}$, and hence the coefficient next to x_{ij} can be strengthened correspondingly. Finally, it is not difficult to see that in the aggregated formulation (18), the remaining constraints (18c) (i.e., those for which $\delta \leq 0$) are redundant as they are dominated by $\phi \geq \mathcal{L}$.

5. Implementation details

This section provides some implementation details that play an important role in the design of effective implementations of the models proposed in Sections 3.2 and 4.2. The major bottleneck for Benders reformulations derived from the model DOMP_θ are three-index variables θ_{ij}^k. Correspondingly, a significant computational improvement is only achieved when considering Benders decomposition approach just for $k \in \Delta^-$, i.e., when just projecting out those variables from the model as in $\mathsf{DOMP}_{\theta 1}$, or its aggregated counterpart. Therefore, in this section we will restrict ourselves to presenting the implementation details for $\mathsf{DOMP}_{\theta 1}$ and $\mathsf{DOMP}_{\theta 1a}$ models, together with the model DOMP_{r0B}.

5.1. Branch-and-Benders-cut implementations

For a more efficient performance, for each of our reformulations, we embed Benders cuts into a branch-and-cut framework, also known as branch-and-Benders-cut. For the formulation DOMP_{r0B}, we initialize the model without the family of Benders cuts (4b) and with lower and upper bounds for θ_h given as $0 \leq \theta_h \leq \sum_{\ell \in N} \lambda_\ell$, $h \in G$. Benders cuts (4b) are then separated (in polynomial time) in a branch-and-bound (B&B) scheme. The pseudocode of the separation algorithm is shown in Algorithm 1. In this algorithm, we are given a solution $(\bar{\theta}, \bar{x})$ found in a current node of the B&B tree. For each $h \in G$, following the rationale given in the proof of Proposition 1, we must first compute the value of $\bar{x}_h^*$. Next, we determine the critical item index k as in (7). We sum unit capacity items starting from the very last index until the knapsack capacity is full, this is, until $\bar{x}_h^*$ minus this sum is less than one. Once the index has been identified, we can propose a cut in the form of (8) computing the values of the cut expression and if the $\bar{\theta}$ value obtained as the node solution is less than the expression plus an ε tolerance, we insert the cut in the master LP.

For the disaggregated and aggregated Benders reformulations, $\mathsf{DOMP}_{\theta 1}$ and $\mathsf{DOMP}_{\theta 1a}$, the separation is performed in a similar way. The respective families of Benders optimality cuts, namely (10c) and (18c), can be also separated in polynomial time. As for the aggregated reformulation the number of Benders optimality cuts (18c) is exponential, the branch-and-Benders-cut is the only computationally viable option for tackling instances of realistic size. The separation algorithms are summarized in Algorithms 2 and 3, respectively. These algorithms differ from Algorithm 1 in two aspects. First, the critical item index h is now computed summing the respective capacity for these cases, which is $\bar{x}_{ij(h)}$. Actually, for $\mathsf{DOMP}_{\theta 1a}$ we need to determine a vector of indexes $h = (h_k)_{k\in\Delta^-}$. We remark that when searching for the values of h_k, $k \in \Delta^-$, we need to process an alternate sequence of indexes k, not necessarily consecutive, since we only consider indexes k for which $\Delta_k < 0$. Second, the proposed Benders cuts are now in the form of (10c) and (18c).

The separation algorithms given in Algorithms 1-3 have to be applied at integer points, and alternatively, they can also be called for fractional LP-solutions within the branching tree. The latter case can be proven to be computationally profitable but if we do not set conditions limiting the number of fractional cuts to include, these cuts can overload the master problem reducing the efficiency of our algorithm. In order to manage this trade-off, the separation of fractional points is done only at the root node of the search tree.

Algorithm 1: Branch-and-Benders-cut separation algorithm for DOMP_{r0B}

```
1 Consider a solution $(\bar\theta_h, \bar x_{ij})$ in a node of the branching tree
2 for $h \in G$ do
3     Let $\bar x^*_h = \sum_{\substack{i,j \in N:\\ c_{ij} \geq c_{(\bar h)}}} \bar x_{ij}$
4     Determine the largest $k \in N$ such that $\bar x^*_h - \sum_{k'=k}^{n} 1 < 1$
5     Let $RHS(\bar x_{ij}, k) = \lambda_k \bar x^*_h - \sum_{\ell' > k} (\lambda_k - \lambda_{\ell'})$
6     if $\bar\theta_h < RHS(\bar x_{ij}, k) - \varepsilon$ then
7         Add cut $\theta_h \geq \lambda_k \sum_{\substack{i,j \in N:\\ c_{ij} \geq c_{(h)}}} x_{ij} - \sum_{\ell' > k} (\lambda_k - \lambda_{\ell'})$
```

Algorithm 2: Branch-and-Benders-cut separation algorithm for $\mathsf{DOMP}_{\theta 1}$

```
1 Consider a solution $(\bar\varphi_k, \bar x_{ij})$ in a node of the branching tree
2 for $k \in \Delta^-$ do
3     Determine the largest $h \in H$ such that $\sum_{h' \geq h, h' \in H} \bar x_{ij(h')} \geq n - k + 1$
4     Let $RHS_k(\bar x_{ij}, h) = (n - k + 1) c_{ij(h)} + \sum_{c_{ij} > c_{ij(h)}} (c_{ij} - c_{ij(h)}) \bar x_{ij}$
5     if $\bar\varphi_k > RHS_k(\bar x_{ij}) + \varepsilon$ then
6         Add cut $\varphi_k \leq (n - k + 1) c_{ij(h)} + \sum_{c_{ij} > c_{ij(h)}} (c_{ij} - c_{ij(h)}) x_{ij}$
```

Algorithm 3: Branch-and-Benders-cut separation algorithm for $\mathsf{DOMP}_{\theta 1a}$

```
1 Consider a solution $(\bar\phi, \bar x_{ij})$ in a node of the branching tree
2 Determine $h = (h_k)_{k \in \Delta^-}$ where $h_k \in H$ is the largest s.t. $\sum_{h' \geq h_k, h' \in H} \bar x_{ij(h')} \geq n - k + 1$
3 Let $RHS(\bar x_{ij}, h) = \sum_{k \in \Delta^-} \Delta_k \big[(n - k + 1) c_{ij(h_k)} + \sum_{c_{ij} > c_{ij(h_k)}} (c_{ij} - c_{ij(h_k)}) \bar x_{ij}\big]$
4 if $\bar\phi < RHS(\bar x_{ij}, h) - \varepsilon$ then
5     Add cut $\phi \geq \sum_{k \in \Delta^-} \Delta_k \big[(n - k + 1) c_{ij(h_k)} + \sum_{c_{ij} > c_{ij(h_k)}} (c_{ij} - c_{ij(h_k)}) x_{ij}\big]$
```

5.2. In-and-out stabilization at the root node

Following observations in Fischetti et al. (2016, 2017), an in-and-out stabilization procedure can be applied at the root node, before starting the branch-and-cut procedure. Its aim is to quickly determine a small set of Benders cuts that brings the master LP-relaxation value as close as possible to the "real optimal value".

We now explain how is this procedure implemented for the formulation $\mathsf{DOMP}_{\theta 1}$ (similar ideas are applied to $\mathsf{DOMP}_{\theta 1a}$ as well). We start from a feasible LP-solution $\hat u = (\hat x, \hat y, \hat\varphi, \hat z, \hat t)$ used as the core or stabilizing point. In our case, we compute the core point as the analytic center (see, e.g., Bonami et al., 2020) obtained from solving the LP relaxation of the model DOMP_θ without the objective and using the barrier algorithm with crossover. If the computation of the LP relaxation takes much effort, then the core point is computed as a convex combination of feasible solutions. These feasible solutions are the result of randomly selecting p facilities, assigning the clients to its closest open facility and properly valuing the rest of continuous variables. This approach, which is highly dependable of the randomness in the facility selection, has been applied to the largest instances from our dataset.

At each cut loop iteration, we start with two points in the u space: the optimal solution of the current master LP, $u^* = (x^*, y^*, \varphi^*, z^*, t^*)$ as in Kelley (1960) method, and the core point $\hat u$. After initializing the parameter $\lambda \in [0, 1]$, we start moving the current master LP solution u^* towards the core point, obtaining $u^* = \lambda u^* + (1 - \lambda)\hat u$. We perform the Benders separation to the current u^*, and if the corresponding $Benders\_cut(u^*)$ is violated, we insert it into the pool of cuts $\mathcal{T}$. This procedure is iterated a given number of times ($iter_2$) without reoptimizing the current master LP, just adding the cuts into $\mathcal{T}$. Afterwards, the LP relaxation is computed including all cuts from $\mathcal{T}$ and the process is repeated with the new master LP-solution u^*. The pseudo-code of this procedure is given in Algorithm 4.

The procedure ends after a given number of iterations ($iter_1$). Also, to speed up the computation, non-binding cuts from the current LP are also removed (cf. Step 13 of Algorithm 4). This procedure has been computationally proven to be very helpful when solving instances of larger size, but not for the small ones. For this reason we have used this stabilization procedure for instances with 100 nodes or more. For the model based on the reformulation DOMP_{r0B} we implemented a lighter and faster warm-start phase. In this case, we only computed the LP relaxation of DOMP_{r0} once and then we added $|G|$ cuts in the form of (8) by means of the separation algorithm given in Algorithm 1.

Algorithm 4: In-and-out stabilization at the root node for DOMP Benders decomposition

```
1  Input:
2      Scaling parameter $\lambda$
3      The core point $\hat{u} = (\hat{x}, \hat{y}, \hat{\varphi}, \hat{z}, \hat{t})$
4      Initial pool of Benders cuts $\mathcal{T} = \emptyset$
5      $iter_1, iter_2$, iteration counters
6  Procedure:
7  while $iter_1$ do
8      Solve to optimality the current master LP including cuts from $\mathcal{T}$.
9      Let $u^* = (x^*, y^*, \varphi^*, z^*, t^*)$ be its optimal solution.
10     while $iter_2$ and violated cuts added to $\mathcal{T}$ do
11         $u^* = \lambda u^* + (1-\lambda)\hat{u}$
12         Perform Benders separation on $u^*$ generating $Benders\_cut(u^*)$ which (if violated) is added to $\mathcal{T}$
13 Remove from the master LP the cuts from $\mathcal{T}$ with positive slack
```

6. Computational study

This section summarizes the results obtained from our computational experiments performed in order to empirically compare the proposed DOMP formulations. Our algorithms were programmed in Python language (version 3.10) using Gurobi Optimizer 9.5 as a MIP solver. The experiments were run in a MacPro server with a 2,7 GHz Intel Xeon W processor of 24 cores and 192 GB RAM using a single thread and with all presolve and heuristics preprocessings deactivated for unifying the comparisons. The time limit for computations was set to 7200 seconds per instance. For the violation threshold when separating Benders cuts (see Algorithms 1-3), we set $\varepsilon = 0.01$. For the stabilization procedure given in Algorithm 4, we set $\lambda = 0.9$, $iter_1 = 10$ and $iter_2 = 5$. This way, we are adding in the best case scenario around 50 cuts by this procedure.

Following experiments in Labbé et al. (2017) and Marín et al. (2020), we considered two sets of benchmark instances. The first dataset, to which we will refer as `Beasley` instances, consists in p-median instances from Beasley (2015). In order to obtain the cost matrix we use the procedure explained in Labbé et al. (2017) which gives rise to instances of sizes $n \in \{50, 100, 150, 200\}$ without symmetry but with (possibly repeated) integer costs from $[1, 1000]$. For the second dataset (to which we refer as `Random` instances), we use instances proposed in Labbé et al. (2017). For each complete graph of size $n \in \{20, 30, 40, 50, 100, 140, 180, 200\}$ a set of five instances is generated by drawing costs c_{ij} uniformly at random from $[100, 1000]$ (with up to two decimal digits). Following similar setting as in the literature, the number of facilities for each group of instances was chosen as $p \in \{\lfloor \frac{n}{4} \rfloor, \lfloor \frac{n}{3} \rfloor, \lfloor \frac{n}{2} \rfloor\}$.

Concerning the chosen criteria, i.e, the λ-vectors to be tested, our purpose was to broaden as possible the criteria selection including non-monotone weight vectors and also negative weights. In fact, that was the reason to introduce vectors not yet used in the existing DOMP literature. The final selection of λ-vectors (see Table 1) is a representative sample that depicts the variety of problems that can be generalized within the ordered median optimization framework. Among them, we find some fairness criteria ($\lambda_1, \lambda_3, \lambda_7, \lambda_{11}, \lambda_{12}$), other particular customer preferences (λ_5, λ_9), and the obnoxious versions of most of them, where the ordered median objective is used to model the undesired facility location problems ($\lambda_2, \lambda_4, \lambda_6, \lambda_8, \lambda_{10}$). Furthermore, these vectors are among the most complex ones within the ordered median literature, primarily due to their non-monotone nature and the presence of negative weights when the minimization of the largest costs is required. Since λ-vectors with negative weights are tested, we define $\Lambda^+ = \{\lambda_1, \lambda_3, \lambda_5, \lambda_7, \lambda_9, \lambda_{11}, \lambda_{12}\}$ and $\Lambda^- = \{\lambda_2, \lambda_4, \lambda_6, \lambda_8, \lambda_{10}\}$ in order to compute the optimality gap as

$$\texttt{gap} = \begin{cases} (\texttt{UB} - \texttt{LB})/\texttt{UB} & \text{for } \Lambda^+, \\ (\texttt{UB} - \texttt{LB})/|\texttt{LB}| & \text{for } \Lambda^-, \end{cases}$$

where UB and LB stand for the best upper and lower bound found at termination, respectively.

Notation	λ-vector	Criterion name	Purpose
λ_1	$(0, ..., 0, 1)$	p-center	Fairness
λ_2	$(0, ..., 0, -1)$	Obnoxious p-center	Obnoxious
λ_3	$(0, \dots, 0, 0, \underbrace{1, \dots, 1}_{k})$	k-centrum, $k = n/2$	Fairness
λ_4	$(0, \dots, 0, 0, \underbrace{-1, \dots, -1}_{k})$	Obnoxiuos k-centrum, $k = n/2$	Obnoxious
λ_5	$(0, \dots, 0, \underbrace{1}_{k-th\ position}, 0, \dots, 0)$	k-max	General preferences
λ_6	$(0, \dots, 0, \underbrace{-1}_{k-th\ position}, 0, \dots, 0)$	Obnoxious k-max	Obnoxious
λ_7	$(-1, 0, ..., 0, 1)$	Range	Fairness
λ_8	$(1, 0, ..., 0, -1)$	Obnoxious range	Obnoxious
λ_9	$(0, -1, 0, ..., 0, 1, 0)$	Second range	General preferences
λ_{10}	$(0, 1, 0, ..., 0, -1, 0)$	Second obnoxious range	Obnoxious
λ_{11}	$(n, n-1, \dots, 2, 1)$	Reverse	Fairness
λ_{12}	$(2(2i - n - 1))_{i \in N}$	Sum of absolute differences	Fairness

Table 1: λ-vectors tested

First, we started by comparing DOMP_{r0} and its radius-Benders reformulation, DOMP_{r0B}, in both versions, implemented as a compact model and embedded within a Branch-and-Benders-cut algorithm (BBC) considering Algorithm 1. As remarked in Proposition 1, this reformulation is only valid for monotone non-increasing λ-vector. In our experimentation setup we decided to test λ_{11} given that it is the only strictly monotone non-increasing λ-vector and because we realized by performing some preliminary tests that DOMP_{r0} was outperforming DOMP_{θ} for λ_{11}. Table 2 shows cpu times for Beasley and Random instances in the first (Table 2a) and second block (Table 2b), respectively. For the Beasley instances, each instance is indexed as (Ins-n-p), where Ins identifies the number of the instance tested, and n and p indicates the number of nodes and the number of open facilities, respectively. For the Random instances, each row (n-p) shows average results over five random instances with the same number of nodes (n) and open facilities (p). From Table 2 we conclude that DOMP_{r0} outperforms DOMP_{r0B} in mostly all cases. The compact reformulation DOMP_{r0B} (whose runtimes are given in the last column) is up to two orders of magnitude slower than the compact formulation DOMP_{r0}. This can be explained by the large number of constraints ($O(|G||N|)$) of type (4b). On the other hand, for non-increasing λ vectors, the radius ordering constraints (3c)-(3d) are redundant (they can be removed by the MIP solver) so that the resulting DOMP_{r0} model requires only $O(|G|)$ constraints of type (3b). This difference in the number of constraints, along with the structure of the constraint matrix, explains the inferior performance of DOMP_{r0B} for λ_{11}. Comparing the BBC implementation of DOMP_{r0B} with its compact counterpart, we notice that speed-ups of an order of magnitude can be obtained. Nevertheless, the model DOMP_{r0} remains a clear winner in this setting. The significant differences in the cpu times for Beasley and Random instances are mainly explained by the fact that there are more repetitions in the cost matrix for the former, resulting in a performance of the models in that case. For all these reasons, in the remainder of this section, we focus on the Benders approaches for the state-of-the-art formulation DOMP_{θ}.

As commented in Section 5, Benders decomposition approaches tend to be promising when they result in reformulations with a significantly reduced number of decision variables. For the model DOMP_{θ}, this means projecting out θ^k_{ij} variables for $k \in \Delta^-$, $i, j \in N$. For this reason, we decided to test in our experiments two state-of-the-art compact formulations (DOMP_{r0} and DOMP_{θ}) against the branch-and-Benders-cut algorithms based on our reformulations $\mathsf{DOMP}_{\theta 1}$ and $\mathsf{DOMP}_{\theta 1a}$. Due to the large number of experiments, we decided to

	DOMP$_{r0}$	DOMP$_{r0B}$ (BBC)	DOMP$_{r0B}$
1-50-5	0.33	2.25	12.69
2-50-10	0.53	3.03	8.55
3-50-10	0.30	3.02	11.22
4-50-20	0.75	2.20	9.06
5-50-33	0.44	1.89	7.86
6-100-5	1.46	6.41	55.29
7-100-10	14.73	18.60	127.91
8-100-20	0.86	8.94	48.98
9-100-40	0.74	6.97	47.06
10-100-67	0.76	4.48	36.65
11-150-5	4.00	13.84	127.56
12-150-10	2.69	14.07	191.44
13-150-30	2.71	14.85	121.74
14-150-60	1.77	17.52	126.61
15-150-100	2.41	6.98	117.30
16-200-5	6.71	23.87	264.49
17-200-10	7.31	20.65	267.84
18-200-40	4.80	29.58	278.88
19-200-80	4.09	22.47	258.27
20-200-133	3.97	13.65	27.45

(a) `Beasley` instances

	DOMP$_{r0}$	DOMP$_{r0B}$ (BBC)	DOMP$_{r0B}$
20-5	0.09	0.39	0.41
20-6	0.27	0.47	0.76
20-10	0.19	0.32	0.39
30-7	4.08	4.50	8.89
30-10	65.76	2.92	5.35
30-15	0.57	2.12	3.75
40-10	51.39	12.89	93.83
40-13	1.03	6.47	203.82
40-20	0.92	8.49	22.43
50-12	7.74	212.08	792.50
50-16	2.17	201.28	393.59
50-25	1.70	12.56	189.62
100-25	227.04	2984.71	7200.00
100-33	55.96	1543.90	7200.00
100-50	53.79	585.79	7200.00
140-35	959.51	4656.04	7200.00
140-46	273.07	5680.54	7200.00
140-70	136.73	3695.69	7200.00
180-45	1722.72	7200.00	7200.00
180-60	956.11	7133.07	7200.00

(b) `Random` instances

Table 2: `cpu` times for Reverse criterion (λ_{11})

summarize the obtained results using the empirical cumulative distribution functions (ECDFs) and bar charts. We report ECDFs w.r.t. the runtimes and percentage gaps at termination, respectively. The ECDFs with e.g., runtimes can be interpreted as the number of instances (shown in y-axis) that can be solved within a certain amount of time (depicted in the x-axis).

For the `Beasley` instances, the ECDF in Fig. 1 shows the cumulative percentage of instances solved within a time limit and Fig. 2 shows the cumulative percentage of instances solved within a gap value. We decided to include both ECDFs since they capture the overall nature of our results, including those instances that were solved within the time limit and those that were not. In both cases, DOMP$_{\theta 1}$ and DOMP$_{\theta 1a}$ report improvements with respect to DOMP$_{r0}$ and DOMP$_{\theta}$. That is, our reformulations were able to solve more instances in less time and were able to obtain lower gaps in those instances for which we could not certify optimality. The reader may observe that, compared to other formulations, DOMP$_{r0}$ improves its performance with an increasing value of `n`, or in other words, with an increasing number of ties in the cost matrix (see Pozo et al. (2023) for a deeper analysis of this effect). We recall that `Beasley` instances include integer c_{ij} costs drawn uniformly in [1,100] for large instances. In order to have a better comparison between the different formulations, Fig. 3 shows results disaggregated by `n` and Fig. 4 shows results disaggregated by λ-vector. Similarly to Fig. 1, we conclude from Fig. 3 that DOMP$_{\theta 1}$ and DOMP$_{\theta 1a}$ are consistently better than DOMP$_{r0}$ and DOMP$_{\theta}$ for all sizes, solving more instances in less time. From Fig. 4 we conclude that DOMP$_{\theta 1}$ and DOMP$_{\theta 1a}$ are able to compute the same or more instances to optimality for mostly all different criteria in less running time. In this case, we highlight that, while the DOMP$_{\theta}$ is not able to prove optimality for any of the instances with λ_{11}, our reformulations can, although as already noted in **Table 2**, DOMP$_{r0}$ performs very well for this specific criterion.

~~Regarding to~~ **Regarding** `Random` instances, the cumulative percentage of instances solved within a time limit and the cumulative percentage of instances solved within a gap value are shown in Fig. 5 and Fig. 6, respectively. A first comment on these results is that, as already pointed in Section 3, we expected DOMP$_{r0}$ performance to be much worse for the `Random` set of instances than for `Beasley` due to the fact that DOMP$_{r0}$ leverages from having a considerable number of repetitions in the cost matrix. On the other hand, it can also be observed that for this set of instances where we have almost no repeated costs, the performance of our approaches is very similar in general to the compact DOMP$_{\theta}$ formulation. However, from Fig. 7 where the number of instances and the time to solve them are shown, we observe that our contribution for this set of `Random` instances is that for large `n` values (180 and 200) DOMP$_{\theta 1}$ and DOMP$_{\theta 1a}$ are able to certify optimality for more instances and in less time than DOMP$_{r0}$ and DOMP$_{\theta}$.

For the sake of completeness and in order to give some more insights on the performance of the models based on $\mathsf{DOMP}_{\theta 1}$ and $\mathsf{DOMP}_{\theta 1a}$, we include some detailed results in Table 3 and Table 4 focusing on the strength of our cuts and their efficiency. Here, columns `lazy` and `user` indicate the total number of integer (or lazy) cuts and fractional (or user) cuts added until termination, respectively. Since we have not made uniform use of the stabilization procedures, we only show in these tables the number of cuts added only within the Branch-and-Benders-cut algorithm. A first comment on these results (a result which is also known in the standard facility location literature) is that, the greater the number p of facilities to locate, the better results can be obtained in terms of running time or final gaps, in general. The reader might notice that there are certain λ-vectors for which we are not adding Benders cuts. These are the cases for which Δ^- is empty, i.e., ~~for the λ_1~~ **for** λ_1 (shown in the summary result tables) and λ_3. However, even in these cases where no cuts are added, our reformulations $\mathsf{DOMP}_{\theta 1}$ and $\mathsf{DOMP}_{\theta 1a}$ are still making a contribution in the sense that they are lighter models with a smaller number of both variables and constraints. Overall, there is a correlation between the number of cuts we are able to separate and the size of the set Δ^-. When $|\Delta^-|$ is large, i.e., we have many negative "jumps" in our Δ vector, in every separation call of the model $\mathsf{DOMP}_{\theta 1}$, we potentially add up to $|\Delta^-|$ cuts. In an extreme case, when λ_{11}, we have $|\Delta^-| = |\Delta|$. In that case, many more cuts are separated for $\mathsf{DOMP}_{\theta 1a}$ than for $\mathsf{DOMP}_{\theta 1}$, further emphasizing the difference between these two models. On the contrary, when the size of Δ^+ is large, then $|\Delta^-|$ is relatively small. In that case, in every node we are adding fewer Benders cuts since the master LP contains all the constraints coming from Δ^+ part of the problem.

7. Conclusions

This paper has ~~proved~~ **demonstrated** that using Benders decomposition techniques brings new features to the analysis and solution techniques of DOMP. Building from two different state-of-the-art formulations of DOMP, namely the one based on radius variables (3) and the other based on partial monotonicity (9), we have developed three different Benders decompositions for solving this problem. We have also tried to enlarge the catalogue of problems previously solved in the literature of DOMP and thus, we have conducted computational experiments including very difficult λ-vectors of weights. Those include weight vectors with positive and negative entries, but also allow the objective function of the problem to be negative. This last feature was known to be very challenging and not fully addressed up to this paper. By strengthening bounds and exploiting various implementation enhancement techniques, we have improved the performance of our methods to the point of being comparable or superior to state-of-the-art approaches. Extensive computational results are included in the paper supporting our claims and also as source for comparison with other future approaches.

The proposed Benders decomposition approaches focus on projecting out variables that model the "ordering" aspect of DOMP, while the classical facility opening and customer allocation variables are kept in the master problem. That way, we were able to obtain fast combinatorial procedures for deriving Benders cuts. However, there is a (yet unexplored) potential to further "thin-out" the proposed models and obtain new Benders reformulations in which also allocation variables x are projected out. This remains an interesting research direction for potential future work.

Acknowledgements

The authors want to thank the three anonymous reviewers for their constructive comments that have helped in improving the contribution of the paper. The authors also acknowledge financial support by the reasearch project PID2020-114594GB-C21 funded by the Spanish Ministerio de Ciencia e Innovación and Agencia Estatal de Investigación (MCIN/ AEI /10.13039/501100011033). The authors also acknowledge partial support from NetmeetData: Ayudas Fundación BBVA a equipos de investigación científica 2019 with reference "COMPLEX NETWORK".

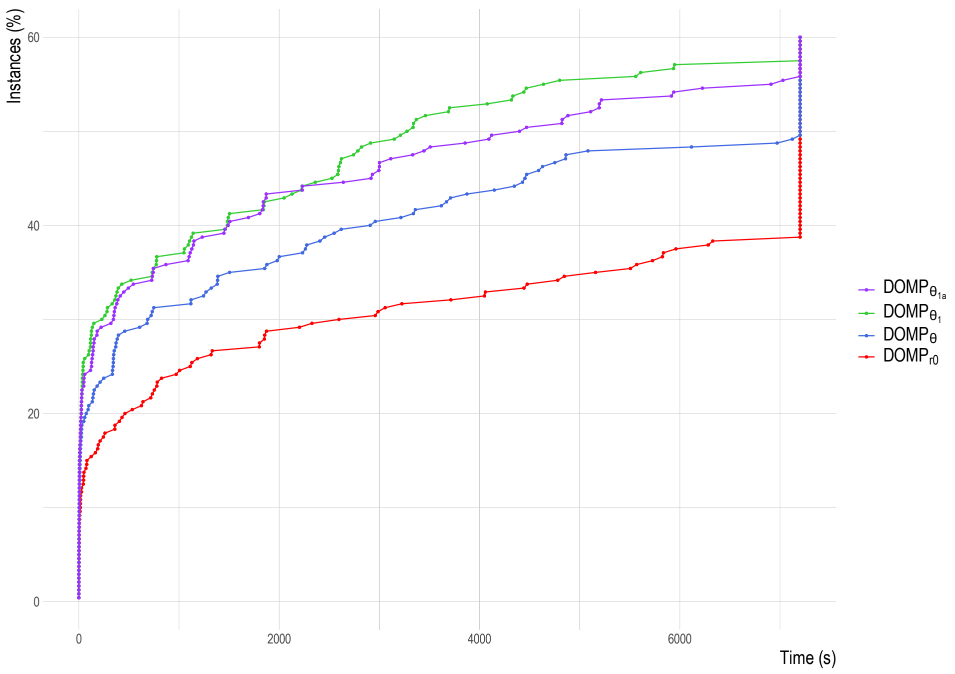

Fig. 1: Cumulative % of `Beasley` instances solved **to optimality** to a given time

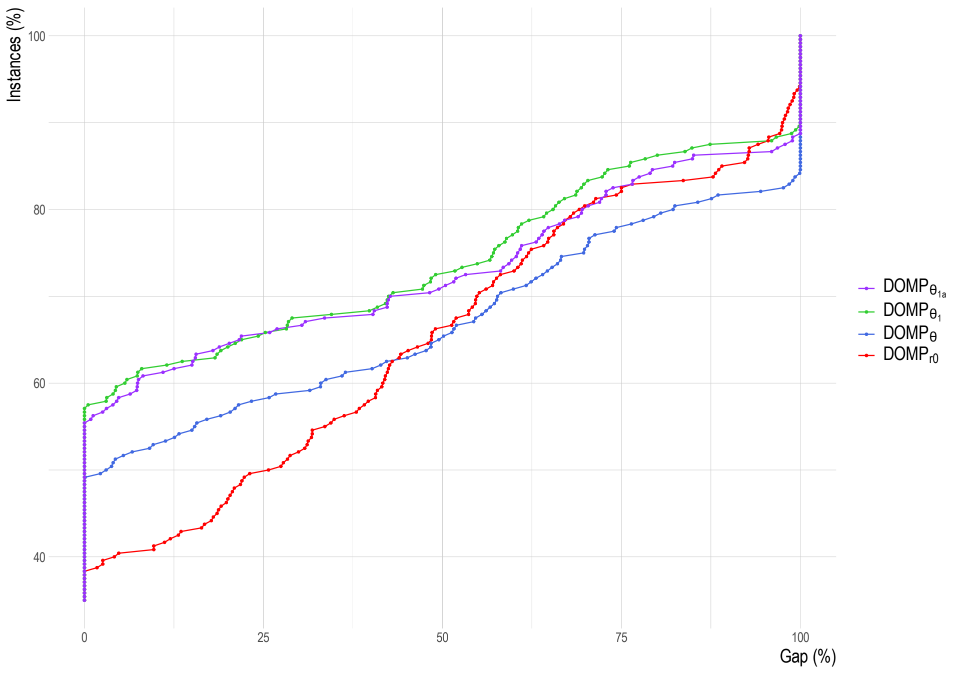

Fig. 2: Cumulative % of `Beasley` instances solved within a gap value

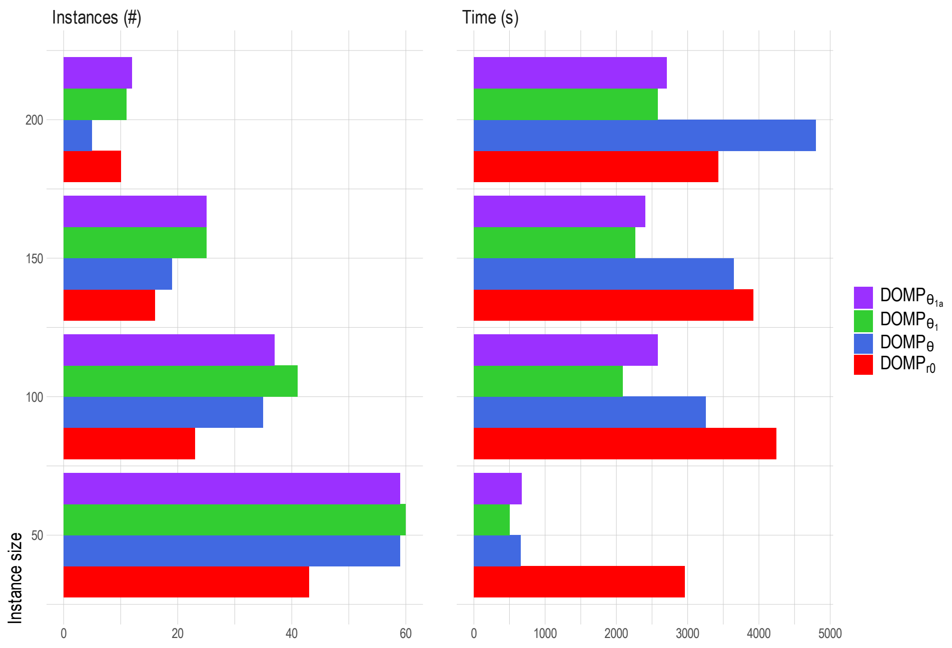

Fig. 3: The number of `Beasley` instances solved **to optimality** and running time per instance size

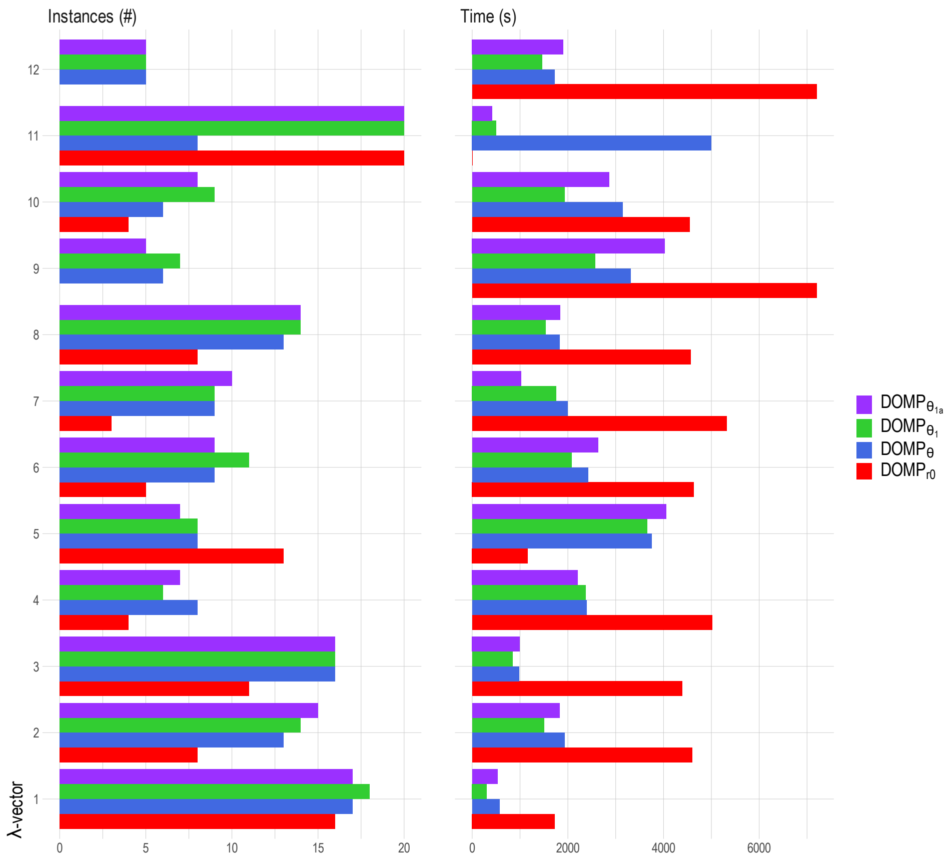

Fig. 4: The number of `Beasley` instances solved **to optimality** and running time per criterion

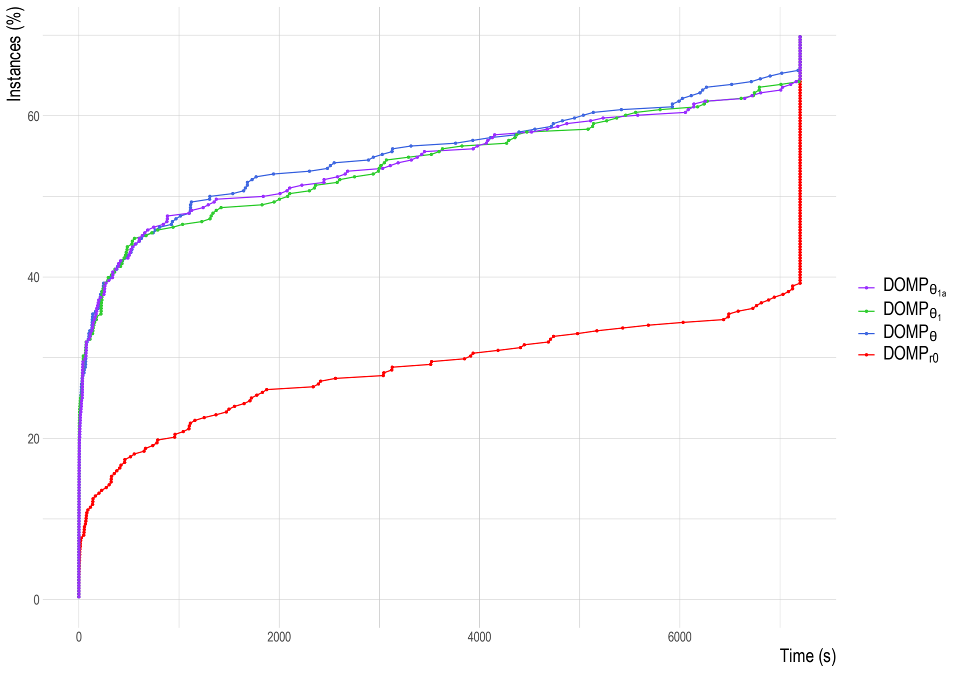

Fig. 5: Cumulative % of `Random` instances solved **to optimality** to a given time

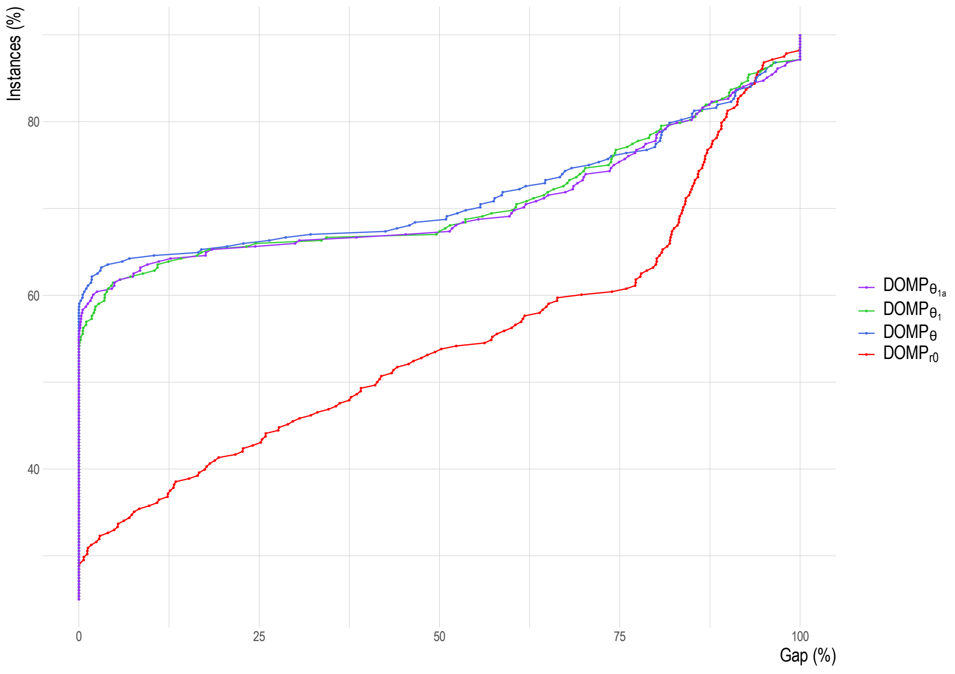

Fig. 6: Cumulative % of `Random` instances solved within a gap value

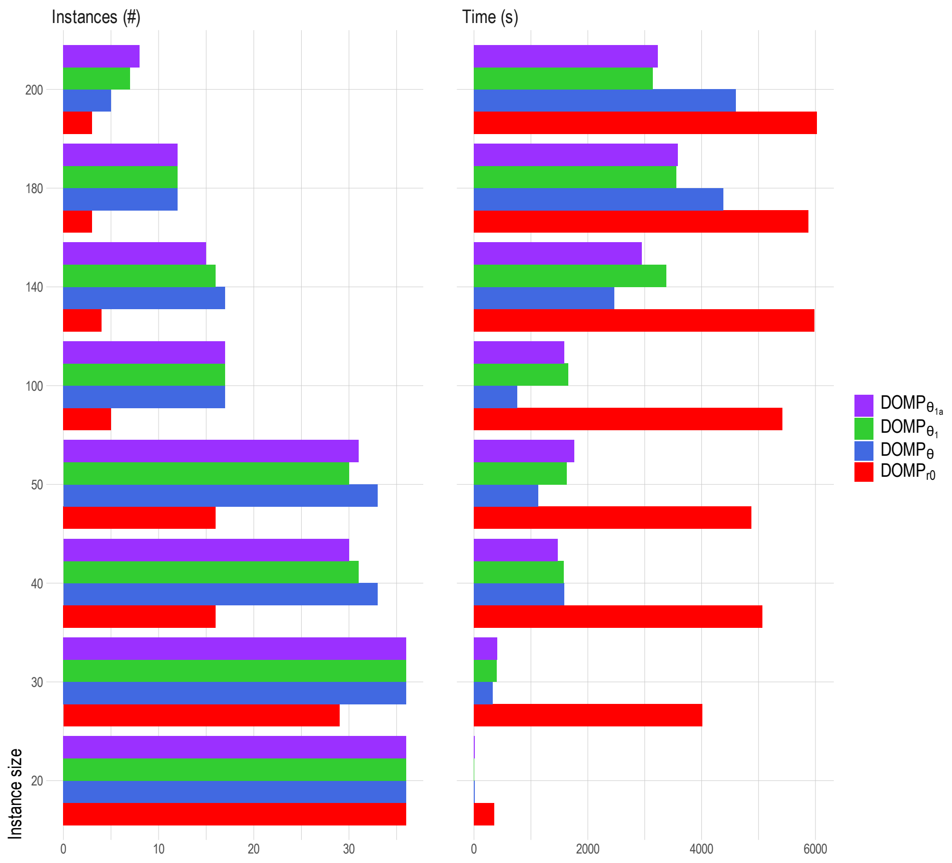

Fig. 7: The number of Random instances solved **to optimality** and running time per instance size

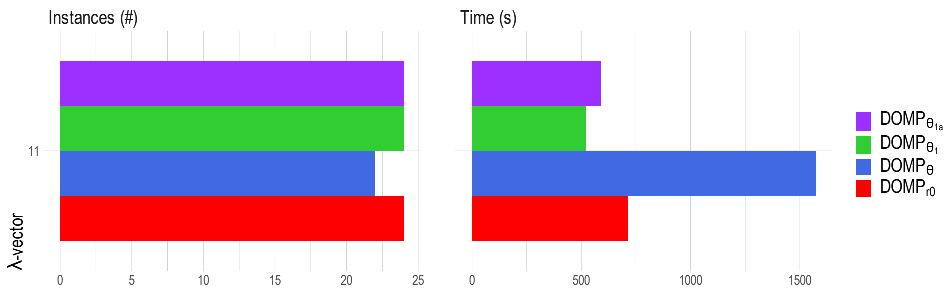

Fig. 8: The number of Random instances solved **to optimality** and running time for λ_{11}

λ	DOMP$_{r0}$		DOMP$_{\theta}$		DOMP$_{\theta 1}$				DOMP$_{\theta 1a}$			
λ_3	cpu	gap	cpu	gap	cpu	gap	lazy	user	cpu	gap	lazy	user
1-50-5	624.74	0.00	1.49	0.00	1.40	0.00	0	0	1.43	0.00	0	0
2-50-10	2202.05	0.00	2.15	0.00	2.15	0.00	0	0	2.01	0.00	0	0
3-50-10	6282.73	0.00	3.17	0.00	2.64	0.00	0	0	6.37	0.00	0	0
4-50-20	778.56	0.00	0.08	0.00	0.07	0.00	0	0	0.07	0.00	0	0
5-50-33	26.20	0.00	0.03	0.00	0.03	0.00	0	0	0.02	0.00	0	0
6-100-5	7200.00	18.52	4858.98	0.00	2621.46	0.00	0	0	5216.46	0.00	0	0
7-100-10	7200.00	13.12	2233.24	0.00	2221.03	0.00	0	0	2230.18	0.00	0	0
8-100-20	4475.95	0.00	1387.51	0.00	1482.35	0.00	0	0	126.72	0.00	0	0
9-100-40	5568.64	0.00	0.49	0.00	0.44	0.00	0	0	0.51	0.00	0	0
10-100-67	2326.94	0.00	0.34	0.00	0.33	0.00	0	0	0.54	0.00	0	0
11-150-5	7200.00	20.66	4751.08	0.00	4799.92	0.00	0	0	4469.07	0.00	0	0
12-150-10	3057.69	0.00	2453.91	0.00	2360.36	0.00	0	0	3855.47	0.00	0	0
13-150-30	7200.00	20.03	7200.00	3.02	7200.00	0.51	0	0	7200.00	2.53	0	0
14-150-60	4780.37	0.00	1.10	0.00	1.08	0.00	0	0	1.12	0.00	0	0
15-150-100	4048.94	0.00	1.49	0.00	1.47	0.00	0	0	2.26	0.00	0	0
16-200-5	7200.00	51.92	7200.00	4.32	7200.00	4.32	0	0	7200.00	3.08	0	0
17-200-10	7200.00	42.30	7200.00	4.00	7200.00	3.99	0	0	7200.00	3.97	0	0
18-200-40	7200.00	19.09	7200.00	3.78	7200.00	7.37	0	0	7200.00	7.37	0	0
19-200-80	7200.00	11.18	4.02	0.00	2.46	0.00	0	0	3.83	0.00	0	0
20-200-133	7200.00	18.73	3.42	0.00	4.73	0.00	0	0	3.34	0.00	0	0
λ_7	cpu	gap	cpu	gap	cpu	gap	lazy	user	cpu	gap	lazy	user
1-50-5	7200.00	76.53	382.94	0.00	359.25	0.00	3	0	354.57	0.00	3	0
2-50-10	7200.00	89.05	25.14	0.00	15.77	0.00	3	1	17.82	0.00	3	1
3-50-10	7200.00	71.11	368.02	0.00	22.93	0.00	3	0	25.48	0.00	3	0
4-50-20	1803.42	0.00	10.19	0.00	6.38	0.00	3	0	6.39	0.00	3	0
5-50-33	187.68	0.00	4.78	0.00	4.02	0.00	3	0	3.93	0.00	3	0
6-100-5	7200.00	97.12	7200.00	63.07	7200.00	56.62	4	0	7200.00	18.84	4	0
7-100-10	7200.00	92.65	7200.00	59.91	7200.00	48.33	3	1	7200.00	63.93	4	0
8-100-20	7200.00	75.00	3214.78	0.00	4638.68	0.00	3	0	3444.02	0.00	3	0
9-100-40	7200.00	68.32	4453.26	0.00	1851.68	0.00	3	0	1870.44	0.00	3	0
10-100-67	825.64	0.00	145.38	0.00	125.62	0.00	3	0	126.26	0.00	3	0
11-150-5	7200.00	98.21	7200.00	100.00	7200.00	100.00	3	0	7200.00	100.00	3	0
12-150-10	7200.00	83.64	7200.00	13.21	7200.00	8.00	4	0	3507.46	0.00	4	0
13-150-30	7200.00	88.14	7200.00	85.67	7200.00	65.45	3	0	7200.00	82.17	3	0
14-150-60	7200.00	12.00	4146.97	0.00	3368.56	0.00	3	0	869.16	0.00	3	0
15-150-100	7200.00	53.66	7200.00	41.38	7200.00	28.21	3	0	7200.00	42.28	3	0
16-200-5	7200.00	100.00	7200.00	100.00	7200.00	99.34	4	0	7200.00	100.00	4	0
17-200-10	7200.00	100.00	7200.00	100.00	7200.00	100.00	4	0	7200.00	100.00	4	0
18-200-40	7200.00	100.00	7200.00	100.00	7200.00	100.00	3	0	7200.00	100.00	3	0
19-200-80	7200.00	74.29	7200.00	88.51	7200.00	84.86	3	0	7200.00	84.92	3	0
20-200-133	7200.00	46.51	7200.00	42.19	7200.00	42.49	3	0	7200.00	42.44	3	0
λ_8	cpu	gap	cpu	gap	cpu	gap	lazy	user	cpu	gap	lazy	user
1-50-5	5960.74	0.00	347.76	0.00	18.53	0.00	18	4	26.71	0.00	12	3
2-50-10	7200.00	9.68	10.92	0.00	21.26	0.00	12	6	12.36	0.00	9	3
3-50-10	4846.48	0.00	336.09	0.00	22.41	0.00	18	6	388.86	0.00	7	3
4-50-20	122.78	0.00	9.49	0.00	6.85	0.00	14	4	5.63	0.00	6	3
5-50-33	44.78	0.00	7.35	0.00	5.55	0.00	8	3	4.03	0.00	3	3
6-100-5	7200.00	16.77	3618.33	0.00	3340.95	0.00	14	6	1089.98	0.00	8	3
7-100-10	7200.00	37.97	2261.79	0.00	1846.71	0.00	13	6	1869.62	0.00	5	3
8-100-20	717.39	0.00	98.54	0.00	115.99	0.00	7	3	115.83	0.00	5	1
9-100-40	1331.89	0.00	248.32	0.00	260.38	0.00	8	4	222.06	0.00	4	3
10-100-67	639.79	0.00	182.25	0.00	134.55	0.00	8	3	142.86	0.00	9	3
11-150-5	7200.00	36.28	7200.00	9.09	7200.00	11.50	13	5	7200.00	8.18	8	3
12-150-10	7200.00	19.82	1269.31	0.00	1124.14	0.00	13	6	2994.33	0.00	7	3
13-150-30	7200.00	41.53	4429.17	0.00	5559.28	0.00	10	4	7200.00	0.88	6	3
14-150-60	4442.84	0.00	211.80	0.00	429.01	0.00	8	3	319.23	0.00	5	3
15-150-100	7200.00	62.04	7200.00	62.39	2911.70	0.00	7	3	6224.55	0.00	8	3
16-200-5	7200.00	34.41	7200.00	5.43	7200.00	7.45	6	6	7200.00	21.92	5	3
17-200-10	7200.00	43.96	7200.00	49.50	7200.00	42.05	8	6	7200.00	49.50	3	3
18-200-40	7200.00	57.01	7200.00	58.18	7200.00	29.00	9	4	6909.11	0.00	5	2
19-200-80	7200.00	21.79	7200.00	64.00	7200.00	60.60	7	3	7200.00	98.93	3	0
20-200-133	7200.00	20.93	7200.00	55.52	7200.00	57.88	6	3	7200.00	51.58	3	3
λ_{11}	cpu	gap	cpu	gap	cpu	gap	lazy	user	cpu	gap	lazy	user
1-50-5	0.33	0.00	29.55	0.00	4.47	0.00	147	0	3.48	0.00	3	0
2-50-10	0.53	0.00	344.28	0.00	4.30	0.00	178	0	4.49	0.00	4	0
3-50-10	0.30	0.00	26.97	0.00	2.19	0.00	166	0	2.50	0.00	4	0
4-50-20	0.75	0.00	17.99	0.00	2.14	0.00	147	0	2.08	0.00	3	0
5-50-33	0.44	0.00	344.76	0.00	4.10	0.00	147	0	3.96	0.00	3	0
6-100-5	1.46	0.00	7200.00	100.00	25.71	0.00	297	0	24.96	0.00	3	0
7-100-10	14.73	0.00	7200.00	100.00	229.08	0.00	566	0	137.14	0.00	6	3
8-100-20	0.86	0.00	3874.37	0.00	31.62	0.00	321	0	25.72	0.00	4	0
9-100-40	0.74	0.00	7200.00	100.00	40.53	0.00	302	0	51.23	0.00	4	0
10-100-67	0.76	0.00	7200.00	100.00	32.46	0.00	297	0	344.73	0.00	3	0
11-150-5	4.00	0.00	7200.00	100.00	1105.17	0.00	534	0	1127.73	0.00	4	0
12-150-10	2.69	0.00	4347.19	0.00	117.41	0.00	603	0	150.65	0.00	5	0
13-150-30	2.71	0.00	7200.00	100.00	285.98	0.00	503	0	182.58	0.00	4	0
14-150-60	1.77	0.00	4628.60	0.00	102.81	0.00	514	0	132.19	0.00	4	0
15-150-100	2.41	0.00	7200.00	100.00	777.31	0.00	447	0	1111.14	0.00	3	0
16-200-5	6.71	0.00	7200.00	100.00	772.47	0.00	1020	0	543.95	0.00	4	1
17-200-10	7.31	0.00	7200.00	100.00	332.55	0.00	796	0	413.16	0.00	4	0
18-200-40	4.80	0.00	7200.00	100.00	3210.45	0.00	712	0	493.90	0.00	6	3
19-200-80	4.09	0.00	7200.00	100.00	393.22	0.00	655	0	3113.80	0.00	6	0
20-200-133	3.97	0.00	7200.00	100.00	2591.39	0.00	597	0	448.69	0.00	3	0

Table 3: `Beasley` instances result tables for some λ-vectors

λ	DOMP$_{r0}$		DOMP$_{\theta}$		DOMP$_{\theta 1}$				DOMP$_{\theta 1a}$			
λ_3	cpu	gap	cpu	gap	cpu	gap	lazy	user	cpu	gap	lazy	user
20-5	24.23	0.00	0.12	0.00	0.13	0.00	0.00	0.00	0.12	0.00	0.00	0.00
20-6	139.24	0.00	0.16	0.00	0.18	0.00	0.00	0.00	0.17	0.00	0.00	0.00
20-10	69.48	0.00	0.06	0.00	0.08	0.00	0.00	0.00	0.07	0.00	0.00	0.00
30-7	6582.21	8.38	1.81	0.00	1.83	0.00	0.00	0.00	3.12	0.00	0.00	0.00
30-10	3935.77	0.02	2.05	0.00	2.53	0.00	0.00	0.00	1.86	0.00	0.00	0.00
30-15	1042.16	0.00	0.61	0.00	0.62	0.00	0.00	0.00	0.56	0.00	0.00	0.00
40-10	7200.00	12.35	11.30	0.00	111.12	0.00	0.00	0.00	111.15	0.00	0.00	0.00
40-13	7200.00	6.23	3.80	0.00	8.94	0.00	0.00	0.00	73.46	0.00	0.00	0.00
40-20	2391.86	0.00	1.07	0.00	1.24	0.00	0.00	0.00	1.73	0.00	0.00	0.00
50-12	7200.00	13.20	386.72	0.00	241.66	0.00	0.00	0.00	172.03	0.00	0.00	0.00
50-16	6885.43	2.82	16.02	0.00	10.21	0.00	0.00	0.00	8.77	0.00	0.00	0.00
50-25	3044.59	0.69	1.63	0.00	1.87	0.00	0.00	0.00	1.65	0.00	0.00	0.00
100-25	7200.00	30.60	5923.57	0.76	5558.21	0.53	0.00	0.00	4780.92	0.19	0.00	0.00
100-33	7200.00	24.11	624.45	0.00	529.72	0.00	0.00	0.00	502.74	0.00	0.00	0.00
100-50	7200.00	17.44	40.16	0.00	35.01	0.00	0.00	0.00	41.35	0.00	0.00	0.00
140-35	7200.00	25.91	7200.00	1.80	7200.00	1.78	0.00	0.00	7200.00	1.79	0.00	0.00
140-46	7200.00	22.76	5991.38	0.26	6242.15	0.26	0.00	0.00	6083.40	0.24	0.00	0.00
140-70	7200.00	16.46	130.91	0.00	136.21	0.00	0.00	0.00	135.58	0.00	0.00	0.00
180-45	7200.00	38.53	7200.00	1.78	7200.00	1.81	0.00	0.00	7200.00	2.00	0.00	0.00
180-60	7200.00	18.16	7200.00	0.54	7200.00	0.55	0.00	0.00	7200.00	0.54	0.00	0.00
180-90	7200.00	13.12	567.99	0.00	472.68	0.00	0.00	0.00	511.89	0.00	0.00	0.00
200-50	7200.00	84.16	7200.00	2.58	7200.00	2.22	0.00	0.00	7200.00	2.51	0.00	0.00
200-66	7200.00	83.32	7200.00	1.02	7200.00	1.03	0.00	0.00	7200.00	1.23	0.00	0.00
200-100	7200.00	82.52	3760.83	0.06	2997.36	0.08	0.00	0.00	3187.14	0.18	0.00	0.00
λ_7	cpu	gap	cpu	gap	cpu	gap	lazy	user	cpu	gap	lazy	user
20-5	650.92	0.00	1.09	0.00	0.90	0.00	1.60	0.40	33.03	0.00	3.00	1.20
20-6	553.33	0.00	0.99	0.00	0.70	0.00	1.80	0.30	32.79	0.00	3.40	0.60
20-10	515.31	0.00	0.56	0.00	0.52	0.00	1.60	0.10	0.43	0.00	3.20	0.60
30-7	7200.00	57.18	23.48	0.00	142.40	0.00	1.60	0.30	76.24	0.00	3.20	0.40
30-10	6942.96	22.69	10.23	0.00	73.10	0.00	1.60	0.10	70.63	0.00	3.00	0.40
30-15	6436.51	4.02	7.13	0.00	5.12	0.00	1.50	0.10	36.03	0.00	3.00	0.20
40-10	7200.00	94.77	644.03	0.00	466.34	0.00	1.90	0.10	338.63	0.00	3.60	0.00
40-13	7200.00	85.43	356.56	0.00	224.65	0.00	1.50	0.00	179.34	0.00	3.00	0.00
40-20	7200.00	10.84	16.45	0.00	44.38	0.00	1.60	0.10	9.53	0.00	3.20	0.20
50-12	7200.00	78.75	132.87	0.00	113.39	0.00	1.50	0.10	100.48	0.00	3.20	0.00
50-16	7200.00	69.69	50.67	0.00	31.92	0.00	1.50	0.10	26.29	0.00	3.00	0.20
50-25	7200.00	28.97	25.78	0.00	17.04	0.00	1.60	0.00	14.48	0.00	3.20	0.00
100-25	7200.00	100.00	7200.00	10.41	7200.00	10.48	1.80	0.00	7200.00	9.50	2.70	0.00
100-33	7200.00	96.15	934.14	0.00	728.88	0.00	1.70	0.00	687.63	0.00	3.60	0.20
100-50	7200.00	81.92	477.54	0.00	432.68	0.00	1.70	0.00	397.38	0.00	3.40	0.00
140-35	7200.00	100.00	7017.64	42.50	7008.64	53.59	1.50	0.00	6806.11	61.98	1.50	0.00
140-46	7200.00	100.00	4827.15	0.00	6611.55	50.00	1.50	0.00	6140.11	53.27	1.50	0.00
140-70	7200.00	100.00	2891.72	0.00	3049.43	0.00	1.50	0.00	3108.95	0.01	1.50	0.00
180-45	7200.00	100.00	7200.00	100.00	7200.00	83.35	1.70	0.00	7200.00	100.00	1.70	0.00
180-60	7200.00	100.00	7200.00	100.00	6792.30	70.21	1.50	0.00	7106.24	80.08	1.60	0.00
180-90	7200.00	100.00	7200.00	100.00	6793.89	70.00	1.70	0.00	6720.80	70.00	1.70	0.00
200-50	7200.00	100.00	7200.00	100.00	7200.00	100.00	1.60	0.00	7200.00	100.00	1.60	0.00
200-66	7200.00	100.00	7200.00	100.00	7200.00	100.00	1.60	0.00	7200.00	100.00	1.60	0.00
200-100	7200.00	100.00	7200.00	100.00	7200.00	100.00	1.70	0.00	7200.00	100.00	1.70	0.00
λ_8	cpu	gap	cpu	gap	cpu	gap	lazy	user	cpu	gap	lazy	user
20-5	14.95	0.00	0.46	0.00	0.40	0.00	7.10	4.10	0.33	0.00	5.40	3.00
20-6	14.84	0.00	0.52	0.00	0.42	0.00	6.30	3.50	0.26	0.00	4.00	1.40
20-10	141.15	0.00	0.43	0.00	0.44	0.00	6.50	3.00	0.21	0.00	4.20	1.00
30-7	1649.23	0.00	3.03	0.00	34.53	0.00	7.70	3.60	1.76	0.00	6.20	3.00
30-10	1251.32	0.00	3.69	0.00	2.00	0.00	7.20	3.30	1.15	0.00	4.20	2.00
30-15	779.12	0.00	4.23	0.00	1.86	0.00	8.00	3.00	2.12	0.00	5.00	3.00
40-10	6728.33	43.37	71.07	0.00	40.56	0.00	8.60	3.90	7.30	0.00	6.00	2.60
40-13	5685.67	17.72	73.78	0.00	4.41	0.00	7.30	3.30	59.76	0.00	5.20	2.60
40-20	1498.25	0.00	7.97	0.00	36.33	0.00	7.10	3.10	34.96	0.00	5.80	3.00
50-12	4709.14	7.33	14.79	0.00	44.32	0.00	8.10	3.70	24.53	0.00	6.40	3.00
50-16	4978.01	25.22	12.34	0.00	7.07	0.00	7.90	3.30	4.24	0.00	4.40	1.80
50-25	3123.47	2.92	12.25	0.00	7.81	0.00	7.30	3.00	13.15	0.00	5.20	3.00
100-25	7200.00	82.24	294.98	0.00	2364.30	4.06	8.00	3.30	3032.64	4.91	6.80	3.00
100-33	7200.00	85.95	194.07	0.00	1035.15	2.29	6.70	3.40	128.23	0.00	4.80	3.00
100-50	7200.00	91.28	217.63	0.00	3518.56	3.58	6.60	3.00	658.22	0.00	5.60	3.00
140-35	7200.00	86.51	1536.35	0.00	2578.32	3.60	8.80	3.50	3378.45	7.47	5.20	3.00
140-46	7200.00	89.48	1683.21	0.00	5269.45	10.92	9.90	3.30	1125.79	0.00	5.80	3.00
140-70	7200.00	92.25	2510.94	0.00	5802.85	8.88	6.40	3.00	602.07	0.00	5.00	3.00
180-45	7200.00	88.69	6200.90	22.81	5128.99	3.88	7.60	3.90	6252.01	8.54	6.40	3.00
180-60	7200.00	89.95	6265.51	3.11	5137.93	7.17	7.70	3.30	5233.71	11.09	5.20	3.00
180-90	7200.00	94.02	7179.83	16.99	6732.91	18.06	7.50	3.10	6137.94	8.48	6.20	3.00
200-50	7200.00	89.78	7200.00	88.57	7200.00	51.47	7.20	4.30	7007.41	51.76	4.80	3.00
200-66	7200.00	91.78	7200.00	90.81	7200.00	76.74	7.40	3.60	7200.00	79.94	5.20	3.00
200-100	7200.00	93.87	7200.00	93.13	7200.00	76.03	6.50	3.00	7200.00	77.30	5.20	3.00
λ_{11}	cpu	gap	cpu	gap	cpu	gap	lazy	user	cpu	gap	lazy	user
20-5	0.09	0.00	0.13	0.00	0.17	0.00	63.90	0.00	0.19	0.00	4.00	0.00
20-6	0.27	0.00	0.30	0.00	0.36	0.00	67.10	6.00	0.46	0.00	4.60	0.60
20-10	0.19	0.00	0.23	0.00	0.35	0.00	59.70	0.00	0.39	0.00	3.80	0.00
30-7	4.08	0.00	68.02	0.00	1.47	0.00	125.20	20.70	2.97	0.00	6.20	1.80
30-10	65.76	0.00	1.25	0.00	1.09	0.00	103.60	0.00	1.33	0.00	4.40	0.20
30-15	0.57	0.00	1.25	0.00	1.12	0.00	98.40	0.00	1.42	0.00	4.80	0.00
40-10	51.39	0.00	70.69	0.00	5.22	0.00	160.80	31.20	70.01	0.00	5.60	2.40
40-13	1.03	0.00	2.78	0.00	1.39	0.00	137.10	3.70	1.63	0.00	4.20	0.40
40-20	0.92	0.00	2.63	0.00	1.19	0.00	133.40	0.00	1.40	0.00	4.20	0.00
50-12	7.74	0.00	14.12	0.00	5.04	0.00	211.70	34.60	8.35	0.00	10.60	1.20
50-16	2.17	0.00	7.37	0.00	2.67	0.00	183.10	0.00	2.75	0.00	4.40	0.00
50-25	1.70	0.00	6.77	0.00	2.50	0.00	172.40	0.00	2.86	0.00	4.20	0.00
100-25	227.04	0.00	377.72	0.00	229.34	0.00	550.00	196.20	218.29	0.00	23.10	2.50
100-33	55.96	0.00	213.63	0.00	156.31	0.00	402.40	78.40	161.86	0.00	7.10	2.10
100-50	53.79	0.00	204.16	0.00	138.82	0.00	327.50	0.00	139.80	0.00	4.20	1.50
140-35	959.51	0.00	1769.48	0.00	355.25	0.00	806.20	243.60	369.71	0.00	16.10	2.70
140-46	273.07	0.00	969.40	0.00	229.55	0.00	652.60	127.80	255.43	0.00	5.20	2.10
140-70	136.73	0.00	1013.76	0.00	173.47	0.00	468.80	0.00	198.92	0.00	3.70	0.60
180-45	1722.72	0.00	4948.15	0.00	534.47	0.00	796.20	391.70	600.39	0.00	4.50	2.40
180-60	956.11	0.00	3932.68	0.00	415.65	0.00	807.30	86.10	519.57	0.00	5.00	1.90
180-90	378.79	0.00	2939.66	0.00	360.76	0.00	694.10	0.00	392.59	0.00	4.50	1.20
200-50	5429.29	2.41	7200.00	80.86	4353.53	0.06	983.70	230.70	4150.96	0.45	4.90	2.20
200-66	4184.74	0.66	7200.00	80.57	3629.96	0.23	956.10	136.20	4519.16	0.33	5.30	1.80
200-100	2561.62	0.00	6803.08	80.00	1948.59	0.00	765.30	0.00	2582.17	0.00	4.30	0.60

Table 4: `Random` instances result tables for some λ-vectors

References

Beasley, J., 2015. http://people.brunel.ac.uk/~mastjjb/jeb/jeb.html.

Benders, J.F., 1962. Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik* 4, 1, 238–252.

Blanco, V., El Haj Ben Ali, S., Puerto, J., 2013. Minimizing ordered weighted averaging of rational functions with applications to continuous location. *Computers & Operations Research* 40, 5, 1448–1460.

Blanco, V., El Haj Ben Ali, S., Puerto, J., 2014. Revisiting several problems and algorithms in continuous location with ℓ-norms. *Computational Optimization and Applications* 58, 3, 563–595.

Blanco, V., Gázquez, R., Ponce, D., Puerto, J., 2023. A branch-and-price approach for the continuous multifacility monotone ordered median problem. *European Journal of Operational Research* 306, 1, 105–126.

Blanco, V., Puerto, J., Ben-Ali, S.E.H., 2016. Continuous multifacility ordered median location problems. *European Journal of Operational Research* 250, 1, 56–64.

Boland, N., Domínguez-Marín, P., Nickel, S., Puerto, J., 2006. Exact procedures for solving the discrete ordered median problem. *Computers & Operations Research* 33, 11, 3270–3300.

Bonami, P., Salvagnin, D., Tramontani, A., 2020. Implementing automatic Benders decomposition in a modern mip solver. In *Integer Programming and Combinatorial Optimization: 21st International Conference, IPCO 2020, London, UK, June 8–10, 2020, Proceedings*, Springer, pp. 78–90.

Brandenberg, R., Stursberg, P., 2021. Refined cut selection for Benders decomposition: applied to network capacity expansion problems. *Mathematical Methods of Operations Research* pp. 1–30.

Conforti, M., Wolsey, L.A., 2019. "Facet" separation with one linear program. *Mathematical Programming* 178, 1, 361–380.

Coniglio, S., Furini, F., Ljubić, I., 2022. Submodular maximization of concave utility functions composed with a set-union operator with applications to maximal covering location problems. *Mathematical Programming* 196, 1, 9–56.

Cordeau, J., Furini, F., Ljubić, I., 2019. Benders decomposition for very large scale partial set covering and maximal covering location problems. *European Journal of Operational Research* 275, 3, 882–896.

Deleplanque, S., Labbé, M., Ponce, D., Puerto, J., 2020. A Branch-Price-and-Cut procedure for the discrete ordered median problem. *INFORMS Journal on Computing* 32, 3, 582–599. https://doi.org/10.1287/ijoc.2019.0915.

Duran-Mateluna, C., Ales, Z., Elloumi, S., 2023. An efficient Benders decomposition for the p-median problem. *European Journal of Operational Research*. doi: https://doi.org/10.1016/j.ejor.2022.11.033.

Elloumi, S., Labbé, M., Pochet, Y., 2004. A new formulation and resolution method for the p-center problem. *INFORMS Journal on Computing* 16, 1, 84–94.

Espejo, I., Marín, A., Puerto, J., Rodríguez-Chía, A.M., 2009. A comparison of formulations and solution methods for the minimum-envy location problem. *Computers & Operations Research* 36, 6, 1966–1981.

Espejo, I., Marín, A., Rodríguez-Chía, A.M., 2012. Closest assignment constraints in discrete location problems. *European Journal of Operational Research* 219, 1, 49–58.

Fernández, E., Pozo, M.A., Puerto, J., 2014. Ordered weighted average combinatorial optimization: Formulations and their properties. *Discrete Applied Mathematics* 169, 97–118.

Fernández, E., Pozo, M.A., Puerto, J., Scozzari, A., 2017. Ordered weighted average optimization in multiobjective spanning tree problem. *European Journal of Operational Research* 260, 3, 886–903.

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

Fischetti, M., Ljubić, I., Sinnl, M., 2017. Redesigning Benders decomposition for large-scale facility location. *Management Science* 63, 7, 2146–2162.

Fischetti, M., Salvagnin, D., Zanette, A., 2010. A note on the selection of Benders' cuts. *Mathematical Programming* 124, 1, 175–182.

Fortz, B., Poss, M., 2009. An improved Benders decomposition applied to a multi-layer network design problem. *Operations research letters* 37, 5, 359–364.

Gaar, E., Sinnl, M., 2022. A scaleable projection-based branch-and-cut algorithm for the p-center problem. *European Journal of Operational Research* 303, 1, 78–98.

Galand, L., Spanjaard, O., 2012. Exact algorithms for OWA-optimization in multiobjective spanning tree problems. *Computers & Operations Research* 39, 7, 1540–1554.

García, S., Labbé, M., Marín, A., 2011. Solving large p-median problems with a radius formulation. *INFORMS Journal on Computing* 23, 4, 546–556.

Geoffrion, A.M., 1972. Generalized Benders decomposition. *Journal of optimization theory and applications* 10, 4, 237–260.

Kalcsics, J., Nickel, S., Puerto, J., Rodríguez-Chía, A.M., 2010a. Distribution systems design with role dependent objectives. *European Journal of Operational Research* 202, 2, 491–501.

Kalcsics, J., Nickel, S., Puerto, J., Rodríguez-Chía, A.M., 2010b. The ordered capacitated facility location problem. *Top* 18, 1, 203–222.

Kalcsics, J., Nickel, S., Puerto, J., Tamir, A., 2002. Algorithmic results for ordered median problems. *Operations Research Letters* 30, 3, 149–158.

Kelley, J.E. Jr, 1960. The cutting-plane method for solving convex programs. *Journal of the society for Industrial and Applied Mathematics* 8, 4, 703–712.

Labbé, M., Ponce, D., Puerto, J., 2017. A comparative study of formulations and solution methods for the discrete ordered p-median problem. *Computers & Operations Research* 78, 230–242.

Lasdon, L.S., 2002. *Optimization theory for large systems*. Courier Corporation.

Magnanti, T.L., Wong, R.T., 1981. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Operations research* 29, 3, 464–484.

Marín, A., Nickel, S., Puerto, J., Velten, S., 2009. A flexible model and efficient solution strategies for discrete location problems.

Discrete Applied Mathematics 157, 5, 1128–1145.

Marín, A., Nickel, S., Velten, S., 2010. An extended covering model for flexible discrete and equity location problems. *Mathematical Methods of Operations Research* 71, 1, 125–163.

Marín, A., Ponce, D., Puerto, J., 2020. A fresh view on the discrete ordered median problem based on partial monotonicity. *European Journal of Operational Research* 286, 3, 839–848.

Martello, S., Toth, P., 1990. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc.

Martínez-Merino, L.I., Ponce, D., Puerto, J., 2022. Constraint relaxation for the discrete ordered median problem. *TOP*. doi: 10.1007/s11750-022-00651-3.

Mesa, J.A., Puerto, J., Tamir, A., 2003. Improved algorithms for several network location problems with equality measures. *Discrete applied mathematics* 130, 3, 437–448.

Minoux, M., 1986. *Mathematical programming: theory and algorithms*. John Wiley & Sons.

Naoum-Sawaya, J., Elhedhli, S., 2013. An interior-point Benders based branch-and-cut algorithm for mixed integer programs. *Annals of Operations Research* 210, 1, 33–55.

Nickel, S., 2001. Discrete Ordered Weber Problems. In Fleischmann, B., Lasch, R., Derigs, U., Domschke, W. and Rieder, U. (eds), *Operations Research Proceedings*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 71–76.

Nickel, S., Puerto, J., 2006. *Location theory: a unified approach*. Springer Science & Business Media.

Ogryczak, W., Tamir, A., 2003. Minimizing the sum of the k largest functions in linear time. *Information Processing Letters* 85, 3, 117–122.

Pozo, M.A., Puerto, J., Chía, A.M.R., 2021. The ordered median tree of hubs location problem. *TOP* 29, 1, 78–105.

Pozo, M.A., Puerto, J., Torrejón, A., 2023. An extended version of the Ordered Median Tree Location Problem including appendices and detailed computational results. *arXiv* https://arxiv.org/abs/2207.05478.

Puerto, J., 2008. A new formulation of the capacitated discrete ordered median problem with $\{0, 1\}$-assignment. *Operations research proceedings 2007* pp. 165–170.

Puerto, J., Ramos, A., Rodríguez-Chía, A.M., 2011. Single-allocation ordered median hub location problems. *Computers & Operations Research* 38, 2, 559–570.

Puerto, J., Ramos, A., Rodríguez-Chía, A.M., 2013. A specialized branch & bound & cut for single-allocation ordered median hub location problems. *Discrete Applied Mathematics* 161, 16-17, 2624–2646.

Puerto, J., Ramos, A., Rodríguez-Chía, A.M., Sánchez-Gil, M.C., 2016. Ordered median hub location problems with capacity constraints. *Transportation Research Part C: Emerging Technologies* 70, 142–156.

Puerto, J., Rodríguez-Chía, A.M., 2015. Ordered median location problems. In *Location science*. Springer, pp. 249–288.

Puerto, J., Tamir, A., 2005. Locating tree-shaped facilities using the ordered median objective. *Mathematical Programming* 102, 2, 313–338.

Rahmaniani, R., Crainic, T.G., Gendreau, M., Rei, W., 2017. The Benders decomposition algorithm: A literature review. *European Journal of Operational Research* 259, 3, 801–817.

Ramírez-Pico, C., Ljubić, I., Moreno, E., 2023. Benders adaptive-cuts method for two-stage stochastic programs. *Transportation Science, to appear* 0, 0, null.

Redondo, J.L., Marín, A., Ortigosa, P.M., 2016. A parallelized Lagrangean relaxation approach for the discrete ordered median problem. *Ann. Oper. Res.* 246, 1-2, 253–272.