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

Robust min-max regret covering problems

Amadeu A. Coco · Andréa Cynthia Santos · Thiago F. Noronha

Received: date / Accepted: date

Abstract This article deals with two min-max regret covering problems: the min-max regret Weighted Set Covering Problem (*min-max regret* WSCP) and the min-max regret Maximum Benefit Set Covering Problem (*min-max regret* MSCP). These problems are the robust optimization counterparts, respectively, of the Weighted Set Covering Problem and of the Maximum Benefit Set Covering Problem. In both problems, uncertainty in data is modeled by using an interval of continuous values, representing all the infinite values every uncertain parameter can assume. This study has the following major contributions: (i) a proof that MSCP is Σ_p^2-Hard, (ii) a mathematical formulation for the *min-max regret* MSCP, (iii) exact and (iv) heuristic algorithms for the *min-max regret* WSCP and the *min-max regret* MSCP. We reproduce the main exact algorithms for the *min-max regret* WSCP found in the literature: a Logic-based Benders decomposition, an extended Benders decomposition and a branch-and-cut. In addition, such algorithms have been adapted for the *min-max regret* MSCP. Moreover, five heuristics are applied for both problems: two scenario-based heuristics, a path relinking, a pilot method and a linear programming-based heuristic. The goal is to analyze the impact of such methods on handling robust covering problems in terms of solution quality and performance.

Keywords Robust optimization · Covering problems · Heuristics · Exact methods · Uncertainties

1 Introduction

Robust Optimization (RO) is a methodology to deal with data uncertainty where the variability of the data is represented by deterministic values [1,39]. It emerged in the late sixties to deal with financial problems and has been applied, in general, as a way to self-protect against undesirable impacts due to vague approximations or incomplete, imprecise, or ambiguous data. Readers are referred to [1] and [38] for robust optimization theoretical issues, and to [43] and [48] for operations research applications.

Many robust counterparts of classical NP-Hard optimization problems have been studied in the literature [2,43,47,53]. These problems bring several challenges regarding the design of algorithms and formulations, since the complexity to compute the cost of a single solution is at

Amadeu A. Coco
Normandie Université, UNIHAVRE, UNIROUEN, INSA Rouen, LITIS, Le Havre, France.
E-mail: amadeuac1@gmail.com

Andréa Cynthia Santos
Normandie Université, UNIHAVRE, UNIROUEN, INSA Rouen, LITIS, Le Havre, France.

Thiago F. Noronha
Computer Science Department, Federal University of Minas Gerais, Belo Horizonte, Brazil.

least equal to the computational complexity of its classical version. Moreover, the formulations using some RO criteria, such as the *min-max regret*, may have an exponential number of constraints due to the linearization of the objective function nested operators. On the other hand, covering problems are one of the most studied combinatorial optimization problems [12,28], and investigating their robust counterparts is a recent trend to address uncertainties in such problems [9,17,18,19,29,40,42,47].

The min-max regret robust criterion is applied in this study since it is one of the most used RO criterion [1]. The uncertain data are modeled by means of a continuous interval, where any realization of a single value for each uncertain parameter is considered as a scenario that can happen. Whenever the deterministic counterpart is a minimization problem, the regret of a feasible solution X in a scenario s is the difference between the cost of X and the cost of an optimal solution in s. The regret slightly differs if the deterministic counterpart is a maximization problem. In this case, the regret is given by the difference between the cost of an optimal solution in s, and the cost of X. In both cases, the objective of a min-max regret problem is to find a solution in which the maximum regret is minimized.

This article is dedicated to two min-max regret covering problems: the min-max regret Weighted Set Covering Problem (*min-max regret* WSCP) and the min-max regret Maximum Benefit Set Covering Problem (*min-max regret* MSCP). Given an interval of costs associated with all facilities, the *min-max regret* WSCP consists in selecting a set of facilities to be opened, in such a way that all neighborhoods are covered by at least one facility. The aim is to minimizing the maximum regret. The *min-max regret* MSCP looks for opening a set of capacitated facilities, to cover all neighborhoods by using at least one facility, while minimizes the maximum benefit regret. The benefit is the sum of the inhabitants number on all neighborhoods serviced by a facility.

Previous studies are found in the literature for the *min-max regret* WSCP [17,47]. For instance, the study [7] is motivated by a probabilistic version of WSCP that was applied to health care management. Since there are uncertainties in the emergency medical services demands, the authors integrated the stochastic WSCP (PSCP) [8] with a general facility location problem (FLP) [45]. Let $\omega \in (0,1)$ be a constant, this problem, referred here as PSCP-FLP, aims to open a set of Emergency Medical Services (EMS) and to designate a set of vehicles to each EMS such that the expected cost is minimized and each constraint is ensured with a probability of at least ω. The PSCP-FLP can be transformed into the *min-max regret* WSCP by assigning a cost interval to open each EMS, changing the objective function to the one applied to *min-max regret* problems, and setting $\omega = 1$.

For the second problem addressed here, the *min-max regret* MSCP, preliminary results were presented in [19]. The benefit of each column is modeled as an interval of continuous values, and the objective is to select a subset of columns (facilities) that covers all the lines (neighborhoods) without violating a capacity constraint, and such that the maximum regret of using the chosen columns is minimized. The interest of investigating two *min-max regret* covering problems together is to analyze their difficulty, using similar methods.

The MSCP and the *min-max regret* MSCP are inspired by an application of locating field hospitals after large-scale disasters, such as the earthquakes that hit Kathmandu, Nepal in April 2015. It also applies to setting up vaccination centers in a crisis like the one of the COVID-19. MSCP models these type of applications as follows. A set of field hospitals must be placed among a set M of predefined candidate sites, such that a capacity $T \in \mathbb{N}^*$ is not exceeded. Given a set N of neighborhoods, a site $j \in M$ is said to cover a neighborhood $i \in N$ if and only if the distance d_{ij} between the barycenter of neighborhood i and j is smaller than a maximum distance $\hat{d}$. Considering $|N|$ and $|M|$ respectively as the number of lines and columns, a matrix $\{a_{ij}\}$ of size $|N| \times |M|$ is defined as follows: $a_{ij} = 1$ if $d_{ij} \leq \hat{d}$, and $a_{ij} = 0$ otherwise. One may note that each column of a_{ij} corresponds to a potential site to open a field hospital, while the lines stand to neighborhoods. We assume that $\hat{d}$ is such that there is at least one covering of N with a capacity less than or equal to T, as in this application no neighborhood may not be left without aid. The objective is to maximize the sum of b_j for all selected sites, where b_j is the population living up to $\hat{d}$ distance-units away from the site $j \in M$. This objective function aims at attending the most densely populated areas in practical applications, since they are more likely to have more people needing aid after a large-scale emergency. A direct extension of this problem is the *min-max regret* MSCP, where uncertainties are associated with the number of inhabitants requiring medical care, which is unknown in emergencies. These uncertainties can be modeled as an interval $[l_j, u_j]$ for each site $j \in M$, representing the minimum and the maximum number of people requiring aid. The objective is to minimize the maximum regret of inhabitants without access to field hospitals. Modeling this application employing RO, in particular, the *min-max regret* objective function, prevents solutions with high regret, which are not suitable in a crisis context.

This study extends the previous work and brings several contributions. First, the main exact algorithms for the *min-max regret* WSCP found in the literature [46] are reproduced, i.e. a Logic-based Benders decomposition (LBD), an Extended Benders decomposition (EB) and a Branch-and-Cut (B&C). In addition, five heuristics are proposed for the *min-max regret* WSCP: two scenario-based heuristics [17], a path relinking (PR) [33], a pilot method (PM) [52] and a linear programming-based heuristic (LPH) [2]. The main contributions regarding the *min-max regret* MSCP concerns the proof that *min-max regret* MSCP is Σ_p^2-Hard, by showing that it is a generalization of the *min-max regret* Knapsack problem [24]; the adaptation of two exact methods (EB and B&C) and three heuristics (PR, PM and LPH). In addition, one exact method (LBD) and two heuristics (SBA and AMU) are reproduced. To the best of our knowledge, this is the first study dedicated to compare a number of exact and heuristic methods for *min-max regret* covering problems. It is worth mentioning that the developed methods can be generalized to other *min-max regret* optimization problems, by simply exchanging the algorithm that solves the corresponding deterministic counterpart.

The remainder of this paper is organized as follows. Related works are discussed in Section 2. In the sequel, the *min-max regret* covering problems are formally defined in Section 3, followed by the description of the proposed exact and heuristic algorithms in Section 4. Computational experiments are reported in Section 5, while concluding remarks are given in Section 6.

2 Literature review

This section provides a literature review on covering, facility location, and robust optimization problems related with *min-max regret* WSCP (Section 2.1) and *min-max regret* MSCP (Section 2.2). Let us assume the following notation from now on: $\{a_{ij}\}$ is a matrix with a line-set N and a column-set M.

2.1 Studies related to *min-max regret* WSCP

The deterministic counterpart of the *min-max regret* WSCP is the well-known Weighted Set Covering Problem (WSCP) [27]. Let $\{a_{ij}\}$ be a matrix with $|N|$ lines and $|M|$ columns, such that $a_{ij} = 1$ if the column $j \in M$ covers the line $i \in N$, and $a_{ij} = 0$ otherwise. Moreover, each column $j \in M$ has an associated cost c_j. WSCP consists in finding a column-subset $X \subseteq M$ with the minimum cost, such that every line in N is covered by at least one column in X. This problem is NP-hard [32], and exact and heuristic algorithms are found in [4,12]. Practical applications of WSCP are found, among others, in scheduling of railway crews [11] and in locating bus stops for new routes [50].

The *min-max regret* WSCP was introduced by [47], and is an NP-hard problem. These authors proposed a linear formulation, exact methods based on Benders decomposition and branch-and-cut (B&C), a Genetic Algorithm (GA) and a hybrid heuristic (GA coupled to an extended Benders decomposition) for *min-max regret* WSCP. Computational experiments showed that B&C had the best performance among the exact methods, and the hybrid heuristic produced the best upper bounds among the heuristics. In [17], a scenario-based heuristic (SBH) and a Path-Relinking (PR) method were developed. Numerical experiments reported that PR found better solutions than SBH, and that both outperformed the heuristics from the literature.

The Probabilistic Set Covering Problem (PSCP), introduced in [8], is a generalization of WSCP. Let N, M and $\{a_{ij}\}$ be as previously defined. Let also $\lambda \in (0,1)$ be a constant and ξ be a $\{0,1\}$-random vector. PSCP consists in finding a subset $X \subseteq M$ with minimum cost, such that every line in N is covered by at least ξ columns in X with a probability of at least λ. This problem is NP-hard [8], and exact and heuristic algorithms were proposed in [8], [49] and [55].

The NP-Hard *Budget Uncertainty* Weighted Set Covering Problem (BU-WSCP) is a generalization of WSCP proposed in [29], that applies the robust optimization criterion introduced by [10]. Let $\{a_{ij}\}$, N and M be as defined above and λ be as described in PSCP. Let also $[l_j, u_j]$ be the cost interval associated with each column $j \in M$, S be the scenario-set, and a scenario $s \in S$ be an assignment of a cost $c_j^s \in [l_j, u_j]$ for every column $j \in M$. Given a scenario $s \in S$ with $k \leq |M|$ columns set in u_j and a subset of scenarios $v \in S$ that contains all scenarios in which k or less columns are fixed in u_j. The BU-WSCP aims at finding a subset $X \subseteq M$ with the minimum cost sum in s, such that every line in N is covered by at least one column in X with a probability of at least λ and X is feasible for all scenarios in v. A polyhedral study, integer linear formulations, and cutting-plane methods were presented in [29] and [40].

2.2 Studies related to *min-max regret* MSCP

The deterministic counterpart of the *min-max regret* MSCP is the Maximum Benefit Set Covering Problem (MSCP) [19]. MSCP is also defined in a matrix $\{a_{ij}\}$ with $|N|$ lines and $|M|$ columns, such that $a_{ij} = 1$ if the column $j \in M$ covers the line $i \in N$, and $a_{ij} = 0$ otherwise. Besides, each column $j \in M$ is associated with a benefit b_j and a weight w_j. MSCP consists in finding a column-subset $X \subseteq M$ with the maximum benefit, such that every line in N is covered by at least one column in X, and the sum of column weights in X is not larger than a capacity $T \in \mathbb{N}^*$. This problem is clearly NP-Hard, as finding any feasible solution for MSCP is as hard as checking the existence of a T-cover in $\{a_{ij}\}$ [32]. Analogously, one can also see that this problem is NP-Hard as for $a_{ij} = 1$, for all $i \in N$ and $j \in M$, MSCP reduces to the Knapsack problem with capacity T.

The Maximum Coverage Location Problem (MCLP) is also related to MSCP. It was introduced by [14] and is defined in a matrix $\{a_{ij}\}$ with a set N of lines and a set M of columns, such that $a_{ij} = 1$ if the column $j \in M$ covers the line $i \in N$, and $a_{ij} = 0$ otherwise. Given a constant C, MCLP consists in finding a column subset $X \subseteq M$, with $|X| \leq C$, that covers the maximum number of lines in N (not necessarily all of them). This problem is NP-hard [32] and some interesting entry points on exact and heuristics for MCLP are found in [20], [41] and [56]. MCLP has several practical applications, such as in nature reserve [15], e-commerce [34] and bike-sharing systems [44]. MSCP, on the other hand, aims at selecting a subset $X \subseteq M$, with $\sum_{j \in X} w_j \leq T$, such that every line in N is covered by at least one column in X. The objective function seeks the subset X^* with the maximum total benefit, which corresponds to the sum of the selected columns' benefit. The main difference between MCLP and MSCP is that the former aims at finding the solution that maximizes the number of lines covered, while the latter seeks the solution with the maximum benefit that covers all lines in N, and does not violate the capacity constraint.

The *Budget Uncertainty* Dynamic Maximum Coverage Problem (BUDMCLP) [42] is a generalization of MCLP. Let $\{a_{ij}\}$, N, M, and S be as formerly described. Let also H be a constant corresponding to a time horizon, such that $Q = \{0, \ldots, H\}$ is the set of time periods. Given a demand interval $[l_{iq}, u_{iq}]$ associated with each row $i \in N$ in a time period $q \in Q$, a scenario $s \in S$ is defined as an assignment of a demand $d_{iq}^s \in [l_{iq}, u_{iq}]$ for every row $i \in N$ and time period $q \in Q$, where S is the set of all possible scenarios. As MCLP, BUDMCLP computes a subset $X \subseteq M$, with $|X|$ not larger than a maximum cardinality C, that covers a subset $N(X) \subseteq N$ of lines. Given a constant uncertainty budget $k \leq |N|$, $S^k \subseteq S$ is defined as the subset of scenarios that have at most k rows whose demands are different than l_{iq}. A solution X is feasible only if the demands of the rows in $N(X)$ can be satisfied in all scenarios in S^k. The objective is to find the solution X that maximizes the sum of the demands of the rows in $N(X)$ overall H time periods. This problem was proven NP-Hard in [42]. The latter also developed a MILP formulation, and a hybrid metaheuristic combining a Variable Neighborhood Search with a Linear Programming method.

The *min-max regret* p-center [3] is a generalization of the well-known p-center problem [25]. Let $\{a_{ij}\}$, N, M be as formerly described, where N here denotes a set of customers and M a set of facilities. Given a demand interval $[l_i, u_i]$ associated with each row $i \in N$, a scenario $s \in S$ is defined as an assignment of a demand $d_i^s \in [l_i, u_i]$ for every row $i \in N$, where S is the set of all possible scenarios. The *min-max regret* p-center computes a subset $X \subseteq M$ of columns (facilities) not larger than a maximum cardinality p, such that each line $i \in N$ (customer) is served by exactly one column. A scenario $s \in S$ is an assignment of demands $d_i^s \in [l_i, u_i]$ for every customer $i \in N$, where S is the set of all possible scenarios. The objective is to find a solution with the smallest maximum regret over all scenarios in S. This problem is NP-hard [3], and the studies of [22,51] are entry points for facility location problems under uncertainties, covering RO, stochastic programming and chance constraints, together with the corresponding approaches.

3 Robust covering problems

The *min-max regret* WSCP and the *min-max regret* MSCP are formally defined in Sections 3.1 and 3.2, respectively. In both problems, $\{a_{ij}\}$ denotes a matrix with a line-set N and a column-set M, such that $a_{ij} = 1$ if the column $j \in M$ covers the line $i \in N$, and $a_{ij} = 0$ otherwise. Moreover, in the *min-max regret* WSCP, the uncertain cost (resp. the uncertain benefit in the *min-max regret* MSCP) of each column $j \in M$ is associated with an interval $[l_j, u_j]$. A scenario $s \in S$ is an assignment of costs $c_j^s \in [l_j, u_j]$ (resp. benefits $b_j^s \in [l_j, u_j]$ in the *min-max regret* MSCP) for every column $j \in M$, where S is the set of all possible values combinations for the columns' cost (resp. benefit in the *min-max regret* MSCP). Besides, we denote the set of all possible scenarios by S.

3.1 The *min-max regret* WSCP

The *min-max regret* WSCP computes a solution $X \subseteq M$, such that every line in N is covered by at least one column in X. Let $\mathit{\Gamma} \subseteq 2^M$ be the set of feasible solutions, and $\omega^s(X) = \sum_{j \in X} c_j^s$ be the cost of a solution $X \in \mathit{\Gamma}$ for the scenario $s \in S$, where c_j^s is the cost of column $j \in M$ in s. The *regret* of a solution $X \in \mathit{\Gamma}$ for a scenario $s \in S$ is defined as the difference $\omega^s(X) - \omega^s(Y^s)$, where Y^s is the optimal solution for the scenario s, i.e. the regret of using X instead of Y^s if scenario s occurs. The *min-max regret WSCP* consists in finding a solution $X^* \subseteq \mathit{\Gamma}$ with the smallest maximum regret over all scenarios, as shown in Equation (1).

$$X^* = \arg \min_{X \in \mathit{\Gamma}} \max_{s \in S} \left\{ \omega^s(X) - \omega^s(Y^s) \right\} \tag{1}$$

Despite there being infinitely many scenarios in S, given a solution $X \in \mathit{\Gamma}$, the scenario $s(X)$ where the regret of X is maximal can be computed in polynomial time for any *min-max regret* robust optimization problem whose classical counterpart is a $\{0,1\}$ minimization problem. In this case, $s(X)$ is the scenario where $c_j^{s(X)} = u_j$, for all $j \in X$, and $c_j^{s(X)} = l_j$, for all $j \in M \setminus X$, i.e. $s(X)$ is the scenario in which all columns in X have the largest possible cost and all the other ones have the smallest possible cost. An example is given in Figure 1. An instance of the *min-max regret* WSCP is shown in the Table 1(a), where a solution $X = \{1, 3\}$ is highlighted. The scenario $s(X)$ is displayed in the Table 1(b), where the optimal solution $Y^{s(X)} = \{2, 4\}$ is highlighted. In this case, the regret of X in $s(X)$ is $\omega^{s(X)}(X) - \omega^{s(X)}(Y^{s(X)}) = 3$, where $\omega^{s(X)}(X) = 8 + 4 = 12$ and $\omega^{s(X)}(Y^{s(X)}) = 3 + 6 = 9$. It is worth mentioning that X is the optimal solution for the instance shown in Table 1(a).

	1	2	3	4
1	1	1		
2	1			1
3	1			1
4		1	1	
5			1	1
6	1			1
7		1	1	
8			1	1
9		1	1	
$[l_j, u_j]$	[5,8]	[3,7]	[3,4]	[6,9]

(a)

	1	2	3	4
1	1	1		
2	1			1
3	1			1
4		1	1	
5			1	1
6	1			1
7		1	1	
8			1	1
9		1	1	
c_j^s	8	3	4	6

(b)

Fig. 1 An example of a *min-max regret* WSCP instance. The solutions X and $Y^{s(X)}$ are highlighted on tables (a) and (b), respectively.

3.1.1 Mathematical formulation

The authors in [47] proposed a MILP formulation for the *min-max regret* WSCP. Let N, M, $\{a_{ij}\}$ and $[l_j, u_j]$ be as previously defined. Let each solution $X \in \Gamma$ of the *min-max regret* WSCP be associated with a characteristic vector of dimension $|M|$, such that X is represented by a vector x, with $x_j = 1$ if $j \in X$, and $x_j = 0$ otherwise. Let Y and y be, respectively, a feasible solution in Γ and its M-dimensional characteristic vector, where $y_j = 1$ if $j \in Y$, and $y_j = 0$ otherwise. Additionally, let θ be a free variable. The MILP formulation is as follows.

$$\min_{X \in \Gamma} \quad \Big\{ \sum_{j \in M} u_j x_j - \theta \Big\} \tag{2}$$

$$s.t.$$

$$\sum_{j \in M} a_{ij} x_j \geq 1 \qquad \forall i \in N \tag{3}$$

$$\theta \leq \sum_{j \in M} l_j y_j + \sum_{j \in M} y_j \left(u_j - l_j\right) x_j \qquad \forall Y \in \Gamma \tag{4}$$

$$\theta \quad free \tag{5}$$

$$x \in \{0,1\}^{|M|} \tag{6}$$

The objective function (2) seeks to find the covering $X \in \Gamma$ with the smallest maximum regret. Inequalities (3) ensure that every line in N is covered by at least one column in M. Constraints (4) and (5) assure that $\theta = \omega^{s(X)}(Y^{s(X)})$. The domain of variables x is defined in (6). One may note that the size of constraint-set (4) grows exponentially with the cardinality of M.

3.2 The *min-max regret* MSCP

The *min-max regret* MSCP computes a solution $X \subseteq M$, such that every line in N is covered by at least one column in X, and the sum of the weights $w_j \in \mathbb{N}^*$ of the columns $j \in X$ is not greater than a capacity $T \in \mathbb{N}^*$. Let $\Delta \subseteq 2^M$ be the set of feasible solutions, i.e., if $X \in \Delta$ then $\exists j \in X : a_{ij} = 1$, for all $i \in N$, and $\sum_{j \in X} w_j \leq T$. Let also $\psi^s(X) = \sum_{j \in X} b_j^s$ be the benefit of a solution $X \in \Delta$ for the scenario $s \in S$, where b_j^s is the benefit of column $j \in M$ in s. The *regret* of a solution $X \in \Delta$ for a scenario $s \in S$ is defined as the difference $\psi^s(Y^s) - \psi^s(X)$, where Y^s is the optimal solution for the scenario s. The *min-max regret MSCP* consists in finding the solution with the smallest maximum regret over all scenarios, as shown in Equation (7). It is worth noticing that computing the regret of a solution in a single scenario for *min-max regret* MSCP (resp. *min-max regret* WSCP) is NP-Hard, as an instance of MSCP (resp. WSCP) must be solved in order to compute Y^s.

$$X^* = \arg \min_{X \in \Delta} \max_{s \in S} \Big\{ \psi^s(Y^s) - \psi^s(X) \Big\} \tag{7}$$

Theorem 1 *Min-max regret MSCP is Σ_p^2-Hard.*

Proof The proof consists in showing that the *min-max regret* MSCP is a generalization of the *min-max regret* Knapsack Problem (*min-max regret* KP), which was proven to be Σ_p^2-Hard in [24]. The *min-max regret* KP is a generalization of the well-known Knapsack Problem [21]. Let I be the set of items, the *min-max regret* KP computes a solution $X \subseteq I$, such that the sum of the weights w_j of the items $j \in X$ is not greater than a capacity T. Let $\Phi \subseteq 2^I$ be the set of feasible solutions, where 2^I denotes the power set of I, and $\psi^s(X) = \sum_{j \in X} b_j^s$ be the benefit of a solution $X \in \Phi$ for the scenario $s \in S$, where b_j^s is the benefit of item $j \in I$ in s. The *regret* of a solution $X \in \Phi$ for a scenario $s \in S$ is defined as the difference $\psi^s(Y^s) - \psi^s(X)$, where Y^s is the optimal solution for the scenario s. As no information is known about which scenario is happening, it is assumed that a solution is as good as its maximum regret over all the infinitely many scenarios in S. Therefore, the *min-max regret KP* consists in finding the solution with the smallest maximum regret over all scenarios, as shown in Equation (8).

$$X^* = \arg\min_{X \in \Phi} \max_{s \in S} \left\{ \psi^s(Y^s) - \psi^s(X) \right\} \tag{8}$$

As can be observed from equations (7) and (8), the objective function of both the *min-max regret* MSCP and the *min-max regret* KP are similar. Besides, for $M = I$, we have that $\Delta \subseteq \Phi \subseteq 2^M$, since all the subsets of columns in Φ respect the capacity constraint, but not necessarily the covering constraints. When $a_{ij} = 1$, for all $i \in N$ and $j \in M$, we have that $\Delta = \Phi$, as in this case, all subsets of columns in Φ also satisfy the covering constraints. Thus, any instance of *min-max regret* KP can be reduced to an instance of *min-max regret* MSCP, where $M = I$, $|N| = 1$ and $a_{1j} = 1$, for all $j \in M$. Therefore, the latter is at least as hard as the former. □

Although there are an infinite number of scenarios in S, given a solution $X \in \Delta$, the scenario $s(X)$ where the regret of X is the maximum can be computed in polynomial time for any min-max regret robust optimization problem whose classical counterpart is a $\{0, 1\}$ maximization problem. In this case, $s(X)$ is the scenario where $b_j^{s(X)} = l_j$, for all $j \in X$, and $b_j^{s(X)} = u_j$, for all $j \in M \setminus X$, i.e. $s(X)$ is the scenario in which all columns in X have the smallest possible benefit and all other columns have the largest possible benefit. The example of Figure 1 is also applied to the *min-max regret* MSCP, where an instance is shown in the Table 1(a), with the cardinality $T = 2$, $w_j = 1$ for all columns $j \in M$, and a solution $X = \{1, 3\}$ highlighted. Moreover, the scenario $s(X)$ is displayed in Table 1(b), where the optimal solution $Y^{s(X)} = \{2, 4\}$ is highlighted. In this case, the regret of X in $s(X)$ is $\psi^{s(X)}(Y^{s(X)}) - \psi^{s(X)}(X) = 3$, where $\psi^{s(X)}(Y^{s(X)}) = 8 + 4 = 12$ and $\psi^{s(X)}(X) = 3 + 6 = 9$. The solution X is optimal for the instance presented in Table 1(a).

3.2.1 Mathematical Formulation

The MILP formulation for the *min-max regret* MSCP, proposed in this work, is as follows. Given N, M, $\{a_{ij}\}$ and T as previously defined. We refer to b_j as the benefit of selecting column $j \in M$. When b_j is uncertain, we denote b_j^s as the benefit of selecting the column $j \in M$ in the scenario s. Each solution $X \in \Delta$ of MSCP and the *min-max regret* MSCP is associated with a characteristic vector of dimension $|M|$, such that X is represented by a vector x, with $x_j = 1$ if $j \in X$, and $x_j = 0$ otherwise.

The MSCP formulation is given by the objective function (9) and the constraints (10) to (12). The objective function (9) aims at finding the solution $X \in \Delta$ with the maximum total benefit. Inequalities (10) ensure that every line in N is covered by at least one column in M. Moreover, constraint (11) enforces that the capacity does not exceed T. The domain of variables x is defined in (12). It is worth mentioning that the set Δ of feasible solutions is formulated by constraints (10) to (12).

$$\max \quad \sum_{j\in M} b_j x_j \tag{9}$$

$$s.t.$$

$$\sum_{j\in M} a_{ij}x_j \geq 1 \qquad \forall i \in N \tag{10}$$

$$\sum_{j\in M} w_j x_j \leq T \tag{11}$$

$$x \in \{0,1\}^{|M|} \tag{12}$$

The *min-max regret* MSCP is a RO version of MSCP where the uncertainties are associated with the benefits. It can be formulated by the objective function (13) and the constraints (10) to (12), where $\psi^s(Y^s) = \max \sum_{j\in M} b_j^s y_j^s$ is the total benefit of the optimal solution Y^s of MSCP for the scenario s and y^s is the M-dimensional characteristic vector of solution Y^s, with $y_j^s = 1$ if $j \in Y^s$, and $y_j^s = 0$ otherwise.

$$\min_{x\in\Delta} \max_{s\in S} \quad \{\psi^s(Y^s) - \psi^s(X)\} \tag{13}$$

To provide a MILP formulation for this problem, we first rewrite the objective function (13) as (14), in order to explicitly compute the values of $\psi^s(Y^s)$ and $\psi^s(X)$.

$$\min_{x\in\Delta} \max_{s\in S} \quad \left\{ \max_{y\in\Delta} \left\{ \sum_{j\in M} b_j^s y_j^s \right\} - \sum_{j\in M} b_j^s x_j \right\} \tag{14}$$

Let $s(X)$, given in equation (15), be the scenario where the regret of X is the maximum. Additionally, let Y and y be, respectively, a feasible solution in Δ and its M-dimension characteristic vector, where $y_j = 1$ if $j \in Y$, and $y_j = 0$ otherwise. The *min-max regret* MSCP, $s(X)$ is set to the scenario where $b_j^{s(X)} = l_j$, when $x_j = 1$, and $b_j^{s(X)} = u_j$ otherwise. According to this result, the objective function (14) can be rewritten as (16), in such a way that only the worst-case scenario $s(X)$ is considered. In this case, the term (a) of (16) gives the total benefit of the optimal solution in scenario $s(X)$, while the term (b) gives the total benefit of X in $s(X)$.

$$s(X) = \arg\max_{s\in S} \quad \left\{ \max_{y\in\Delta}\{ \sum_{j\in M} b_j^s y_j^s \} - \sum_{j\in M} b_j^s x_j \right\} \tag{15}$$

$$\min_{x\in\Delta} \quad \left\{ \underbrace{\max_{y\in\Delta} \left\{ \sum_{j\in M} (u_j + (l_j - u_j)x_j) y_j \right\}}_{(a)} - \underbrace{\sum_{j\in M} l_j x_j}_{(b)} \right\} \tag{16}$$

Then, equation (16) is linearized following the approach proposed by [31], based on Danzig duality theory [23]. Term (a) is replaced by a free variable μ and the MILP formulation for the *min-max regret* MSCP is given by the objective function (17), constraints (18) and (19), which ensure that $\mu = \psi^{s(X)}(Y^{s(X)})$, and constraints in (10) to (12). It is important to highlight that the number of constraints (18) grows exponentially with $|M|$. This is expected, as Theorem 1 implies that there is no compact MILP formulation for the *min-max regret* MSCP unless $\Sigma_p^2 = NP$.

$$\min_{x \in \Delta} \quad \left\{ \mu - \sum_{j \in M} l_j x_j \right\} \tag{17}$$

$s.t.$

$$\mu \geq \sum_{j \in M} u_j y_j + \sum_{j \in M} y_j \left(l_j - u_j \right) x_j \qquad \forall Y \in \Delta \tag{18}$$

$$\mu \quad free \tag{19}$$

$$\sum_{j \in M} a_{ij} x_j \geq 1 \qquad \forall i \in N \tag{20}$$

$$\sum_{j \in M} w_j x_j \leq T \tag{21}$$

$$x \in \{0,1\}^{|M|} \tag{22}$$

4 Methods for the *min-max regret* WSCP and the *min-max regret* MSCP

This section describes the exact and heuristic algorithms proposed in this work for the *min-max regret* WSCP and the *min-max regret* MSCP. In Section 4.1, the exact algorithms introduced by [47] for *min-max regret* WSCP are reproduced and generalized to *min-max regret* MSCP. In the sequel, heuristic algorithms for *min-max regret* WSCP and *min-max regret* MSCP are proposed and described in Section 4.2.

4.1 Exact algorithms

The first exact algorithm proposed by [47] to the *min-max regret* WSCP, and adapted here for *min-max regret* MSCP, relies on a cutting plane algorithm introduced in [35], inspired by the Benders decomposition [6]. It is usually referred in the literature as Logic-based Benders decomposition algorithm (LBD). LBD master problem is given by a relaxation of the constraints that grow exponentially on the robust problem's formulation, and its sub-problem is the mathematical model of the classical counterpart. It is similar to the methods applied to solve other *min-max* regret problems [31,43].

The LBD method for the *min-max regret* WSCP is based on the mathematical model (2)-(6). As explained in the previous section, the number of constraints (4) increases exponentially with the number of columns. Thus, they are relaxed and replaced by (23) in the master problem as follows. Let $\Gamma^h \subseteq \Gamma$ be the set of solutions that induce the constraints (23). In the algorithm, at each iteration, a new constraint is separated from $\Gamma \setminus \Gamma^h$, by solving a WSCP sub-problem, and added to the master problem. LBD stops when the lower bound obtained by solving the master problem is equal to the upper bound or when a time limit is reached. The upper bound of LBD is computed as follows. The incumbent solution X^h regret of the master problem is given by the difference between its cost on the sub-problem scenario, and the optimal solution cost of this scenario. At the end of each iteration, if the regret is improved, the upper bound is updated. The pseudocode of this algorithm is found in [47].

$$\theta \leq \sum_{j \in M} u_j y_j + \sum_{j \in M} y_j \left(l_j - u_j \right) x_j \quad \forall Y \in \Gamma^h \tag{23}$$

The LBD algorithm to the *min-max regret* MSCP is based on the formulation (10)-(12) and (17)-(19). The number of constraints (18) increases exponentially with the number of columns. Thus, they are relaxed and replaced by (24) in the master problem as follows. Let $\Delta^h \subseteq \Delta$ be the set of solutions that induce the constraints (24). At each iteration of the algorithm, a new constraint is separated from $\Delta \setminus \Delta^h$, by solving a MSCP sub-problem, and added to the master problem. LBD stops when the lower bound obtained by solving the master problem is equal to the upper bound or when a time limit is reached.

$$\mu \geq \sum_{j\in M} u_j y_j + \sum_{j\in M} y_j \left(l_j - u_j\right) x_j \quad \forall Y \in \Delta^h \tag{24}$$

The pseudocode of LBD for the *min-max regret* MSCP is shown in Algorithm 1. Let $\rho(X) = \psi^{s(X)}(Y^{s(X)}) - \psi^{s(X)}(X)$ be the maximum regret of a solution $X \in \Delta$, where $Y^{s(X)}$ is the optimal solution of MSCP in the scenario $s(X)$. Δ^1 is initialized with the solutions X^m and X^u, returned by the Algorithm Mean Upper (AMU) heuristic [37] in order to avoid an unbounded master problem. AMU computes the maximum regret of the optimal solutions found on two different scenarios: (i) one where all uncertain parameters are fixed at their mean values and; (ii) the other in which these parameters are set to their upper values. The loop in lines 4 to 10 is performed until an optimal solution is found or when a time limit is reached. The master problem is run from Δ^h in line 5. Let (X^h, μ^h) be the optimal solution of this problem. We point out that X^h is feasible for the *min-max regret* MSCP, but the lower bound z obtained from the master problem may not be equal to $\rho(X^h)$ because the value of μ^h may not be equal to $\psi^{s(X)}(Y^{s(X)})$. Therefore, a MSCP sub-problem is run in line 6 in order to obtain the optimal solution $Y^{s(X^h)}$ for the scenario $s(X^h)$. This solution is added to Δ^{h+1} in line 7, which induces a new constraint (24) that cuts the solution (X^h, μ^h). The best known solution X^* is updated in line 8, and the iteration counter h is incremented in line 9. If $z = \rho(X^*)$ in line 10, the optimal solution X^* is returned in line 11.

```
   Input: $M, N, \{a_{ij}\}, T, [l_j, u_j]\ \forall j \in M$
   Output: $X^*$
 1 $h \leftarrow 1$
 2 $\{X^m, X^u\} \leftarrow$ AMU$(M, N, \{a_{ij}\}, T, [l_j, u_j]\ \forall j \in M)$
 3 $\Delta^h \leftarrow \{X^m, X^u\}$
 4 do
 5     $(X^h, \mu^h, z) \leftarrow$ MasterProblem$(M, N, \{a_{ij}\}, T, [l_j, u_j]\ \forall j \in M, \Delta^h)$
 6     $Y^{s(X^h)} \leftarrow$ MSCP$(M, N, \{a_{ij}\}, T, s(X^h))$
 7     $\Delta^{h+1} \leftarrow \Delta^h \cup \{Y^{s(X^h)}\}$
 8     $X^* \leftarrow \arg\min_{X\in\{X^h, X^*\}} \rho(X)$
 9     $h \leftarrow h + 1$
10 while $z < \rho(X^*)$ and the time limit is not reached;
11 return $X^*$
```

Algorithm 1: Pseudocode of LBD for *min-max regret* MSCP.

The convergence of LBD may be slow, since only one cut is produced after the run of a difficult master problem [47]. Thus, in order to speed up the convergence of LBD, the authors in [47] developed an extension of LBD called Extended Benders (EB). EB follows a method introduced by [30] where all incumbent solutions found by CPLEX are used to generate new cuts (not only the optimal one), while the master problem is solved. A sub-problem is solved for each incumbent solution, and the solutions returned by each sub-problem are added to the Master Problem. Therefore, the expected number of iterations in EB is smaller than the one of LBD.

The authors of [43] noticed that LBD may be computationally inefficient because at each iteration of this algorithm an ILOG CPLEX branch-and-bound algorithm is run from scratch to solve the MILP formulation of the master problem. Thus, they proposed an approach to the *min-max regret* traveling salesman problem where only one instance of a branch-and-cut (B&C) algorithm is performed. B&C is an optimization method where the optimal solution is sought by means of a branch-and-bound tree in which cutting planes are applied to tighten the linear programming relaxations of the tree [20,54]. Henceforth, the framework developed by [43] is going to be referred to as B&C.

B&C was extended to the *min-max regret* WSCP in [47] and to *min-max regret* MSCP in this work. For *min-max regret* WSCP, it is based on the linear relaxation of the mathematical model (2), (3), (5), (6) and (23) while the formulation (10)-(12), (17), (19) and (24) is the starting point of the B&C to *min-max regret* MSCP. As both methods are similar, only the B&C for *min-max regret* MSCP is explained as follows. This algorithm starts with the formulation given by the subset $\Delta' = \Delta^1$ of constraints (24). When an integer solution (X', μ') is found in a node of the enumeration tree, a new solution $Y^{s(X')}$ is computed by solving the MSCP sub-problem in the scenario $s(X')$. Then, $Y^{s(X')}$ is added to Δ' and a new global cut is propagated to all active nodes in the branch-and-bound tree. Therefore, one does not need to restart the branch-and-bound algorithm from $\Delta' \cup \{Y^{s(X')}\}$. This algorithm is correct since for each solution X' found, a new constraint (18) is generated to enforce the correct value of μ' [43].

4.2 Heuristics

In this section, five heuristics are proposed to *min-max regret* WSCP and *min-max regret* MSCP: two scenario-based algorithms [17,37], a path relinking [33], a pilot method [52] and a linear programming based heuristic [23] are detailed respectively in Sections 4.2.1, 4.2.2, 4.2.3 and 4.2.4. Except the one based in linear programming, they are guaranteed to be 2-approximate.

4.2.1 Scenario-based heuristics

Scenario-based heuristics for *min-max regret* combinatorial optimization problems consist in sampling a subset of scenarios and optimally solving the deterministic problem on each of these scenarios. Then, the maximum regret of each obtained solution is computed, and the one with the smallest maximum regret is returned.

The *Algorithm Mean* (AM) heuristic for *min-max regret* WSCP and *min-max regret* MSCP are instantiations of the framework proposed in [37]. Let the *mean scenario* s^m be the scenario where the cost or benefit of each column $j \in M$ is $b_j^m = (l_j + u_j)/2$. AM consists in solving the WSCP or MSCP in the scenario s^m, and computing the maximum regret, on scenario $s(X)$, of the obtained solution. The proof that this algorithm is 2-approximate for any *min-max regret* combinatorial optimization problem can be found in [37]. Two variations of this approach are also proposed in [37]. The *Algorithm Upper* (AU) consists in solving the WSCP or MSCP in the scenario s^u, where the cost or benefit of each column $j \in M$ is $b_j^u = u_j$ and computing the maximum regret, on scenario $s(X)$, of the returned solution. The *Algorithm Mean Upper* (AMU) simply returns the best solution obtained by AM and AU.

The Scenario Based Algorithm (SBA) heuristic is an instantiation of the framework proposed in [17] and successfully applied in [13]. SBA is a generalization of AMU, where a set Q of scenarios, instead of a single one, is investigated. The algorithm consists of solving one instance of the problem (*e.g.* WSCP or MSCP) for each scenario in Q, and returning the solution with the minimum maximal regret. Let s_p be the scenario where $b_j^{s_p} = \{l_j + (p \times (u_j - l_j))$ for each $j \in M\}$. Given $Q = \{s_p \mid p = \frac{i}{q}$ and $i = 0, 1, 2, 3, \cdots, q\}$, where the number of SBA iterations q was set to 100. Obviously, for an even value of q, the mean scenario is always investigated. Therefore, the solutions obtained by SBA are at least as good as those of AMU.

Since the SBA is similar for both covering problems focused here, only its pseudocode for *min-max regret* MSCP is given (see Algorithm 2). The difference for SBA applied to *min-max regret* WSCP is in line 4, where $X^p \leftarrow \text{WSCP}(M, N, \{a_{ij}\}, s_p)$ is used instead. At each of the q iterations of the for-loop in lines 1 to 6, a MSCP instance is solved in a specific scenario. The value of p and the scenario s_p are computed in lines 2 and 3, respectively. The optimal MSCP solution X^p for the scenario s_p is obtained in line 4, and the best known solution X' is updated in line 5. The best solution found by SBA is returned in line 6.

4.2.2 Path relinking

Path Relinking is a search heuristic that has been successfully applied to a number of optimization problems [33]. Given two solutions X^i and X^f, the path relinking main idea is to gradually transform X^i into X^f, by applying a set of different moves. This mechanism is

```
   Input: $q, M, N, \{a_{ij}\}, T, [l_j, u_j]\ \forall j \in M$
   Output: $X'$
1  for $i$ from 0 to $q$ do
2      $p \leftarrow i/q$
3      Let $s_p$ be the scenario where $b_j^{s_p} \leftarrow l_j + p \times (u_j - l_j)$, $\forall j \in M$
4      $X^p \leftarrow$ MSCP$(M, N, \{a_{ij}\}, T, s_p)$
5      $X' \leftarrow \arg\min_{X \in \{X^p, X'\}} \rho(X)$
6  end
7  return $X'$
```

Algorithm 2: Pseudocode of SBA heuristic.

motivated by the fact that different near-optimal solutions usually share good components. For both *min-max regret* WSCP, and *min-max regret* MSCP, this is translated by two solutions that share a common subset of columns, *i.e.* two solutions X^i and X^f may share a common subset of columns L, where $L = X^i \cap X^f$ and $L \neq \emptyset$. Thus, path relinking uses this information to create a sequence of intermediate solutions between X^i and X^f, in a hope that better solutions will be found.

The authors in [17] proposed a Path Relinking (PR) heuristic framework for *min-max regret* problems and applied it to solve the *min-max regret* WSCP. In this work, the PR proposed by [17] is extended to solve the *min-max regret* MSCP. The solutions that belong to the path between X^i and X^f are created by adding to X^i a column that is in X^f but not in X^i and, after that, by removing from X^i one redundant column that is not in X^f.

The pseudocode of PR for the *min-max regret* MSCP is displayed in Algorithm 3. Let P be a pool with all distinct solutions obtained by the SBA heuristic, and X^{sba} be the best solution found by SBA. The best known solution X' is initialized with X^{sba} in line 1. The loop in lines 2-16 is performed for each solution $P_i \in P$. X^i is initialized with P_i in line 3. The loop in lines 4-15 is repeated for each solution in the pool, except X^i. X^f and the column-set Z containing all columns $j \in X^f$ and $j \notin X^i$ are initialized in lines 5 and 6, respectively. The loop in lines 7-14 is performed for each column in Z by the increasing order of index. First, a column $l \in Z$ is added to X^i, in line 8. Then, the first redundant column $l \in X^i$ that is not in X^f is removed from X^i in line 9. Finally, the best known solution X' is updated in line 10 and, if X' is not in P, then it is added to the pool in line 12. The best solution found by PR is returned in line 17.

```
    Input: $P, X^{sba}, M, N, \{a_{ij}\}, T, [l_j, u_j]\ \forall j \in M$
    Output: $X'$
1   $X' \leftarrow X^{sba}$
2   for $i$ from 0 to $|P|$ do
3       $X^i \leftarrow P_i$
4       for $k$ from $i+1$ to $|P|$ do
5           $X^f \leftarrow P_k$
6           $Z \leftarrow$ Compare$(X^i, X^f)$
7           for $l$ from 0 to $|Z|$ do
8               AddColumn$(X^i, Z[l])$
9               $X^i \leftarrow$ RemoveColumn$(l \in X^i$ and $l \notin X^f)$
10              $X' \leftarrow \arg\min_{X \in \{X^i, X'\}} \rho(X)$
11              if $X' \notin P$ then
12                  $P \leftarrow P \cup \{X'\}$
13              end
14          end
15      end
16  end
17  return $X'$
```

Algorithm 3: PR pseudocode.

4.2.3 Pilot Method

Pilot Method (PM) [26] is a metaheuristic that uses a greedy constructive guiding heuristic H to build a new and more efficient heuristic H' and works as follows. Given a constructive heuristic, it will iteratively insert one element at a time in a partial solution. However, instead of using a local greedy criterion to evaluate the cost of inserting an element in the solution, the criterion used by H' consists in (i) inserting the element individually in the solution (ii) performing the heuristic H until a feasible solution is found, and (iii) using the cost of this solution as the greedy cost of inserting the element. At each iteration, these three steps are performed for all candidate elements and the one with the best greedy cost is inserted on the solution.

A survey on PM heuristics is found in [52]. PM was successfully used to produce upper bounds for NP-hard combinatorial optimization problems, such as the traveling salesman problem [26] and the Steiner tree problem [26]. The authors in [16] proposed a PM framework for *min-max regret* problems and applied it to solve the *min-max regret* shortest path problems. In this work, the PM proposed by [16] is adapted to solve both *min-max regret* WSCP and *min-max regret* MSCP. The heuristic AM [37] works as a guiding heuristic in both algorithms.

The pseudocode of PM for *min-max regret* MSCP is presented in Algorithm 4. The algorithm inputs are: $M, N, \{a_{ij}\}, T, [l_j, u_j]\ \forall j \in M$, defined in Section 3. The partial (or guiding) solution X' and the best known feasible solution X^{PM} are initialized at line 1. The loop on lines 2-12 is performed while X' is not feasible, *i.e.*, while all lines are not covered and the capacity is less than T. Let $i \in N$ be a line and let $\delta^+(i)$ be the set of columns in M that cover line i. The uncovered line i with the highest value of $|\delta^+(i)|$ is identified at line 3, where $\bar{N}(X')$ denotes the set of lines not covered by X'. The loop on lines 4-9 is performed for each column $j \in \delta^+(i)$. The MSCP formulation using the mean scenario $s^m \in S$ is run at line 5 and returns a feasible solution X'_j which contains all columns in $X' \cup \{j\}$. Next, $\rho(X'_j)$ is used as the greedy cost of inserting a column j in the solution X'. Then, if $\rho(X'_j)$ is smaller than the current iteration's best greedy cost $\rho(X'_{j^*})$ or else if the latter is not set yet (line 7), the column $j^* \in \delta^+(i)$ with the smallest maximum regret and its respective covering X'_{j^*} are updated in line 8. Afterwards, j^* is inserted at the end of X' at line 10. PM returns the best solution X^{PM} found throughout the heuristic, which is not necessarily X'. Therefore, the former is updated at line 11, and returned at line 13. One may observe that PM can be straightforwardly extended to *min-max regret* WSCP, by running WSCP formulation in line 5.

```
   Input: $M, N, \{a_{ij}\}, T, [l_j, u_j]\ \forall j \in M$
   Output: $X^{PM}$
 1 $X' \leftarrow \emptyset$ and $X^{PM} \leftarrow \emptyset$
 2 while $X'$ is not a feasible solution do
 3     Let $i = argmax_{i' \in \bar{N}(X')}\ \delta^+(i')$
 4     for $j \in \delta^+(i)$ do
 5         $X'_j \leftarrow$ MSCPFormulation $(X' \cup \{j\},\ s^m)$
 6         if $\rho(X'_j) < \rho(X'_{j^*})$ or $X'_{j^*} = \emptyset$ then
 7             $j^* \leftarrow j$ and $X'_{j^*} \leftarrow X'_j$
 8         end
 9     end
10     Insert $j^*$ in $X'$
11     $X^{PM} \leftarrow \arg\min_{X \in \{X'_{j^*}, X^{PM}\}} \rho(X)$
12 end
13 return $X^{PM}$;
```

Algorithm 4: Pseudocode of PM for *min-max regret* MSCP.

4.2.4 Linear Programming Heuristic

The Linear Programming Heuristic (LPH) was introduced in [2]. In this work, LPH is reproduced for the *min-max regret* WSCP and developed for the *min-max regret* MSCP. As far as we know, there is no modeling approach in the literature that provides compact formulations for *min-max regret* optimization problems, whose deterministic counterpart is NP-hard. This is why LPH makes use of an alternative compact formulation, which returns an upper bound to the maximum regret of a solution.

The authors in [2] developed a LPH heuristic for the *min-max regret* WSCP. Let N, M, $\{a_{ij}\}$, $[l_j, u_j]$, and vector x be as previously defined. Additionally, let ν_i be the variables obtained on the dual formulation of WSCP. The MILP formulation for the LPH heuristic is as follows.

$$\min \quad \sum_{j \in M} u_j x_j - \sum_{i \in N} \nu_i \tag{25}$$

$$s.t.$$

$$\sum_{j \in M} a_{ij} x_j \geq 1 \qquad \forall i \in N \tag{26}$$

$$\sum_{i \in N} a_{ij} \nu_i \leq l_j + (u_j - l_j) x_j \qquad \forall j \in M \tag{27}$$

$$\nu_i \geq 0 \qquad \forall i \in N \tag{28}$$

$$x \in \{0,1\}^{|M|} \tag{29}$$

The objective function (25) aims at finding a covering $X \in \Gamma$ with the smallest maximum regret. Inequalities (26) ensure that every line in N is covered by at least one column in M. Constraints (27) assure that $\sum_{i \in N} a_{ij} \nu_i$ is a lower bound to $\omega^{s(X)}(Y^{s(X)})$. The domain of variables ν and w are defined, respectively, in equations (28) and (29). This formulation is compact, as the number of constraints (27) grows polynomially with the cardinality of M. The LPH heuristic for the *min-max regret* WSCP consists in solving and returning the best solution of this formulation.

LPH for the *min-max regret MSCP* works as follows. First, the non-linear formulation (10)-(12), and (16) is rewritten below as (30)-(33). The MILP formulation of the sub-problem that computes the optimal solution in the scenario $s(X)$ is highlighted in the term (c). It can be seen that this sub-problem is equivalent to a MSCP, where the cost of a column $j \in M$ is equal to $u_j + (l_j - u_j)x_j$.

$$\min \quad \left\{ \underbrace{\max_{y \in \Delta} \quad \left\{ \sum_{j \in M} (u_j + (l_j - u_j)x_j) y_j \right\}}_{(c)} - \underbrace{\sum_{j \in M} l_j x_j}_{(d)} \right\} \tag{30}$$

$$s.t.$$

$$\sum_{j \in M} a_{ij} x_j \geq 1 \qquad \forall i \in N \tag{31}$$

$$\sum_{j \in M} w_j x_j \leq T \tag{32}$$

$$x \in \{0,1\}^{|M|} \tag{33}$$

Then, the linear relaxation of the sub-problem (c) is expanded in (34)-(37).

$$\max \quad \sum_{j \in M} (l_j + (u_j - l_j)x_j)y_j \tag{34}$$

$s.t.$

$$\sum_{j \in M} a_{ij} y_j \geq 1 \qquad \forall i \in N \tag{35}$$

$$\sum_{j \in M} w_j y_j \leq T \tag{36}$$

$$y_j \in [0, 1] \qquad \forall j \in M \tag{37}$$

Inspired on Dantzig duality theory [23], the authors in [2] proved that an upper bound to the non-linear formulation of any *min-max regret* problem can be obtained by replacing the sub-problem (c) by the dual of its linear relaxation. In the case of the *min-max regret* MSCP, the dual of formulation (34)-(37) is shown in (38)-(41).

$$\min \quad T\xi - \sum_{i \in N} \nu_i \tag{38}$$

$s.t.$

$$w_j \xi - \sum_{i \in N} a_{ij} \nu_i \geq u_j + (l_j - u_j)x_j \qquad \forall j \in M \tag{39}$$

$$\nu_i \geq 0 \qquad \forall i \in N \tag{40}$$

$$\xi \geq 0 \tag{41}$$

Finally, the MILP formulation for *min-max regret* MSCP is obtained by replacing the sub-problem (c) in (30) by the dual relaxation (38)-(41). The resulting formulation is given by (31)-(33), (39)-(41) and (42). This formulation is compact, as the number of constraints (39) grows polynomially with the cardinality of M. The LPH heuristic for the *min-max regret* MSCP consists in solving and returning the best solution for this formulation.

$$\min \quad (T\xi - \sum_{i \in N} \nu_i) - \sum_{j \in M} l_j x_j \tag{42}$$

5 Computational experiments

Computational experiments were carried out on an Intel Core i7-4790K with 4.00 GHz clock and 16 GB of RAM, running Ubuntu Linux operating system version 16.04 LTS. Algorithms LBD, EB, B&C, AMU, SBA, PR, LPH and PM were implemented in C++ and compiled with GNU g++ version 4.8.2. The master and the sub-problems of LBD, EB and B&C were solved using IBM/ILOG CPLEX version 12.6.2 with default parameter settings.

A total of 90 theoretical instances were generated for the *min-max regret* WSCP as follows. As in [47], the classical instances BKZ-4, BKZ-5 and BKZ-6 [5] of the set covering problem were used and extended. Furthermore, an instance-set called BKZ-7 was introduced in this work. Table 1 displays the characteristics regarding each instance set. Columns 1 to 5 report the name, the size ($|G|$), the number of lines ($|N|$), the number of columns ($|M|$), and the density of matrix $\{a_{ij}\}$ (δ) of each instance set, respectively. Three different interval data were generated for each instance, as suggested by [36]. For each column $j \in M$, the values of l_j and u_j are chosen, respectively, within the intervals $U[0, \lambda]$ and $U[l_j, l_j + \lambda]$, where $U[a, b]$ denotes a random number uniformly chosen in the range $[a, b]$, and the interval length λ is set to 1000.

The theoretical instances for *min-max regret* MSCP were created based on the set η of 90 instances for *min-max regret* WSCP described above. For each instance $\alpha \in \eta$, three new

Instance	$\|G\|$	$\|N\|$	$\|M\|$	δ
BKZ-4	10	1000	200	2%
BKZ-5	10	2000	200	2%
BKZ-6	5	1000	200	5%
BKZ-7	5	2000	200	5%

Table 1 Characteristics of instance sets BKZ-4, BKZ-5, BKZ-6 and BKZ-7.

instances were generated using the same interval data and the same matrix of α, varying only the parameter T in $T = 0.1 \times |M|$, $T = 0.2 \times |M|$ and $T = 0.3 \times |M|$, and $w_j = 1$ for all $j \in M$. Therefore, a total of 270 instances were used in the experiments for *min-max regret* MSCP.

5.1 Numerical results for min-max regret *WSCP*

The first experiment evaluates the performance of the exact algorithms LBD, EB and B&C [47]. The running times were limited to 900 seconds. The results are reported in Table 2, and values in bold stand for the best ones among all methods, as for Table 3. The first and the second columns present, respectively, the name and the size ($|G|$) of each instance set. The third column reports the number of optimal solutions ($|O|$) out of $|G|$ found by LBD in each instance. The average relative optimality Gap ($\frac{\rho(X^{\text{LBD}})-z^{\text{LBD}}}{\rho(X^{\text{LBD}})}$)(%) between the regret of the solution provided by LBD and the lower bound z^{LBD} is reported in the fourth column, where the latter is obtained by solving at optimality the master problem in the last iteration of LBD. We also show the average relative optimality Gap* ($\frac{\rho(X^{\text{LBD}})-z^*}{\rho(X^{\text{LBD}})}$)(%) in the fifth column, where z^* is the best lower bound obtained by any of the exact algorithms LBD, EB and B&C. The sixth column shows LBD's average running time, while the seventh column displays the average number of iterations, which is equal to the number of cuts added to the master problem. Similar data is reported for EB and B&C in columns 8 to 12 and 13 to 17, respectively. It is worth mentioning that, in B&C, the lower bound $z^{B\&C}$ is obtained by the linear relaxation of formulation (2), (3), (5), (6) and (23) with Γ^h containing one cut for each integer solution found by B&C.

B&C is the algorithm that best performed among the exact methods: it found the optimal solution for 22 out of 90 instances, while EB and BD proved optimality for 16 and 4 instances, respectively, out of 90. These results are similar to the ones shown by [47]. Moreover, it is worth noting that the relative gaps of EB (6.32%) and B&C (6.37%) were almost the same. However, the relative Gap* of B&C (5.56%) was smaller than that of EB (6.08%). This shows that the lower bounds of B&C are worse than those of EB. This probably happens because of the excessive number of constraints (23) generated during the B&C that do not improve its lower bound. It is also noticeable that *min-max regret* WSCP becomes easier to solve as the matrix density increases. This is illustrated by the number of optimal solutions, i.e. 21 out of 30 instances, found using sets BKZ-6 and BKZ-7, which is higher than optimal solutions found using sets BKZ-4 and BKZ-5, i.e. 1 out of 60 instances.

The second experiment assesses the performance of the proposed heuristics (AMU, SBA, PR, LPH and PM) for *min-max regret* WSCP. The results are reported in Table 3. The first column displays the name of each instance set. Columns 2 to 6 show the average relative optimality Gap* ($\frac{\rho(X^{\text{AMU}})-z^*}{\rho(X^{\text{AMU}})}$)(%) of AMU, SBA, PR, LPH and PM, respectively, where z^* is the best lower bound obtained by any of the exact algorithms LBD, EB and B&C. We also report, in columns 7 to 11, the average percentage deviation $\frac{\rho(X^{best})-\rho(X^{\text{AMU}})}{\rho(X^{best})}$(%) of the solutions provided by AMU, SBA, PR, LPH and PM relative to the best solution found by any of the three exact algorithms. Columns 12 to 16 present the respective average running times of AMU, SBA, PR, LPH and PM for each instance set. A negative percentage deviation means that the heuristic found a better feasible solution than 900-second runs of LBD, EB and B&C. Results indicate that LPH is the heuristic that best performed on average, while AMU returned the best average running times. It can also be observed that SBA found solutions as good as those of PR-R-Any and better than the ones returned by PM consuming much less computational time. The performance of SBA shows that an extensive scenario search does not necessarily result in better solutions for *min-max regret* WSCP, since SBA deals with much less scenarios than PR-R-Any and PM.

Figure 2 displays the convergence of AMU, SBA, PR, LPH and PM for *min-max regret* WSCP over the time, in seconds, for the representative instance scp43-2-1000, belonging to the set BKZ-4a. The time is truncated after 140 seconds because the slowest heuristic LPH, stops after almost 136 seconds. LPH took nearly 90 seconds to find its best solution for this instance, despite that it took only 25 seconds, approximately, to find a solution that is better than the ones found by the other heuristics. Finally, it can be noticed that AMU is the fastest heuristic.

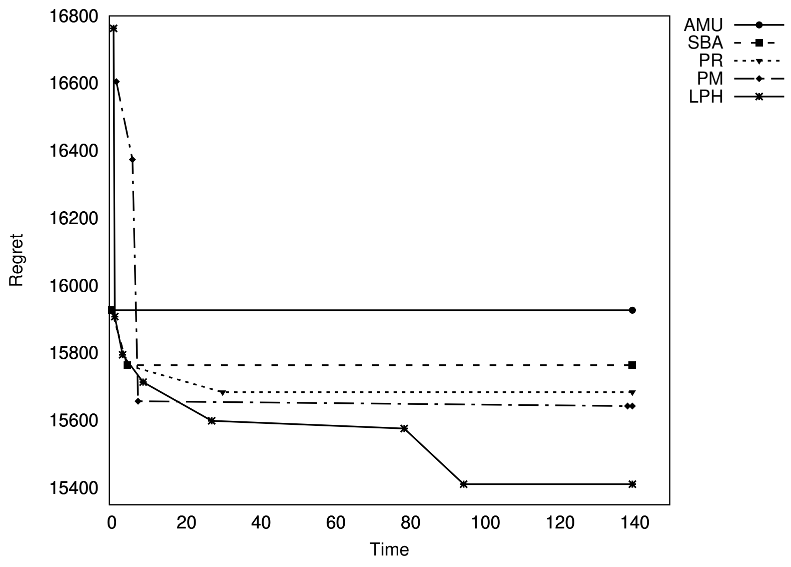

Fig. 2 Convergence of AMU, SBA, PR, LPH and PM for *min-max regret* WSCP to the instance scp43-2-1000.

5.2 Numerical results for *min-max regret* MSCP

The third experiment evaluates the performance of the exact algorithms LBD, EB and B&C using the set of 270 theoretical instances proposed for *min-max regret* MSCP. The running times were limited to 3600 seconds. The results are reported in Table 4. The first column displays the value of T, while the second one presents the name of each instance set. Bold in Tables 4 and 5 stands for the best results obtained by the methods. The average relative optimality Gap $(\frac{\rho(X^{\text{LBD}})-z^{\text{LBD}}}{\rho(X^{\text{LBD}})})(\%)$ between the regret of the solution provided by LBD and the lower bound z^{LBD} is reported in the third column, where the latter is obtained by solving at optimality the master problem in the last iteration of LBD. We also show the average relative optimality gap* $(\frac{\rho(X^{\text{LBD}})-z^{*}}{\rho(X^{\text{LBD}})})(\%)$ in the fourth column, where z^* is the best lower bound obtained by any of the exact algorithms LBD, EB and B&C. The fifth column depicts the average running time of LBD for each set, while the sixth column displays the number of cuts added to the master problem. Similar data is given for EB in columns 7 to 10 and for B&C in columns 11 to 14. For sake of clarity, in B&C, the lower bound $z^{B\&C}$ is obtained by the linear relaxation of formulation (10)-(12), (17), (19) and (24) with Δ^h containing one cut for each integer solution found by B&C.

In Table 4, the gaps of the three algorithms were on average about 38% for the sets with $T = 0.1 \times |M|$, 35% for the sets with $T = 0.2 \times |M|$ and 34% for the sets with $T = 0.3 \times |M|$. Unlike other problems in the literature that used such algorithms as in [31,43,47], LBD, EB and B&C found high gaps for the *min-max regret* MSCP. We have noticed in our numerical experiments that most cuts did not significantly improve the formulation's linear relaxation

		LBD					EB					B&C				
Instance	$\|G\|$	$\|O\|$	Gap	Gap*	Time (s)	Cuts	$\|O\|$	Gap	Gap*	Time (s)	Cuts	$\|O\|$	Gap	Gap*	Time (s)	Cuts
BKZ-4-a	10	0	17.09	12.39	900.00	23.40	0	12.26	12.26	900.00	144.50	0	**12.00**	**10.24**	900.00	809.70
BKZ-4-b	10	0	19.50	15.79	900.00	15.20	0	**15.79**	15.79	900.00	95.20	0	16.66	**14.84**	900.00	873.10
BKZ-4-c	10	0	17.82	13.19	900.00	18.90	0	**13.08**	13.08	900.00	115.60	0	14.02	**12.12**	900.00	810.70
BKZ-5-a	10	0	11.61	6.85	900.00	19.10	0	**6.93**	6.85	900.00	125.50	0	7.73	**6.51**	900.00	401.90
BKZ-5-b	10	0	12.58	8.18	900.00	15.00	0	**8.27**	8.18	900.00	94.40	0	9.11	**7.87**	900.00	336.80
BKZ-5-c	10	0	13.19	8.57	900.00	18.10	0	**8.39**	8.37	900.00	127.50	**1**	8.48	**7.57**	828.37	429.30
BKZ-6-a	5	0	7.66	1.72	900.00	17.00	3	2.09	1.25	759.58	100.60	**4**	**0.74**	**0.74**	484.61	93.80
BKZ-6-b	5	0	10.51	4.66	900.00	14.00	1	5.09	4.17	899.86	88.80	**2**	**4.55**	**3.98**	591.66	117.00
BKZ-6-c	5	0	9.85	3.64	900.00	15.20	0	3.58	2.97	900.00	115.80	**1**	**3.04**	**2.85**	806.78	140.00
BKZ-7-a	5	2	0.91	0.11	665.93	23.80	**5**	**0.00**	**0.00**	238.46	75.00	**5**	**0.00**	**0.00**	118.96	37.40
BKZ-7-b	5	0	4.28	0.21	900.00	15.40	4	0.13	0.01	640.77	72.20	**5**	**0.00**	**0.00**	439.59	52.20
BKZ-7-c	5	2	2.97	0.44	892.25	15.40	3	0.23	**0.04**	676.19	62.80	**4**	**0.13**	**0.04**	288.03	52.20
Average			10.66	6.31	879.85	17.54		**6.32**	6.08	792.91	101.49		6.37	**5.56**	671.50	346.18

Table 2 Comparison among the exact algorithms proposed by [47] to the set BKZ for *min-max regret* WSCP.

	Gap*					Dev (%)					Time (s)				
Instance	AMU	SBA	PR	LPH	PM	AMU	SBA	PR	LPH	PM	AMU	SBA	PR	LPH	PM
BKZ-4-a	12.39	11.23	11.00	**10.05**	10.97	2.49	1.12	0.86	**-0.21**	0.82	2.48	12.59	85.97	118.48	107.05
BKZ-4-b	15.79	14.97	14.74	**14.38**	14.92	1.15	0.15	-0.12	**-0.55**	0.10	4.18	23.72	64.34	270.29	183.14
BKZ-4-c	13.19	12.12	11.98	**11.56**	12.24	1.24	0.01	-0.15	**-0.63**	0.14	2.43	12.66	74.54	137.46	108.54
BKZ-5-a	6.85	6.63	6.61	**6.44**	**6.95**	0.37	0.13	0.12	**-0.07**	0.49	2.21	10.44	59.29	44.82	157.47
BKZ-5-b	8.18	7.91	7.91	**7.68**	7.97	0.33	0.04	0.04	**-0.21**	0.10	2.85	15.56	67.65	73.96	206.12
BKZ-5-c	8.57	7.81	7.74	**7.36**	7.73	1.09	0.25	0.18	**-0.23**	0.16	3.08	15.76	96.58	140.24	277.15
BKZ-6-a	1.72	1.62	1.42	**0.87**	1.82	1.00	0.90	0.81	**0.14**	1.12	11.87	48.99	186.54	123.69	633.95
BKZ-6-b	4.66	4.66	4.66	**3.93**	4.73	0.71	0.71	0.71	**-0.05**	0.80	19.40	49.96	102.64	383.51	742.57
BKZ-6-c	4.01	3.15	3.15	2.92	**2.89**	1.23	0.31	0.31	0.07	**0.04**	17.12	56.79	298.78	273.73	781.95
BKZ-7-a	0.34	0.20	0.20	**0.06**	**0.06**	0.34	0.20	0.20	**0.06**	**0.06**	6.05	22.26	51.82	36.27	547.43
BKZ-7-b	0.21	**0.04**	**0.04**	**0.04**	0.98	0.21	**0.04**	**0.04**	**0.04**	0.98	15.16	89.74	164.14	98.15	884.91
BKZ-7-c	0.45	0.14	0.14	**0.06**	1.14	0.42	0.10	0.10	**0.03**	1.12	18.87	106.46	196.95	173.91	867.50
Average	6.36	5.87	5.80	**5.45**	6.03	0.88	0.33	0.26	**-0.13**	0.49	8.81	38.74	120.77	156.21	458.15

Table 3 Comparison among the proposed heuristics to the set BKZ for *min-max regret* WSCP.

bound. That was specially the case for B&C, which obtained worse lower bounds than LBD and EB, since its gap was 40.46% while its gap* was 37.83%. This opens several avenues of research in terms of development of new cuts for this problem. In general, B&C performed slightly better, on average, for $T = 0.1 \times |M|$ while LBD was the best exact algorithm, on average, for $T = 0.2 \times |M|$ and $T = 0.3 \times |M|$.

The fourth experiment evaluates the proposed heuristics (AMU, SBA, PR, LPH and PM) for *min-max regret* MSCP. The results are reported in Table 5. The first column displays the values of capacity T, while the second one presents the name of each instance set. Columns 3 to 7 report the average relative optimality Gap* ($\frac{\rho(X^{\text{AMU}})-z^*}{\rho(X^{\text{AMU}})}$)(%) of AMU, SBA, PR, LPH and PM, respectively, where z^* is the best lower bound obtained by any of the exact algorithms LBD, EB and B&C. We also present, in columns 8 to 12, the average percentage deviation $\frac{\rho(X^{best})-\rho(X^{\text{AMU}})}{\rho(X^{best})}$(%) of the solutions provided by AMU, SBA, PR, LPH and PM relative to the best solution found by any of the three exact algorithms. Columns 13 to 17 show the respective average running times of AMU, SBA, PR, LPH and PM for each instance set. The best average solutions found for each instance set are highlighted, and a negative percentage deviation means that the heuristic found a better feasible solution than 3600-second runs of LBD, EB and B&C. Results indicate that PR performed better on average for $T = 0.1 \times |M|$ and AMU, SBA, PR found similar solutions, on average, for instances with $T = 0.2 \times |M|$ and $T = 0.3 \times |M|$ while, AMU returned the best average results in terms of running times. PR found the best deviations on average, although running times are higher than those for SBA and AMU. However, AMU and SBA are a good compromise between quality and running times. These results indicate that focusing on representative scenarios is relevant since AMU and SBA explore fewer scenarios, but important ones, and produce good results in a smaller running time. Moreover, it can be observed that LPH, which is the best known heuristic for *min-max regret* WSCP [2] and *min-max regret* Restricted Shortest Path problem [2], had the worst average results for *min-max regret* MSCP. This is due to the quality of the linear relaxation of mathematical formulation (31)-(33), (39)-(41) and (42).

Figure 3 depicts the convergence of the best solution found by AMU, SBA, PR-R-Best, LPH and PM for *min-max regret* MSCP over the time, in seconds, to the representative instance scp41-2-1000 with $T = 0.1 \times |M|$. It is worth mentioning that this instance belongs to the set BKZ-4. The time limit is 3600 seconds, due to the running time limit for all algorithms. Moreover, it is in logarithmic scale due to the huge timeline. It can be seen that PR-I-Any consumed nearly 90 seconds to find the best solution for this instance, while LPH found the worst best solution among all algorithms after almost 3600 seconds. The fastest heuristic is AMU.

The methods' results clearly show that the min-max regret MSCP is more complicated to solve than the min-max regret WSCP. As proven in Section 3.2, the min-max regret MSCP is Σ_p^2-Hard. Moreover, it is a strong indication that the formulation produces bad quality lower bounds.

6 Conclusions and future works

The *min-max regret* WSCP and the *min-max regret* MSCP are addressed in this study. Despite both problems aim at minimizing the maximum regret, the proposed exact methods have a different performance. In particular, the exact algorithms which work well for the former do not produce good results for the latter. A possible explanation is that the *min-max regret* MSCP belongs to another complexity class in the polynomial hierarchy.

The heuristics proposed to the *min-max regret* WSCP in this work returned better solutions, on average, than the AMU heuristic of [47]. Moreover, the heuristics developed in this work have found better upper bounds than a 900-second run of exact algorithms in several instances. Finally, it is worth mentioning the heuristics LPH found, on average, the best solutions for the *min-max regret* WSCP among the heuristics, and SBA which found solutions as good as the exact algorithms in a smaller amount of time.

The MSCP and the *min-max regret* MSCP were motivated by an application in disaster relief logistics, where field hospitals must be placed after large-scale emergencies such

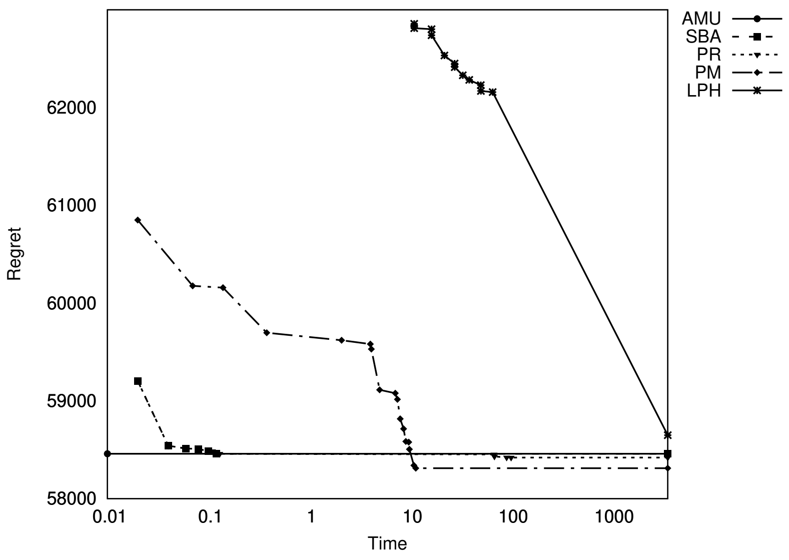

Fig. 3 Convergence of AMU, SBA, PR, LPH and PM to *min-max regret* MSCP to the instance scp41-2-1000 with T = $0.1 \times M$.

		LBD				EB				B&C			
T	Instance	Gap	Gap*	Time (s)	Cuts	Gap	Gap*	Time (s)	Cuts	Gap	Gap*	Time (s)	Cuts
$0.1 \times \lvert M\rvert$	BKZ-4	39.35	**38.50**	3600.00	8.00	39.26	**38.50**	3600.00	137.57	**38.54**	**38.50**	3600.00	4454.37
	BKZ-5	41.65	**40.69**	3600.00	7.30	41.41	**40.69**	3600.00	142.17	**40.70**	**40.69**	3600.00	5137.77
	BKZ-6	**39.21**	**39.18**	3600.00	7.20	39.39	**39.18**	3600.00	140.93	40.20	**39.18**	3600.00	5233.67
	BKZ-7	**42.17**	**42.12**	3600.00	7.93	42.55	**42.12**	3600.00	141.78	45.30	43.68	3600.00	4014.67
$0.2 \times \lvert M\rvert$	BKZ-4	**35.68**	**35.67**	3600.00	7.13	36.01	**35.67**	3600.00	116.27	39.21	**35.67**	3600.00	1494.30
	BKZ-5	**39.14**	**39.13**	3600.00	7.20	39.51	**39.13**	3600.00	101.57	42.39	**39.13**	3600.00	3047.10
	BKZ-6	**35.41**	**35.38**	3600.00	6.53	35.78	**35.38**	3600.00	98.73	39.00	**35.38**	3600.00	3022.53
	BKZ-7	**38.92**	**38.92**	3600.00	8.20	39.07	**38.92**	3600.00	129.23	42.85	**38.92**	3600.00	2405.40
$0.3 \times \lvert M\rvert$	BKZ-4	**34.96**	**34.94**	3600.00	6.70	35.33	**34.94**	3600.00	102.30	37.65	**34.94**	3600.00	765.97
	BKZ-5	**36.65**	**36.65**	3600.00	7.50	37.20	**36.65**	3600.00	115.77	40.52	**36.65**	3600.00	2556.40
	BKZ-6	**34.77**	**34.76**	3600.00	6.60	35.19	**34.76**	3600.00	102.40	37.76	**34.76**	3600.00	1582.27
	BKZ-7	**36.45**	**36.45**	3600.00	8.13	36.62	**36.45**	3600.00	119.67	41.40	**36.45**	3600.00	1938.53
Average		**37.86**	**37.70**	3600.00	7.37	38.11	**37.70**	3600.00	120.70	40.46	37.83	3600.00	2971.08

Table 4 Comparison among the exact algorithms to the set BKZ for the *min-max regret* MSCP.

		Gap*					Dev (%)					Time (s)				
T	Instance	AMU	SBA	PR	LPH	PM	AMU	SBA	PR	LPH	PM	AMU	SBA	PR	LPH	PM
$0.1 \times \lvert M\rvert$	BKZ-4	38.50	38.50	**38.42**	38.53	39.08	0.00	0.00	**-0.15**	0.06	0.94	0.13	2.15	337.56	3183.70	15.13
	BKZ-5	40.69	40.69	40.58	**40.53**	41.10	0.00	0.00	-0.18	**-0.26**	0.71	0.24	4.00	553.90	3600.00	50.20
	BKZ-6	**39.18**	**39.18**	**39.18**	40.72	39.92	**0.00**	**0.00**	**0.00**	2.60	1.23	0.15	2.79	521.07	60.18	22.12
	BKZ-7	42.12	42.11	**42.10**	42.93	42.44	0.00	-0.01	**-0.02**	1.42	0.55	0.29	1.21	2141.20	392.22	144.80
$0.2 \times \lvert M\rvert$	BKZ-4	35.67	35.65	**35.64**	43.97	37.16	0.00	-0.02	**-0.04**	14.90	2.38	0.18	2.63	3600.00	11.26	20.28
	BKZ-5	39.13	39.12	**39.11**	53.82	39.81	0.00	-0.02	**-0.03**	31.84	1.14	0.33	4.88	3600.00	253.29	68.15
	BKZ-6	35.38	**35.37**	**35.37**	44.12	36.97	0.00	**-0.01**	**-0.01**	15.79	2.54	0.22	3.34	3600.00	7.95	27.13
	BKZ-7	**38.92**	**38.92**	**38.92**	41.03	39.70	**0.00**	**0.00**	**0.00**	3.55	1.30	0.39	1.64	3600.00	75.49	200.49
$0.3 \times \lvert M\rvert$	BKZ-4	**34.94**	**34.94**	**34.94**	41.73	37.79	**0.00**	**0.00**	**0.00**	11.69	4.60	0.24	3.22	3600.00	11.26	25.70
	BKZ-5	**36.65**	**36.65**	**36.65**	39.76	37.87	**0.00**	**0.00**	**0.00**	5.17	1.96	0.46	5.96	3600.00	253.29	84.64
	BKZ-6	**34.76**	**34.76**	**34.76**	42.01	37.58	**0.00**	**0.00**	**0.00**	12.58	4.53	0.28	3.95	3600.00	7.95	32.96
	BKZ-7	**36.45**	**36.45**	**36.45**	39.45	37.53	**0.00**	**0.00**	**0.00**	4.96	1.73	0.50	2.09	3600.00	68.37	257.94
Average		37.70	37.70	**37.68**	42.65	38.85	0.00	-0.01	**-0.04**	8.69	1.97	0.28	3.16	2696.14	660.41	79.13

Table 5 Comparison among the proposed heuristics to the set BKZ for the *min-max regret* MSCP.

as earthquakes, hurricanes and floods. For the latter, uncertainties are associated with the number of inhabitants affected after the disaster.

The numerical results indicate that the exact algorithms usually applied to *min-max regret* problems [31,43,47] have not obtained good results for *min-max regret* MSCP. In addition, the heuristics proposed for *min-max regret* MSCP found competitive results when compared to the ones produced by a one-hour run of the exact algorithms. It is worth mentioning that *min-max regret* MSCP is a challenging problem, requiring further studies on its structure and particularities. We conjecture that *min-max regret* MSCP belongs to Σ_2^p-complete class.

As future works, it is relevant to investigate the structure of the *min-max regret* MSCP mathematical formulation and to develop new cuts and valid inequalities. These future avenues of work may help to improve lower bounds for this problem. In addition, there is room to investigate representative scenarios and the reduction of scenarios by using dominance rules.

Acknowledgements This study was funded by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), the Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG), and Campus France.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Aissi, H., Bazgan, C., Vanderpooten, D.: Min-max and min-max regret versions of combinatorial optimization problems: A survey. European Journal of Operational Research **197**, 427–438 (2009)
2. Assunção, L., Santos, A.C., Noronha, T.F., Andrade, R.: A linear programming based heuristic framework for min-max regret combinatorial optimization problems with interval costs. Computers & Operations Research **81**, 51–66 (2017)
3. Averbakh, I., Berman, O.: Minmax regret p-center location on a network with demand uncertainty. Location Science **5**, 247–254 (1997)
4. Beasley, J.E.: A lagrangian heuristic for set-covering problems. Naval Research Logistics (NRL) **37**(1), 151–164 (1990)
5. Beasley, J.E.: OR-library: distributing test problems by electronic mail. The Journal of the Operational Research Society **41**, 1069–1072 (1990)
6. Benders, J.F.: Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik **4**, 238–252 (1962)
7. Beraldi, P., Bruni, M.E., Conforti, D.: Designing robust emergency medical service via stochastic programming. European Journal of Operational Research **158**, 183–193 (2004)
8. Beraldi, P., Ruszczyński, A.: The probabilistic set-covering problem. Operations Research **50**, 956–967 (2002)
9. Berman, O., Wang, J.: The minmax regret gradual covering location problem on a network with incomplete information of demand weights. European Journal of Operational Research **208**, 233–238 (2011)
10. Bertsimas, D., Sim, M.: The price of robustness. Operations Research **52**, 35–53 (2004)
11. Caprara, A., Fischetti, M., Toth, P.: A heuristic method for the set covering problem. Operations Research **47**, 730–743 (1999)
12. Caprara, A., Toth, P., Fischetti, M.: Algorithms for the set covering problem. Annals of Operations Research **98**, 353–371 (2000)
13. Carvalho, I.A., Noronha, T.F., Duhamel, C., Vieira, L.F.M.: A scenario based heuristic for the robust shortest path tree problem. In: IFAC-PapersOnLine, vol. 49, pp. 443–448 (2016)
14. Church, R., ReVelle, C.: The maximal covering location problem. Papers in Regional Science **32**, 101–118 (1974)
15. Church, R., Stoms, D.M., Davis, F.W.: Reserve selection as a maximal covering location problem. Biological Conservation **76**, 105–112 (1996)
16. Coco, A.A., Júnior, J.C.A., Noronha, T.F., Santos, A.C.: An integer linear programming formulation and heuristics for the minmax relative regret robust shortest path problem. Journal of Global Optimization **60**, 265–287 (2014). DOI 10.1007/s10898-014-0187-x
17. Coco, A.A., Santos, A.C., Noronha, T.F.: Scenario-based heuristics with path-relinking for the robust set covering problem. In: Proceedings of MIC 2015: The XI Metaheuristics International Conference, pp. 1–10 (2015)
18. Coco, A.A., Santos, A.C., Noronha, T.F.: Coupling scenario-based heuristics to exact methods for the robust set covering problem with interval data. In: IFAC-PapersOnLine, vol. 49, pp. 455–460 (2016)
19. Coco, A.A., Santos, A.C., Noronha, T.F.: Formulation and algorithms for the robust maximal covering location problem. In: Electronic Notes in Discrete Mathematics, vol. 64, pp. 145–154 (2018)

20. Cordeau, J.F., Furini, F., Ljubić, I.: Benders decomposition for very large scale partial set covering and maximal covering location problems. European Journal of Operational Research **275**, 882 – 896 (2019)
21. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 3rd edn. The MIT Press (2009)
22. Correia, I., da Gama, F.S.: Facility location under uncertainty. In: Location Science, pp. 177–203. Springer (2015)
23. Dantzig, G.B.: Linear programming and extensions. Princeton University Press (1963)
24. Deineko, V.G., Woeginger, G.J.: Pinpointing the complexity of the interval min–max regret knapsack problem. Discrete Optimization **7**, 191–196 (2010)
25. Drezner, Z., Hamacher, H.W.: Facility location: applications and theory. Springer Science & Business Media (2001)
26. Duin, C., Voss, S.: The Pilot Method: A strategy for heuristic repetition with application to the Steiner problem in graphs. Networks **34**, 181–191 (1999)
27. Edmonds, J.: Covers and packings in a family of sets. Bulletin of the American Mathematical Society **68**, 494–499 (1962)
28. Farahani, R.Z., Asgari, N., Heidari, N., Hosseininia, M., Goh, M.: Covering problems in facility location: A review. Computers & Industrial Engineering **62**, 368 – 407 (2012)
29. Fischetti, M., Monaci, M.: Cutting plane versus compact formulations for uncertain (integer) linear programs. Mathematical Programming Computation **4**, 239–273 (2012)
30. Fischetti, M., Salvagnin, D., Zanette, A.: A note on the selection of Benders' cuts. Mathematical Programming **124**, 175–182 (2010)
31. Furini, F., Iori, M., Martello, S., Yagiura, M.: Heuristic and exact algorithms for the interval min-max regret knapsack problem. INFORMS Journal on Computing **27**, 392–405 (2015)
32. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, USA (1979)
33. Glover, F., Laguna, M., Martí, R.: Fundamentals of scatter search and path relinking. Control and cybernetics **29**, 653–684 (2000)
34. Hammar, M., Karlsson, R., Nilsson, B.J.: Using maximum coverage to optimize recommendation systems in e-commerce. In: Proceedings of the 7th ACM Conference on Recommender Systems, RecSys '13, pp. 265–272. ACM, New York, NY, USA (2013)
35. Hooker, J.N., Ottosson, G.: Logic-based benders decomposition. Mathematical Programming **96**, 33–60 (2003)
36. Kasperski, A., Zieliňki, P.: Minimizing maximal regret in the linear assignment problems with interval costs. Preprint **7** (2004)
37. Kasperski, A., Zieliński, P.: An approximation algorithm for interval data minmax regret combinatorial optimization problems. Information Processing Letters **97**, 177–180 (2006)
38. Kasperski, A., Zieliński, P.: Robust discrete optimization under discrete and interval uncertainty: A survey, chap. 6, pp. 113–143. Springer International Publishing (2016). DOI 10.1007/978-3-319-33121-8_6
39. Kouvelis, P., Yu, G.: Robust discrete optimization and its applications. Kluver Academic Publishers (1997)
40. Lutter, P., Degel, D., Busing, C., Koster, A., Werners, B.: Improved handling of uncertainty and robustness in set covering problems. European Journal of Operational Research **263**, 35–49 (2017)
41. Máximo, V.R., Nascimento, M.C.V., Carvalho, A.C.P.L.F.: Intelligent-guided adaptive search for the maximum covering location problem. Computers & Operations Research **78**, 129–137 (2017)
42. Mišković, S.: A VNS-LP algorithm for the robust dynamic maximal covering location problem. OR Spectrum **39**, 1011–1033 (2017)
43. Montemanni, R., Barta, J., Mastrolilli, M., Gambardella, L.M.: The robust traveling salesman problem with interval data. Transportation Science **41**, 366–381 (2007)
44. Muren, Li, H., Mukhopadhyay, S.K., Wu, J., Zhou, L., Du, Z.: Balanced maximal covering location problem and its application in bike-sharing. International Journal of Production Economics **223**(4), 107513 (2020)
45. Owen, S.H., Daskin, M.S.: Strategic facility location: A review. European journal of operational research **111**, 423–447 (1998)
46. Pereira, J., Averbakh, I.: Exact and heuristic algorithms for the interval data robust assignment problem. Computers & Operations Research **38**, 1153–1163 (2011)
47. Pereira, J., Averbakh, I.: The robust set covering problem with interval data. Annals of Operations Research **207**, 217–235 (2013). DOI 10.1007/s10479-011-0876-5
48. Roy, B.: Robustness in operational research and decision aiding: A multi-faceted issue. European Journal of Operational Research **200**, 629–638 (2010)
49. Saxena, A., Goyal, V., Lejeune, M.A.: MIP reformulations of the probabilistic set covering problem. Mathematical Programming **121**(1), 1–31 (2010)
50. Schilling, D.A., Jayaraman, V., Barkhi, R.: A review of covering problem in facility location. Location Science **1**, 25–55 (1993)
51. Snyder, L.V.: Facility location under uncertainty: A review. IIE Transactions **38**, 537–554 (2006)
52. Voss, S., Fink, A., Duin, C.: Looking ahead with the Pilot Method. Annals of Operations Research **136**, 285–302 (2005)
53. Wang, S., Cui, W., Chu, F., Yu, J.: The interval min–max regret knapsack packing-delivery problem. International Journal of Production Research **0**, 1–17 (2020)
54. Wolsey, L.A.: Integer programming. Wiley-Interscience, New York, NY, USA (1998)
55. Wu, H.H., Kucukyavuz, S.: Probabilistic partial set covering with an oracle for chance constraints. SIAM Journal on Optimization **29**, 690–718 (2019)
56. Xia, L., Xie, M., Xu, W., Shao, J., Yin, W., Dong, J.: An empirical comparison of five efficient heuristics for maximal covering location problems. In: 2009 IEEE/INFORMS International Conference on Service Operations, Logistics and Informatics, pp. 747–753 (2009). DOI 10.1109/SOLI.2009.5204032