Mathematical Models and Solving Methods for Diversity and Equity Optimization

RAFAEL MARTÍ
Departamento de Estadística e Investigación Operativa, Universidad de Valencia, Spain
Rafael.Marti@uv.es

FRANCISCO PARREÑO
Escuela Superior de Ingeniería Informática, Universidad de Castilla La Mancha, Spain
Francisco.Parreno@uclm.es

JORGE MORTES
Département Automatique, Productique et Informatique, IMT Atlantique, France
jorge.mortes-alcaraz@imt-atlantique.fr

ABSTRACT

Discrete diversity optimization basically consists in selecting a subset of elements of a given set in such a way that the sum of their pairwise distances is maximized. Equity, on the other hand, refers to minimize the difference between the maximum and the minimum of the distances in the subset of selected elements to balance their diversity. Both problems have been studied in the combinatorial optimization literature, but recently major drawbacks in their classic mathematical formulations have been identified. We propose new mathematical models to overcome these limitations, and heuristic methods to solve large size instances of them. In particular, we propose a matheuristic based on the CMSA methodology for diversity, and a GRASP heuristic for equity. Our extensive experimentation compares the original models with the new proposals by analyzing the solutions of our heuristics and those of the previous approaches. We also evaluate their quality with respect to the optimal solutions, size permitting. Statistical analysis allows us to draw significant conclusions.

Keywords: Maximum diversity, heuristic optimization, mathematical programming.

Original version: March 2022

1. Introduction

Maximizing diversity by means of combinatorial optimization models has gained prominence in Operations Research (OR) over the last two decades, and constitutes nowadays a well-established research area. Continuous diversity models were very popular in the mathematical programming literature in the sixties and seventies, and in the late eighties they were adapted to discrete models. Terms like diversity, dispersion, and equity are nowadays widely applied in discrete optimization, especially in areas related to logistics and planning.

Kuby (1988) proposed the first integer models to state in mathematical terms the rather ambiguous concept of diversity. To formulate it, the author considered a base graph $G = (V, E)$, where V is the set of n nodes or elements, and E is the set of edges, and d_{ij} is the inter-element distance between any two elements i and j. The maximum diversity problem (MaxSum model) consists in selecting a subset of m elements $M \subseteq V$, in such a way that their sum of distances is maximized. In mathematical terms, it can be easily formulated with binary variables, $x_i \in \{0,1\}$, indicating whether element i is selected or not, as follows:

$$\begin{array}{lll} \text{(MaxSum model)} & \text{Maximize} & \sum_{i<j} d_{ij} x_i x_j \\ & \text{subject to} & \sum_{i=1}^{n} x_i = m \\ & x_i \in \{0,1\} & i = 1, \dots, n \end{array}$$

Kuby (1988) also introduced the p-dispersion problem (MaxMin model), in which instead of maximizing the sum of distances among the selected elements, it maximizes the minimum distance between the selected elements.

Since these early proposals, many researchers have developed models and methods, mainly heuristics and metaheuristics (Glover et al., 2021), to provide high-quality solutions to these two problems. From Erkut and Neuman (1989) to Martínez-Gavara et al. (2021), we can find more than 50 papers published in top ranked journals proposing solving methods for these problems and their variants, where the MaxSum model is the most widely applied.

A very interesting alternative to diversity was introduced by Prokopyev et al. (2009) under the term equity, which incorporates the concept of fairness among candidates. The associated models appear not only in group selection, but also in facility location or sub-graph identification, in which one may address fair diversification among members of a network. The Minimum Differential Dispersion model, MinDiff, minimizes extreme equity values of the selected elements, namely the maximum and the minimum sum of distances for each selected element. It can be formulated as:

$$\begin{array}{lll} \text{(MinDiff model)} & \text{Minimize} & \max_{i \in M} \sum_{j:j \neq i} d_{ij} x_j - \min_{i \in M} \sum_{j:j \neq i} d_{ij} x_j \\ & \text{subject to} & \sum_{i=1}^{n} x_i = m \\ & x_i \in \{0,1\} & i = 1, \dots, n \end{array}$$

Parreño et al. (2021) recently identified an important issue in the MaxSum and the MinDiff models when analyzing their solutions. In particular, although the optimal solution of the MaxSum has an overall large diversity value, as measured by its objective function, some of its selected elements might be relatively close, which is not in line with the objective of maximizing diversity. Figure 1 shows the optimal solution of an Euclidean instance with with $n = 50$ elements from which we select $m = 5$ to maximize their inter-distances sum.

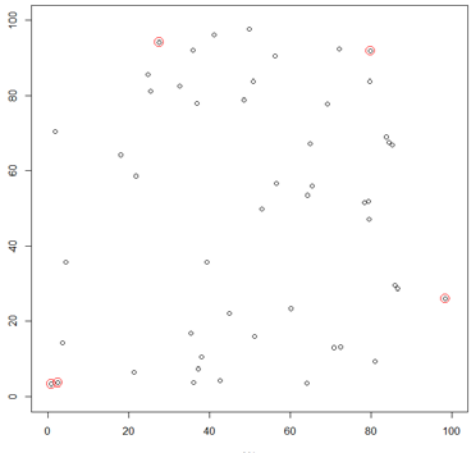

Figure 1. Optimal MaxSum solution of an instance with $n = 50$ and $m = 10$.

Figure 1 shows the 50 points in the set (depicted with small black circles), and the 5 points selected in the optimal solution of the MaxSum model (depicted with larger red circles). Parreño et al. (2021) identified two important characteristics of the geometry in the MaxSum solutions represented in their study: the selected points are located close to the border of the diagram avoiding the central region, and some of the selected points are very close to each other (as the two selected points in the left-bottom part of Figure 1). This second characteristic can be considered an issue, since the objective of the model is to achieve diversity, and there is no justification to select close points. The authors also pointed out that a similar situation, if not even worse, can be found in the MinDiff model. Figure 2 shows the optimal MinDiff solution of an instance with $n = 25$ elements from which we have to select $m = 3$ of them to minimize the difference of their inter-distances.

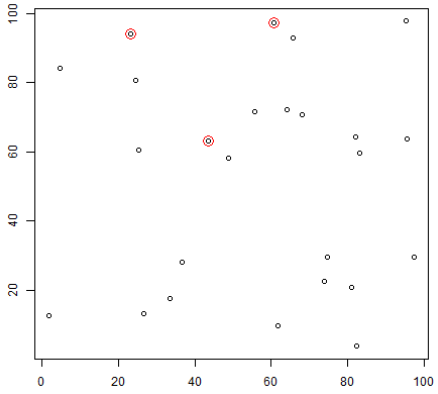

Figure 2. Optimal MinDiff solution of an instance with $n = 25$ and $m = 3$.

The location of the three selected points in Figure 2 clearly illustrates the drawback of the MinDiff model. These points are located in the upper part of the diagram, and they are relatively close to each other. This objective function seeks for inter-distance equality among the selected points, which is the desired objective in terms of equity; however, it ignores how large or small these distances are. Considering that the equity model belongs to the family of diversity problems, there is no justification to select points very close to each other, when we could select points with similar inter-distances, but at a larger distance from each other. We can find applications in facility location theory in which, together with the equal-distance distribution, we need that the selected points are relatively distant to cover a territory (Teitz, 1968). Consider for example the case in which the facilities provide a service, and they have to be scattered over a territory, and we have to avoid their concentration in a specific zone. At the same time, they may need to be within a specific distance, and none of them can be very far from the rest. This type of situation may occur in emergency health care in which portable clinics are installed to provide service and at the same time they depend among them to share supplies. In these location models, the MinDiff properly reflects the idea of being at a similar distance but the standard model fails on scatter them over the territory.

The limitations found with the two models described above, have been also identified in the recent review by Martí et al. (2022) in which the research on diversity problems is classified in three periods: the early period (1980-2000), the expansion period (2000 -2010), and the development period (2010 – now). As a matter of fact, in the first paper on discrete diversity models, Kuby (1988) already mentioned the potential limitations of the MaxMin model due to its multiple optimal solutions and pointed out a possible solution combining it with the MaxSum model. In line with that, Porumbel et al. (2011) proposed a heuristic that considers both objectives simultaneously. We view our proposal to extend the classical models on diversity and equity, as built from these previous studies.

In this paper we propose extended formulations for the MaxSum and MinDiff models to optimize their respective objective functions while avoiding the selection of very close elements. In this way, we overcome their limitations and offer robust models to achieve diversity and equity respectively. We test these models with CPLEX on the instances in the public domain benchmark library MDPLIB. As expected, in some cases CPLEX cannot solve large size instances in practical CPU times, so we propose heuristic methods to target them. Specifically, we consider a matheuristic for the Extended MaxSum, based on the CMSA methodology (Construct, Merge, Solve and Adapt by Blum et al. 2016), and a GRASP (Feo and Resende, 1989) for the Extended MinDiff.

2. Extended mathematical formulations

In our diversity and equity models, we want to avoid the selection of close points. This can be easily achieved by adding a threshold constraint, in which we do not allow the selection of two points if its inter-distance is lower than a value. In mathematical terms, this can be easily included in the models above based on binary variables, with the constraints:

$$d_{ij} \geq d^* x_i x_j \qquad i, j = 1, \dots, n$$

The problem is how to set the threshold value d^*. It is clear that the larger the value the better the solution; however, if we select it too large, we may obtain an unfeasible model. The best d^* is given by the solution of the MaxMin model, since it is the maximum value with feasible solutions. This

model, also known as p-dispersion problem (Kuby, 1988), can be trivially formulated by simply considering its objective function over the set of selected elements M:

$$\text{Maximize} \qquad d^* = \min_{i,j \in M} d_{ij}$$

$$\text{subject to} \qquad M \subseteq V,\ |M| = m.$$

The MaxMin diversity problem has been exhaustively studied, and there are many efficient models and heuristic algorithms to solve it (see for example Resende et al., 2010). We build our extended model on the MaxSum problem by merging the standard models on the MaxSum and MaxMin as follows:

$$\begin{aligned}
\text{(Extended MaxSum)} \quad \text{Maximize} \quad & \textstyle\sum_{i<j} d_{ij} x_i x_j \\
\text{subject to} \quad & \textstyle\sum_{i=1}^{n} x_i = m \\
& d_{ij} \geq d^* x_i x_j \qquad i, j = 1, \dots, n \\
& d^* = \text{Max} \min_{i,j \in M'} d_{ij} \\
& \qquad \text{subject to} \qquad M' \subseteq V,\ |M'| = m. \\
& x_i \in \{0,1\} \quad i = 1, \dots, n
\end{aligned}$$

In the Extended MaxSum model, the set M' represents the solution of the problem MaxMin problem (that we call the lower level problem), and M represents the solution of the constrained MaxSum defined by the x-variables (upper level problem). Note that if we consider the Extended Model as a bi-level model, it is straightforward to solve, since the variables of the two problems are separated. We only have to solve the lower level model, MaxMin in this case, and then, with its optimal objective function value d^*, solve the upper level model. However, considering that the two problems are NP-hard, the extended problem is indeed difficult to solve.

If we analyze the extended model from the perspective of the set of solutions, we can see that we are solving the upper level problem, the MaxSum, in the restricted feasible region given by the optimal solutions of the MaxMin problem. As a matter of fact, a straightforward (inefficient) way to obtain the optimal solution of the extended problem, would be to enumerate all the feasible solutions (selections of m elements) and keep all the optimal solutions of the MaxMin. Then, evaluate them in terms of the MaxSum objective function, and return the one with maximum value. This is clearly impracticable in medium or large size instances due to the large number of solutions, but it helps to understand the nature of the model. For example, we may conclude that if the MaxMin model has many alternate optimal solution, it makes sense to search the best of them in terms of a secondary objective, which helps to break ties. This will obviously depend on each particular instance, but we have empirically found that, in general, the MaxMin model has many alternate optimal solutions and it makes sense to consider a secondary objective.

Another important aspect of the extended model is to compute how much it deteriorates the upper objective function with respect to the original (upper) model. If we solve the MaxSum over a restricted set of solutions, it is expected that the optimal value may be worse than the optimal value of the standard MaxSum model in which the feasible region is significantly larger. However, we have to keep in mind that both models are highly correlated, and when we restrict the search of the MaxSum solution to the MaxMin optimal solutions, we are actually having very good candidate solutions, and therefore we do not expect a significant deterioration in the MaxSum final value. However, the Extended MinDiff problem exhibits a very different situation.

Sandoya et al. (2018) computed the correlation of the best values of four diversity and equity models (MaxSum, MaxMin, MinDiff, and MaxMinSum) over 30 instances with $n = 20$ elements from which $m = 5$ have to be selected. The correlation obtained in their study between the MaxSum and MaxMin is 0.78, while the correlation between the MinDiff and the MaxMin is 0.03. Our hypothesis is then that in the Extended MaxSum, we do not expect an important deterioration of the objective compared with the original MaxSum, but in the Extended MinDiff it is likely to occur. From a theoretical perspective, we can model the Extended MinDiff in similar terms as we modeled above the Extended MaxSum:

$$
\begin{aligned}
\text{(Extended MinDiff)} \quad \text{Minimize} \quad & \max_{i \in M} \textstyle\sum_{j:j \neq i} d_{ij} x_j - \min_{i \in M} \sum_{j:j \neq i} d_{ij} x_j \\
\text{subject to} \quad & \textstyle\sum_{i=1}^{n} x_i = m \\
& d_{ij} \geq d^* x_i x_j \qquad i, j = 1, \ldots, n \\
& d^* = \text{Max} \min_{i,j \in M'} d_{ij} \\
& \qquad \text{subject to} \quad M' \subseteq V, \ |M'| = m. \\
& x_i \in \{0,1\} \quad i = 1, \ldots, n
\end{aligned}
$$

In our computational testing, we will see that the extended model provides poor solutions due to the low correlation between the objectives (MinDiff and MaxMin) mentioned above. We will therefore propose a method to relax the lower level problem (MaxMin) in search for a compromise between both objectives. The Extended MaxSum is, on the other hand, simpler to analyze, since we can directly consider it as shown in the formulation. We will solve both extended models with CPLEX and compare them with their original counterparts respectively.

3. Previous methods

We have identified a previous effort in line with the Extended MaxSum, so we will describe it in this section. On the other hand, there is no previous method for the Extended MinDiif, and therefore we will use a competitive MinDiff solver as a reference in our comparisons.

3.1 The MaxSum model

Porumbel et al. (2011) proposed a fast tabu search (Glover et al., 2021) for a model that combines the MaxMin and the MaxSum problems, which can be considered a first approach to solve the Extended

MaxSum. In particular, the authors minimize the MaxMin objective function and consider the MaxSum as a secondary objective. The inclusion of this secondary objective is motivated by the fact that there may be a relative large number of solutions that qualify as optimal for the MaxMin, and it makes sense to choose the best one among them in terms of the MaxSum objective. This is exactly the point of departure of our research. Although not mentioned by these authors, and ignored by researchers in discrete diversity optimization, we can find this proposal in the very first paper published for these problems. In fact, Kuby (1988) not only introduced the MaxSum and the MaxMin as described above, but this author also introduced what he called a multi-criteria approach, which is the same problem solved by Porumbel et al., arguing that the MaxSum model is an appropriate way to choose among the many alternate optima of the MaxMin problem.

The tabu search by Porumbel et al. (2011) has two phases. In the first one a greedy algorithm constructs an initial solution. At each iteration, it adds to the partial solution under construction the element with the largest contribution in terms of the MaxMin objective, and it breaks ties according to the MaxSum. The second phase is a short term tabu search based on Add and Drop moves. An interesting characteristic of this method is its implicit use of memory. Instead of the typical tabu list, the method employs a tabu rule in which it always drops from the solution its oldest point. Then, it adds to the solution the best element in a similar way than its greedy construction. In this way, the authors avoid the quadratic complexity that usually exhibit local search methods based on exchanges. Their empirical analysis shows that this simple tabu search performs better than previous approaches for the MaxMin model.

An alternative way to approach this problem came from the exact domain. Sayyady and Fathi (2016) proposed an efficient method to solve the MaxMin to optimality. The authors consider the node packing problem, in which given a threshold value l, a graph $G(l)$ is defined with the set V of n nodes of graph $G = (V, E)$, and the set of edges $E(l) = \{(i,j) \in E: d_{ij} < l\}$. The node packing problem consists in finding a maximum cardinality subset of nodes so that no two nodes in this subset are adjacent to each other. In this way, an optimal solution of the node packing problem in G provides a set of $v(l)$ points with minimum distance larger than or equal to l. Sayyady and Fathi proposed to solve a sequence of node packing problems for different values of l until it obtains a set of $v(l) = m$ points (the optimal solution of the MaxMin model). In mathematical terms, the method iteratively solves the following problem:

$$
\begin{array}{lll}
v(l)\text{=Maximize} & \sum_{i=1}^{n} x_i & \\
\text{subject to} & x_i + x_j \leq 1 & \forall (i,j) \in E(l) \\
 & x_i \in \{0,1\} & i = 1, \ldots, n
\end{array}
$$

The authors propose a systematic search in the interval $l \in [d_{min}, d_{max}]$, where d_{min} and d_{max} are the minimum and maximum values respectively among all the distances in the graph. They base the search on the property that if we have two l –values, l_1 and l_2 such that $l_1 < l_2$, $v(l_1) \geq m$ and $v(l_2) < m$ then, the optimal value of the MaxMin problem z^* verifies $l_1 \leq z^* < l_2$. The method performs a binary search over the ordered set of different distances in the graph. To that end, the minimum distance between consecutive values is computed to divide the search into 2^q equal subintervals, performing at most q steps (i.e., solving the node packing problem a maximum of q

times). We will apply this method in CPLEX as a first step, to obtain the optimal solution of the MaxMin problem, to then solve the upper level problem in the extended model.

To sum it up, we have identified two previous approaches that can be applied to solve the Extended MaxSum, the heuristic by Porumbel et al. (2011), and the exact method by Sayyady and Fathi (2016). In this latter case, we first obtain the MaxMin optimal solution value d^*, and then solve with CPLEX the MaxSum formulation with the additional constraint that excludes the pairs of elements with distance lower than d^*. Since this can be impractical for large size instances, we propose a matheuristic, CMSA, to efficiently solve medium and large instances. In our empirical testing we compare these approaches.

3.2 The MinDiff model

Duarte et al. (2015) proposed a heuristic algorithm to find good solutions for the MinDiff problem. The authors adapted a Greedy Randomized Adaptive Search Procedure (GRASP) in terms of this equity problem. GRASP (Feo and Resende, 1989) is a multi-start algorithm where each iteration consists in two steps: constructive method and improvement method. In this way, in the first step, the algorithm constructs a feasible solution using a greedy randomized procedure and, in the second step, it uses a local search to improve the constructed solution. We make below a brief description of the two constructive methods (C1 and C2), and the three local search methods (LS1, LS2, and LS3) proposed in Duarte et al. (2015).

The first constructive method (C1) is based on the standard GRASP design. It first creates a candidate list (CL) with all the elements that could be added to the solution (M). After that, it selects two elements at random from CL, and adds them to M. An iteration finishes by removing the elements from CL.

To compute the greedy evaluation function, we calculate for each element $u \in CL$, the change in the objective function that it may cause if added to the solution. To do that, we first compute, for each element v in M, the sum of the distances to the rest of the elements in M, and we include in this computation the element u. In mathematical terms,

$$s(v) = \sum_{w \in M} d_{wv} + d_{uv}.$$

Then, we compute the maximum and minimum of these sum of distance (s-values), s_{max} and s_{min} respectively, and we include the sum of distances from u to the elements in the solution. In other words, we compute the maximum and minimum of the sum of distances in the solution considering that u is already part of the solution.

$$s_{max} = \max\left\{\sum_{v \in M} d_{uv}\,, \max_{v \in M} s(v)\right\},$$

and

$$s_{min} = \min\left\{\sum_{v \in M} d_{uv}\,, \min_{v \in M} s(v)\right\}.$$

Finally, the greedy function is computed as $g(u) = s_{max} - s_{min}$.

As it is customary in GRASP, C1 creates the restricted candidates list (RCL) as

$$RCL = \{v \in CL \mid g(v) \leq g_{min} + \alpha_1 \cdot (g_{max} - g_{min})\},$$

where $g_{max} = \max_{u \in CL} g(u)$, $g_{min} = \min_{u \in CL} g(u)$, and $\alpha \in [0,1]$. This list contains all the elements in CL whose greedy function does not exceed an α-percentage of its best value. Lastly, the method selects one element from RCL at random, includes it into M and removes it from CL.

The second constructive method (C2) is based on the strategy proposed in Resende and Werneck (2004). C2 creates the RCL by selecting $\alpha_2|CL|$ elements at random, evaluates the greedy function on all these elements, and selects the element with best value to become part of the partial solution. In the same way that C1, the method iterates until m elements are selected. We may say that C1 first implements the greedy strategy and then the random one, while C2 applies them in the other way around.

The authors proposed three different local search procedures to improve the solution provided by the constructive methods. They are based on trying to exchange an element $u \in M$ with another element v not in the solution. In this way, the algorithm computes the objective function value of the new solution $M' = M \cup \{u\} \setminus \{v\}$ in an incremental way. For each $w \in M \setminus \{u\}$,

$$s(w) = \sum_{a \in M \setminus \{u\}} d_{wa} + d_{wv}$$

$$s_{max} = \max \left\{ \sum_{w \in M \setminus \{u\}} d_{vw}\, , \max_{w \in M \setminus \{u\}} s(w) \right\},$$

$$s_{min} = \min \left\{ \sum_{w \in M \setminus \{u\}} d_{vw}\, , \min_{w \in M \setminus \{u\}} s(w) \right\}.$$

The objective function of the new solution is then computed as $f(M') = s_{max} - s_{min}$. So, if the new solution improves upon the previous one (i.e., $f(M') < f(M)$), the method performs the move. Based on this algorithm, the authors define three local search procedures: The first one (LS1) implements the so-called best improvement strategy, in which the entire neighborhood is scanned in each iteration to identify the best move; the second one (LS2) does not make an exhaustive search, but it performs the first move that improves the solution (first improvement strategy), and the last procedure (LS3) applies a first strategy after sorting the vertices of M in descending order according to their distance values with respect to the elements already in the solution (i.e., LS3 explores first the most promising moves).

Duarte et al. (2015) performed an exhaustive experimentation to disclose the best configuration of their GRASP, and concluded that the best constructive method is C2 run with the parameter α_2 set to 0.5, and the best local search is LS2. We therefore include this method in our comparison.

4. The Extended MaxSum Diversity Problem

The CMSA method (Construct, Merge, Solve, and Adapt) proposed by Blum et al. (2016) is based on solving a reduced instance (a sub-instance of reduced size) of a given instance with an exact method, and then use it to solve the original instance. It requires that any solution to the sub-instance is also a solution to the original instance. In the diversity problems considered here, a solution is a selection of m points, and therefore, if we solve the problem on a reduced instance in which a subset of the original points is considered, the output solution can be directly considered a solution of the original instance. Since we have an efficient exact method to solve medium size instances of the Extended Max-Sum problem, we can apply CMSA to target large instances of this problem.

As described in Blum et al. (2016), the first step of each CMSA iteration (Construct) consists in generating a number of feasible solutions to the original problem instance in a probabilistic way. In our case, we apply a heuristic and collect a set of good solutions. In a second step (Merge), we populate the set S with the nodes selected in these solutions. We expect that the nodes in S are a sample of diverse points of the original instance. In the third step, we apply CPLEX to the sub-instance formed with the nodes in S, and solve it to optimality. Finally, in the last step (Adapt), we remove some of the elements in S applying an aging mechanism. In this way, the set S with the "good" nodes evolves during the search process, since at each iteration new nodes are added and the old ones are dropped. The method performs iterations, keeping the best solution found so far, which is returned as its final output.

When a node x in a constructed solution is added to the set S, its age is set to 0 ($age(x) = 0$). Then, at the end of each iteration, the age of all the nodes in S is incremented by one unit, with the exception of those in the best solution found so far, which are reset to 0. When the age of a node reaches the maximum age (max_age), it is dropped from S.

It is well documented that the good performance of metaheuristics is based on a balance between search intensification and diversification. The intensification in the CMSA method comes from the number of repetitions k of the heuristic method, which will determine the number of nodes added to S together with the exact resolution of the mathematical model. Search diversification, on the other hand, is managed by the maximum age, max_age, which will determine the number of elements dropped from S, in a way that the size of S permits to be solved to optimality.

Considering that the tabu search method by Porumbel et al. (2011), TS, is an efficient way to obtain heuristic solutions for the MaxMin with a relative good value for the MaxSum, we apply it in the ***Construct*** step of our CMSA algorithm. We adapt TS to produce several good solutions to populate S. In particular, we divide its search into two parts. In the first one, we simply identify the best solution found, x_{best}, in terms of the MaxMin objective, and during the second half we include in the elite set, ES, all the solutions M visited with a MaxMin value larger than or equal to the x_{best}. Then, in the ***Merge*** step, we include in S all the elements in the solutions in ES. In mathematical terms:

$$S = \{x \in V : \ x \in M, \ M \in ES\}$$

We now perform the ***Solve*** step of CMSA, in which we solve our problem on the reduced instance formed with the elements in S. In this step, we first apply the binary search method by Sayyady and Fathi (2016) to solve the MaxMin problem. In this way, we try to improve (increase) the MaxMin value

obtained with the tabu search in the Construct step. Let d^* be the best MaxMin value obtained (i.e., the minimum distance between the selected elements in the solution in which this value is maximum). Then, we solve the MaxSum model restricted to the distances larger than or equal to d^*. The output of this phase is the best MaxSum solution in this set. This reduced MaxSum model can be expressed in mathematical terms as:

$$\text{(Solve step)} \qquad \text{Maximize} \quad \sum_{i,j \in S} d_{ij} x_i x_j$$

$$\text{subject to} \quad \sum_{i \in S} x_i = m$$

$$d_{ij} \geq d^* x_i x_j \qquad i, j \in S$$

$$x_i \in \{0,1\} \qquad i \in S$$

Note that in large instances this ***Solve*** step may require excessive long running times (since we are solving two exact models). We therefore limit its total execution time to keep the computational effort relatively low. In particular, we devote a third of the running time of this step to the first part to improve the MaxMin value with the binary search. The remaining two thirds of time are devoted to solve the MaxSum over the set of good MaxMin solutions. An early termination of these steps may obviously result in missing the optimal solution, and therefore we have to keep in mind that we are applying here a heuristic method, that balances solution quality with running time.

The last step of a global iteration is ***Adapt***, in which we remove from S its oldest elements according to the max_age counter to keep its size relatively small, since in subsequent applications of Construct new elements will be added to S. It therefore manages the size of S, and in consequence the computational effort of CPLEX to solve the two models. In our empirical experimentation we will adjust it to obtain a good balance between search intensification (exploitation) and diversification (exploration).

The central part of the CMSA method is the ***Solve*** step, in which we first apply the binary search solver to improve the MaxMin value, and then the MaxSum mathematical model. Considering that the TS applied in the ***Construct*** step may produce a very good MaxMin value d^* (eventually optimal), we propose an alternative to this step, in which we only apply the MaxSum mathematical model shown above, skipping the binary search. We call CMSA2 to the algorithm with this variant of the solve step (keeping the rest of the method as described above).

5. The Extended MinDiff Equity Problem

In this section, we adapt the GRASP methodology to the Extended MinDiff and call Extended GRASP (EXG) to the resulting method. In particular, we first solve the MaxMin problem to identify a set of high-quality solutions for this model. This first phase targets the lower level of the extended problem, since the elements in these high-quality MaxMin solutions form the elite set of good candidates to solve the upper-level problem. In the second phase, we solve the MinDiff over the subset of elite elements previously identified. In this way, we achieve a compromise between the MaxMin and MinDiff models. It is worth mentioning that we do not solve the MinDiff over the set of optimal MaxMin solutions, as specified in the extended model, but we are proposing a heuristic approach in which we sequentially solve these two models in an approximate way.

In order to get the MaxMin elite set in the **first phase** of EXG, we implement a standard GRASP algorithm to solve the MaxMin problem, and then select all the elements in the best five solutions. The MaxMin GRASP works in the following way: The constructive method selects the first two elements at random, includes them into the partial solution, and creates a candidate list CL with the elements that are not in the solution. Then, it computes the so-called restricted candidate list, RCL, selecting $\alpha_3|CL|$ elements. Lastly, it selects from RCL the element with maximum distance to the partial solution and adds it. This algorithm iterates until it selects m elements. On the other hand, to improve the solution provided by the constructive phase, the algorithm applies a local search post-processing to the constructed solution. This procedure considers the two elements in the solution with the minimum inter-distance to remove the one with lower sum of distances to the rest of elements in the solution. In particular, we scan the elements not in the solution in the canonical order, and replace it with the first one that improves the MaxMin value of the solution (which constitutes the so-called first improvement strategy). The algorithm iterates until no further improvement is possible. At this stage, it returns the final solution as the local optimum and stops.

The **second phase** of EXG considers the elements in the elite set (i.e., those that form part of the good MaxMin solutions identified in the first phase). Now, we apply a second GRASP heuristic to solve the MinDiff problem over this set of points. However, this could lead to low quality MaxMin solutions since some of these elements can be relatively close. We therefore propose an adaptation of the GRASP to avoid the selection of close elements. In particular, in the C1 constructive method described in Section 3.2, instead of directly selecting an element from RCL, we compute

$$md(u) = \min_{w \in M} d_{wu}\,, \forall\, u \in RCL,$$

calculate

$$md_{max} = \max_{u \in RCL} md(u), \text{ and } md_{min} = \min_{u \in RCL} md(u),$$

and create a refined restricted candidate list (RCL2) from which the algorithm selects an element at random to add it to the solution.

$$RCL2 = \{v \in RCL \mid md(v) \geq md_{max} - \beta \cdot (md_{max} - md_{min})\}.$$

We propose a similar adaptation of C2. Specifically, we first create RCL selecting $\alpha_2|CL|$ elements at random, calculate $md(u)\ \forall\, u \in RCL$, md_{max}, and md_{min}, and create a RCL2 in the same way to C1. The algorithm then proceeds in the same way as described in Section 3.2, evaluating $g(u)\ \forall\, u \in RCL2$, and adding to the solution the $u \in RCL2$ with best value.

Finally, we adapt the local search by modifying the typical move selection in diversity problems. In particular, we restrict the neighborhood of a solution, to those moves that do not deteriorate the MaxMin value of the current solution. In this way, the local search of the Extended GRASP explores this reduced neighborhood in search for an exchange of elements that improves the MinDiff value without reducing the MaxMin value.

6. Computational Experiments

This section describes the computational experiments performed to assess the merit of the methods described in the previous sections. In particular, we consider our CMSA method for the Extended MaxSum, described in Section 4 and implemented in CPLEX, and the 2-phase GRASP for the Extended MinDiff (EXG), described in Section 5 and implemented in Python 3.8. All the experiments were conducted in a 2.8 Ghz Intel Core i5 8400 with 16 GB RAM.

As it is customary in heuristic papers, we divide our experimentation into two parts. A first part devoted to analyze the elements of our algorithms, usually called scientific testing, and a second part to compare our methods with the best published so far, usually called comparative testing. To avoid the overtraining of the methods, we analyze their elements and set up the values of their parameters in a subset of instances. The comparative testing, on the other hand, is performed over the entire benchmark of instances (described in Section 6.1).

This section is organized as follows. After the description of the instances and statistics used to report our experimentation in Section 6.1, we consider separate sections for each problem. In particular, Section 6.2 describes our findings on the Extended MinDiff, and Section 6.3 is devoted to the Extended MaxSum.

6.1 Problem instances and statistics

In line with previous papers on diversity problems, we consider the benchmark library called MDPLIB (Martí et al., 2013) to perform our empirical analysis. This set has been recently updated (MDPLIB 2.0, Martí et al. 2022) adding more challenging instances, and removing the nowadays trivial ones. The library is divided into the following three groups: SOM, GKD, and MDG. A brief description of the sets is included here:

- **SOM:** This benchmark set contains the 70 original matrices with integer random numbers between 0 and 9 generated from an integer uniform distribution included in MDPLIB. In MDPLIB 2.0, 80 new larger instances are included with integer random numbers between 0 and 100 generated in the same way.

- **GKD:** This benchmark set contains the 140 original matrices for which the values were calculated as the Euclidean distances from randomly generated points with coordinates in the 0 to 10. 300 new matrices are included in the MDPLIB 2.0, for which the values were calculated in the same way but with two coordinates in the 0 to 100 range.

- **MDG:** This benchmark set contains the 100 original matrices with real numbers randomly selected between 0 and 10 from a uniform distribution.

In our experimentation, we do not include the smallest instances in these sets (because they do not permit to differentiate among methods), and the largest ones (because CPLEX cannot solve them). Our testbed thus contains 500 instances. We considered a subset of 100 of them to perform the preliminary experimentation (scientific testing).

In all our experiments, we report the following statistics:

- **Value:** Average objective function value of the method.
- **DevB:** Average deviation from the best value of the experiment.
- **#Best:** The number of times that the best solution was reached by this method.
- **Time:** The average computing time of each method in seconds.

6.2 The Extended MaxSum

In this section we first compare the two CMSA variants, to choose the best one of our proposals, and then compare the best one with the previous methods.

6.2.1 Scientific testing

We consider the CMSA method described in Section 4, and the reduced variant, CMSA2, which skips the binary search and only applies the MaxSum mathematical model in the Solve step. Table 1 shows the results of this comparison.

	MaxSum			**MaxMin**			
	Value	**DevB**	**#Best**	**Value**	**DevB**	**#Best**	**Time**
CMSA	100124.0	0.11%	419	65.79	0.01%	492	220.69
CMSA2	100359.8	0.02%	466	65.78	0.06%	440	218.31

Table 1. CMSA variants

Results in Table 1 have to be carefully read. In terms of the MaxMin, the CMSA obtains better results than the reduced version CMSA2, with 492 best solutions of the former method versus 440 of the latter one. In terms of the MaxSum, CMSA seems worse than CMSA2, with 419 best solutions of the former method versus 466 of the latter one. However, considering that these methods solve the MaxSum model restricted to the distances larger than or equal to the best MaxMin value d^* found, we may say that this comparison is somehow unfair. If we compare the MaxSum value only in the instances in which both methods obtain the same MaxMin value, we will measure their ability to find good MaxSum values under similar conditions. According to this, both methods present similar results, since CMSA is able to obtain 319 best solutions and CMSA2 328. This fact together that both methods present similar running times, makes us to select the CMSA for the rest of our experimentation.

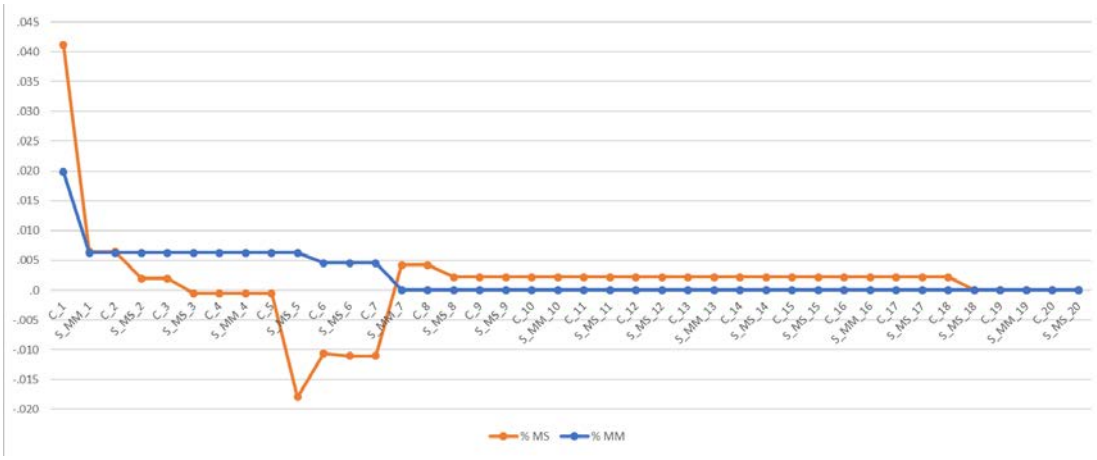

Figure 3. CMSA search profile

In our second experiment of this section we evaluate the contribution of the main elements of the CMSA method. Figure 3 shows the evolution of the best solution over a standard run on a representative instance with $n = 250$. In this diagram the x-axis represents iterations, and the y-axis optimal value deviations. Since we know the optimal solution for this instance, we represent the deviation of both values, MaxSum (%MS) and MaxMin (%MM), of the best solution so far, the incumbent solution, with respect to the optimal values. It must be noted that the optimal values correspond to the output of the exact method that computes the best MaxSum value over the set of the optimal MaxMin values. Therefore, a solution with a worse (lower) MaxMin value could exhibit a better (larger) MaxSum value than the reference optimal solution. In this figure, C_i refers to the best solution of the Construct step at iteration i. Similarly, S_MM_i represents the best solution obtained with the application of the MaxMin optimization in the Solve step at iteration i (with the binary search method), and S_MS_i the best solution obtained with the application of the MaxSum optimization (with the restricted mathematical model). This diagram clearly shows the evolution of the best solution found, and the contribution of the optimizations performed in the CMSA method.

Figure 3 shows that during the initial iterations, the best solution significantly improves both values, MaxSum and MaxMin, by means of the combination of the optimizations applied in the Construct and Solve steps. Then, the method stagnates, and it is not able to improve the solutions for several iterations (from 8 to 18). However, after that period, it is able to marginally improve the MaxSum objective without deteriorating the MaxMin one, reaching the reference optimal solution (identified with an exact method in significantly larger running time).

We finish this section by showing the CMSA solution of the instance represented in Figure 1. As we can see in Figure 4, this solution (represented with red circles) maximizes the total dispersion while avoiding the selection of close points, and compares favorably with the optimal MaxSum solution depicted in Figure 1, in which two elements were very close.

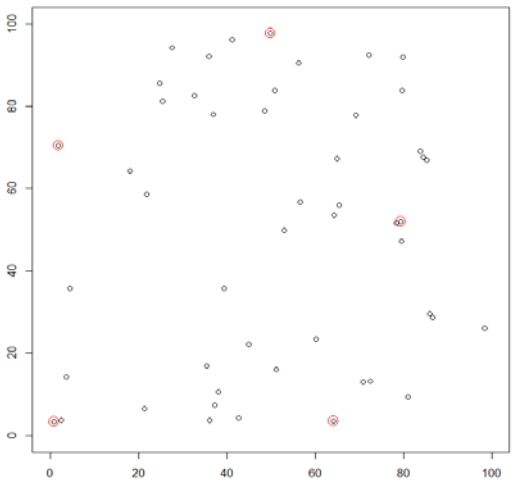

Figure 4. CMSA solution

6.2.2 Comparative testing

In our final experiment, we compare our CMSA method with the previous Tabu Search heuristic, TS (Porumbel et al., 2011), and a truncated exact method, Exact. This method first applies the binary

search by Sayyady and Fathi (2016) to obtain the MaxMin optimal solution value d^*, and then solves with CPLEX the MaxSum formulation with the additional constraint that excludes the pairs of elements with distance lower than d^*. We run CPLEX for a maximum of 3,600 seconds, which results in an early termination in the large instances, and therefore the output solution is not guaranteed to be optimal.

To analyze the results in Table 2, we focus on solution quality, as measured by average deviation from the best (DevB) and number of best solutions found (Best). The results in this table show the advantage of our CMSA over the previous TS heuristic and the truncated Exact method. In particular, CMSA exhibits a remarkable 0.49% average percentage deviation on the MaxMin and 0.17% on the MaxSum, while TS obtains 0.55% and 0.35% respectively. On the other hand, the truncated Exact method obtains 1.05% and 0.65% respectively, although it requires significantly larger running times (1616.59 seconds on average versus 218.31 for CMSA and 151.88 for TS). The numbers of best solutions found with each method are in line with these results, confirming the superiority of the CMSA over its competitors.

		MaxSum			**MaxMin**			
Method	**Instances**	**Value**	**DevB**	**Best**	**Value**	**DevB**	**Best**	**Time**
CMSA	GKD	78564.2	0.05%	231	73.77	0.35%	218	92.31
	MDG	181592.4	0.22%	65	150.32	0.07%	76	137.83
	SOM	94763.4	0.35%	106	6.61	0.96%	145	424.78
	All	**100124.0**	**0.17%**	**402**	**65.79**	**0.49%**	**439**	**218.31**
TS	GKD	78044.7	0.30%	184	73.76	0.42%	200	49.85
	MDG	180688.9	0.37%	46	150.34	0.03%	77	37.01
	SOM	94651.1	0.41%	94	6.61	1.06%	145	368.78
	All	**99665.9**	**0.35%**	**324**	**65.79**	**0.55%**	**422**	**151.88**
Exact	GKD	78527.6	0.13%	216	73.81	0.02%	249	774.01
	MDG	181487.7	0.29%	52	149.69	1.14%	48	1797.80
	SOM	94691.8	1.76%	63	6.39	2.83%	111	2277.95
	All	**100065.9**	**0.65%**	**331**	**65.64**	**1.05%**	**408**	**1616.59**

Table 2. Performance comparison

We applied the non-parametric Friedman test for multiple correlated samples to the best solutions obtained by each of the 3 methods in Table 2. This test computes, for each instance, the rank value of each method according to solution quality (where rank 3 is assigned to the best method and rank 1 to the worst one). Then, it calculates the average rank values of each method across all the instances solved. If the averages differ greatly, the associated p-value or significance will be small. We apply this test for the two objectives in the resulting solutions of the 3 methods. The p-value of 0.00 of the Friedman test clearly establish that, in general terms, the algorithms produce different solutions, and thus CMSA obtains better results than the previous methods.

6.3 The Extended MinDiff

In this section we first perform an analysis to disclose the contribution of the different elements of our heuristic, and to set the appropriate values for its parameters for the MinDiff problem. Then, in the second subsection, we compare the best variant of our method with the previous GRASP for the MinDiff problem.

6.3.1 Scientific testing

Our first experiment with the MinDiff problem undertakes to choose the best values of the parameters of the algorithms. For the MinDiff GRASP introduced in Duarte et al. (2015), we selected the constructive method C2 and the local search LS2 with $\alpha_2 = 0.5$, as suggested by the authors. For the MaxMin GRASP, we selected $\alpha_3 = 0.9$ as proposed by Resende et al (2010). In order to calibrate β for our improved version of the GRASP (Section 5), we tried three values : $0.25, 0.5$, and 0.75:

	MinDiff			MaxMin			
β	**Value**	**DevB**	**#Best**	**Value**	**DevB**	**#Best**	**Time**
0.25	347.42	29%	29	50.27	7%	82	1632.12
0.5	310.77	21%	33	43.63	21%	35	1657.21
0.75	271.49	10%	72	40.28	26%	23	1556.82

Table 3. GRASP calibration

The results in Table 3 show that the minimum deviation is reached with $\beta = 0.75$ for the MinDiff, and with $\beta = 0.25$ for the MaxMin. Since the global objective of our extended problem is to achieve a balance between the MaxMin and the MinDiff, we select $\beta = 0.5$ since it exhibits the best tradeoff between the both objectives.

	MinDiff			MaxMin			
p	**Value**	**DevB**	**#Best**	**Value**	**DevB**	**#Best**	**Time**
5	385.42	53%	18	49.37	15%	46	537.65
10	338.24	33%	28	48.43	16%	45	1046.40
15	335.10	16%	40	51.43	14%	39	1216.39
20	319.93	7%	63	51.64	12%	44	1340.18

Table 4. Elite set size

In our second experiment, we undertake to study the elite set of the Extended GRASP, composed with the p best solutions found with the GRASP that solves the MaxMin problem. Thus, we calibrate the best value for p (i.e., how many of the good solutions found when solving the MaxMin are used to populate the elite set). Table 4 shows the results of the Extended GRASP with different values of p. As we can see in this table, there is a marginal improvement in terms of both MaxMin and MinDiff while increasing the value of p. On the other hand, lower values of p create smaller elite sets that result in lower computing times. We therefore select $p = 5$ considering that the difference between running times is more significant than the change of MaxMin and MinDiff values.

We evaluate now the contribution of each phase of our GRASP for the Extended MinDiff, EXG, to the quality of its output solution. As described in Section 5, EXG has two phases, a first one, in which we compute a set of elite MaxMin solutions, and a second one, in which we solve the MinDiff over the subset of points contained in these elite solutions. Considering that the second phase of EXG takes into account both the MaxMin and MinDiff, we want to evaluate if it is necessary to apply the first phase, or we could directly apply the second phase to the original problem. In this experiment, we compare the EXG method described in Section 5, with a simplified version, EXG2, in which only the second phase is applied.

	MinDiff			MaxMin			
Method	**Value**	**DevB**	**#Best**	**Value**	**DevB**	**#Best**	**Time**
EXG2	287.15	0%	477	39.99	12%	274	1887.42
EXG	366.26	148%	56	43.28	14%	304	628.48

Table 5. Contribution of EXG phases

Table 5 shows that EXG2 obtains very good results considering its simplicity; however, it requires very long running times. Comparing EXG and EXG2 results we can see that the selection of candidates in the elite set performed by EXG results in a significant reduction of the total running time, but also in a slight deterioration of the MinDiff final value. Both points are to be expected since in EXG we are solving the MinDiff problem over a small set of points (compared with EXG2) which clearly involves lower running times. At the same time, this reduction may result in missing some good candidate points not identified in the elite set. We select EXG as our final method considering its good balance between quality and CPU time.

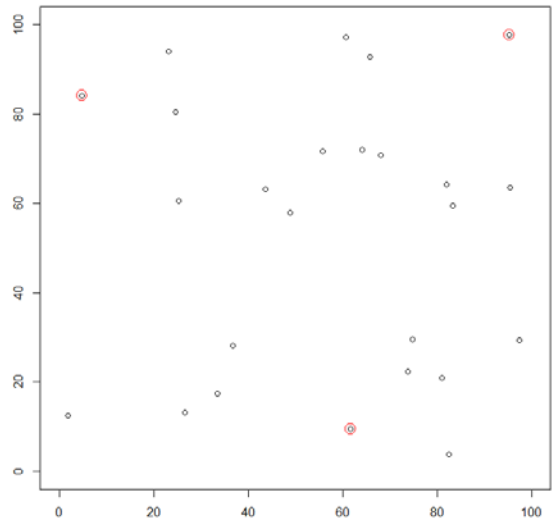

Figure 5. EXG solution

To show the structure of the solution obtained with the EXG method, we represent a Euclidean instance. Figure 5 shows the EXG solution of the instance with $n = 25$ and $m = 3$ depicted in Figure 2. As we can see in this solution, EXG selects three disperse and equidistant points, as indicated by the new MinDiff extended model, which makes more sense than the selection of three close points of the optimal original MinDiff solution represented in Figure 2.

6.3.2 Comparative testing

In this section we compare our EXG heuristic with the previous GRASP (Duarte et al., 2015). We also include in this comparison an exact method to obtain, if possible, the optimal solution. In particular, we apply a binary search to solve the MaxMin problem (Sayyady and Fathi, 2016) and collect all its optimal solutions. Then, we select the best MinDiff solution in this set of optimal MaxMin solutions. This exact algorithm returns the solution that has minimum MinDiff value in this set as its output. The average results of the three methods, EXG, GRASP, and Exact, over the MDPLIB are shown in Table 6.

		MinDiff			MaxMin			
Method	**Instances**	**Value**	**DevB**	**#Best**	**Value**	**DevB**	**#Best**	**Time**
GRASP	GKD	134.71	0%	265	47.36	56%	27	739.28
	MDG	470.12	0%	78	14.40	94%	4	116.04
	SOM	113.96	5%	125	0.21	82%	0	968.73
	All	182.63	1%	468	27.74	68%	23	708.09
Exact	GKD	849.25	763%	10	73.81	0%	265	465.68
	MDG	2139.20	355%	0	149.69	0%	80	1805.20
	SOM	421.72	207%	30	6.39	0%	150	2633.31
	All	928.17	523%	40	65.64	0%	495	1339.03
EXG	GKD	386.34	438%	13	63.57	28%	66	935.61
	MDG	676.10	45%	2	54.54	66%	0	58.49
	SOM	165.54	43%	15	1.42	67%	28	389.86
	All	366.26	253%	30	43.28	45%	94	628.48

Table 6. Performance comparison

As expected, the fact that the GRASP (Duarte et al., 2015) only targets the MinDiff problem, makes it to obtain the best values on this objective (it exhibits a 1% average deviation with respect to the best known values in this objective). On the other hand, since this algorithm does not optimize the MaxMin objective, those solutions have the worst MaxMin values (68% of average deviation). Symmetrically, the Exact algorithm is known to give optimal solutions for the MaxMin problem, and therefore it obtains the best solutions in this objective (0% of average deviation); but on the other hand, it presents very large MinDiff values (with 523% average deviation), since it is computed on a reduced set (consisting on the optimal MaxMin solutions). These results confirm that good MaxMin solutions do not usually have good MinDiff values, as also reported in Sandoya et al. (2018).

When comparing the results of our EXG method with the Exact and the GRASP, we can conclude that it provides a good balance between both objectives. On one hand, the average percentage deviation of the three methods in the MinDiff objective is: 1% (GRASP), 523% (Exact), and 253% (EXG). On the other hand, the average percentage deviation of the three methods in the MaxMin objective is: 68% (GRASP), 0% (Exact), and 45% (EXG). The average number of the best solutions found with each method confirms this pattern, in which EXG provides an in-between performance on both objectives. If we analyze the results in Table 6 on each set of instances, we do not observe important differences, and the results seem to be quite robust across the different sets.

We finally apply in this experiment the Friedman test. The resulting p-value of 0.000 obtained for the MinDiff in this experiment clearly indicates that there are statistically significant differences among the 3 methods tested. When we apply this test to the MaxMin values, we also obtain a p-value of 0.000.

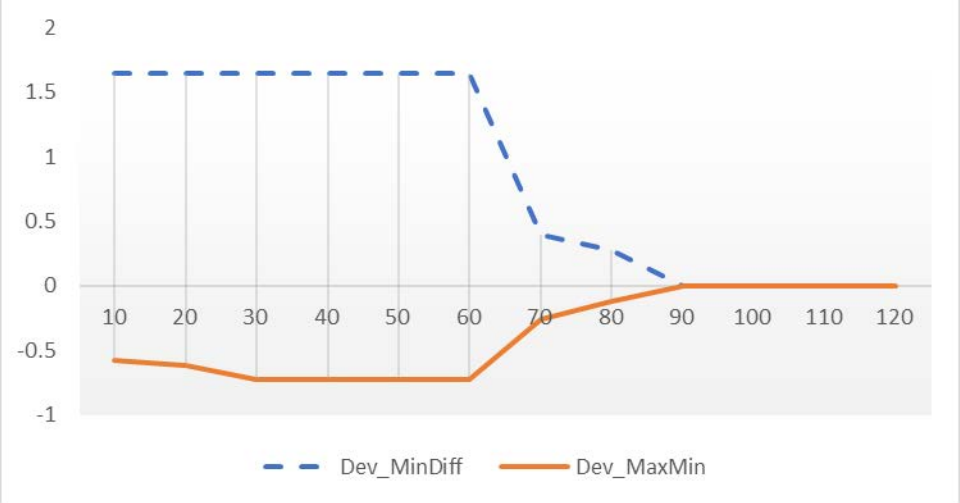

Figure 6. EXG search profile

Our final experiment has the goal of showing how the best solution obtained with GRASP evolves over time. Considering that we have two objectives, we represent in Figure 6 a line with each one. In this diagram the x-axis represents the global iterations, and the y-axis depicts the percentage deviations with respect to the best known values. It is easy to identify in this figure that the in first iterations our heuristic applies an improvement procedure with respect to the MaxMin, and at a certain point (iteration 60) it applies an improvement GRASP for the MinDiff. This clearly explains the evolution pattern of both objectives. Note that in the first 60 iterations, the method is able to improve the MaxMin significantly, and when the second part of the heuristic is applied, then it deteriorates this value to favor the MinDiff (note that deviations are computed with respect to the best final value, and this is why in the first iterations the MaxMin objective has a negative deviation). As a matter of fact, this second phase is able to reduce the MinDiff to its final deviation of 0.00%.

7. Conclusions

We had a twofold goal for this work, to propose a extended model for two discrete location problems, diversity and equity maximization and, to develop state-of-the-art procedures for the two associated models. We believe that we have achieved the first goal with the two extended designs. The merit of these models is that they overcome the important issues recently identified for the standard MaxSum and MinDiff models. The extended models incorporate the well-known MaxMin objective as a sub-problem to guarantee that the selected points are not too close. Although this adds an extra complexity to the standard models, our empirical analysis reveals that they can be efficiently solved with heuristic methods.

In terms of our second goal, the results reported in Tables 2 and 6 are very strong in favor of our proposals for the two extended models. On one hand, our adaptation of GRASP, EXG, to the extended MinDiff (equity model) improves upon the previous GRASP for the standard MinDiff. On the other hand, the CMSA matheuristic that combines a previous tabu search with an exact method, shows that this clever combination of both technologies outperforms them when targeting the Extended MaxSum (diversity model). We have established benchmarks for the instances of the well-known MDPLIB, and

we believe that they will help other researchers test additional search strategies on these interesting combinatorial optimization problems.

Acknowledgement

This research has been partially supported by the Spanish Ministry of Science, Innovation, and Universities with grant refs. PGC2018-0953322-B-C21/MCIU/AEI/FEDER-UE and RTI2018-094940-B-100, and the Junta de Comunidades de Castilla-La Mancha project SBPLY/17/180501/0 0 0282, partially financed with FEDER-UE funds. The authors thank Prof. Jin-Kao Hao and his team for making their TS code available.

References

Blum, C., Pinacho, P., López-Ibañez, M., and Lozano, J.A. "Construct, Merge, Solve, & Adapt: A New General Algorithm for Combinatorial Optimization" Computers and Operations Research **68**, 75-88, 2016.

Duarte, A., Sánchez-Oro, J., Resende, M., Glover, F., Martí, R. "GRASP with Exterior Path Relinking for Differential Dispersion Minimization", Information Sciences **296**, 46-60, 2015.

Erkut, E., Neuman, S. "Analytical models for locating undesirable facilities", *European Journal of Operational Research* **40,** 275–291, 1989.

Feo, T.A. and Resende, M.G.C. "A probabilistic heuristic for a computationally difficult set covering problem", *Operations Research Letters* **8**(2), 67-71, 1989.

Glover, F., Campos, V. Martí, R. "Tabu search tutorial. A Graph Drawing Application", TOP **29**, 319–350, 2021.

Kuby, M. J.; "Programming models for facility dispersion: the p-dispersion and maxisum dispersion problems", Mathematical and Computer Modelling **10**, 792, 1988.

Martí, R., Gallego, M., Duarte, A., Pardo, E. G. "Heuristics and metaheuristics for the maximum diversity problem", Journal of Heuristics **19**, 591-615, 2013.

Martí, R., Martínez-Gavara, A., Pérez-Peló, S., Sánchez-Oro, J. A review on discrete diversity and dispersión maximization from an OR perspective, European Journal of Operational Research **299**, 795-813. 2022

Martínez-Gavara, A., Corberán, T., Martí, R.; "GRASP and Tabu Search for the Generalized Dispersion Problem", Expert Systems with Applications **173**, 114703. 2021

Parreño, F., Álvarez-Valdés, R., Martí, R.; "Measuring diversity. a review and an empirical analysis", European Journal of Operational Research **289**, 515–532. 2021.

Porumbel, D. C., Hao, J. K., Glover, F. "A simple and effective algorithm for theMaxMin diversity problem", Annals of Operations Research **186**, 275–293, 2011.

Prokopyev, O. A., Kong, N., & Martinez-Torres, D. L. "The equitable dispersion problem". European Journal of Operational Research **197**, 59-67, 2009.

Resende, M. G. C., Martí, R. Gallego, M., Duarte, A.; "GRASP and path relinking for the max–min diversity problem". *Computers and Operations Research*, **37**(3): 498-508, 2010.

Resende, M.G.C. and Werneck, R.F.; "A hybrid heuristic for the p-median problem", *Journal of Heuristics*, **10**(1):59-88, 2004.

Sandoya, F., Aceves, R., Martínez-Gavara, A., Duarte, A., Martí, R. Diversity and Equity Models, Handbook of Heuristics, Martí, Resende, Pardalos (Eds.), 979-998 Springer, 2018.

Sayyady, F., Fathi, Y.; "An integer programming approach for solving the p-dispersion problem". *European Journal of Operational Research*, 253: 216-225, 2016.

Teitz, M.B.; Toward a theory of public facility location, Papers in regional science 21, 35-51, 1968.