Heuristic Solution Approaches for the Maximum MinSum Dispersion Problem

ANNA MARTÍNEZ-GAVARA
Departamento de Estadística e Investigación Operativa, Universidad de Valencia, Spain
gavara@uv.es

VICENTE CAMPOS
Departamento de Estadística e Investigación Operativa, Universidad de Valencia, Spain
Vicente.Campos@uv.es

MANUEL LAGUNA
Leeds School of Business, University of Colorado at Boulder, USA
laguna@colorado.edu

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

ABSTRACT

The Maximum Minsum Dispersion Problem (*Max-Minsum DP*) is a strongly NP-Hard problem that belongs to the family of equitable dispersion problems. When dealing with dispersion, the operations research literature has focused on optimizing efficiency-based objectives while neglecting, for the most part, measures of equity. The most common efficiency-based functions are the sum of the inter-element distances or the minimum inter-element distance. Equitable dispersion problems, on the other hand, attempt to address the balance between efficiency and equity when selecting a subset of elements from a larger set. The objective of the Max-Minsum DP is to maximize the minimum aggregate dispersion among the chosen elements. We develop tabu search and GRASP solution procedures for this problem and compare them against the best in the literature. We also apply LocalSolver, a commercially available black-box optimizer, to compare our results. Our computational experiments show that we are able to establish new benchmarks in the solution of the Max-Minsum DP.

Keywords: Equitable dispersion problems, tabu search, GRASP, metaheuristics.

Version: October 10, 2014

1. Introduction

The problem of maximizing diversity deals with selecting a subset of elements from a given set in such a way that the diversity among the elements is maximized (Glover et al., 1995). Several models have been proposed to deal with this combinatorial optimization problem. All of them require a diversity measure, typically a distance function in the space where the objects belong. The definition of this distance between elements is customized to specific applications. As described in (Glover, 1998), these models have applications in plant breeding, social problems, ecological preservation, pollution control, product design, capital investment, workforce management, curriculum design, and genetic engineering.

Given a graph $G = (V, E)$, where V is the set of n nodes and E is the set of edges, let d_{ij} be the inter-element distance between any two elements i and j, let $M \subseteq V$ be the set of m selected elements, and define $U = V \backslash M$ as the set of unselected elements. Then, the Maximum Minsum Dispersion Problem (*Max-Minsum DP*) consists of selecting a set $M \subseteq V$ of m elements such that the smallest total dispersion associated with each selected element i is maximized. The problem is formulated in Prokopyev et al. (2009) as follows:

$$\text{Maximize} \quad \{\min_{i:x_i=1} \textstyle\sum_{j:j\neq i} d_{ij}x_j\} \tag{1}$$

$$\text{subject to} \quad \textstyle\sum_{i=1}^{n} x_i = m \tag{2}$$

$$x_i \in \{0,1\} \qquad i = 1, \dots, n$$

The set of the m selected elements is $M = \{i: x_i = 1\}$ and the objective function $f(M)$ is based on measuring the total dispersion associated with each $i \in M$, denoted by $c(M, i)$. In other words, $c(M, i) = \sum_{j\in M, j\neq i} d_{ij}$ and the objective is to maximize its minimum value by a judicious selection of M.

The contributions of our work can be summarized as follows:

- Implementation of GRASP variants for the *Max-Minsum DP*
- Development and implementation of a tabu search approach for the *Max-Minsum DP*
- Exploration of hybrid approaches that combine GRASP and TS
- Comparison of new and existing methods on general instances for the *Max-MinSum DP*

We start by exploring the existing *Max-Minsum DP* literature.

2. Existing Solution Approaches

The two most popular dispersion problems in the literature consist of maximizing the sum of the diversity (*Maxsum*) and maximizing the minimum diversity (*Maxmin*) in M. The *Maxsum* is characterized by the maximization of $\sum_{i,j\in M, i<j} d_{ij}$ while the *Maxmin* is characterized by the maximization of $\min_{i,j\in M} d_{ij}$. Clearly, both of these objectives are based on measures of efficiency. As pointed out by Prokopyev et al. (2009):

> "The maximum dispersion problem primarily focuses on operational efficiency of locating facilities according to distance, accessibility, impacts, etc. It also arises in various other contexts including maximally diverse/similar group selection (e.g., biological diversity, admissions policy formulation, committee formation, curriculum design, market planning, etc.), and densest subgraph identification."

The *Maxsum* and *Maxmin* literature includes extensive surveys (Ağca, Eksioglu, Ghosh, 2000; Erkut, Neuman, 1989; Kuo, Glover, Dhir, 1993), exact methods (Ağca, Eksioglu, Ghosh, 2000; Ghosh, 1996; Pisinger, 2006), and heuristics (Ghosh, 1996; Hassin, Rubinstein, Tamir, 1997; Kincard, 1992; Ravi, Rosenkrantz, Tayi, 1994; Resende et al., 2010).

Measures of equity in dispersion problems are the counterpart of measures of efficiency. Prokopyev et al. (2009) introduced several models to describe various aspects of the equitable dispersion problem. In particular, they introduced mathematical programming formulations for the following dispersion problems:

Max-Mean — the maximization of the average dispersion (i.e., $\frac{\sum_{i>j} d_{ij} x_i x_j}{\sum_i x_i}$)
Max-Minsum — the maximization of the minimum $c(M, i)$
Min-Diff — the minimization of the differential dispersion (i.e., $\max_{i \in M} \sum_{j:j \neq i} d_{ij} x_j - \min_{i \in M} \sum_{j:j \neq i} d_{ij} x_j$)

The computational experiments conducted by Prokopyev et al. (2009) include the exact solution (via CPLEX) of small instances of the *Maxsum*, *Maxmin*, *Max-Minsum*, and *Min-Diff* dispersion problems. They also include results of applying GRASP to the *Max-Minsum DP*.

In their GRASP implementation, Prokopyev et al. (2009) define M_k as a partial solution with k selected elements ($1 \leq k < m$). Each construction phase of GRASP starts by randomly selecting an element in order to initialize M_1. Then, in each iteration, the method builds a candidate list CL that consists of all the unassigned elements, i.e., $CL = V \backslash M_k$. For each element i in CL, the method computes a marginal contribution of the element toward the objective function associated with M_{k+1}:

$$\Delta f^k(i) = \min\left\{\min_{j \in M_k}\{ s^k(j) + d_{ij}\}, s^k(i)\right\} - f(M_k) \tag{3}$$

where $f(M_k)$ is the value of the objective function corresponding to the partial solution M_k and

$$s^k(i) = \sum_{j \in M_k} d_{ij}. \tag{4}$$

Elements in CL are ordered according to the marginal contributions, that is from largest to smallest Δf^k values. Then, a reduced candidate list RCL is constructed with the top α elements in CL. The value of α is selected at random, in each construction step, from a uniform distribution with parameters 1 and $|CL|$. The element to be included in the partial solution M_{k+1} is randomly chosen from RCL. The construction procedure stops after $m - 1$ steps.

An improvement phase is executed after a solution M has been constructed. The improvement phase consists of an exchange mechanism in which an element i in M is replaced with an element j in $V\backslash M$. The method randomly selects both elements and exchanges them if and only if the objective function value of the resulting solution improves; otherwise, the (i, j) selection is discarded. The improvement phase finishes after 100 iterations without any improvement. We will refer to this procedure as *Prokopyev*.

3. Proposed Solution Methods for the *Max-MinSum DP*

Our main goal is the development and implementation of a procedure to obtain high quality solutions for the *Max-MinSum DP*. In particular, we explore a new application of GRASP and the first adaptation of tabu search to this diversity problem.

3.1 Basic GRASP

Typically in GRASP, a construction consists of evaluating each candidate element with a greedy function in order to identify the best elements and add them in the so-called Restricted Candidate List (RCL). Then the next element to be included in the partial solution is chosen from RCL (see e.g., Resende et al., 2001). However, Resende et al. (2004) describe an alternative design in which the RCL is constructed totally at random from elements in the candidate list (CL). Therefore, RCL can be thought of as a random sample of CL of a pre-established size. Then, the greedy function is applied to evaluate all the elements in RCL in order to choose the best. In this alternative design, the sequence of applying randomness followed by greediness is inverted. Martí and Sandoya (2013) employed this design in a GRASP for the *Max-Mean DP*.

We too use this design in the construction phase of our basic GRASP implementation, which we formalize as follows. Given a partial solution M_k and a corresponding $CL = \{i: i \in V\backslash M_k\}$, RCL consists of $\alpha\%$ of the elements of CL chosen at random. The element in RCL to be included in M_{k+1} is the one that maximizes Δf^k. The value of α is fixed throughout the entire construction process. We refer to this construction method as CM1.

For the improvement phase, we consider two different designs. Both designs are pure local searches in the sense that they terminate when the exploration of the entire neighborhood of the current solution does not yield a move that improves the objective function value. Given a solution M, the IM1 improvement method consists of the exhaustive exploration of the neighborhood defined by all (i, j) exchanges, where $i \in M$ and $j \in U$. An exchange results in a neighbor solution $M' = M\backslash\{i\} \cup \{j\}$. Let M^* be the neighbor solution M' with the best objective function value. Then if $f(M^*) > f(M)$ then the search moves to M^* and the new neighborhood is explored. Otherwise, the improvement method ends.

Instead of exploring all possible exchanges, our second design (IM2) considers the contribution of the selected elements as well as the potential contribution of the unselected elements to create a priority list. For this purpose, we define:

$$s^m(i) = \sum_{j \in M} d_{ij} \tag{5}$$

The elements j in U are ordered in descending s^m values while the elements i in M are ordered in ascending s^m values. That is, the first unselected element in the ordered list has the largest potential contribution while the first selected element in the ordered list has the smallest current contribution. The priority list of (i, j) exchanges is built by scanning both ordered lists and paring each unselected element with a selected element. The order of exchanges in the priority list is important because IM2 employs the first-improving strategy in which the search makes an immediate transition from the current solution M to the neighbor solution M' when $f(M') > f(M)$. In other words, the search does not explore all possible exchanges in order to determine the best one (according to the change in the objective function value). In contrast, under the best improving strategy (as done in IM1), the order in which the exchanges are considered is not important since all of the exchanges will be evaluated before selecting the best one. In our implementation, we do not update the priority list every time an exchange is made. Instead, we finish exploring the list and update it if an exchange has been made. The IM2 search ends when no improving exchange is identified, that is, when the entire priority list has been scanned and no improving exchange has been found.

3.2 Additional Construction Strategies

The RCL in CM1 is built by randomly selecting $\alpha\%$ of CL. We refine this process by narrowing down the choices with a threshold value τ, which is calculated as follows:

$$\tau = \min_{i \in M_k} s^k(i) + \alpha \left(\max_{i \in M_k} s^k(i) - \min_{i \in M_k} s^k(i) \right) \tag{6}$$

As before, α is a search parameter that has a fixed value throughout the search. The RCL consists of all the elements in CL with s^k values that meet or exceed the established threshold:

$$RCL = \{i \in CL : s^k(i) \geq \tau\} \tag{7}$$

As indicated by (7), RCL is not chosen at random like in CM1. In this case, RCL consists of elements that according to the greedy function achieve some minimum potential contribution to the objective function value, as established by (6). In order to start the construction process and choose an element for M_1, we define:

$$s^0(i) = \sum_{j \in V} d_{ij} \tag{8}$$

After applying a greedy criterion to select the elements in RCL, the selection of the next element to be included in M is random from those elements in RCL. In other words, we are back to the more classical GRASP design in which greediness is followed by randomness. The process continues until $|M| = m$. We refer to this construction method as CM2.

Instead of constructing solutions by adding elements to a partial solution too few elements, a feasible solution can be found by deleting elements from an infeasible solution with too many elements. This is the basis of our next construction method (CM3). The procedure starts with $M_k = V$, that is, it starts with all the elements being selected. Then, at each step of the process, one element is chosen to be removed

from M_k. This is done until $|M_k| = m$. Making $CL = M_k$, the RCL for this construction method is given by:

$$RCL = \{i \in CL: s^k(i) \leq \tau\} \tag{9}$$

An element is selected at random from RCL and it is eliminated from M_k. In the context of dispersion problems, the idea of constructing a feasible solution by eliminating elements from a solution that has too many elements was first proposed by Erkut (1990). A summary of our adaptation of this idea in the framework of a GRASP construction is shown in Algorithm 1.

1. Let $M_k = V$ be the initial solution, and $k = n$, the size of V
2. Let m be the number of elements to select from V

while $(k > m)$ {

   a. Make $CL = M_k$
   b. Compute $s^k(i)$ for all $i \in CL$
   c. Compute $\tau = \min_{i \in M_k} s^k(i) + \alpha \left(\max_{i \in M_k} s^k(i) - \min_{i \in M_k} s^k(i) \right)$
   d. Construct $RCL = \{i \in CL: s^k(i) \leq \tau\}$
   e. Randomly select an element $i^* \in RCL$
   f. $M_{k+1} = M_k \backslash \{i^*\}$
   g. $k = k - 1$.

}

Algorithm 1. GRASP construction phase for CM3.

We create variants of CM2 and CM3 with a modified greedy function based on a minimum distance instead of the sum of the distances. Specifically, we redefine (4) as follows:

$$s^k(i) = \min_{j \in M_k} d_{ij} \tag{10}$$

We refer to CM4 as the construction method that uses (10) instead of (4) within the framework of CM2. Likewise, we refer to CM5 as the construction method that uses (10) instead of (4) within the framework of CM3.

3.3 GRASP with Strategic Oscillation

Glover (1977) introduced the notion of strategic oscillation (SO), which consists of orienting moves (or exchanges) in relation to a critical boundary, as identified by a stage of construction or a chosen interval of functional values. As summarized in Glover et al. (1997), this critical boundary identifies regions of the search space that are expected to contain solutions of particular interest. Our main research inquiry is to apply SO in the context of a GRASP framework customized for the *Max-MinSum DP*. We focus on the construction phase of GRASP and oscillate around the feasibility boundary defined by constraint (2). The constraint indicates that a feasible solution must have m elements. Therefore, the boundary around which we define our SO separates infeasible M_k solutions into those with $k < m$ and those with $k > m$.

As described in Glover and Laguna (1997), the following decisions must be made in order to implement strategic oscillation:

Macro level decisions

1. Select an oscillation guidance function
2. Choose a target level for the function
3. Choose a pattern of oscillation

Micro level decisions

4. Choose a target rate of change (for moving toward or away from the target level)
5. Choose an oscillation amplitude
6. Identify aspiration criteria to override target restrictions

In our case, the guiding function is simply k, the number of elements in the current solution, and its target level is m. We will test three oscillation patterns: single-sided for $k > m$, single-sided for $k < m$, and double-sided. That is, given a feasible solution M, we create an infeasible solution M_k by either adding $k - m$ elements to M or removing $m - k$ elements from M. In both cases, we will use a constant amplitude of oscillation. In terms of the micro level decisions, we choose a rate of change of 1 unit. That is, at each step the oscillation moves one element closer to the target level. The amplitude (given in terms of number of elements) is controlled by the search parameter β. This parameter is a fraction between 0 and 1 and it is multiplied by m to determine the number of elements to be removed from or added to a feasible solution. For small β values, in both cases, the strategic oscillation amplitude is set to at least 1 element. In other words, the oscillation amplitude is given by $\max(1, \beta m)$. We use no aspiration criteria to override any of these choices. Figure 1 shows a graphical representation of the three oscillation patterns that we tested.

The strategic oscillation pattern in Figure 1.a starts from a feasible solution M to which βm elements have been randomly added. Then, CM3 or CM5 can be applied until the target level is reached. Process is repeated by once again randomly adding βm elements to the current feasible solution. The mirror image of this pattern is shown in Figure 1.b. In this case, βm elements are removed from a feasible solution with m elements and either CM2 or CM4 can be applied to restore feasibility (i.e., to reach the target level). The third graph, Figure 3.c, shows the double-sided oscillation pattern. This pattern starts from a feasible solution M to which βm elements are added, one at a time, following the criteria in CM2 or CM4. Once the peak amplitude above the target level is reached, CM3 or CM5 is applied to delete $2\beta m$ elements and reach the peak amplitude below the target level. Note that the double-sided oscillation pattern visit a feasible solution (i.e., crosses the target level) every $2\beta m$ steps while the single-sided oscillation patterns reach the target level every βm. Hence, the single-sided oscillation patterns may be seen as intensification mechanisms while the two-sided oscillation pattern favors search diversification.

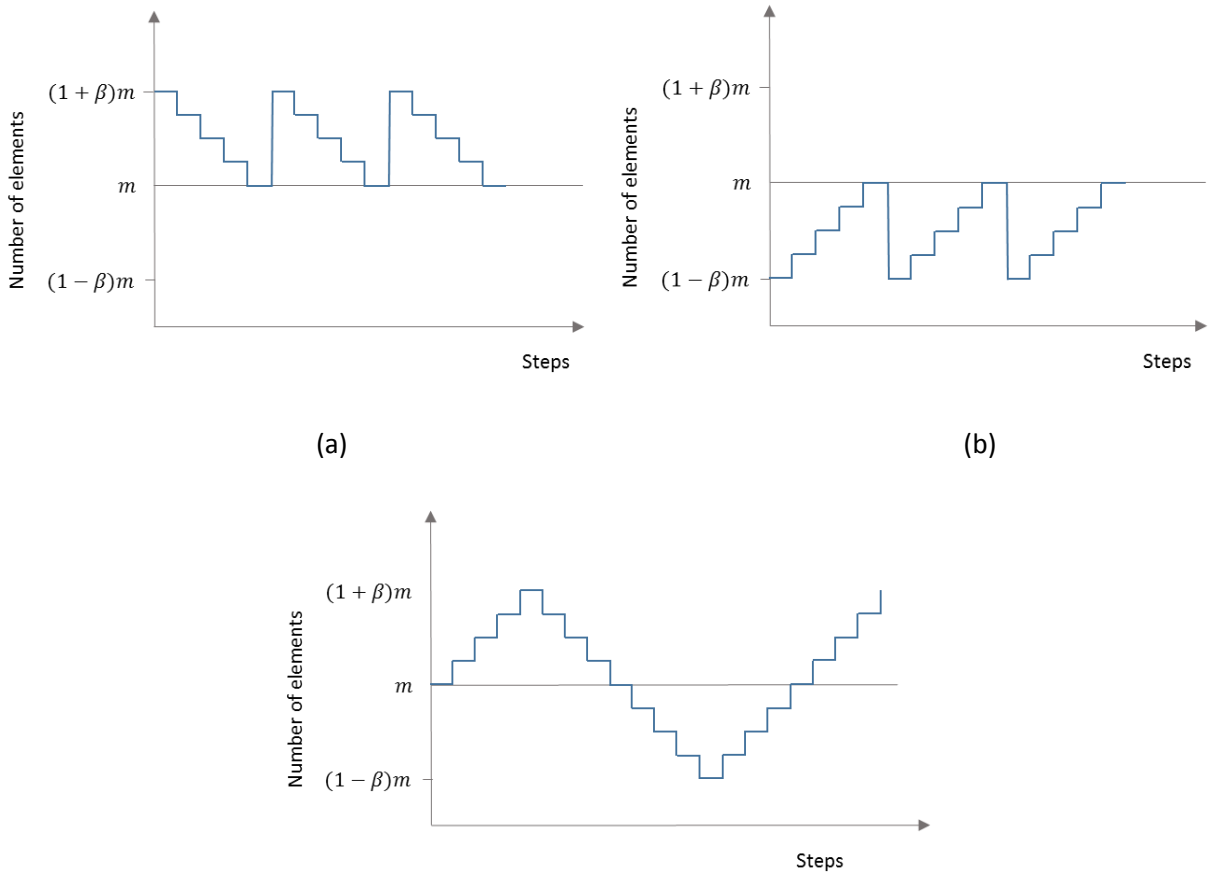

Figure 1. Strategic oscillation patterns.

Algorithm 2 shows the procedural steps associated with implementing the one-sided oscillation pattern depicted in Figure 1.a

1. Let M be an initial solution and $M^* = M$ the best solution
2. Set $l = 0$ and $U = V \backslash M$

while $(l < \gamma n)$ {

   a. Let R be βm randomly selected elements in U
   b. Obtain M' by applying CM3 or CM5 to $M \cup R$ for βm steps
   c. Make $M = M'$ and $U = V \setminus M$
   d. If $f(M) > f(M^*)$ then $M^* = M$ and $l = 0$, else $l = l + 1$.

}

Algorithm 2. Single-sided oscillation strategy with CM3 or CM5

Unlike pure local search, strategic oscillation does not have a natural termination. Therefore, as shown in Algorithm 2, we stop the SO process after a number of cycles without improvement (controlled by the

search γ parameter). We define an SO cycle, in a single-sided oscillation pattern, as the pair consisting of the infeasibility step (i.e., adding to or removing elements from M) and the restoration step (i.e., applying an appropriate construction method to restore feasibility). The stopping limit is an additional search parameter.

3.4 Tabu Search

To complement the GRASP variants described above, we developed a specialized tabu search for the *Max-MinSum DP*. The starting point of the search is an initial solution constructed in the same way as suggested by Prokopyev et al. (2009). The neighborhood search around the current solution is the same one used in the improvement methods IM1 and IM2. That is, the search attempts to identify an exchange of (i, j) for which $i \in M$ and $j \in V \backslash M$. In the same way as IM1 and IM2, we create two TS variants, one for which the best exchange is identified by exploring the entire neighborhood (TS1) and a second one in which a priority list is used to evaluate the exchanges and select one as soon as an improvement of the current objective function value is identified (TS2). The tabu memory structure after executing an (i, j) exchange consists of recording the index j, that is, the index of the unselected element that became part of the solution. The element becomes tabu-active and remains in such a status for a specified number of iterations controlled by the search parameter θ. Exchanges that include tabu-active elements are declared tabu and therefore are not allowed. As customary, the tabu status of an exchange is overridden if the exchange leads to a solution that is better than the incumbent. The search stops after a number of iterations without improvement (controlled by the search parameter γ).

In our computational experiment, we test this simple TS implementation as a standalone procedure as well as an improvement method within GRASP.

4. Computational Experiments

The computational experiments described in this section were performed to test the effectiveness and efficiency of the procedures discussed above. All procedures were implemented in C and the experiments were conducted on a computer equipment with a 2.8 Ghz Intel Core i7 processor.

We employed 255 instances in our experimentation. The group of instances, referred to as **MDPLIB**, is available at http://www.optsicom.es/mdp and contains three sets. The **SOM** set consists of 70 matrices with random numbers between 0 and 9 generated from an integer uniform distribution. The **GKD** set consists of 145 matrices for which the values were calculated as the Euclidean distances from randomly generated points with coordinates in the 0 to 10 range. Finally, the **MDG** set consists of 100 matrices with real numbers randomly selected between 0 and 10 from a uniform distribution. Our experiments do not include instances with $n \geq 2000$, which result in excessively long runs. Details of these sets are provided in Table 1.

Set	Reference	Number of instances	Size
SOM-a	Generated by Martí et al. (2010) with a generator developed by Silva et al. (2004).	50	$n = 25, m = 2,7$ $n = 50, m = 5,15$ $n = 100, m = 10,30$ $n = 125, m = 12,37$ $n = 150, m = 15,45$
SOM-b	Generated by Silva et al. (2004) and used by many, including Aringhieri et al. (2008).	20	$n = 100, m = 10,20,30,40$ $n = 200, m = 20,40,60,80$ $n = 300, m = 30,60,90,120$ $n = 400, m = 40,80,120,160$ $n = 500, m = 50,100,150,200$
GKD-a	Glover et al. (1998)	75	$n = 10, m = 2,3,4,6,8$ $n = 15, m = 3,4,6,9,12$ $n = 30, m = 6,9,12,18,24$
GKD-b	Martí et al. (2010)	50	$n = 25, m = 2,7$ $n = 50, m = 5,15$ $n = 100, m = 10,30$ $n = 125, m = 12,37$ $n = 150, m = 15,45$
GKD-c	Duarte and Martí (2007)	20	$n = 500, m = 50$
MDG-a	Developed by Duarte and Martí (2007) and used by Palubeckis (2007)	20	$n = 500, m = 50$
MDG-b	Developed by Duarte and Martí (2007) and used by Palubeckis (2007) and Gallego et al. (2009)	20	$n = 500, m = 50$

Table 1. Characteristics of the data sets.

A series of preliminary experiments were conducted to determine effective values of the key search parameters shown in Table 2. All of the parameters operate in the range $]0, 1]$.

Parameter	Description
$\boldsymbol{\alpha}$	Used by all GRASP constructions methods (i.e., CM1 to CM5) to trade off randomization and greediness.
$\boldsymbol{\beta}$	Percentage of the elements in a feasible solution (i.e., m) that are added to or removed from a solution in SO. This parameter defines the amplitude of the oscillation, given by $\max(1, \beta m)$.
$\boldsymbol{\gamma}$	Controls the termination criterion for either SO or TS. The number of cycles or iterations without improvement is given by $\max(1, \gamma n)$.
$\boldsymbol{\theta}$	Controls the number of iterations that a selected element remains tabu-active in TS. The number of iterations is given by $\max(1, \theta m)$

Table 2. Description of the key search parameters.

For these preliminary experiments, we employ a training set of 20 representative instances from MDPLIB. Our test set consists of all 255 problems in MDPLIB. The GRASP that we tested along with the best parameter values are shown in Table 3. The SSO search strategy refers to the single-sided strategic oscillation. The SSO for GRASP3 and GRASP5 corresponds to the oscillation pattern in Figure 1.a, while the SSO for GRASP2 and GRASP4 corresponds to the oscillation pattern in Figure 1.b. The table shows the best parameter values identified in the set $\{0.1,0.3,0.5,0.7\}$. We utilized a sequential design in which we set the value of each a parameter one at a time, while the others are kept constant. Each run consisted of 100 GRASP iterations, where an iteration includes both phases; construction and improvement.

Procedure	Search strategies	Best parameter values
GRASP0	CM1, IM1	$\alpha = 0.5$
GRASP1	CM1, IM2	$\alpha = 0.3$
GRASP2	CM2, IM2, SSO	$\alpha = 0.7, \beta = 0.5, \gamma = 0.5$
GRASP3	CM3, IM2, SSO	$\alpha = 0.1, \beta = 0.1, \gamma = 0.3$
GRASP4	CM4, IM2, SSO	$\alpha = 0.7, \beta = 0.5, \gamma = 0.5$
GRASP5	CM5, IM2, SSO	$\alpha = 0.3, \beta = 0.1, \gamma = 0.3$

Table 3. GRASP variants and best parameter values.

We use the following metrics to measure the merit of each procedure:

Dev Average percent deviation from the best-known objective function values. These deviations are calculated against the best solution that is known for each problem in the data set and therefore can be compared across different experiments.

$Best$ Fraction of instances in a set for which a procedure is able to match the best-known solution. Since this fraction is calculated against the best-known solutions, the performance metric is absolute and can be compared across tables.

$Score$ Fraction of the instances for which the competing procedures "win" (i.e., they produce better solutions than the procedure being scored). It is calculated as $(q(p-1)-r)/(q(p-1))$, where p is the number of procedures being compared, q is the number of instances, and r is the number of instances in which the $p-1$ competing procedures find a better result. Hence, the best score is 1 (when $r = 0$) and the worst is 0 ($r = q(p-1)$). This is a relative measure of performance that is only meaningful within a table of results and not across tables.

Table 4 shows a comparison of the GRASP configurations when applied to the training set. The termination criterion was set to 5 seconds for the small instances, and 15 seconds for the large instances. The results in Table 4 indicate that GRASP3 seems to be the best configuration to tackle instances of the *Max-MinSum DP*. In the training set, this procedure obtains the smallest average deviation value and the highest score. It also yields the highest fraction of best solutions (i.e., 0.25), although none of the methods in this experiment performs particularly well on this metric.

Procedure	Dev	$Best$	$Score$	$Time$
GRASP0	1.83%	0.20	0.50	11.05
GRASP1	1.56%	0.20	0.74	11.05
GRASP2	1.14%	0.30	0.78	11.00
GRASP3	0.18%	0.40	0.95	11.00
GRASP4	1.56%	0.20	0.74	11.05
GRASP5	1.84%	0.20	0.46	11.00

Table 4. Performance comparison of GRASP variants.

In the experiment reported in Table 4, we limited the application of the improvement methods to the solution that results after the SSO stops. A variant of this results when the improvement method is applied every time the target level is reached during SSO. We focus on GRASP3 in order to test this idea and refer to the resulting method as GRASP3.1. Table 5 shows the results of comparing GRASP3 with its GRASP3.1 variant.

Procedure	Dev	$Best$	$Score$	$Time$
GRASP3	0.18%	0.40	0.90	11.00
GRASP3.1	0.23%	0.45	0.75	15.20

Table 5. GRASP3 vs. GRASP3.1 (multiple applications of the improvement method).

The Dev and $Score$ values in Table 5 do not favor the idea of multiple application of the improvement method. These multiple applications of the improvement method within a single GRASP iteration increase the time that the procedure spends intensifying the search within some particular regions of the solution space (i.e., those established by the constructed solutions). This extra time precludes the method from exploring other regions because the number of constructions decreases, resulting in a tradeoff that seems somewhat detrimental to the performance of the entire procedure. We note however that the method is able to find a higher fraction of best-known solutions.

In our next experiment, we once again focus on GRASP3 and replace SSO with the double-sided strategic oscillation pattern shown in Figure 1.c. We refer to this method as GRASP 3.2 and compare it with the original GRASP3 using the training set. The results are shown in Table 6.

Procedure	Dev	$Best$	$Score$	$Time$
GRASP3	0.18%	0.40	0.55	11.00
GRASP3.2	0.05%	0.75	1.00	11.00

Table 6. GRASP3 vs. GRASP3.2 (double-sided strategic oscillation).

The double-sided strategic oscillation pattern seems like an effective search strategy within the GRASP framework for the *Max-MinSum DP.* Given these results, we declare GRASP3.2 our best GRASP variant to be tested against other competing procedures. Before comparing it to methods in the literature, we perform two additional tests. The first one consists of comparing GRASP3.2 and the simple TS procedure described in section 3.4. We do not show results of this comparison because GRASP3.2 turns out to be

widely superior to the simple TS implementation. We then tried a final variant in which TS replaces IM2 within GRASP3.2. That is, we have CM3 with a double-sided strategic oscillation for the construction phase and TS for the improvement phase. This configuration turns out to be also inferior (by a wide margin) to GRASP 3.2.

These results and those obtained when testing GRASP3.1 point to the conjecture that the better strategy to search the solution space of *Max-MinSum DP* instances is to favor increasing the number of constructions with limited improvement over fewer constructions with additional intensification around those solutions.

We now compare the performance of GRASP3.2 with *Prokopyev*, Cplex, and LocalSolver. The Cplex runs correspond to the MIP formulation of the *Max-MinSum DP* suggested by Prokopyev et al. (2009), which is derived from (1) and (2). The formulation consists of a single continuous variable (that is also the objective function to be maximized), n binary variables, n "Big-M" constraints, and constraint (2). Cplex 12.6 was configured to stop after 3600 seconds.

LocalSolver is a commercially available optimization software for combinatorial problems (localsolver.com). In order to apply LocalSolver to the *Max-MinSum DP*, we define `x[i]` variables within the LocalSolver model and use them to calculate `dsum[i]` for each element. We then identify the minimum `dsum` value for those elements that are selected (i.e., for `i` such that `x[i]` equals 1). The minimum value is then maximized. The LocalSolver model is shown in Algorithm 3.

```
function model()
{
	// x[i] is 1 if element i is chosen
	x[0..n] <- bool();

	// choose m elements
	constraint sum[i in 0..nodes] (x[i]) == m;

	//calculate the sum of distances for each chosen element
	dsum[i in 0..n] <- 0;
	for [i in 0..n]
		dsum[i] <- sum[j in 0..n : j != i] (d[i][j] * x[i] * x[j]);

	// find the minimum sum
	minsum <- 100000000;
	for [i in 0..n]
		minsum <- (x[i] == 1 && dsum[i] < minsum) ? dsum[i] : minsum;
	maximize minsum;
}
```

Algorithm 3. LocalSolver model.

Since LocalSolver uses metaheuristic methodologies, the nonlinearities in the model do not represent a problem for this optimization software. Table 7 compares the performance of the 5 approaches using the metrics described above. It also shows the average computational effort (given in seconds of computing time).

Procedure	Dev	$Best$	$Score$	$Time$
GRASP3.2	0.05%	0.75	0.97	11.00
Prokopyev	11.08%	0.00	0.00	11.15
Cplex	2.64%	0.25	0.68	2888
LocalSolver	3.93%	0.10	0.45	600

Table 7. Performance comparison of current and previous approaches.

The results in Table 7 support the hypothesis that GRASP3.2 is significantly better than its competitors at solving instances of the *Max-MinSum DP*. It is worth mentioning that LocalSolver's performance is fairly robust when considering that it is a generic solver, although it does require relative long running times to produce high quality solutions. On the other hand, Prokopyev provides low quality solutions as compared with the new proposal. We now perform our final experiment with all the 255 instances in the test set. We compare GRASP3, GRASP3.2, and *Prokopyev*, and report the results in Table 8.

Procedure	Dev	$Best$	$Score$	$Time$
Small instances ($n \leq 200$)				
GRASP3	0.01%	0.99	0.94	2.13
GRASP3.2	0.02%	0.99	0.91	2.13
Prokopyev	2.63%	0.60	0.47	2.13
Large instances ($n > 200$)				
GRASP3	0.36%	0.21	0.64	10.00
GRASP3.2	0.13%	0.34	0.90	10.00
Prokopyev	14.71%	0.00	0.00	10.00

Table 8. GRASP variants vs. *Prokopyev* on test set.

Results in Table 8 are in line with those reported in Table 7 and confirm the superiority of the new method proposed in this paper. An interesting result is that our variant with the single-sided strategic oscillation (GRASP3) shows a slightly better performance than the one with the double-sided strategic oscillation (GRASP3.2) in the small instances. Statistical tests, however, did not detect any significant differences in performance on the small instances between GRASP3 and GRASP3.2. We applied the non-parametric statistical test of Friedman to the data associated with the large instances in Table 7 and obtained a $p < 0.01$, which indicates the existence of significant performance differences among the three methods. Additionally, we applied the Wilcoxon test between GRASP3 and GRASP3.2 also to large-instance data and obtained a $p = 0.001$, indicating that there are significant performance differences between these two methods.

5. Conclusions

The *Max-MinSum DP* is a difficult combinatorial optimization problem and a perfect platform to study the effectiveness of search mechanisms. We studied how to integrate the strategic oscillation paradigm within the construction phase of the GRASP framework. We tested six different GRASP variants in which we considered different ways to generate and improve a solution. We performed several experiments with

instances previously used in the literature. Our experiments show that the double-sided strategic oscillation with the CM3 constructive method and IM2 improvement method provides the best outcomes overall. Moreover, the results indicate that the proposed hybrid heuristic compares favorably to an existing specialized procedure and a general-purpose optimizer (LocalSolver).

The *Max-MinSum DP* gave us the opportunity to test search strategies in a way that has not been described or reported in the literature. In particular, while strategic oscillation has its origins in tabu search, we were able to apply it in the context of GRASP. And although the most logical place to embed SO was either within the simple TS implementation or the neighborhood-search-driven improvement methods within GRASP, we decided to integrate it as part of GRASP's construction phase. Our SO design was such that we could test three variants of the oscillation pattern and we were able to discover how the double-sided pattern has some clear advantage over the one-sided counterpart when tackling large instance of the *Max-MinSum DP.* We believe that our findings can be translated to other settings and will help in the development of robust searches of combinatorial spaces.

Acknowledgments

This research has been partially supported by the Ministerio de Economía y Competitividad of Spain (Grant Ref. TIN2012-35632-C02), the Generalitat Valenciana (ACOMP/2014/A/241 and Prometeo 2013/049), and the University of Valencia (UV-INV-PRECOMP13-115334).

References

Ağca, S., B. Eksioglu, J. B. Ghosh; "Lagrangian solution of maximum dispersion problems". *Naval Research Logistics*, **47**: 97–114, 2000.

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

Erkut, E; "The discrete p-dispersion problem". *European Journal of Operational Research*, **46**:48–60, 1990.

Ghosh, J. B; "Computational aspects of the maximum diversity problem". *Operations Research Letters,* **19:** 175–181, 1996.

Glover, F; "Heuristics for Integer Programming using Surrogate Constraints". *Decision Sciences*, **8**(1): 156-166, 1977.

Glover, F., C. C. Kuo, K.S. Dhir; "A discrete optimization model for preserving biological diversity". *Applied Mathematical Modeling,* **19**: 696-701, 1995.

Glover, F., M. Laguna; "Tabu Search". *Kluwer Academic Publishers*, 1997.

Glover, F., C. C. Kuo, K. S. Dhir; "Heuristic algorithms for the maximum diversity problem". *Journal of Information and Optimization Sciences,* **19**(1): 109-132, 1998.

Hassin, R., S. Rubinstein, A. Tamir; "Approximation algorithms for maximum dispersion". *Operations Research Letters*, **21:** 133–137, 1997.

Kincard, R. K; "Good solutions to discrete noxious location problems via metaheuristics". *Annals of Operations Research*, **40**: 265-281, 1992.

Kuo, C. C., F. Glover, K. S. Dhir; "Analyzing and modeling the maximum diversity problem by zero-one programming". *Decision Sciences*, **24:**1171–1185, 1993.

Martí, R., F. Sandoya; "GRASP and path relinking for the equitable dispersion problem". *Computers and Operations Research,* **40**: 3091-3099, 2013.

Pisinger, D; "Upper bounds and exact algorithms for p-dispersion problems". *Computers and Operations Research*, **33**: 1380–1398, 2006.

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

Ravi, S. S., D. J. Rosenkrantz, G.K. Tayi; "Heuristic and special case algorithms for dispersion problems". *Operations Research*, **42**: 299-310, 1994.

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

Resende, M. G. C., C. C. Ribeiro; "Greedy randomized adaptive search procedures". *Metaheuristics*, F. Glover, G. Kochenberger, editors. Boston: Kluwer Academic Publishers, 219-250, 2001.

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