Les Cahiers du GERAD ISSN: 0711–2440

Less is more: Basic variable neighborhood search for Minimum differential dispersion problem

N. Mladenović, R. Todosijević,
D. Urošević

G–2015–39

April 2015

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Less is more: Basic variable neighborhood search for Minimum differential dispersion problem

Nenad Mladenović [a,b]
Raca Todosijević [b]
Dragan Urošević [d]

[a] *LAMIH – Université de Valenciennes et du Hainaut-Cambrésis, 59313 Valenciennes Cedex 9, France*

[b] *GERAD, HEC Montréal, Montréal (Québec) Canada, H3T 2A7*

[c] *Mathematical Institute, Serbian Academy of Sciences and Arts, 11000 Belgrade, Serbia*

nenad.mladenovic@gerad.ca
racatodosijevic@gmail.com
draganu@turing.mi.sanu.ac.rs

April 2015

Les Cahiers du GERAD
G–2015–39

Abstract: Large size optimization problems are usually successfully solved by using some metaheuristic approach. Nowadays, there is a trend to combine several metaheuristics into a new hybrid method in order to demonstrate that it is superior to each of its constituents. In this paper, we apply Basic variable neighborhood search for solving Minimum differential dispersion problem using only the Swap neighborhood structure in both descent (intensification) and shaking (diversification) steps. Despite the simplicity of the method, results obtained by our heuristic significantly outperforms the results obtained by a hybrid heuristic that combines GRASP and exterior path relinking rules. This fact confirms that *simplicity* is not just user friendly desirable property of heuristic, but it could also contribute to get more efficient and effective method than by using complex hybrid metaheuristics.

Key Words: Optimization, differential dispersion, heuristic, variable neighborhood search.

Acknowledgments: The work by Nenad Mladenović was conducted at National Research University, Higher School of Economics, Russia and supported by RSF grant 14-41-00039.

1 Introduction

Given a set N of n elements and the distances d_{ij} between any two elements i and j, the dispersion or diversity problems (DP) consist of finding a subset $S \subset N$ such that an objective function based on the distances between elements in S is maximized or minimized. The objective function may represent either efficiency-based measure that considers some dispersion quantity for the entire selection S, or equity-based measure that guarantees equitable dispersion among the selected elements. Widely studied problems that use efficiency-based objective functions are the Maximum diversity problem (MDP), in which the goal is to find subset S so that the sum of the distances between the selected elements is maximized, and The Max-Min diversity problem (MMDP), in which the goal is to find subset S so that the minimum distance between the selected elements is maximized. The problems that consider equity-based measures have been introduced by Prokopyev et al. [22]. They are: Maximum mean dispersion problem (Max-Mean DP), Minimum differential dispersion problem (Min-Diff DP), and Maximum min-sum dispersion problem (Max-Min-sum DP). The first problem includes finding a subset S, so that the average distance between the selected elements is maximized; the second deals with finding a subset S so that the difference between the maximum sum and the minimum sum of the distances to the other selected elements is minimized. Finally, the Max-Min-sum DP consists of finding a subset S so that the minimum sum of the distances to the other selected elements is maximized. In all aforementioned problems, except Max-Mean DP, the cardinality of the subset S must be equal to a given number m.

Some applications of diversity problems that use efficiency-based measures arise in the context of facility location (locating facilities according to distance, accessibility, impacts, etc) [7, 6, 14, 23], maximally diverse/similar group selection (e.g., biological diversity, admissions policy formulation, committee formation, curriculum design, market planning, etc.) [1, 8, 9, 15, 26], and densest subgraph identification [13]. On the other hand, diversity problems that use equity-based measures have applications in the context of urban public facility location, where the fairness among candidate facility locations is important [25], selection of homogeneous groups [2], dense/regular subgraph identification [13], and equity-based measures in network flow problems [3].

In this paper we study the Minimum differential dispersion problem (Min-Diff DP). Formally, Min-Diff DP may be formulated in the following way. Let S be a subset of a given set N whose cardinality is equal to m. The differential dispersion of this subset, $\delta(S)$ is calculated as

$$\delta(S) = \max_{i \in S} \Delta(i) - \min_{j \in S} \Delta(j)$$

where $\Delta(i) = \sum_{k \in S, k \neq i} d_{ik}$ represents the sum of distances of element i from all remaining elements in S. Therefore, the combinatorial formulation of the Min-Diff DP is as follows: find a subset $S^* \subset N$, $|S^*| = m$ with the minimum differential dispersion, i.e.,

$$S^* = \operatorname*{argmin}_{S \subset N, |S|=m} \delta(S) \tag{1}$$

Mathematical programming formulation of the Min-Diff DP may be stated in the following way. Let x_i be a binary variable that indicates whether element i belongs to S or not. Further, let L_i and U_i denote lower and upper bounds on the value of $\sum_{j \neq i, j \in N} d_{ij}$ calculated as $L_i = \sum_{j \neq i, j \in N} \min\{0, d_{ij}\}$ and $U_i = \sum_{j \neq i, j \in N} \max\{0, d_{ij}\}$. Finally, let M^+ and M^- denote an upper bound on U_i and a lower bound on the L_i values, respectively. Then Min-Diff DP may be formulated as 0-1 Mixed Integer Program as follows (for details see [22]):

$$\min_{t,r,s,x} t \tag{2}$$

subject to

$$t \geq r - s \tag{3}$$

$$r \geq \sum_{j, j \neq i} d_{ij} x_j - U_i(1 - x_i) + M^-(1 - x_i), \qquad i \in N; \tag{4}$$

$$s \leq \sum_{j,j \neq i} d_{ij}x_j - L_i(1-x_i) + M^+(1-x_i), \qquad i \in N; \tag{5}$$

$$\sum_{i \in N} x_i = m; \tag{6}$$

$$x \in \{0,1\}^n \tag{7}$$

Min-Diff DP is a NP-hard problem [22]. For solving it, several approaches are proposed in the literature. Prokopyev et al. [22] used CPLEX 9.0 MIP solver to solve the above MIP formulation. CPLEX solver succeeded to solve only small size instances up to $|N| = 40$ and $m = 15$, consuming more than 2500 seconds. For solving larger instances, they proposed generic GRASP heuristic (for solving dispersion problems using equity-based measure). More recently, Duarte et al. [5] proposed specialized GRASP heuristic, as well as a hybrid approach that combines GRASP and exterior path relinking. The last mentioned hybrid heuristic may be considered as a state-of-the-art heuristic for solving Min-Diff DP.

In this paper we suggest Basic variable neighborhood search for solving *Min-Diff DP*. Only a swap neighborhood structure is used in both the descent and the perturbation of an incumbent solution. Despite its simplicity, the results obtained at benchmark test instances significantly outperforms the state-of-the-art results, obtained by hybrid of GRASP and exterior path relinking based heuristic, published recently in *Information Sciences* journal [5]. Therefore, we can conclude that, sometimes, inclusion of many ideas in the search is not necessary to get excellent computational results: the less is more.

The rest of the paper is organized as follows. In the next section, we give rules of our heuristic, and in section 3 we report on computational results. Section 4 concludes the paper.

2 Variable neighborhood search for Min-Diff DP

Finding an optimal solution for large size *Min-Diff DP* is unlikely to be possible in reasonable time, thus, heuristic methods are a preferable option for finding good or near-optimal solutions. For that reason, we propose an efficient variable neighborhood Search (VNS) [18, 11] based heuristic to tackle *Min-Diff DP*. VNS is a flexible framework for building heuristics for approximately solving combinatorial and continuous global optimization problems. The main idea is systematical exploration of several neighborhood structures during the search for an optimal (or near-optimal) solution. The foundations of VNS are based on the following observations: (i) A local optimum relatively to one neighborhood structure is not necessarily the local optimal for another neighborhood structure; (ii) A global optimum is a local optimum with respect to all neighborhood structures; (iii) Empirical evidence shows that for many problems all local optima are relatively close to each other.

The work of a VNS based heuristic consists of applying alternately an improvement procedure and a shaking procedure, together with neighborhood change step, until reaching predefined stopping condition. An improvement procedure used within VNS heuristic may be either simple local search,that explores one neighborhood structure, or some more advanced procedure that explores several neighborhood structures. Such explorations could also be organized in different ways: (i) sequential variable neighborhood descent; (ii) Composite (or Nested) VND; (iii) Mixed nested [12]. On the other hand, a shaking procedure is used to possibly resolve local optima traps in which the used improvement procedure may be stuck. Typical stopping criteria for VNS heuristic are maximal number of iterations that may be performed, or maximum CPU time allowed to be consumed. The VNS based heuristics have been successfully applied to solving many optimization problems (see e.q., [19, 20, 16] for recent successful applications).

The proposed VNS heuristic, named `VNS_MinDiff`, uses one neighborhood structure within both improvement procedure and shaking procedure. In what follows, we give thorough description of the proposed heuristic. More precisely, we provide a description of a procedure for creating an initial solution, the definition of the used neighborhood structure, the description of the used shaking procedure, as well as the outline of entire heuristic.

An initial solution for our heuristic is created in the random fashion (see Algorithm 1). Namely, an initial solution is created choosing m random elements from the set N. Such a solution is further improved applying alternately local search procedure and shaking procedure together with neighborhood change step, as shown in Algorithm 2. The whole process is repeated until, the imposed time limit of t_{max} seconds is reached. Besides this parameter, `VNS_MinDiff` has parameter p_{max}, which defines the maximal value of the parameter of the shaking procedure, which will be described later.

Algorithm 1: Procedure for creating an initial solution

Function `Initial_solution()`;
$S = \emptyset$;
for $i = 1$ *to* m **do**
  Select j in $N \setminus S$ at random;
  $S \leftarrow S \cup \{j\}$;
end

Algorithm 2: VNS heuristic for solving Min-Diff DP

Function `VNS_MinDiff`(S, p_{max}, t_{max});
$S \leftarrow$ `Initial_solution ()`;
repeat
  $k \leftarrow 1$;
  while $p \le p_{max}$ **do**
    $S' \leftarrow$ `Shake`(S, p); /* Shaking */
    $S'' \leftarrow$ `LS`(S'); /* Local search */
    $p \leftarrow p + 1$; /* Next neighborhood */
    if S'' *is better then* S **then**
      $S \leftarrow S''$; $p \leftarrow 1$; /* Make a move */
    end
  end
  $t \leftarrow$ `CpuTime()`;
until $t > t_{max}$;
Return S**;**

Local search used within `VNS_MinDiff` is based on the exploration of the swap neighborhood structure defined as:

$$Swap(S) = \{S' \subset N || S \cap S'| = |S| - 1, \; |S'| = |S|\}.$$

This neighborhood structure is defined by the move that involves exchanging one selected element by the one which does not belong to S. In order to evaluate efficiently each solution in that neighborhood, we use an auxiliary array (already mentioned in the Introduction), denoted by Δ, that enable us to deduce the value of a solution S' in $O(m)$ time complexity. Namely, each element in the array Δ represents the sum of the distances of an element $i \in N$ to the selected elements in the set S (i.e., $\Delta(i) = \sum_{j \in S, j \neq i} d_{ij}$). Hence, in order to evaluate the value of the solution S' obtained by replacing a selected element k with an unselected element l, it suffices to determine the minimum and the maximum of values $\delta(i) = \Delta(i) - d_{ik} + d_{il}$,$i \in S \cup \{l\}$, $i \neq k$. Note that these two values determine the value of the solution S', as being the difference between them.

Depending on a search strategy used to explore this neighborhood structure, we distinguish *the first improvement* local search (denoted by `LS_FI`) which uses the first improvement search strategy (as soon as an improving solution is detected it is set to be the new incumbent solution), and *the best improvement local search* (denoted by `LS_BI`), which uses the best improvement search strategy (the best among all improving solutions (if any) is set to be the new incumbent solution). Regardless of the used search strategy, if the change of incumbent solution occurs, the search is resumed to start from the new incumbent solution, otherwise the procedure finishes its work. Note that each change of the incumbent solution requires update of the array Δ, which may be performed in $O(n)$ since each element $\Delta(i)$ may be updated in the constant time.

Shaking In order to escape from a local optima trap generated by a local search procedure, VNS heuristic employs the shaking procedure `Shake(S,p)`, presented at Algorithm 3.

Algorithm 3: Shaking procedure

Function `Shake(S,p)`;
for $i = 1$ *to* p **do**
  Select S' in $Swap(S)$ at random;
  $S \leftarrow S'$;
end

The shaking procedure has two parameters: a solution S and a parameter p. The parameter p determines the number of iterations performed within the shaking procedure. At each of p iterations, the shaking procedure generates a random solution from the swap neighborhood of the current solution. At the output, the procedure returns the last generated solution.

3 Computational results

In this section we evaluate performances of the proposed `VNS_MinDiff` heuristic, which has been coded in C++ language and run on a computer with an Intel Core i7 2600 CPU (3.4 GHz) and 16GB of RAM. For testing purposes, we use benchmark test instances, usually referred to as MDPLIB, that are publicly available at http://www.optsicom.es/mdp/mdplib_2010.zip. Instances are divided into three groups (having in total 190 instances):

- **SOM data set.** This data set consists of 20 test instances whose sizes range from $n = 25$ and $m = 2$ to $n = 500$ and $m = 200$. These instances were created with a generator developed by Silva et al. [24].
- **GKD data set.** This data set contains 70 test instances whose sizes range from $n = 10$ and $m = 2$ to $n = 500$ and $m = 50$. The instances are created by randomly choosing points from the square $[0, 10] \times [0, 10]$, while the distance between each two points is calculated as the Euclidean distance. These instances were introduced in Glover et al. [9].
- **MDG data set.** This data set consists of 100 test instances, and their sizes range from $n = 500$ and $m = 50$ to $n = 3000$ and $m = 600$. The distance matrices in these instances are generated by selecting real numbers between 0 and 10 from a uniform distribution. For extensive description of these instances, refer to Duarte and Marti [4], Marti et al. [17], and Palubeckis [21].

3.1 First vs best search strategy

The first part of experiments is devoted to discovering the most suitable search strategy for exploration of swap neighborhood structure regarding overall performance of `VNS_MinDiff`. Thus, we distinguish `VNS_MinDiff_BI` that uses `LS_BI` as a local search, and `VNS_MinDiff_FI` that uses `LS_FI` as a local search. Regardless of the used search strategy, after extensive testing, we set `VNS_MinDiff` parameter p_{max} to the smaller value between n and 30. The time limit, i.e., parameter t_{max}, is set to n seconds. Both VNS variants have been executed ten times, with different random seeds on each instance.

Comparative results are summarized in Tables 1 and 2. For each VNS variant, we report the average values of the best, the average and the worst solution values found on a ceratin data set regarding ten runs (columns 'best', 'avg.' and 'worst', respectively). In columns 'times', the average CPU times consumed by VNS variants to solve an instance from a certain data set are provided. On each instance, the percentage deviation of the best found solution value by `VNS_MinDiff_BI` over ten runs from the corresponding best found solution value attained by `VNS_MinDiff_FI` (in ten runs) is calculated using the formula:

$$\frac{\texttt{VNS\_MinDiff\_BI} - \texttt{VNS\_MinDiff\_FI}}{\texttt{VNS\_MinDiff\_BI}} \cdot 100\%.$$

In the similar way, on each test instance, the percentage deviation of the average solution value found by `VNS_MinDiff_BI` from the corresponding average solution value found by `VNS_MinDiff_FI`, and the percentage deviation of the worst solution value found VNS_MinDiff_BI from the corresponding worst solution value found by VNS_MinDiff_FI are computed. Hence, in the last three columns of Table 1, we report the average of these values over all instances from the same data set. In Table 1, the row before the last one contains the averages of the average values reported for each data set, while the last row provides average values calculated considering the union of those three data set as one data set. Since data sets contain unequal number of instances, the average values calculated considering the union of those three data sets as one data set do not coincide with the average values calculated as the averages of the average values over each data set.

Table 1: First versus best improvement search strategy within basic VNS

	VNS_MinDiff_BI				VNS_MinDiff_FI				(%)dev.		
Data set	best	avg.	worst	time	best	avg.	worst	time	best	avg.	worst
SOM	18.40	20.09	21.75	121.47	18.45	20.32	22.15	112.70	0.18	-2.83	-4.02
GKD	45.99	49.67	55.12	111.34	45.08	46.85	49.13	129.44	4.74	5.78	7.18
MDG	3052.07	3290.12	3521.92	1077.87	3114.59	3281.61	3451.08	1097.54	-6.62	-6.05	-5.80
Average:	1038.82	1119.96	1199.59	436.89	1059.38	1116.26	1174.12	446.56	-0.57	-1.03	-0.88
Total average:	1625.23	1752.05	1876.24	621.105	1657.81	1746.56	1836.79	637.20	-1.72	-1.36	-0.83

In Table 2, for each data set, we report the number of instances (# wins) where: the best solution offered by one VNS variant is better than the best solution found by another (Column 'best'); the average solution offered by one VNS variant is better than the average solution found by another (Column 'avg.'); and the worst solution offered by one VNS variant is better than the worst solution found by another (Column 'worst').

Table 2: First vs. best improvement search strategies - number of wins

		VNS_MinDiff_BI			VNS_MinDiff_FI		
Data set	# of instances	best	avg.	worst	best	avg.	worst
SOM	20	8	15	12	5	4	3
GKD	70	7	12	13	30	35	33
MDG	100	82	85	78	15	15	20
Total	190	97	112	103	50	54	56

From the results presented in Tables 1 and 2, the following interesting observations may be derived:

(i) VNS_MinDiff_BI performs better than VNS_MinDiff_FI on non-Euclidean instances (i.e., on SOM and MDG sets) in terms of both precision and cpu time spent in the search. The opposite is true for the GKD instances (where the average solution value found by VNS_MinDiff_FI is 5.78% better than the average solution value found by VNS_MinDiff_BI). The similar pattern regarding comparison of the first and the best search strategies (in solving travelling salesman problem) is observed in [10]. This is the reason why our final heuristic VNS_MinDiff uses the best improvement local search strategy for non-Euclidean instances and the first improvement for Euclidean.

(ii) On the entire set of instances VNS_MinDiff_BI performs better, since there were more non-Euclidean than Euclidean instances in all 3 data sets. Namely, the number of wins achieved by VNS_MinDiff_BI regarding all 190 instances is about two times larger than the number of wins achieved by VNS_MinDiff_FI.

(iii) On MDG data set containing the largest instances, VNS_MinDiff_BI performs much better than VNS_MinDiff_FI. On 82 (out of 100) instances VNS_MinDiff_FI the best solution found by VNS_MinDiff_BI is better than the best one of VNS_MinDiff_FI.

3.2 Comparison with the state-of-the-art approach

In this section we compare the results obtained by VNS_MinDiff with the results found by a hybrid heuristic that combines GRASP and exterior path relinking (GRASP_EPR) [5]. The comparison on each data set is performed comparing the results of `GRASP_EPR` with those of `VNS_MinDiff_BI` or with those of `VNS_MinDiff_FI`. On data sets SOM and MDG, where `VNS_MinDiff_BI` exhibits better performance than `VNS_MinDiff_FI`, the results of `GRASP_EPR` are compared with the results of `VNS_MinDiff_BI`, while on the data set GKD, the results of `GRASP_EPR` are compared with the results of `VNS_MinDiff_FI`. `GRASP_EPR` were tested on a computer with an Intel Core i7 2600 CPU (3.4 GHz) and 4 GB of RAM. It was executed a single time on each instance. On the other hand, `VNS_MinDiff_BI` and `VNS_MinDiff_BI` have been executed ten times, with different random seeds on each instance.

The comparison is presented in Tables 3–7. In these tables, we report the following values for each test instance: solution value found by `GRASP_EPR` (column '`GRASP_EPR`'); CPU time consumed by `GRASP_EPR` until reaching that solution (column '`GRASP_EPR` time'); the best, the average, and the worst solution values found by a considered `VNS_MinDiff` variant over ten runs (columns '`VNS_MinDiff` best', '`VNS_MinDiff` avg.' and '`VNS_MinDiff` worst', respectively); the deviation of these values from the corresponding value reported in column '`GRASP_EPR`' (columns '(%)imp. best', (%)imp. avg.' and '(%)imp. worst', respectively); and finally, the average CPU time consumed by a considered `VNS_MinDiff` variant over ten runs to solve the considered test instance (column '`VNS_MinDiff time`'). The values in columns '(%)imp. best', '(%)imp. avg.', '(%)imp. worst' are computed using the formula

$$\frac{\texttt{GRASP\_EPR} - \texttt{VNS\_MinDiff}}{\texttt{GRASP\_EPR}} \cdot 100\%,$$

and '`VNS_MinDiff` best', '`VNS_MinDiff` avg.' and '`VNS_MinDiff` worst', values instead of `VNS_MinDiff`, respectively.

From the results presented in Tables 3–7, we may infer the following:

(i) `VNS_MinDiff` significantly outperforms `GRASP_EPR`. Except on 20 small instances in GKD data, where the same solution by two heuristics are obtained, for all other instances, but one, our `VNS_MinDiff` heuristic established new best known solutions. We found 169 new best known solutions, we had 20 ties and on instance `MDG-a_18_n500_m50` we did not reach the best solution found by another method. In fact, for the MDG instances, we found 99 (out of 100) new best known solutions and just one was worst. We did not make much efforts to improve best known solutions (by increasing maximum cpu

Table 3: Computational results on SOM data set

Test instance	`GRASP_EPR`	time	`VNS_MinDiff` best	avg.	worst	time	(%)imp. best	avg.	worst
SOM-b_01_n100_m10	2	0.70	1	1	1	1.24	50.00	50.00	50.00
SOM-b_02_n100_m20	6	3.04	4	4.5	5	23.80	33.33	25.00	16.67
SOM-b_03_n100_m30	10	5.80	8	8.6	9	18.06	20.00	14.00	10.00
SOM-b_04_n100_m40	13	8.72	12	12.2	13	30.55	7.69	6.15	0.00
SOM-b_05_n200_m20	5	5.93	3	3.9	4	68.52	40.00	22.00	20.00
SOM-b_06_n200_m40	13	24.92	10	10.5	11	87.63	23.08	19.23	15.38
SOM-b_07_n200_m60	19	51.93	16	16.7	18	75.09	15.79	12.11	5.26
SOM-b_08_n200_m80	27	74.15	22	24	26	64.26	18.52	11.11	3.70
SOM-b_09_n300_m30	9	23.38	7	7.4	8	83.01	22.22	17.78	11.11
SOM-b_10_n300_m60	17	88.86	15	16.2	17	97.65	11.76	4.71	0.00
SOM-b_11_n300_m90	27	173.61	22	24.1	26	94.03	18.52	10.74	3.70
SOM-b_12_n300_m120	36	300.00	29	31.9	34	135.42	19.44	11.39	5.56
SOM-b_13_n400_m40	12	53.34	10	10.4	11	80.32	16.67	13.33	8.33
SOM-b_14_n400_m80	24	239.43	19	21.3	23	165.73	20.83	11.25	4.17
SOM-b_15_n400_m120	38	400.00	30	31.7	34	204.59	21.05	16.58	10.53
SOM-b_16_n400_m160	54	400.00	40	43.4	47	255.04	25.93	19.63	12.96
SOM-b_17_n500_m50	13	114.40	12	12.8	13	195.87	7.69	1.54	0.00
SOM-b_18_n500_m100	26	500.00	23	25.1	27	232.08	11.54	3.46	-3.85
SOM-b_19_n500_m150	47	500.00	36	39.6	45	248.05	23.40	15.74	4.26
SOM-b_20_n500_m200	69	500.00	49	56.4	63	268.57	28.99	18.26	8.70
Average:	23.35	173.41	18.40	20.09	21.75	121.47	21.82	15.20	9.32

Table 4: Computational results on GKD data set

			VNS_MinDiff				(%)imp.		
Test instance	GRASP_EPR	time	best	avg.	worst	time	best	avg.	worst
GKD-b_01_n25_m2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GKD-b_02_n25_m2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GKD-b_03_n25_m2	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GKD-b_04_n25_m2	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GKD-b_05_n25_m2	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GKD-b_06_n25_m7	12.72	0.17	12.72	12.72	12.72	0.32	0.00	0.00	0.00
GKD-b_07_n25_m7	14.10	0.16	14.10	14.10	14.10	0.01	0.00	0.00	0.00
GKD-b_08_n25_m7	16.76	0.16	16.76	16.76	16.76	0.00	0.00	0.00	0.00
GKD-b_09_n25_m7	17.07	0.17	17.07	17.07	17.07	0.00	0.00	0.00	0.00
GKD-b_10_n25_m7	23.27	0.31	23.27	23.27	23.27	0.66	0.00	0.00	0.00
GKD-b_11_n50_m5	1.93	0.19	1.93	1.93	1.93	0.03	0.00	0.00	0.00
GKD-b_12_n50_m5	2.12	0.17	2.05	2.05	2.05	1.08	3.29	3.29	3.29
GKD-b_13_n50_m5	2.36	0.19	2.36	2.36	2.36	0.44	0.00	0.00	0.00
GKD-b_14_n50_m5	1.66	0.19	1.66	1.66	1.66	0.02	0.00	0.00	0.00
GKD-b_15_n50_m5	2.85	0.19	2.85	2.85	2.85	0.05	0.00	0.00	0.00
GKD-b_16_n50_m15	42.75	1.39	42.75	42.75	42.75	4.84	0.00	0.00	0.00
GKD-b_17_n50_m15	48.11	1.61	48.11	48.11	48.11	9.19	0.00	0.00	0.00
GKD-b_18_n50_m15	43.20	1.34	43.20	43.20	43.20	1.28	0.00	0.00	0.00
GKD-b_19_n50_m15	46.41	1.36	46.41	46.41	46.41	7.06	0.00	0.00	0.00
GKD-b_20_n50_m15	47.72	1.27	47.72	47.72	47.72	8.27	0.00	0.00	0.00
GKD-b_21_n100_m10	13.83	1.17	9.43	9.57	10.14	51.96	31.82	30.79	26.66
GKD-b_22_n100_m10	13.66	1.17	8.04	9.24	10.88	56.85	41.16	32.36	20.40
GKD-b_23_n100_m10	15.35	1.08	7.59	8.48	9.95	28.25	50.51	44.74	35.17
GKD-b_24_n100_m10	8.64	1.20	6.60	7.27	8.79	47.40	23.58	15.91	-1.79
GKD-b_25_n100_m10	17.20	1.33	6.91	9.44	10.43	47.72	59.80	45.12	39.36
GKD-b_26_n100_m30	168.73	9.44	159.19	159.19	159.19	19.18	5.65	5.65	5.65
GKD-b_27_n100_m30	127.10	9.72	124.17	124.17	124.17	32.57	2.30	2.30	2.30
GKD-b_28_n100_m30	106.38	10.42	106.38	106.38	106.38	28.50	0.00	0.00	0.00
GKD-b_29_n100_m30	137.45	10.05	135.85	135.85	135.85	44.95	1.17	1.17	1.17
GKD-b_30_n100_m30	127.48	9.28	127.27	128.64	134.11	39.71	0.16	-0.91	-5.20
GKD-b_31_n125_m12	11.75	3.14	11.05	11.05	11.05	40.15	5.89	5.89	5.89
GKD-b_32_n125_m12	18.79	2.22	11.79	13.42	15.02	69.67	37.25	28.60	20.04
GKD-b_33_n125_m12	18.53	2.50	9.76	11.86	14.44	68.55	47.35	36.02	22.10
GKD-b_34_n125_m12	19.49	2.26	10.79	13.82	15.60	77.14	44.65	29.10	19.95
GKD-b_35_n125_m12	18.11	2.31	7.53	10.54	12.24	70.00	58.43	41.83	32.45
GKD-b_36_n125_m37	155.43	17.74	125.55	127.96	135.18	69.46	19.23	17.68	13.03
GKD-b_37_n125_m37	198.89	19.44	195.80	197.20	201.01	81.01	1.56	0.85	-1.06
GKD-b_38_n125_m37	187.97	18.71	184.27	184.39	185.43	90.94	1.97	1.91	1.35
GKD-b_39_n125_m37	168.59	18.43	155.39	161.29	171.36	68.91	7.83	4.33	-1.64
GKD-b_40_n125_m37	178.19	18.18	161.68	173.08	174.34	89.29	9.27	2.87	2.16
GKD-b_41_n150_m15	23.35	4.39	16.46	19.95	22.13	54.41	29.50	14.53	5.19
GKD-b_42_n150_m15	26.79	4.59	15.16	19.28	21.83	66.04	43.39	28.03	18.52
GKD-b_43_n150_m15	26.75	4.15	13.30	16.94	19.80	77.15	50.31	36.70	26.01
GKD-b_44_n150_m15	25.94	4.32	15.31	17.97	20.70	66.67	40.99	30.71	20.19
GKD-b_45_n150_m15	27.77	4.36	14.38	19.23	22.73	79.91	48.23	30.75	18.16
GKD-b_46_n150_m45	227.75	34.37	207.81	212.65	232.53	120.67	8.76	6.63	-2.10
GKD-b_47_n150_m45	228.60	34.57	212.97	215.69	223.16	92.61	6.84	5.65	2.38
GKD-b_48_n150_m45	226.75	30.27	177.29	179.51	185.34	93.64	21.81	20.83	18.26
GKD-b_49_n150_m45	226.41	36.04	197.88	214.13	231.90	111.81	12.60	5.42	-2.42
GKD-b_50_n150_m45	248.86	33.04	220.76	229.22	243.67	116.62	11.29	7.89	2.08
GKD-c_01_n500_m50	16.85	186.20	8.54	10.07	12.11	397.95	49.33	40.25	28.17
GKD-c_02_n500_m50	16.53	189.93	8.57	10.40	11.29	383.66	48.13	37.08	31.70
GKD-c_03_n500_m50	18.50	181.71	8.01	10.23	12.70	345.83	56.72	44.73	31.35
GKD-c_04_n500_m50	18.87	173.69	8.54	10.01	11.23	397.43	54.75	46.94	40.46
GKD-c_05_n500_m50	18.45	182.91	9.27	11.32	12.94	274.06	49.77	38.63	29.83
GKD-c_06_n500_m50	17.92	183.83	8.74	10.02	13.69	403.46	51.21	44.09	23.61
GKD-c_07_n500_m50	17.54	173.60	10.05	11.05	12.10	422.10	42.68	37.02	31.00
GKD-c_08_n500_m50	19.86	186.61	9.52	10.94	13.35	393.58	52.04	44.90	32.77
GKD-c_09_n500_m50	17.96	169.37	8.93	10.23	12.12	348.37	50.28	43.07	32.54
GKD-c_10_n500_m50	17.10	180.95	9.12	10.57	12.06	303.23	46.65	38.18	29.47
GKD-c_11_n500_m50	15.77	184.50	8.09	9.54	11.58	392.04	48.73	39.47	26.55
GKD-c_12_n500_m50	17.71	179.79	8.00	10.59	12.34	353.99	54.85	40.24	30.36
GKD-c_13_n500_m50	17.04	184.04	9.06	10.56	13.56	338.66	46.86	38.05	20.44
GKD-c_14_n500_m50	19.27	181.15	9.62	10.63	12.31	344.63	50.09	44.84	36.13
GKD-c_15_n500_m50	17.65	177.48	8.61	10.58	12.60	274.57	51.20	40.05	28.60
GKD-c_16_n500_m50	16.32	179.78	8.81	10.54	12.60	379.18	45.99	35.42	22.77
GKD-c_17_n500_m50	17.56	180.31	8.83	9.92	10.52	359.28	49.75	43.49	40.08
GKD-c_18_n500_m50	19.03	180.01	9.62	11.30	13.75	317.63	49.45	40.64	27.76
GKD-c_19_n500_m50	18.15	192.12	8.41	10.18	11.49	301.02	53.64	43.90	36.69
GKD-c_20_n500_m50	18.53	182.48	8.14	10.23	12.75	294.80	56.05	44.80	31.20
Average:	52.57	56.99	45.08	46.85	49.13	129.44	24.78	19.46	13.70

Table 5: Computational results on MDG data set

			VNS_MinDiff				(%)imp.		
Test instance	GRASP_EPR	time	best	avg.	worst	time	best	avg.	worst
MDG-a_01_n500_m50	13.53	179.47	11.34	12.26	12.67	151.84	16.19	9.39	6.36
MDG-a_02_n500_m50	12.99	176.56	11.67	12.45	12.94	189.10	10.16	4.14	0.38
MDG-a_03_n500_m50	13.34	172.91	11.71	12.22	12.82	311.65	12.22	8.43	3.90
MDG-a_04_n500_m50	13.41	178.02	11.56	12.34	12.94	258.35	13.80	8.01	3.50
MDG-a_05_n500_m50	13.50	164.69	12.05	12.50	12.77	233.59	10.74	7.39	5.41
MDG-a_06_n500_m50	12.95	180.56	10.87	12.15	12.74	328.59	16.06	6.19	1.62
MDG-a_07_n500_m50	13.09	173.27	10.95	12.14	13.17	292.69	16.35	7.30	-0.61
MDG-a_08_n500_m50	13.89	170.31	11.80	12.41	13.00	204.47	15.05	10.68	6.41
MDG-a_09_n500_m50	13.61	176.66	11.54	12.37	12.80	248.57	15.21	9.12	5.95
MDG-a_10_n500_m50	12.56	175.50	11.60	12.33	13.00	185.04	7.64	1.82	-3.50
MDG-a_11_n500_m50	13.21	174.29	11.25	12.12	12.68	117.75	14.84	8.29	4.01
MDG-a_12_n500_m50	13.01	182.68	12.17	12.53	12.87	251.91	6.46	3.70	1.08
MDG-a_13_n500_m50	12.70	170.06	12.05	12.41	12.99	298.51	5.12	2.31	-2.28
MDG-a_14_n500_m50	12.89	181.77	11.60	12.42	13.06	164.87	10.01	3.69	-1.32
MDG-a_15_n500_m50	13.51	178.36	11.55	12.39	12.91	221.15	14.51	8.33	4.44
MDG-a_16_n500_m50	13.19	176.83	12.15	12.64	13.12	240.76	7.88	4.19	0.53
MDG-a_17_n500_m50	12.48	180.14	11.76	12.32	12.73	276.39	5.77	1.32	-2.00
MDG-a_18_n500_m50	11.49	169.06	11.95	12.42	12.90	317.89	-4.00	-8.11	-12.27
MDG-a_19_n500_m50	13.50	177.66	11.50	12.34	12.93	241.46	14.81	8.58	4.22
MDG-a_20_n500_m50	13.20	175.63	11.66	12.18	12.60	253.48	11.67	7.75	4.55
MDG-a_21_n2000_m200	68.00	2000.00	50.00	53.10	57.00	1359.44	26.47	21.91	16.18
MDG-a_22_n2000_m200	70.00	2000.01	51.00	53.60	56.00	1490.53	27.14	23.43	20.00
MDG-a_23_n2000_m200	63.00	2000.00	52.00	54.30	57.00	959.98	17.46	13.81	9.52
MDG-a_24_n2000_m200	63.00	2000.00	48.00	53.00	58.00	1348.31	23.81	15.87	7.94
MDG-a_25_n2000_m200	57.00	2000.00	51.00	54.30	58.00	1255.08	10.53	4.74	-1.75
MDG-a_26_n2000_m200	68.00	2000.00	49.00	53.00	57.00	1136.47	27.94	22.06	16.18
MDG-a_27_n2000_m200	62.00	2000.00	50.00	54.70	58.00	1196.13	19.35	11.77	6.45
MDG-a_28_n2000_m200	64.00	2000.00	48.00	53.10	57.00	1280.44	25.00	17.03	10.94
MDG-a_29_n2000_m200	63.00	2000.01	51.00	53.00	56.00	1097.72	19.05	15.87	11.11
MDG-a_30_n2000_m200	65.00	2000.00	50.00	54.00	57.00	804.95	23.08	16.92	12.31
MDG-a_31_n2000_m200	67.00	2000.00	50.00	54.50	60.00	1084.71	25.37	18.66	10.45
MDG-a_32_n2000_m200	57.00	2000.00	51.00	54.60	61.00	1049.13	10.53	4.21	-7.02
MDG-a_33_n2000_m200	67.00	2000.01	49.00	53.70	60.00	1315.45	26.87	19.85	10.45
MDG-a_34_n2000_m200	59.00	2000.00	49.00	53.30	57.00	1058.42	16.95	9.66	3.39
MDG-a_35_n2000_m200	67.00	2000.00	53.00	54.70	56.00	1020.57	20.90	18.36	16.42
MDG-a_36_n2000_m200	57.00	2000.00	51.00	54.10	57.00	1300.08	10.53	5.09	0.00
MDG-a_37_n2000_m200	57.00	2000.00	49.00	52.90	56.00	1294.62	14.04	7.19	1.75
MDG-a_38_n2000_m200	65.00	2000.00	48.00	53.60	57.00	1263.25	26.15	17.54	12.31
MDG-a_39_n2000_m200	60.00	2000.00	51.00	54.00	58.00	1234.13	15.00	10.00	3.33
MDG-a_40_n2000_m200	62.00	2000.00	50.00	53.20	56.00	1056.70	19.35	14.19	9.68
MDG-b_01_n500_m50	1350.08	178.54	1185.11	1246.78	1296.49	266.17	12.22	7.65	3.97
MDG-b_02_n500_m50	1368.54	189.36	1182.48	1256.77	1322.03	245.14	13.60	8.17	3.40
MDG-b_03_n500_m50	1286.01	186.81	1070.87	1243.84	1310.09	331.21	16.73	3.28	-1.87
MDG-b_04_n500_m50	1300.24	185.34	1153.93	1240.57	1287.46	239.95	11.25	4.59	0.98
MDG-b_05_n500_m50	1258.79	185.03	1209.80	1262.90	1317.82	186.06	3.89	-0.33	-4.69
MDG-b_06_n500_m50	1272.73	182.13	1071.61	1227.71	1319.86	298.82	15.80	3.54	-3.70
MDG-b_07_n500_m50	1279.10	193.63	1099.68	1215.38	1311.55	256.32	14.03	4.98	-2.54
MDG-b_08_n500_m50	1315.79	185.12	1185.59	1245.45	1316.97	247.01	9.90	5.35	-0.09
MDG-b_09_n500_m50	1346.91	175.09	1154.33	1232.61	1261.83	243.90	14.30	8.49	6.32
MDG-b_10_n500_m50	1339.82	179.28	1198.08	1242.15	1289.55	272.05	10.58	7.29	3.75
Average:	292.82	907.10	254.90	274.72	288.81	619.62	14.97	9.07	4.11

Table 6: Computational results on MDG data set-continued

			VNS_MinDiff				(%)imp.		
Test instance	GRASP_EPR	time	best	avg.	worst	time	best	avg.	worst
MDG-b_11_n500_m50	1305.28	182.65	1145.73	1221.54	1275.68	249.64	12.22	6.42	2.27
MDG-b_12_n500_m50	1274.36	169.72	1165.43	1238.15	1294.60	252.95	8.55	2.84	-1.59
MDG-b_13_n500_m50	1337.33	185.02	1180.43	1238.00	1280.44	157.10	11.73	7.43	4.25
MDG-b_14_n500_m50	1291.06	191.77	1166.81	1247.25	1315.79	150.65	9.62	3.39	-1.92
MDG-b_15_n500_m50	1278.86	186.00	1220.83	1273.74	1314.10	281.51	4.54	0.40	-2.76
MDG-b_16_n500_m50	1328.66	180.79	1176.16	1248.31	1317.30	295.13	11.48	6.05	0.85
MDG-b_17_n500_m50	1299.00	179.15	1174.66	1252.52	1297.05	319.81	9.57	3.58	0.15
MDG-b_18_n500_m50	1321.87	174.22	1187.82	1267.94	1338.04	152.27	10.14	4.08	-1.22
MDG-b_19_n500_m50	1333.22	172.76	1175.26	1257.81	1291.88	369.35	11.85	5.66	3.10
MDG-b_20_n500_m50	1328.53	172.66	1151.34	1233.04	1285.53	271.70	13.34	7.19	3.24
MDG-b_21_n2000_m200	5073.98	2000.00	4083.16	4468.62	4737.59	1025.80	19.53	11.93	6.63
MDG-b_22_n2000_m200	5062.07	2000.00	4187.77	4540.42	4952.63	1039.79	17.27	10.30	2.16
MDG-b_23_n2000_m200	4899.35	2000.00	4237.38	4489.07	5171.39	1327.39	13.51	8.37	-5.55
MDG-b_24_n2000_m200	4780.51	2000.00	4212.28	4452.33	4708.87	1002.08	11.89	6.87	1.50
MDG-b_25_n2000_m200	5021.93	2000.00	4152.88	4435.36	4713.25	1375.81	17.31	11.68	6.15
MDG-b_26_n2000_m200	4959.65	2000.00	4039.92	4497.39	4798.83	1317.73	18.54	9.32	3.24
MDG-b_27_n2000_m200	4874.36	2000.00	4010.77	4486.90	4855.86	1079.71	17.72	7.95	0.38
MDG-b_28_n2000_m200	5245.69	2000.00	4206.07	4498.25	4798.32	844.44	19.82	14.25	8.53
MDG-b_29_n2000_m200	4955.58	2000.00	4214.79	4505.51	4809.00	1037.28	14.95	9.08	2.96
MDG-b_30_n2000_m200	5045.63	2000.00	4272.07	4564.38	4786.12	1022.86	15.33	9.54	5.14
MDG-b_31_n2000_m200	4962.72	2000.00	4328.97	4474.43	4710.96	1248.66	12.77	9.84	5.07
MDG-b_32_n2000_m200	4833.29	2000.00	4226.55	4484.07	4664.58	1069.63	12.55	7.23	3.49
MDG-b_33_n2000_m200	4973.32	2000.39	4037.50	4387.64	4786.52	1281.73	18.82	11.78	3.76
MDG-b_34_n2000_m200	4880.74	2000.00	4279.58	4480.58	4850.85	1038.31	12.32	8.20	0.61
MDG-b_35_n2000_m200	5061.54	2000.00	4018.60	4367.05	4679.23	1582.57	20.61	13.72	7.55
MDG-b_36_n2000_m200	4963.93	2000.00	4231.38	4433.05	4674.14	1067.68	14.76	10.69	5.84
MDG-b_37_n2000_m200	4801.03	2000.00	4100.54	4472.45	4834.64	1479.05	14.59	6.84	-0.70
MDG-b_38_n2000_m200	4946.67	2000.00	4136.67	4506.89	4802.26	1262.67	16.37	8.89	2.92
MDG-b_39_n2000_m200	5095.33	2000.44	4242.30	4450.95	4635.06	1219.16	16.74	12.65	9.03
MDG-b_40_n2000_m200	5001.89	2000.00	4249.76	4556.52	4804.78	1422.06	15.04	8.90	3.94
MDG-c_01_n3000_m300	7429.00	3001.50	6344.00	6595.40	7135.00	1455.45	14.60	11.22	3.96
MDG-c_02_n3000_m300	7781.00	3001.59	6109.00	6651.50	7183.00	1320.28	21.49	14.52	7.69
MDG-c_03_n3000_m300	7438.00	3001.63	6365.00	6828.70	7221.00	1639.72	14.43	8.19	2.92
MDG-c_04_n3000_m300	7212.00	3001.71	6304.00	6787.10	7215.00	1294.78	12.59	5.89	-0.04
MDG-c_05_n3000_m300	7346.00	3001.48	5954.00	6729.30	7282.00	1648.19	18.95	8.40	0.87
MDG-c_06_n3000_m400	10559.00	3002.86	8403.00	9422.10	10592.00	1861.19	20.42	10.77	-0.31
MDG-c_07_n3000_m400	9738.00	3003.16	8606.00	9308.60	9770.00	1847.39	11.62	4.41	-0.33
MDG-c_08_n3000_m400	10262.00	3002.85	8217.00	9206.80	10219.00	2009.81	19.93	10.28	0.42
MDG-c_09_n3000_m400	10202.00	3003.00	8478.00	9140.50	10337.00	2082.95	16.90	10.40	-1.32
MDG-c_10_n3000_m400	9266.00	3003.02	8244.00	9372.30	10129.00	1821.70	11.03	-1.15	-9.31
MDG-c_11_n3000_m500	13203.00	3005.46	11145.00	11998.90	13151.00	3014.47	15.59	9.12	0.39
MDG-c_12_n3000_m500	13458.00	3005.06	11366.00	12001.40	12709.00	3008.85	15.54	10.82	5.57
MDG-c_13_n3000_m500	11930.00	3004.86	10942.00	11832.40	12427.00	3012.91	8.28	0.82	-4.17
MDG-c_14_n3000_m500	13734.00	3005.04	10903.00	11455.20	12095.00	2736.52	20.61	16.59	11.93
MDG-c_15_n3000_m500	12091.00	3004.80	11051.00	12311.90	13282.00	3008.82	8.60	-1.83	-9.85
MDG-c_16_n3000_m600	16682.00	3007.55	13934.00	14732.10	15278.00	3006.15	16.47	11.69	8.42
MDG-c_17_n3000_m600	16673.00	3007.45	14086.00	14882.70	16184.00	3009.47	15.52	10.74	2.93
MDG-c_18_n3000_m600	15307.00	3007.09	13415.00	14515.20	15385.00	3006.68	12.36	5.17	-0.51
MDG-c_19_n3000_m600	14812.00	3007.68	13850.00	14821.90	15976.00	3005.60	6.49	-0.07	-7.86
MDG-c_20_n3000_m600	14462.00	3007.16	13532.00	14651.80	15396.00	3013.64	6.43	-1.31	-6.46
Average:	6842.45	2037.61	5849.23	6305.52	6755.03	1460.98	14.23	7.79	1.68

Table 7: Average results on each data set

			VNS_MinDiff				(%)imp.		
Data set	GRASP_EPR	time	best	avg.	worst	time	best	avg.	worst
SOM	23.35	173.41	18.40	20.09	21.75	121.47	21.82	15.20	9.32
GKD	52.57	56.99	45.08	46.85	49.13	129.44	24.78	19.46	13.70
MDG	3567.63	1472.35	3052.07	3290.12	3521.92	1040.30	14.60	8.43	2.89
Average:	1214.52	567.58	1038.52	1119.02	1197.60	430.40	20.40	14.37	8.64

time or by increasing the number of 10 trials). However, for the curiosity, we just wanted to check on that single instance, if we could improve the best known solution on it as well. We first increased the t_{max} parameter from 500 seconds to 550 seconds. Once in 10 trials we got the new best value again (equal to 11.34, the previous one was 11.49). It has been obtained after 504 seconds.

(ii) These new best known solutions are significantly better than the previous ones. This is especially true on data set GKD, where `VNS_MinDiff` improves the previous best known solutions values about 25% on best known. Also, the improvements achieved on data sets SOM and MDG are remarkable, and their % improvement are about 22% and 15%, respectively.

(iii) On each data set, the average improvement of `VNS_MinDiff` achieved over `GRASP_EPR` is greater or equal to 14.37%.

(iv) On data sets SOM, GKD and MDG the worst improvements of `VNS_MinDiff` achieved over `GRASP_EPR` are 9.32%, 13.70% and 2.89%, respectively.

(v) Regarding the average CPU time consumed, `VNS_MinDiff` is faster than `GRASP_EPR` on data sets MDG and SOM, while on data set GKD, `GRASP_EPR` is faster. However, regarding the average CPU time on all test instances, it follows that `VNS_MinDiff` needs less CPU time than `GRASP_EPR` on average to solve an instance (compare 430.40 seconds of `VNS_MinDiff` and 567.58 of `GRASP_EPR`).

All observations from above undoubtedly confirm superiority of `VNS_MinDiff` over the current state-of-the-art heuristic `GRASP_EPR`.

4 Conclusion

In this paper we addressed the minimum differential dispersion problem. For solving this NP-hard optimization problem, we propose basic Variable Neighborhood Search (VNS) based heuristic that uses just interchange neighborhood structure in both intensification and diversification phases. The proposed VNS based heuristic is tested on 190 benchmark instances. The results have been compared with those of a hybrid heuristic that combines GRASP and exterior path relinking (GRASP_EPR). The comparative analysis show that our heuristic succeeded to establish 170 (out of 190) new best known solutions, improving the quality of previous ones for about 20% on average! Additionally, the computational results disclosed that our VNS is faster than (GRASP_EPR) heuristics. All these facts indicate that the basic VNS, despite its simplicity and user friendliness, significantly outperforms recent approach that combines GRASP and exterior path relinking. We believe that our results will return area of heuristics to its original track: make an efficient and effective algorithm to be as simple as possible: the less is more.

Future work may include development of either basic or more advanced VNS based heuristics for other dispersion problems.

References

[1] G.K. Adil and J.B. Ghosh. Maximum diversity/similarity models with extension to part grouping. International Transactions in Operational Research, 12(3):311–323, 2005.

[2] J.R. Brown. The knapsack sharing problem. Operations Research, 27(2):341–355, 1979.

[3] J.R. Brown. The sharing problem. Operations Research, 27(2):324–340, 1979.

[4] A. Duarte and R. Martí. Tabu search and grasp for the maximum diversity problem. European Journal of Operational Research, 178(1):71–84, 2007.

[5] A. Duarte, J. Sánchez-Oro, M.G. Resende, F. Glover, and R. Martí. Greedy randomized adaptive search procedure with exterior path relinking for differential dispersion minimization. Information Sciences, 296:46–60, 2015.

[6] E. Erkut. The discrete p-dispersion problem. European Journal of Operational Research, 46(1):48–60, 1990.

[7] E. Erkut and S. Neuman. Analytical models for locating undesirable facilities. European Journal of Operational Research, 40(3):275–291, 1989.

[8] J.B. Ghosh. Computational aspects of the maximum diversity problem. Operations Research Letters, 19(4):175–181, 1996.

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

[10] P. Hansen and N. Mladenović. First vs. best improvement: An empirical study. Discrete Applied Mathematics, 154(5):802–817, 2006.

[11] P. Hansen, N. Mladenović, and J.A.M. Pérez. Variable neighbourhood search: methods and applications. Annals of Operations Research, 175(1):367–407, 2010.

[12] A. Ilić, D. Urošević, J. Brimberg, and N. Mladenović. A general variable neighborhood search for solving the uncapacitated single allocation p-hub median problem. European Journal of Operational Research, 206(2):289–300, 2010.

[13] G. Kortsarz and D. Peleg. On choosing a dense subgraph. In Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on, pages 692–701. IEEE, 1993.

[14] M.J. Kuby. Programming models for facility dispersion: The p-dispersion and maxisum dispersion problems. Geographical Analysis, 19(4):315–329, 1987.

[15] C.-C. Kuo, F. Glover, and K.S. Dhir. Analyzing and modeling the maximum diversity problem by zero-one programming. Decision Sciences, 24(6):1171–1185, 1993.

[16] J. Lazić, R. Todosijević, S. Hanafi, and N. Mladenović. Variable and single neighbourhood diving for mip feasibility. Yugoslav Journal of Operations Research, 2014. doi:10.2298/YJOR140417027L.

[17] R. Martí, M. Gallego, A. Duarte, and E.G. Pardo. Heuristics and metaheuristics for the maximum diversity problem. Journal of Heuristics, 19(4):591–615, 2013.

[18] N. Mladenović and P. Hansen. Variable neighborhood search. Computers & Operations Research, 24(11):1097–1100, 1997.

[19] N. Mladenović, R. Todosijević, and D. Urošević. An efficient general variable neighborhood search for large travelling salesman problem with time windows. Yugoslav Journal of Operations Research, 23(1):19–31, 2013.

[20] N. Mladenović, D. Urošević, and D. Perez-Brito. Variable neighborhood search for minimum linear arrangement problem. Yugoslav Journal of Operations Research, 2014. doi:10.2298YJOR140928038M.

[21] G. Palubeckis. Iterated tabu search for the maximum diversity problem. Applied Mathematics and Computation, 189(1):371–383, 2007.

[22] O.A. Prokopyev, N. Kong, and D.L. Martinez-Torres. The equitable dispersion problem. European Journal of Operational Research, 197(1):59–67, 2009.

[23] M. Rahman and M. Kuby. A multiobjective model for locating solid waste transfer facilities using an empirical opposition function. Location Science, 4(4):277–278, 1996.

[24] G.C. Silva, L.S. Ochi, and S.L. Martins. Experimental comparison of greedy randomized adaptive search procedures for the maximum diversity problem. In Experimental and Efficient Algorithms, pages 498–512. Springer, 2004.

[25] M.B. Teitz. Toward a theory of urban public facility location. Papers in Regional Science, 21(1):35–51, 1968.

[26] R. Weitz and S. Lakshminarayanan. An empirical comparison of heuristic methods for creating maximally diverse groups. Journal of the operational Research Society, pages 635–646, 1998.