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

Nenad Mladenovic, Raca Todosijević, Dragan Urošević

Information Sciences 326 (2016) 160–171

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

Nenad Mladenović [a,b,c,*], Raca Todosijević [b,c], Dragan Urošević [c]

[a] *Institute of Information and Computer Technologies, Kazakhstan*
[b] *LAMIH, Université de Valenciennes et du Hainaut Cambrésis, Valenciennes, France*
[c] *Mathematical Institute, Serbian Academy of Science and Arts, Serbia*

article info

Keywords:
Optimization
Differential dispersion
Heuristic
Variable neighborhood search

abstract

In this paper, we propose a basic variable neighborhood search for solving Minimum differential dispersion problem using only the *swap* neighborhood structure in both descent (intensification) and shaking (diversification) steps. It has become a trend in the metaheuristic literature to use hybrid metaheuristics, i.e., combination of several metaheuristic paradigms, for solving some particular optimization problem. We show that our simple method, which relies on the basic Variable neighborhood search, significantly outperforms the hybrid one that combines GRASP, Variable neighborhood search, and Exterior path relinking metaheuristics. Thus, simplicity is not only the desired user friendly property of a heuristic but can lead to more efficient and effective method than if complex hybrid metaheuristic is used: less is more.

1. Introduction

The dispersion, or diversity problems (DP) consist of finding a subset $S \subset N$, where a set N of n elements and distances between pairs of elements are given, such that the objective function based on the distances between elements in S is maximized or minimized. The objective function may represent either efficiency-based measure, considering some dispersion quantity for the entire selection S, or an equity-based measure, guaranteeing equitable dispersion among the selected elements. Widely studied problems that use efficiency-based objective functions are: the Maximum diversity problem (MDP), where the goal is to find a subset S maximizing the sum of the distances between the selected elements, and the Max–Min diversity problem (MMDP), whose goal is to maximize the minimal distance between the selected elements. The problems considering equity-based measures, introduced by Prokopyev et al. [21], 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 goal of finding a subset S in the first mentioned problem is to maximize the average distance between the selected elements; in the second, it is to minimize the difference between the maximum sum and the minimum sum of the distances to the other selected elements. Finally, in the Max–Min-sum DP, finding a subset S is done so to maximize the minimum sum of the distances to the other selected elements. Except Max-Mean DP, the cardinality of the subset S must be equal to a given number m.

DOI : 10.1016/j.ins.2015.07.044

Diversity problems that use efficiency-based measures find their application in the context of facility location (locating facilities according to distance, accessibility, impacts, etc) [7,8,15,22], maximally diverse/similar group selection (e.g., biological diversity, admissions policy formulation, committee formation, curriculum design, market planning, etc.) [1,9,10,16,26], and densest subgraph identification [14]. 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 [3], dense/regular subgraph identification [14], and equity-based measures in network flow problems [4].

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 the remaining elements in S. Therefore, the combinatorial formulation of the Min-Diff DP is as follows: find a subset $S^* \subset N$ containing m elements ($|S^*| = m$) 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, indicating whether an element i belongs to S or not. Further, let L_i and U_i denote the lower and the upper bounds of the value of $\Sigma_{j \neq i, j \in N} d_{ij}$, 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 the upper bound of U_i and the lower bound of L_i values, respectively. Then, by using decision variables t, r, and s, created in order to make the problem linear, the Min-Diff DP may be formulated as the following 0–1 Mixed Integer Program:

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

subject to

$$r \geq \sum_{j, j \neq i} d_{ij} x_j - U_i(1 - x_i) + M^-(1 - x_i), \ i \in N; \tag{3}$$

$$s \leq \sum_{j, j \neq i} d_{ij} x_j - L_i(1 - x_i) + M^+(1 - x_i), \ i \in N; \tag{4}$$

$$t \geq r - s \tag{5}$$

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

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

Constraints (3)/(4) ensure that the value of variable r/s is greater/less than the maximum (minimum) sum of distances of an element $i \in S$ from the remaining elements in the selected set S. Constraint (5) together with the objective function require that the difference between r and s is minimal (for details see [21]). Constraint (6) assures that the cardinality of the set S equals to m.

Min-Diff DP is a NP-hard problem [21]. For solving it, several approaches are proposed in the literature. Prokopyev et al. [21] used CPLEX 9.0 MIP solver to solve the above MIP formulation. CPLEX solver succeeded to solve only small size instances, those 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. [6] proposed a specialized GRASP heuristic, and a hybrid approach that combines GRASP, variable neighborhood search, 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 a 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 the simplicity of the method, the results obtained at benchmark test instances significantly outperform the state-of-the-art results, obtained by hybrid of GRASP, Variable Neighborhood Search and Exterior path relinking based heuristic, published recently in *Information Sciences* journal [6]. Therefore, we can conclude that including many ideas in the search does not necessarily lead to better computational results, on the contrary, sometimes "less can yield 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) [12,19] based heuristic to tackle *Min-Diff DP*. VNS is a flexible framework for building heuristics to solve combinatorial and continuous global optimization problems approximately. The main idea is to systematically explore 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) For many problems, empirical evidence shows that all local optima are relatively close to each other.

The VNS based heuristic consists of applying alternately an improvement procedure, a shaking procedure, and a neighborhood change step, until reaching predefined stopping condition. The 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 (VND); (ii) Composite (or Nested) VND; (iii) Mixed nested [13]. On the other hand, 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 performed iterations, or maximum allowed CPU time, t_{max}. The VNS based heuristics have been successfully applied to solving many optimization problems (see e.g., [2,17,23] for recent successful applications).

The pseudocode of the proposed VNS heuristic, named `VNS_MinDiff`, is given in Algorithm 1. The whole process is repeated until the imposed time limit of t_{max} seconds is reached (outer loop that starts from step 2). Besides t_{max}, `VNS_MinDiff` has p_{max} parameter, which defines the maximum number of the neighborhoods that will be used in the shaking or the diversification procedure (see neighborhood loop that starts from step 4). The choice of t_{max} and p_{max} values will be described later in computational result section.

Algorithm 1: VNS heuristic for solving Min-Diff DP.

```
   Function VNS_MinDiff(S, p_max, t_max);
1  S ← Initial_solution ();
2  repeat
3  |  p ← 1;
4  |  while p ≤ p_max do
5  |  |  S' ← Shake(S, p);              /* Shaking */
6  |  |  S'' ← LS(S');                  /* Local search */
7  |  |  p ← p + 1;                     /* Next neighborhood */
8  |  |  if S'' is better then S then
9  |  |  |  S ← S''; p ← 1;            /* Make a move */
   |  |  end
   |  end
10 |  t ← CpuTime();
   until t > t_max;
11 Return S;
```

`VNS_MinDiff` uses one neighborhood structure within both key steps of VNS that are iterated: improvement procedure and shaking procedure (see steps 5 and 6). Moreover, the *Move or not* step is also the simplest possible one (steps 7–9): move is made only if the better solution in the local search (step 6) is found.

In what follows, we give a 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, as well as the description of the used shaking procedure.

An initial solution for our heuristic is obtained by choosing m elements from the set N at random. Hence, no attempt is made to design some greedy constructive heuristic to get an initial solution of good quality. This fact makes implementation of our `VNS_MinDiff` even more simple. Its steps are given in Algorithm 2 .

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 substituting one selected element with the element which does not belong to S. In order to efficiently evaluate the objective function value of each solution in that neighborhood, we use an auxiliary array (already mentioned in the Introduction), denoted by Δ. It enables 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 $i \in N$ from the selected elements in the set S: $\Delta(i) = \sum_{j \in S, j \neq i} d_{ij}$. Hence, in order to find (update) the value of the solution S' obtained by replacing a

Algorithm 2: Procedure for creating an initial solution.

```
Function Initial_solution();
S = ∅;
for i = 1 to m do
    Select j in N \ S at random;
    S ← S ∪ {j};
end
```

selected element k with an element l that is not in S, it suffices to determine the maximum and the minimum values of $\delta(i)$:

$$(\max)/(\min)\ \delta(i) = \Delta(i) - d_{ik} + d_{il},\ i \in S \cup \{l\},\ i \neq k.$$

Note that these two values determine the value of the solution S', as it is the difference between them: $f(S') = \max_i \delta(i) - \min_i \delta(i)$.

We distinguish two different search strategies to explore this single neighborhood structure:

(i) **the first improvement** local search (`LS_FI`), where the new incumbent solution is obtained as soon as an improved solution is detected, and

(ii) **the best improvement** local search (`LS_BI`), where 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 an incumbent solution occurs, the search is resumed to start from the new incumbent solution, otherwise the procedure is finished, a local minimum is reached. Note that each change of the incumbent solution requires updating 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 avoid a local optima trap generated by a local search procedure, VNS heuristic employs the shaking procedure `Shake(S,p)`, presented in Algorithm 3 .

Algorithm 3: Shaking procedure.

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

The shaking procedure has two formal parameters: solution S and neighborhood index p. In fact, the parameter p determines the number of iterations performed within the shaking procedure. At each iteration a random solution from the swap neighborhood of the current solution is generated. Note that the substitution can include two elements already substituted in some of the previous iterations.

3. Computational results

In this section we evaluate the performance 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, publicly available at http://www.optsicom.es/mdp/mdplib_2010.zip. The total number of 190 instances are divided into three groups:

- **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. [10].
- **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 [5], Marti et. al. [18], and Palubeckis [20].

3.1. First vs. best search strategy

First part of the experiments is devoted to discovering the most suitable search strategy for exploration of swap neighborhood structure regarding overall performance of `VNS_MinDiff`. We distinguish two VNS based heuristics: `VNS_MinDiff_BI`

Table 1
First vs. best improvement search strategy within basic VNS.

Data set		VNS_MinDiff_BI					VNS_MinDiff_FI					(%)dev.		
Name	Size	Best	Avg.	Worst	σ	Time	Best	Avg.	Worst	σ	Time	Best	Avg.	Worst
GKD	[25–100]	33.76	34.03	34.35	0.33	8.75	33.76	34.12	34.76	0.45	11.72	0.00	−0.33	−1.34
GKD	[125–150]	98.02	108.26	124.45	7.88	50.61	97.61	107.37	121.92	6.32	64.84	−3.82	−0.75	−2.55
GKD	500	9.21	12.72	15.83	1.95	328.36	8.80	13.49	16.01	1.92	344.43	4.21	−6.19	−1.08
MDG-a	500	11.30	12.45	12.88	0.44	241.90	11.41	12.83	14.64	0.46	267.96	−1.05	−3.08	−13.64
MDG-a	2000	49.20	55.25	57.45	2.84	1208.47	49.65	55.62	59.25	2.74	1216.91	−0.94	−0.67	−3.18
MDG-b	500	1139.14	1247.04	1300.81	47.29	251.70	1134.65	1277.94	1439.77	48.14	283.60	0.33	−2.48	−10.71
MDG-b	2000	4153.52	4534.25	4782.79	197.54	1201.52	4173.45	4684.10	5192.38	194.94	1260.44	−0.49	−3.30	−8.61
MDG-c	3000	9800.35	10890.24	11433.15	541.07	2649.05	9367.00	10564.95	11454.75	459.10	2179.86	3.23	1.95	−0.50
SOM	[100–500]	17.60	20.05	21.75	1.16	125.73	17.45	20.21	22.30	1.16	127.53	0.02	−1.11	−4.40
Average:		1701.34	1879.36	1975.94	88.94	674.01	1654.86	1863.40	2039.53	79.47	639.70	0.16	−1.77	−5.11
Total average:		1613.58	1782.24	1873.75	84.28	639.00	1569.54	1767.12	1934.02	75.31	606.65	0.16	−1.70	−4.91

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

Data set	Size	# of instances	VNS_MinDiff_BI			VNS_MinDiff_FI		
			Best	Avg.	Worst	Best	Avg.	Worst
GKD	[25–100]	30	0	7	6	0	2	0
GKD	[125–150]	20	4	11	9	4	8	4
GKD	500	20	7	20	3	13	0	0
MDG-a	500	20	11	20	20	4	0	0
MDG-a	2000	20	7	14	14	2	6	0
MDG-b	500	20	6	20	20	5	0	0
MDG-b	2000	20	3	20	20	0	0	0
MDG-c	3000	20	3	9	5	12	11	3
SOM	[100–500]	20	5	17	12	5	3	2
	Total	190	46	138	109	45	30	9

that uses LS_BI as a local search, and VNS_MinDiff_FI that uses LS_FI as a local search. After extensive testing, we set VNS_MinDiff parameter p_{max} to 30, regardless of the used search strategy. However, since the minimum number of random swap moves, required to replace all currently selected elements by the new ones equals to m (the required cardinality), we set the value of p_{max} to $\min(m, 30)$. The time limit, i.e., parameter t_{max}, is set to n seconds, where n is the number of elements in the considered test instance (the value of $t_{max} = n$ has also been used in [6]). Both VNS variants have been executed forty times with different random seeds on each instance.

Comparative results are summarized in Tables 1 and 2. In each table, the first two columns provide the name and the size of the considered data set, respectively. In Table 1, for both VNS variants, we report the average values of the best, average, and the worst solution values found on a certain data set obtained in forty runs (columns 'best', 'avg.' and 'worst', respectively). In columns 'time', the average CPU times consumed by VNS variants are provided. Columns 'σ' contain the corresponding average standard deviations obtained in forty runs. In addition, the percentage deviations of the best, the average, and the worst solution values obtained by VNS_MinDiff_BI from the corresponding best, average, and worst solution values obtained by VNS_MinDiff_FI are calculated for each instance by the formula:

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

Hence, in the last three columns of Table 1, we report the average of these values over all 40 generated instances from the same data set. Therefore the negative sign in the last three columns indicate that the best improvement search strategy outperformed the first improvement. The opposite is true if the corresponding number in the last three columns has the positive sign.

The row 'Average' of Table 1, contains the averages of the average values reported for each data set. The last row provides average values calculated considering all 190 instances 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 all data sets.

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 variant (column 'best'); the average solution offered by one VNS variant is better than the average solution found by another variant (column 'avg.'); and the worst solution offered by one VNS variant is better than the worst solution found by another variant (column 'worst').

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

Table 3
Statistical comparison of VNS_MinDiff_BI and VNS_MinDiff_FI heuristics

Data set	Size	Number	R^+	R^-	R_{crit}	Sign
GKD	[25, 100]	30	232.5	1232.5	137	−
GKD	[125, 150]	20	103	107	52	−
GKD	500	20	43	167	52	+
MDG-a	500	20	146.5	63.5	52	−
MDG-a	2000	20	144	66	52	−
MDG-b	500	20	104.5	105.5	52	−
MDG-b	2000	20	133.5	76.5	52	−
MDG-c	3000	20	34.5	175.5	52	+
SOM	[100, 500]	20	105.5	104.5	52	−

Table 4
Computational results on SOM data set.

Test instance	GRASP_EPR	Time	VNS_MinDiff					(%)imp.		
			Best	Avg.	Worst	σ	Time	Best	Avg.	Worst
SOM-b_01_n100_m10	2.00	0.70	**0.00**	0.93	1.00	0.26	5.43	100.00	53.75	50.00
SOM-b_02_n100_m20	6.00	3.04	**4.00**	4.60	5.00	0.49	17.66	33.33	23.33	16.67
SOM-b_03_n100_m30	10.00	5.80	**6.00**	8.13	9.00	0.64	31.58	40.00	18.75	10.00
SOM-b_04_n100_m40	13.00	8.72	**10.00**	11.90	13.00	0.62	30.17	23.08	8.46	0.00
SOM-b_05_n200_m20	5.00	5.93	**3.00**	3.98	4.00	0.27	56.90	40.00	20.50	20.00
SOM-b_06_n200_m40	13.00	24.92	**10.00**	10.78	11.00	0.52	66.80	23.08	17.12	15.38
SOM-b_07_n200_m60	19.00	51.93	**15.00**	16.80	18.00	0.75	70.59	21.05	11.58	5.26
SOM-b_08_n200_m80	27.00	74.15	**21.00**	23.53	26.00	1.41	79.10	22.22	12.87	3.70
SOM-b_09_n300_m30	9.00	23.38	**6.00**	7.18	8.00	0.49	94.86	33.33	20.28	11.11
SOM-b_10_n300_m60	17.00	88.86	**15.00**	16.40	17.00	0.77	83.48	11.76	3.53	0.00
SOM-b_11_n300_m90	27.00	173.61	**21.00**	23.78	26.00	1.31	149.70	22.22	11.94	3.70
SOM-b_12_n300_m120	36.00	300.00	**29.00**	33.05	34.00	2.07	169.31	19.44	8.19	5.56
SOM-b_13_n400_m40	12.00	53.34	**9.00**	10.25	11.00	0.54	93.46	25.00	14.58	8.33
SOM-b_14_n400_m80	24.00	239.43	**19.00**	20.75	23.00	1.02	154.15	20.83	13.54	4.17
SOM-b_15_n400_m120	38.00	400.00	**28.00**	31.48	34.00	1.52	181.22	26.32	17.17	10.53
SOM-b_16_n400_m160	54.00	400.00	**40.00**	44.18	47.00	2.88	237.13	25.93	18.19	12.96
SOM-b_17_n500_m50	13.00	114.40	**11.00**	12.65	13.00	0.53	140.12	15.38	2.69	0.00
SOM-b_18_n500_m100	26.00	500.00	**22.00**	25.00	27.00	1.40	268.06	15.38	3.85	−3.85
SOM-b_19_n500_m150	47.00	500.00	**34.00**	39.20	45.00	2.15	298.15	27.66	16.60	4.26
SOM-b_20_n500_m200	69.00	500.00	**49.00**	56.40	63.00	3.61	286.79	28.99	18.26	8.70
Average:	23.35	173.41	17.60	20.05	21.75	1.16	125.73	28.75	15.76	9.32

(i) Comparing average solution values and CPU time spent in the search, it appears that VNS_MinDiff_BI performs better than VNS_MinDiff_FI on each data set, except on MDG-c and GKD data sets, containing instances whose sizes are within interval [125–150]. On the data set MDG-c, VNS_MinDiff_FI significantly outperforms VNS_MinDiff_BI regarding both solution quality and consumed CPU time. On the other hand, on GKD instances with 125 and 150 elements, VNS_MinDiff_FI provides better solutions than VNS_MinDiff_BI but consumes more CPU time.

(ii) On the entire set of instances, the VNS_MinDiff_FI heuristic performs slightly better (see row 'Total Average' in Table 1, i.e., compare 1767.12 and 1782.24 for total average values for VNS_MinDiff_FI and VNS_MinDiff_BI, respectively). The advantage of VNS_MinDiff_FI basically comes from the results obtained on the largest MDG-c instances. Indeed, VNS_MinDiff_BI may spend more time exploring a whole neighborhood of a current solution before making a move. The similar pattern regarding comparison of the first and the best search strategies in solving travelling salesman problem has been observed in [11].

(iii) Average solution values offered by VNS_MinDiff_BI are better than those found by VNS_MinDiff_FI on all instances from data sets GKD 500, MDG-a 500, MDG-b 500 and MDG-b 2000 (see the number of wins in Table 2). Moreover, only on data set MDG-c, VNS_MinDiff_FI succeeded to provide larger number of better average solution values than VNS_MinDiff_BI. In addition, on 138 out of total 190 instances, VNS_MinDiff_BI provides better average solution values, while VNS_MinDiff_FI do so on just 30 instances. However, VNS_MinDiff_BI provides better best found solution value than VNS_MinDiff_FI on only 46 instances, while VNS_MinDiff_FI do so on 45 instances.

In order to detect if there is a significant difference between VNS_MinDiff_BI and VNS_MinDiff_FI, we apply the Wilcoxon signed-rank test [27] on the results obtained by using two different strategies within the local search step of the basic VNS. The outcome is given in Table 3. Columns 1, 2, and 3 contain names of data sets, size of instances, and the number of instances in each data set, respectively. Columns 4 and 5 provide the sum of ranks for the instances where VNS_MinDiff_BI outperforms VNS_MinDiff_FI (R^+), and the sum of ranks for the instances where VNS_MinDiff_FI

Table 5
Computational results on GKD data set.

Test instance	GRASP_EPR	Time	VNS_MinDiff					(%)imp.		
			Best	Avg.	Worst	σ	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	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	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	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	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	0.00
GKD-b_06_n25_m7	12.72	0.17	12.72	12.72	12.72	0.00	0.00	0.00	0.00	0.00
GKD-b_07_n25_m7	14.10	0.16	14.10	14.10	14.10	0.00	0.00	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	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	0.00
GKD-b_10_n25_m7	23.27	0.31	23.27	23.27	23.27	0.00	0.00	0.00	0.00	0.00
GKD-b_11_n50_m5	1.93	0.19	1.93	1.93	1.93	0.00	0.00	0.00	0.00	0.00
GKD-b_12_n50_m5	2.12	0.17	**2.05**	2.05	2.05	0.00	0.06	3.29	3.29	3.29
GKD-b_13_n50_m5	2.36	0.19	2.36	2.36	2.36	0.00	0.02	0.00	0.00	0.00
GKD-b_14_n50_m5	1.66	0.19	1.66	1.66	1.66	0.00	0.00	0.00	0.00	0.00
GKD-b_15_n50_m5	2.85	0.19	2.85	2.85	2.85	0.00	0.00	0.00	0.00	0.00
GKD-b_16_n50_m15	42.75	1.39	42.75	42.75	42.75	0.00	0.01	0.00	0.00	0.00
GKD-b_17_n50_m15	48.11	1.61	48.11	48.11	48.11	0.00	0.00	0.00	0.00	0.00
GKD-b_18_n50_m15	43.20	1.34	43.20	43.20	43.20	0.00	0.02	0.00	0.00	0.00
GKD-b_19_n50_m15	46.41	1.36	46.41	46.41	46.41	0.00	0.21	0.00	0.00	0.00
GKD-b_20_n50_m15	47.72	1.27	47.72	47.72	47.72	0.00	0.01	0.00	0.00	0.00
GKD-b_21_n100_m10	13.83	1.17	**9.33**	9.36	9.50	0.05	31.48	32.53	32.35	31.32
GKD-b_22_n100_m10	13.66	1.17	**8.04**	8.85	10.22	0.75	34.01	41.16	35.23	25.24
GKD-b_23_n100_m10	15.35	1.08	**6.91**	7.44	9.10	0.66	33.50	54.99	51.52	40.68
GKD-b_24_n100_m10	8.64	1.20	**5.79**	7.52	7.94	0.83	44.24	32.98	13.01	8.08
GKD-b_25_n100_m10	17.20	1.33	**6.91**	8.70	10.43	1.20	42.85	59.80	49.39	39.36
GKD-b_26_n100_m30	168.73	9.44	**159.19**	160.65	163.85	2.26	8.37	5.65	4.79	2.89
GKD-b_27_n100_m30	127.10	9.72	**124.17**	124.61	127.10	1.04	21.95	2.30	1.96	0.00
GKD-b_28_n100_m30	106.38	10.42	**106.38**	106.38	106.38	0.00	10.82	0.00	0.00	0.00
GKD-b_29_n100_m30	137.45	10.05	**135.85**	136.01	135.85	0.53	25.56	1.17	1.05	1.17
GKD-b_30_n100_m30	127.48	9.28	**127.27**	128.47	127.27	2.60	9.50	0.16	−0.78	0.16
GKD-b_31_n125_m12	11.75	3.14	**11.05**	11.05	11.05	0.00	19.44	5.89	5.89	5.89
GKD-b_32_n125_m12	18.79	2.22	**8.48**	12.72	15.02	1.33	61.49	54.87	32.31	20.04
GKD-b_33_n125_m12	18.53	2.50	**9.18**	11.44	14.44	1.26	52.72	50.48	38.28	22.10
GKD-b_34_n125_m12	19.49	2.26	**10.79**	13.25	15.60	0.99	57.92	44.65	31.99	19.95
GKD-b_35_n125_m12	18.11	2.31	**7.53**	10.02	12.24	1.49	60.52	58.43	44.66	32.45
GKD-b_36_n125_m37	155.43	17.74	**125.55**	141.33	180.18	12.80	55.63	19.23	9.08	−15.92
GKD-b_37_n125_m37	198.89	19.44	**194.22**	200.02	221.51	7.95	59.08	2.35	−0.57	−11.37
GKD-b_38_n125_m37	187.97	18.71	**184.27**	188.48	213.88	8.26	51.47	1.97	−0.27	−13.79
GKD-b_39_n125_m37	168.59	18.43	**155.39**	167.90	181.13	7.79	59.95	7.83	0.41	−7.44
GKD-b_40_n125_m37	178.19	18.18	**161.68**	184.00	205.13	10.39	62.61	9.27	−3.26	−15.12
GKD-b_41_n150_m15	23.35	4.39	**16.25**	19.77	22.13	1.73	72.11	30.38	15.31	5.19
GKD-b_42_n150_m15	26.79	4.59	**12.38**	19.44	21.83	2.33	75.89	53.78	27.44	18.52
GKD-b_43_n150_m15	26.75	4.15	**11.83**	17.80	19.93	2.01	69.99	55.79	33.45	25.52
GKD-b_44_n150_m15	25.94	4.32	**12.54**	18.20	20.70	1.75	69.99	51.66	29.81	20.19
GKD-b_45_n150_m15	27.77	4.36	**14.38**	19.20	22.73	2.17	79.06	48.23	30.88	18.16
GKD-b_46_n150_m45	227.75	34.37	**207.81**	224.09	257.52	11.94	77.03	8.76	1.61	−13.07
GKD-b_47_n150_m45	228.60	34.57	**212.97**	224.90	259.86	11.84	77.04	6.84	1.62	−13.67
GKD-b_48_n150_m45	226.75	30.27	**177.29**	203.18	239.52	16.41	76.49	21.81	10.39	−5.64
GKD-b_49_n150_m45	226.41	36.04	**197.88**	219.22	247.26	15.03	73.08	12.60	3.18	−9.21
GKD-b_50_n150_m45	248.86	33.04	**220.76**	241.30	256.71	8.96	85.23	11.29	3.03	−3.15
GKD-c_01_n500_m50	16.85	186.20	**9.08**	11.89	13.63	1.56	318.08	46.10	29.42	19.14
GKD-c_02_n500_m50	16.53	189.93	**9.89**	12.51	15.41	1.70	331.93	40.17	24.27	6.76
GKD-c_03_n500_m50	18.50	181.71	**8.84**	12.21	16.29	2.13	330.98	52.24	34.03	11.98
GKD-c_04_n500_m50	18.87	173.69	**9.42**	12.42	15.99	2.12	358.39	50.10	34.16	15.26
GKD-c_05_n500_m50	18.45	182.91	**9.44**	13.22	16.00	2.04	296.34	48.84	28.32	13.28
GKD-c_06_n500_m50	17.92	183.83	**9.58**	12.11	14.49	1.68	341.38	46.53	32.45	19.17
GKD-c_07_n500_m50	17.54	173.60	**10.10**	13.38	16.65	1.93	277.31	42.40	23.71	5.09
GKD-c_08_n500_m50	19.86	186.61	**10.04**	13.43	18.38	2.35	342.03	49.43	32.37	7.44
GKD-c_09_n500_m50	17.96	169.37	**8.30**	13.43	17.64	2.44	348.68	53.77	25.21	1.81
GKD-c_10_n500_m50	17.10	180.95	**9.06**	13.44	17.82	2.56	328.39	47.03	21.42	−4.24
GKD-c_11_n500_m50	15.77	184.50	**8.42**	11.56	13.70	1.37	352.71	46.63	26.68	13.12
GKD-c_12_n500_m50	17.71	179.79	**9.06**	12.57	15.06	1.87	310.23	48.85	29.02	15.01
GKD-c_13_n500_m50	17.04	184.04	**8.66**	12.84	15.38	1.96	329.91	49.19	24.63	9.75
GKD-c_14_n500_m50	19.27	181.15	**9.50**	12.75	15.60	2.09	309.20	50.72	33.86	19.04
GKD-c_15_n500_m50	17.65	177.48	**9.34**	13.73	17.59	2.63	338.73	47.06	22.21	0.33
GKD-c_16_n500_m50	16.32	179.78	**8.13**	12.18	15.11	1.77	335.54	50.20	25.34	7.41

(*continued on next page*)

Table 5 (*continued*)

Test instance	GRASP_EPR	Time	VNS_MinDiff					(%)imp.		
			Best	Avg.	Worst	σ	Time	Best	Avg.	Worst
GKD-c_17_n500_m50	17.56	180.31	**8.72**	11.51	13.29	1.18	372.66	50.32	34.44	24.34
GKD-c_18_n500_m50	19.03	180.01	**9.40**	13.59	17.06	2.05	296.20	50.61	28.59	10.34
GKD-c_19_n500_m50	18.15	192.12	**9.27**	12.80	16.36	1.87	323.27	48.91	29.44	9.86
GKD-c_20_n500_m50	18.53	182.48	**9.96**	12.79	15.12	1.62	325.25	46.23	30.95	18.39
Average:	52.57	56.99	44.99	48.89	54.08	2.50	116.09	25.08	15.39	6.50

outperforms VNS_MinDiff_BI (R^-), respectively. Column 6 provides a critical value for the corresponding number of instances in the data set. If $\min\{R^+; R^-\}$ is less than or equal to the critical value, then the test detects significant differences between the algorithms. So, the last column indicates whether the Wilcoxon test found statistical differences between these algorithms or not ($'+'$ if a significant difference is found, and $'-'$ otherwise).

From this table, we can conclude that, in almost all of the groups, there are no significant differences between the two strategies. Exceptions are two groups GKD instances with $n = 500$ elements, and MDG-c instances with $n = 3000$ elements. This is where the first improvement strategy significantly outperforms the best improvement strategy (with respect to the best found solutions).

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

In this section, we compare results obtained by our VNS_MinDiff (either VNS_MinDiff_FI or VNS_MinDiff_BI) with the results obtained by the hybrid heuristic that combines GRASP, VNS, and exterior path relinking (GRASP_EPR) [6]. Detailed computational results of GRASP_EPR are taken from http://www.optsicom.es/mindiff/. On GKD instances with $n = 125$ and $n = 150$ elements, and on MDG-c instances with $n = 3000$ elements, VNS_MinDiff_FI is used for comparison because on those instances, it exhibits better performance than VNS_MinDiff_BI. On all other instances, VNS_MinDiff_BI is compared with GRASP_EPR. GRASP_EPR is coded in JAVA, tested on a computer with an Intel Core i7 2600 CPU (3.4 GHz) and 4 GB of RAM. Each instance was executed just once. On the other hand, VNS_MinDiff_BI and VNS_MinDiff_BI have been executed forty times, each time using different random seeds. Therefore, fair comparison should include values obtained by GRASP_EPR and our average objective values.

The comparison is presented in Tables 4, 5, 6, 7 and 8. In these tables, we report the following values for each test instance: values 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 forty 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); standard deviation for the considered VNS_MinDiff variant (columns 'σ') and finally, average CPU time consumed by a particular VNS_MinDiff variant over forty runs to solve the considered test instance (column 'VNS_MinDiff time'). The values in columns '(%)imp. best', '(%)imp. avg.', '(%)imp. worst' are computed by 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 4, 5, 6, 7 and 8, we may infer the following:

(i) VNS_MinDiff significantly outperforms GRASP_EPR. Except on 20 small instances in GKD data, where two heuristics obtained the same solution, our VNS_MinDiff heuristic did not establish new best known solution only on one instance. We found 169 new best known solutions (which can be downloaded from the website http://www.mi.sanu.ac.rs/~nenad/mddp/), 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 only one the worse than by GRASP_EPR. We did not make much efforts to improve best known solutions (by increasing maximum cpu time or by increasing the number of 40 trials). However, for the curiosity, we wanted to check on a single instance if we could improve the best known solution. We first increased the t_{max} parameter from 500 s to 550 s. In the first 10 trials, we got one new best value again (equal to 11.34, the previous one was 11.49). It was obtained after 504 s.

(ii) These new best known solutions are significantly better than the previous ones. This is especially true on GKD with $n = 500$ elements, where VNS_MinDiff improves the previous best known values for about 48% on the best known!. The improvements achieved on the remaining data sets are also remarkable. They are greater or equal to 7.80%.

Table 6
Computational results on MDG data set.

Test instance	GRASP_EPR	Time	VNS_MinDiff					(%)imp.		
			Best	Avg.	Worst	σ	Time	Best	Avg.	Worst
MDG-a_01_n500_m50	13.53	179.47	**11.34**	12.46	12.67	0.40	205.60	16.19	7.93	6.36
MDG-a_02_n500_m50	12.99	176.56	**11.29**	12.54	12.94	0.47	207.24	13.09	3.47	0.38
MDG-a_03_n500_m50	13.34	172.91	**11.71**	12.55	12.82	0.38	262.88	12.22	5.89	3.90
MDG-a_04_n500_m50	13.41	178.02	**11.40**	12.43	12.94	0.40	261.13	14.99	7.32	3.50
MDG-a_05_n500_m50	13.50	164.69	**11.33**	12.55	12.77	0.36	270.22	16.07	7.04	5.41
MDG-a_06_n500_m50	12.95	180.56	**10.87**	12.39	12.74	0.50	294.05	16.06	4.33	1.62
MDG-a_07_n500_m50	13.09	173.27	**10.95**	12.40	13.17	0.50	257.94	16.35	5.27	−0.61
MDG-a_08_n500_m50	13.89	170.31	**11.34**	12.44	13.00	0.42	220.49	18.36	10.46	6.41
MDG-a_09_n500_m50	13.61	176.66	**11.53**	12.37	12.80	0.43	227.59	15.28	9.08	5.95
MDG-a_10_n500_m50	12.56	175.50	**11.13**	12.33	13.00	0.46	191.74	11.39	1.85	−3.50
MDG-a_11_n500_m50	13.21	174.29	**10.06**	12.28	12.68	0.60	205.85	23.85	7.01	4.01
MDG-a_12_n500_m50	13.01	182.68	**11.48**	12.49	12.87	0.45	248.28	11.76	4.00	1.08
MDG-a_13_n500_m50	12.70	170.06	**11.56**	12.47	12.99	0.40	264.26	8.98	1.85	−2.28
MDG-a_14_n500_m50	12.89	181.77	**11.14**	12.51	13.06	0.50	228.95	13.58	2.94	−1.32
MDG-a_15_n500_m50	13.51	178.36	**11.55**	12.45	12.91	0.40	262.24	14.51	7.82	4.44
MDG-a_16_n500_m50	13.19	176.83	**11.65**	12.50	13.12	0.44	238.99	11.68	5.22	0.53
MDG-a_17_n500_m50	12.48	180.14	**11.33**	12.44	12.73	0.48	233.96	9.21	0.31	−2.00
MDG-a_18_n500_m50	11.49	169.06	11.55	12.49	12.90	0.38	267.76	−0.52	−8.69	−12.27
MDG-a_19_n500_m50	13.50	177.66	**11.50**	12.49	12.93	0.44	235.63	14.81	7.47	4.22
MDG-a_20_n500_m50	13.20	175.63	**11.27**	12.45	12.60	0.46	253.18	14.62	5.72	4.55
MDG-a_21_n2000_m200	68.00	2000.00	**49.00**	54.70	57.00	2.36	1364.63	27.94	19.56	16.18
MDG-a_22_n2000_m200	70.00	2000.01	**50.00**	55.35	56.00	2.86	1361.71	28.57	20.93	20.00
MDG-a_23_n2000_m200	63.00	2000.00	**49.00**	55.50	57.00	2.90	1140.64	22.22	11.90	9.52
MDG-a_24_n2000_m200	63.00	2000.00	**48.00**	54.80	58.00	3.12	1312.65	23.81	13.02	7.94
MDG-a_25_n2000_m200	57.00	2000.00	**50.00**	54.83	58.00	2.13	1296.24	12.28	3.82	−1.75
MDG-a_26_n2000_m200	68.00	2000.00	**49.00**	54.60	57.00	2.84	1197.82	27.94	19.71	16.18
MDG-a_27_n2000_m200	62.00	2000.00	**50.00**	55.55	58.00	3.09	1196.20	19.35	10.40	6.45
MDG-a_28_n2000_m200	64.00	2000.00	**47.00**	55.58	57.00	3.56	1343.74	26.56	13.16	10.94
MDG-a_29_n2000_m200	63.00	2000.01	**47.00**	54.40	56.00	2.91	1230.54	25.40	13.65	11.11
MDG-a_30_n2000_m200	65.00	2000.00	**50.00**	55.43	57.00	2.54	1094.32	23.08	14.73	12.31
MDG-a_31_n2000_m200	67.00	2000.00	**50.00**	55.53	60.00	2.87	1133.58	25.37	17.13	10.45
MDG-a_32_n2000_m200	57.00	2000.00	**49.00**	56.08	61.00	3.17	1155.74	14.04	1.62	−7.02
MDG-a_33_n2000_m200	67.00	2000.01	**49.00**	55.75	60.00	3.01	1153.34	26.87	16.79	10.45
MDG-a_34_n2000_m200	59.00	2000.00	**49.00**	55.33	57.00	3.23	1063.56	16.95	6.23	3.39
MDG-a_35_n2000_m200	67.00	2000.00	**52.00**	56.43	56.00	3.07	1168.43	22.39	15.78	16.42
MDG-a_36_n2000_m200	57.00	2000.00	**51.00**	56.13	57.00	2.56	1152.45	10.53	1.54	0.00
MDG-a_37_n2000_m200	57.00	2000.00	**49.00**	54.38	56.00	2.51	1203.72	14.04	4.61	1.75
MDG-a_38_n2000_m200	65.00	2000.00	**48.00**	55.25	57.00	2.85	1154.78	26.15	15.00	12.31
MDG-a_39_n2000_m200	60.00	2000.00	**48.00**	54.70	58.00	2.53	1209.16	20.00	8.83	3.33
MDG-a_40_n2000_m200	62.00	2000.00	**50.00**	54.78	56.00	2.65	1236.16	19.35	11.65	9.68
MDG-b_01_n500_m50	1350.08	178.54	**1185.11**	1268.89	1296.49	40.07	271.08	12.22	6.01	3.97
MDG-b_02_n500_m50	1368.54	189.36	**1182.48**	1256.77	1322.03	46.10	245.14	13.60	8.17	3.40
MDG-b_03_n500_m50	1286.01	186.81	**1070.87**	1243.84	1310.09	67.82	331.21	16.73	3.28	−1.87
MDG-b_04_n500_m50	1300.24	185.34	**1153.93**	1240.57	1287.46	42.46	239.95	11.25	4.59	0.98
MDG-b_05_n500_m50	1258.79	185.03	**1209.80**	1262.90	1317.82	37.06	186.06	3.89	−0.33	−4.69
MDG-b_06_n500_m50	1272.73	182.13	**1071.61**	1227.71	1319.86	76.32	298.82	15.80	3.54	−3.70
MDG-b_07_n500_m50	1279.10	193.63	**1099.68**	1215.38	1311.55	65.90	256.32	14.03	4.98	−2.54
MDG-b_08_n500_m50	1315.79	185.12	**1185.59**	1245.45	1316.97	39.07	247.01	9.90	5.35	−0.09
MDG-b_09_n500_m50	1346.91	175.09	**1154.33**	1232.61	1261.83	31.06	243.90	14.30	8.49	6.32
MDG-b_10_n500_m50	1339.82	179.28	**1143.26**	1241.29	1289.55	40.96	260.86	14.67	7.35	3.75
Average:	292.82	907.10	253.33	275.79	288.81	11.05	631.75	16.63	7.76	4.11

(iii) On all data sets, the average improvement achieved by VNS_MinDiff is greater or equal to 11.58% in comparison with GRASP_EPR.
(iv) On all data sets, the average worst improvement of VNS_MinDiff achieved over GRASP_EPR is greater or equal to 4.88%.
(v) Regarding the average CPU time consumed, VNS_MinDiff is faster than GRASP_EPR on large scale instances (MDG instances with 2000 and 3000 elements) and on SOM instances. However, regarding the average CPU time on all test instances, VNS_MinDiff needs less CPU time than GRASP_EPR, on average, to solve an instance (compare 623.46 seconds of VNS_MinDiff and 859.29 of GRASP_EPR).

In order to confirm the superiority of VNS_MinDiff over GRASP_EPR heuristic, we again use the Wilcoxon signed-rank test [27]. The results are given in Table 9. The column headings are defined in the same way as in Table 3. The results from Table 9 clearly confirm significant superiority of VNS_MinDiff approach over GRASP_EPR. Indeed, all signs in the last column have '+' signs.

Table 7
Computational results on MDG data set-continued.

Test instance	GRASP_EPR	Time	VNS_MinDiff					(%)imp.		
			Best	Avg.	Worst	σ	Time	Best	Avg.	Worst
MDG-b_11_n500_m50	1305.28	182.65	**1145.73**	1244.33	1275.68	42.74	239.02	12.22	4.67	2.27
MDG-b_12_n500_m50	1274.36	169.72	**1113.73**	1241.27	1294.60	43.45	245.68	12.60	2.60	−1.59
MDG-b_13_n500_m50	1337.33	185.02	**1180.28**	1252.67	1280.44	39.24	193.14	11.74	6.33	4.25
MDG-b_14_n500_m50	1291.06	191.77	**1157.36**	1250.73	1315.79	42.66	237.13	10.36	3.12	−1.92
MDG-b_15_n500_m50	1278.86	186.00	**1096.89**	1256.17	1314.10	49.46	258.33	14.23	1.77	−2.76
MDG-b_16_n500_m50	1328.66	180.79	**1176.16**	1248.17	1296.36	35.00	260.23	11.48	6.06	2.43
MDG-b_17_n500_m50	1299.00	179.15	**1131.63**	1253.39	1297.05	45.42	258.75	12.88	3.51	0.15
MDG-b_18_n500_m50	1321.87	174.22	**1106.29**	1259.19	1331.03	50.32	240.24	16.31	4.74	−0.69
MDG-b_19_n500_m50	1333.22	172.76	**1112.51**	1243.71	1291.88	52.72	277.20	16.55	6.71	3.10
MDG-b_20_n500_m50	1328.53	172.66	**1105.61**	1255.79	1285.53	57.86	243.99	16.78	5.48	3.24
MDG-b_21_n2000_m200	5073.98	2000.00	**4083.16**	4628.39	4737.59	258.14	1295.31	19.53	8.78	6.63
MDG-b_22_n2000_m200	5062.07	2000.00	**4098.84**	4597.18	4952.63	271.88	1101.76	19.03	9.18	2.16
MDG-b_23_n2000_m200	4899.35	2000.00	**4094.62**	4642.83	5052.41	237.18	1206.04	16.43	5.24	−3.12
MDG-b_24_n2000_m200	4780.51	2000.00	**4212.28**	4594.41	4708.87	174.52	1185.88	11.89	3.89	1.50
MDG-b_25_n2000_m200	5021.93	2000.00	**3986.03**	4595.29	4713.25	238.92	1225.85	20.63	8.50	6.15
MDG-b_26_n2000_m200	4959.65	2000.00	**4039.92**	4675.40	4798.83	243.85	1190.70	18.54	5.73	3.24
MDG-b_27_n2000_m200	4874.36	2000.00	**4010.77**	4626.19	4855.86	224.16	1033.47	17.72	5.09	0.38
MDG-b_28_n2000_m200	5245.69	2000.00	**4206.07**	4641.80	4798.32	256.83	1059.71	19.82	11.51	8.53
MDG-b_29_n2000_m200	4955.58	2000.00	**4214.79**	4505.51	4809.00	174.51	1037.28	14.95	9.08	2.96
MDG-b_30_n2000_m200	5045.63	2000.00	**4272.07**	4564.38	4786.12	185.52	1022.86	15.33	9.54	5.14
MDG-b_31_n2000_m200	4962.72	2000.00	**4328.97**	4474.43	4710.96	114.99	1248.66	12.77	9.84	5.07
MDG-b_32_n2000_m200	4833.29	2000.00	**4226.55**	4484.07	4664.58	114.09	1069.63	12.55	7.23	3.49
MDG-b_33_n2000_m200	4973.32	2000.39	**4037.50**	4387.64	4786.52	230.33	1281.73	18.82	11.78	3.76
MDG-b_34_n2000_m200	4880.74	2000.00	**4279.58**	4480.58	4850.85	153.11	1038.31	12.32	8.20	0.61
MDG-b_35_n2000_m200	5061.54	2000.00	**4018.60**	4367.05	4679.23	209.12	1582.57	20.61	13.72	7.55
MDG-b_36_n2000_m200	4963.93	2000.00	**4231.38**	4433.05	4674.14	145.73	1067.68	14.76	10.69	5.84
MDG-b_37_n2000_m200	4801.03	2000.00	**4100.54**	4472.45	4834.64	237.49	1479.05	14.59	6.84	−0.70
MDG-b_38_n2000_m200	4946.67	2000.00	**4136.67**	4506.89	4802.26	205.01	1262.67	16.37	8.89	2.92
MDG-b_39_n2000_m200	5095.33	2000.44	**4242.30**	4450.95	4635.06	123.17	1219.16	16.74	12.65	9.03
MDG-b_40_n2000_m200	5001.89	2000.00	**4249.76**	4556.52	4804.78	152.26	1422.06	15.04	8.90	3.94
MDG-c_01_n3000_m300	7429.00	3001.50	**6344.00**	7031.26	7873.00	370.52	1720.76	14.60	5.35	−5.98
MDG-c_02_n3000_m300	7781.00	3001.59	**6109.00**	7015.89	7340.00	345.03	1655.57	21.49	9.83	5.67
MDG-c_03_n3000_m300	7438.00	3001.63	**6346.00**	6999.57	7290.00	281.21	1862.79	14.68	5.89	1.99
MDG-c_04_n3000_m300	7212.00	3001.71	**6304.00**	7017.68	7291.00	274.16	1845.44	12.59	2.69	−1.10
MDG-c_05_n3000_m300	7346.00	3001.48	**5954.00**	6937.69	7282.00	273.63	1984.06	18.95	5.56	0.87
MDG-c_06_n3000_m400	10559.00	3002.86	**8307.00**	9389.43	10458.00	423.70	1995.74	21.33	11.08	0.96
MDG-c_07_n3000_m400	9738.00	3003.16	**8606.00**	9459.86	9770.00	401.64	2093.73	11.62	2.86	−0.33
MDG-c_08_n3000_m400	10262.00	3002.85	**8217.00**	9383.06	10219.00	401.52	2001.05	19.93	8.57	0.42
MDG-c_09_n3000_m400	10202.00	3003.00	**8378.00**	9285.48	10047.00	441.53	2103.56	17.88	8.98	1.52
MDG-c_10_n3000_m400	9266.00	3003.02	**8244.00**	9307.53	10129.00	503.13	1909.07	11.03	−0.45	−9.31
MDG-c_11_n3000_m500	13203.00	3005.46	**10443.00**	11842.83	13151.00	576.14	2451.43	20.90	10.30	0.39
MDG-c_12_n3000_m500	13458.00	3005.06	**10568.00**	11728.98	12709.00	466.83	2165.63	21.47	12.85	5.57
MDG-c_13_n3000_m500	11930.00	3004.86	**10504.00**	11832.75	12427.00	523.31	2090.24	11.95	0.82	−4.17
MDG-c_14_n3000_m500	13734.00	3005.04	**10355.00**	11675.71	12095.00	581.49	2253.02	24.60	14.99	11.93
MDG-c_15_n3000_m500	12091.00	3004.80	**10322.00**	11744.02	12986.00	531.42	2428.99	14.63	2.87	−7.40
MDG-c_16_n3000_m600	16682.00	3007.55	**12653.00**	14046.10	15278.00	534.15	2387.35	24.15	15.80	8.42
MDG-c_17_n3000_m600	16673.00	3007.45	**12127.00**	14080.93	16184.00	626.93	2723.07	27.27	15.55	2.93
MDG-c_18_n3000_m600	15307.00	3007.09	**12592.00**	14116.56	15385.00	537.52	2538.64	17.74	7.78	−0.51
MDG-c_19_n3000_m600	14812.00	3007.68	**12415.00**	14251.09	15785.00	502.06	2917.07	16.18	3.79	−6.57
MDG-c_20_n3000_m600	14462.00	3007.16	**12552.00**	14152.63	15396.00	586.03	2469.97	13.21	2.14	−6.46
Average:	6842.45	2037.61	5634.73	6289.79	6754.67	271.83	1401.63	16.40	7.35	1.65

4. Conclusion

In this paper we addressed the minimum differential dispersion problem. For solving this NP-hard optimization problem, we propose a basic Variable Neighborhood Search (VNS) based heuristic, where only interchange neighborhood structure is used, both in intensification and diversification phases. The proposed VNS based heuristic is tested on 190 benchmark instances. The results are compared with the results obtained by one hybrid heuristic that combines GRASP, variable neighborhood search, and exterior path relinking (GRASP_EPR). The comparative analysis shows that our heuristic succeeded to establish 170 (out of 190) new best known solutions, so improving the quality of the previous ones for about 21%, on average! Additionally, the computational results show that our VNS is faster than GRASP_EPR heuristic. All these facts indicate that the basic VNS, despite its simplicity, significantly outperforms recent approach that combines GRASP, variable neighborhood search, and exterior path

Table 8
Average results on each data set.

Data set	Size	GRASP_EPR	Time	VNS_MinDiff					(%)imp.		
				Best	Avg.	Worst	σ	Time	Best	Avg.	Worst
GKD	[25–100]	35.29	2.13	33.76	34.03	34.35	0.33	8.75	7.80	6.39	5.07
GKD	[125–150]	113.24	14.75	97.61	107.37	121.92	6.32	64.84	27.81	15.76	3.98
GKD	500	17.83	181.52	9.21	12.72	15.83	1.95	328.36	48.27	28.53	11.16
MDG-a	500	13.10	175.72	11.30	12.45	12.88	0.44	241.90	13.62	4.81	1.52
MDG-a	2000	63.05	2000.00	49.20	55.25	57.45	2.84	1208.47	21.64	12.00	8.48
MDG-b	500	1310.81	181.75	1139.14	1247.04	1300.81	47.29	251.70	13.08	4.82	0.70
MDG-b	2000	4971.96	2000.04	4153.52	4534.25	4782.79	197.54	1201.52	16.42	8.76	3.75
MDG-c	3000	11479.25	3004.25	9367.00	10564.95	11454.75	459.10	2179.86	17.81	7.36	−0.06
SOM	[100–500]	23.35	173.41	17.60	20.05	21.75	1.16	125.73	28.75	15.76	9.32
	Average:	2003.10	859.29	1653.15	1843.12	1978.06	79.66	623.46	21.69	11.58	4.88

Table 9
Statistical comparison of VNS_MinDiff and GRASP_EPR heuristics.

Data set	Size	Number	R^+	R^-	R_{crit}	Sign
GKD	[25, 100]	30	360	105	137	+
GKD	[125, 150]	20	210	0	52	+
GKD	500	20	210	0	52	+
MDG-a	500	20	209	1	52	+
MDG-a	2000	20	210	0	52	+
MDG-b	500	20	209	1	52	+
MDG-b	2000	20	210	0	52	+
MDG-c	3000	20	210	0	52	+
SOM	[100, 500]	20	210	0	52	+

relinking. We believe that our results will be a reminder of what the original goal of heuristics is: to create an efficient and effective algorithm so to be as simple as possible, or to put it as a moto, "less is more".

Future work may include application of either basic or more advanced VNS based heuristics to other dispersion problems. Also, much more effort should be made to moderate the actual strong trend towards complex and complicated hybrid metaheuristics.

Acknowledgements

This paper is partially supported by Ministry of Education and Science, Republic of Kazakhstan (Institute of Information and Computer Technologies), project number 0115PK00546, and also by Ministry of Education, Science and Technological Development of Serbia, project number 174010.

References

[1] G.K. Adil, J.B. Ghosh, Maximum diversity/similarity models with extension to part grouping, Int. Trans. Operational Res. 12 (3) (2005) 311–323.
[2] J. Brimberg, N. Mladenović, D. Urošević, Solving the maximally diverse grouping problem by skewed general variable neighborhood search, Inf. Sci. 295 (2015) 650–675.
[3] J.R. Brown, The knapsack sharing problem, Operations Res. 27 (2) (1979) 341–355.
[4] J.R. Brown, The sharing problem, Operations Res. 27 (2) (1979) 324–340.
[5] A. Duarte, R. Martí, Tabu search and grasp for the maximum diversity problem, European J. Operational Res. 178 (1) (2007) 71–84.
[6] A. Duarte, J. Sánchez-Oro, M.G. Resende, F. Glover, R. Martí, Greedy randomized adaptive search procedure with exterior path relinking for differential dispersion minimization, Inf. Sci. 296 (2015) 46–60.
[7] E. Erkut, The discrete p-dispersion problem, European J. Operational Res. 46 (1) (1990) 48–60.
[8] E. Erkut, S. Neuman, Analytical models for locating undesirable facilities, European J. Operational Res. 40 (3) (1989) 275–291.
[9] J.B. Ghosh, Computational aspects of the maximum diversity problem, Operations Res. Lett. 19 (4) (1996) 175–181.
[10] F. Glover, C.-C. Kuo, K.S. Dhir, Heuristic algorithms for the maximum diversity problem, J. Inf. Opt. Sci. 19 (1) (1998) 109–132.
[11] P. Hansen, N. Mladenović, First vs. best improvement: an empirical study, Discrete Appl. Math. 154 (5) (2006) 802–817.
[12] P. Hansen, N. Mladenović, J.A.M. Pérez, Variable neighbourhood search: methods and applications, Ann. Operations Res. 175 (1) (2010) 367–407.
[13] A. Ilić, D. Urošević, J. Brimberg, N. Mladenović, A general variable neighborhood search for solving the uncapacitated single allocation p-hub median problem, European J. Operational Res. 206 (2) (2010) 289–300.
[14] G. Kortsarz, D. Peleg, On choosing a dense subgraph, in: Proceedings of the 34th Annual Symposium on Foundations of Computer Science, 1993, pp. 692–701.
[15] M.J. Kuby, Programming models for facility dispersion: the p-dispersion and maxisum dispersion problems, Geograp. Anal. 19 (4) (1987) 315–329.
[16] C.-C. Kuo, F. Glover, K.S. Dhir, Analyzing and modeling the maximum diversity problem by zero-one programming, Decision Sci. 24 (6) (1993) 1171–1185.
[17] J. Lazić, R. Todosijević, S. Hanafi, N. Mladenović, Variable and single neighbourhood diving for mip feasibility, Yugoslav J. Operations Res. (2014), doi:10.2298/YJOR140417027L.
[18] R. Martí, M. Gallego, A. Duarte, E.G. Pardo, Heuristics and metaheuristics for the maximum diversity problem, J. Heuristics 19 (4) (2013) 591–615.
[19] N. Mladenović, P. Hansen, Variable neighborhood search, Comput. Operations Res. 24 (11) (1997) 1097–1100.
[20] G. Palubeckis, Iterated tabu search for the maximum diversity problem, Appl. Math. Comput. 189 (1) (2007) 371–383.

[21] O.A. Prokopyev, N. Kong, D.L. Martinez-Torres, The equitable dispersion problem, European J. Operational Res. 197 (1) (2009) 59–67.
[22] M. Rahman, M. Kuby, A multiobjective model for locating solid waste transfer facilities using an empirical opposition function, Location Sci. 4 (4) (1996) 277–278.
[23] S. Salhi, A. Imran, N.A. Wassan, The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation, Comput. Operations Res. 52 (2014) 315–325.
[24] G.C. Silva, L.S. Ochi, S.L. Martins, Experimental comparison of greedy randomized adaptive search procedures for the maximum diversity problem, Experimental and Efficient Algorithms, Springer, 2004, pp. 498–512.
[25] M.B. Teitz, Toward a theory of urban public facility location, Pap. Regional Sci. 21 (1) (1968) 35–51.
[26] R. Weitz, S. Lakshminarayanan, An empirical comparison of heuristic methods for creating maximally diverse groups, J. operational Res. Soc. 49 (1998) 635–646.
[27] F. Wilcoxon, Individual comparisons by ranking methods, Biomet. Bull. 6 (1) (1945) 80–83.