Optimization Letters
https://doi.org/10.1007/s11590-020-01690-0

ORIGINALPAPER

A multi-objective parallel variable neighborhood search for the bi-objective obnoxious p-median problem

Jesús Sánchez-Oro[1] · Ana D. López-Sánchez[2] · J. Manuel Colmenar[1]

Received: 29 July 2020 / Accepted: 14 December 2020

Abstract
Researchers and practitioners have addressed many variants of facility locations problems. Each location problem can be substantially different from each other depending on the objectives and/or constraints considered. In this paper, the bi-objective obnoxious p-median problem (*Bi-OpM*) is addressed given the huge interest to locate facilities such as waste or hazardous disposal facilities, nuclear power or chemical plants and noisy or polluting services, among others. The *Bi-OpM* aims to locate p facilities maximizing two different objectives: the distance between each customer and their nearest facility center and the dispersion among facilities. To address the *Bi-OpM* problem a Multi-objective Parallel Variable Neighborhood Search approach (Mo-PVNS) is implemented. Computational results indicate the superiority of the Mo-PVNS compared to the state-of-art algorithms.

Keywords Facility location problem · Obnoxious p-median problem · Multi-objective optimization · Variable neighborhood search

1 Introduction

According to [7], location problems can be classified into four categories regarding the objective function criteria: *facility location* problems, which seek to find a place to locate a facility in order to minimize the total cost between demand points and facilities; *p-median* problems, which determine the locations of p facilities in order

✉ J. Manuel Colmenar
josemanuel.colmenar@urjc.es

Jesús Sánchez-Oro
jesus.sanchezoro@urjc.es

Ana D. López-Sánchez
adlopsan@upo.es

1 Rey Juan Carlos University, C/ Tulipán s/n. 28933 Móstoles, Madrid, Spain

2 Pablo de Olavide University, Ctra. Utrera Km 1. 41013, Seville, Spain

Published online: 02 January 2021

to minimize the total cost between demand points and facilities; *p-center* problems, which minimize the maximum distance between each demand point and its assigned facility; and *covering* problems whose objective is to find the minimum number of facilities to cover all the demand points or to maximize the number of demand points covered by a given number of facilities. All those problems can also consider capacities in the facilities and demands in the demand points. In those cases they are known as *capacitated* and *uncapacitated* problems, when these features are not included [5,23]. Furthermore, location problems can be considered on the *discrete* space, when facilities can be only placed at specific locations [19], or on the *continuous* space, in which facilities can be placed at any location of a given region [1].

This work deals with an uncapacitated discrete facility location problem. Specifically, we focus on a variant named the bi-objective obnoxious p-median problem, *Bi-OpM*. The *Bi-OpM* was first introduced in [4] and seeks to locate a set of obnoxious facilities as far as possible from the set of demand points and at the same time from other obnoxious facilities. This situation appears when the interest is to locate facilities such as waste or hazardous disposal facilities, nuclear power or chemical plants and noisy or polluting services like airports, among others, and that is why the facilities are called obnoxious. There are many other real applications that can be modeled as the *Bi-OpM*. For instance, it can be considered that the obnoxious facilities to locate are power plants or mobile phone antennas given that they can produce radiation, and can make neighbor citizens uncomfortable.

The *Bi-OpM* can be formally stated as follows. Let I be the set of demand points that could represent, for instance, customers or cities, and J the set of candidate facilities, where $|I| = n$ and $|J| = m$. Furthermore, let $d(a, b)$, be the distance between two locations, where $a, b \in I \cup J$. The aim of the *Bi-OpM* is to locate a subset $P \subseteq J$ of facilities, having $|P| = p$ and $p < m$ in order to maximize two objective functions: f_1, the distance from each demand point to the facilities, computed as the sum of the minimum distances from each demand point and the nearest facility; and f_2, the dispersion among the facilities, computed as the sum of the minimum distances from each facility to the other selected facilities. Some authors name facilities in P as *open* facilities and facilities in $J \setminus P$ as *closed* or *unopened* facilities. More precisely, these objective functions can be described in the following way:

$$\begin{aligned} \max_{j \in P} f_1 &= \sum_{i \in I} \min d_{ij} \\ \max_{k \in P, k \neq j} f_2 &= \sum_{j \in P} \min d_{jk} \\ &\text{s.t.} P \subseteq J \\ &|P| = p \end{aligned}$$

Once the problem has been defined, it is important to emphasize that we are dealing with a multi-objective optimization problem. Therefore, we consider an efficient solution to be the one where no single-objective function value can be improved without deteriorating another objective function value. It is said that a solution P^* dominates another solution P if P^* is not worse than P in all the objectives, and P^* is better than

P in at least one objective. Similarly, we say that P^* weakly dominates P if P^* is not worse than P in all the objectives [2]. Formally, as we are maximizing the objectives, a solution P^* dominates another solution P, if $f_k(P^*) \geq f_k(P)$ for all $k = 1, 2$ and $f_k(P^*) > f_k(P)$ for at least one $k = 1, 2$. According to this, a solution is *efficient* if there is no other solution that dominates it. The aim is to achieve the set of efficient solutions, also known as the efficient frontier or the Pareto front.

As stated before, the *Bi-OpM* was first introduced in [4]. The authors proposed a Multi-Objective Memetic Algorithm (MOMA) defining two new variants of the crossover and mutation operators and studying three local search strategies applied in the MOMA. Furthermore, they performed a comparison using two multi-objective state-of-the-art methods, specifically the Non-dominated Sorting Genetic Algorithm II, (NSGA-II, see [8]), and the Strength-Pareto Evolutionary Algorithm 2 (SPEA2, see [25]), and also adding a single-objective Genetic Algorithm (GA) which combines the objectives under study through a weighted sum of their values. The authors proved that the MOMA approaches obtained better non-dominated solutions within less execution time. However, there is not a clear decision among the different local search strategies implemented in the MOMA framework but the specific selection will depend on the decision maker since two of the three MOMA variants, namely Dominance-Based Local Search (DBLS) and Alternate Objectives Local Search (AOLS), provide a higher number of efficient solutions, many of them close to higher values for each objective function separately and the other local search provides solutions with a good compromise between both objectives. Besides, these algorithms depend on the values stated for each one of the parameters they need such as the population size, number of generations, crossover and mutation probabilities and also two different probabilities related to the execution of the local search process. As we will show later, our approach is easier to adjust to the target set of instances. Furthermore, we refer the reader to [3,4] where an extensive and recent review of the literature covering obnoxious situations in both cases, single-objective and multi-objective optimizations is done.

In this paper, a Multi-objective Parallel Variable Neighborhood Search approach (Mo-PVNS) is specifically designed to solve the *Bi-OpM* in order to overcome the limitations found by the algorithms proposed in [4]. Our purpose is to find a good approximation to the Pareto Front (PF) in terms of convergence and/or diversity. That is, we aim for solutions close to the exact PF (if known) or to the best-known PF which, in the best of the cases, will be uniformly spread to cover all the regions of the search space. Hence, the main contributions of this paper are the adaption of the VNS schema to this problem in a very efficient way, and the final results obtained, which overcome the previous state-of-the-art.

The rest of the paper is organized as follows. Section 2 describes the proposed algorithm, Mo-PVNS, and details how the algorithm has been adapted and implemented to deal with *Bi-OpM*. Section 3 presents the computational results and finally, Sect. 4 summarizes the paper and discusses future work.

2 Multi-objective parallel variable neighborhood search algorithm

Our Multi-objective Parallel Variable Neighborhood Search (Mo-PVNS) tackles the *Bi-OpM* problem taking into account both the objectives of this problem in a non-aggregated way. However, before going deeper into the details of our proposal, we will first briefly describe the VNS methodology.

VNS is a metaheuristic, firstly introduced in [20], that relies on the idea of systematic changes in the neighborhood structures in order to explore different search spaces and avoid to be trapped by a local optimum. The flexibility of the VNS methodology has resulted in several variants in recent years (see [16] for a recent survey on the methodology), which has led to several successful applications for a variety of difficult optimization problems, such as those in [10,21]. Particularly, [9] or [17], among others, have dealt other variants of the p-median problem using the VNS methodology.

In single-objective optimization, the basic version of the VNS algorithm (BVNS) requires an initial solution as a starting point for the search, as well as the maximum number of neighborhoods to be explored. Then, the algorithm iterates from the first neighborhood until reaching the maximum considered neighborhood. In each iteration, BVNS obtains a random solution in the neighborhood under exploration, and improves it with a local search method. If the improved solution outperforms the incumbent one, then it is updated, restarting the search from the first neighborhood again. Otherwise, the search continues in the next considered neighborhood. This is repeated until a certain stopping condition is reached. Nevertheless, we are addressing a multi-objective optimization problem and the concept of "improving a solution" is different. In multi-objective optimization, instead of a single solution, a set of efficient solutions will be the "starting point" for the VNS algorithm. In this case, it is considered that the current solution has improved if a new generated solution is included in the approximate set of efficient solutions. We have followed the indications proposed by [11], where an innovative way to design the shake, the improvement, and the acceptance criterion procedures for multiple objective problems is presented. They propose to consider the approximate PF found during the search process as the incumbent solution.

In this work, not only VNS is adapted to solve this bi-objective optimization problem but also a parallel design of the algorithm is proposed, which is able to explore a wider search space. There are mainly four parallel designs for VNS that have been previously proposed in the literature [6,14]. The Syncrhonous Parallel VNS (SPVNS) is focused on parallelizing the local search phase, since it is usually the most time-consuming part of the search. Replicated Parallel VNS (RPVNS) tries to execute a complete and independent VNS procedure in each available processor, with the aim of having different starting points for VNS. Replicated Shaking VNS (RSVNS) parallelizes the shake method in such a way that the algorithm explores more than one single solution in each considered neighborhood, exploring a wider portion of the search space. Finally, in Cooperative Neighborhood VNS (CNVNS) several processors explore the considered neighborhood simultaneously, collaborating among them when a new best solution is found.

In this work, the SPVNS has not been considered since it is focused on reducing the computation time by parallelizing the local search. This is not necessary for the

Bi-OpM since the proposed local search method is indeed quite fast, as it can be seen in the results of Sect. 3. RPVNS has also been discarded for solving the *Bi-OpM* mainly due to the quality of the starting point found by the constructive procedure, that is not a single solution but a set of efficient solutions (see Sect. 2.1), which makes unnecessary starting the search in a new solution. Finally, CNVNS is not suitable for the proposal since we do not consider collaborations among processors for obtaining better solutions to allow more diversity in the search. On the other hand, the ability of RSVNS to explore wider solution spaces makes it the most suitable framework for the tackled problem, and allows us to obtain better solutions in the same computation time. Therefore, in this paper a RSVNS is considered with the aim of exploring a wider portion of the search space.

The pseudocode of the Mo-PVNS algorithm is briefly described in Algorithm 1.

Algorithm 1 Mo-PVNS(PF, k_{max})

1: $k \leftarrow 1$
2: **while** $k \neq k_{\max}$ or a time limit is reached **do**
3: $PF' \leftarrow PF$
4: **for all** $P \in PF'$ **do**
5: **for** $p_i \in 1 \ldots AVP$ **do** ▷ Parallel Section
6: $P' \leftarrow Shake(P, k)$
7: $PF' \leftarrow Insert\&Update(P')$
8: $P'' \leftarrow LocalSearch(P')$
9: $PF' \leftarrow Insert\&Update(P'')$
10: **end for**
11: **if** $PF' \neq PF$ **then** ▷ Improve in the Pareto front
12: $k \leftarrow 1$
13: $PF \leftarrow PF'$
14: **else**
15: $k \leftarrow k + 1$
16: **end if**
17: **end for**
18: **end while**
19: **return** PF

Our Mo-PVNS requires two input parameters: a set of non-dominated solutions PF and the number of the largest neighborhood to be explored, k_{max}. Starting from the first neighborhood (step 1), the method iterates until reaching the maximum predefined neighborhood $k_{\max}$ (steps 2–18). Notice that, in addition to the maximum predefined neighborhood, we have included another stopping criterion which is a maximum time limit. At each iteration, two different phases are applied to every efficient solution in the approximate Pareto front, which are explored in each available processor (*AVP*), denoted as p_i (step 5). In particular, every efficient solution is randomly perturbed in the current neighborhood k using the *Shake* procedure (step 6). The proposed *Shake* algorithm consists in randomly interchanging k open facilities with k closed facilities from the set of candidate facilities, generating solution P', which is added to the updated Pareto front, PF' (step 7). It is worth mentioning that the shake method performs random movements (whose sizes depend on the current neighborhood) which are not considered in the local search (i.e., interchange $k \leq p$ facilities simultaneously,

while the local search only interchanges a single facility). Moreover, the shake method accepts solutions of lower quality that will eventually allows us to explore further regions of the search space, while the local search only considers improved solutions. In the context of the multi-objective optimization it could happen that the perturbed solution is not dominated and, therefore, it must be inserted in the Pareto Front. Since the update of the Pareto Front is implemented in a very efficient way, step 7 assures that we are not losing any non-dominated solution during the search. After that, a local search method is responsible of locally improving the perturbed solution P', obtaining solution P'' (step 8). Notice that every feasible solution generated during the search is a candidate solution for entering in the set of non-dominated solutions. The method *Insert & Update* (steps 7 and 9) performs this verification, inserting the solution if it is non-dominated by others already in the set, removing those solutions dominated by the new one. Regarding this behavior, any modification in the Pareto front is considered as an improvement since a new non-dominated solution has been included in it. The complexity of the process that compares both fronts is linear with respect to the size of PF. Therefore, if the Pareto front has been modified, the search starts again from the first neighborhood (step 11), updating the incumbent Pareto front. Otherwise, the method explores the next neighborhood (step 15) until reaching the largest considered neighborhood. Mo-PVNS ends up returning the set of non-dominated solutions generated in the search.

VNS performs a single shake and local search to each solution of the Pareto front. However, with the aim of exploring a wider portion of the search space, the proposed algorithm performs several shake and local search steps in each iteration. Specifically, a different shake and local search step is performed in each available processor. This feature allows the algorithm to explore more than one single solution in the current neighborhood, which will eventually lead it to find better quality solutions, improving the Pareto Front.

Several parallelization techniques are suitable for designing and implementing parallel algorithms, such as threads, OpenMP, or CUDA, among others. The parallelization considered in the Mo-PVNS algorithm is focused on the use of threads, which can be defined as fragments of code that are independently executed in a processor. Since we are using Java programming language, we propose the use of Java threads, which can be easily used to tackle parallel task applications. In brief, the parallelization of an algorithm can have two different objectives: reduce the computing time or explore a wider portion of the search space. In this work we are focused in the latter, allowing VNS to perform several simultaneous shake and local search methods in the same neighborhood without increasing the computing time required to execute the complete algorithm. Thus, this parallelization allows us to explore more than one solution in each neighborhood spending similar computing times, which may eventually lead to a better set of non-dominated solutions.

2.1 Constructive method

The input for the Mo-PVNS algorithm is an initial approximation to the Pareto front named PF. To obtain this set of efficient solutions, we have opted for a constructive

procedure inspired by the construction phase of the *Greedy Randomized Adaptive Search Procedure* (GRASP) methodology, firstly introduced by [13]. This procedure combines greediness (intensification) and randomness (diversification) to build a variety of initial efficient solutions. In our case, as we are dealing with two objective functions, the procedure generates a predefined number of initial solutions for each objective function separately. Each solution is evaluated for entering in the approximated set of efficient solutions, PF, which is initially empty. This set is updated and, therefore, the output of the constructive phase is then the set of efficient solutions, PF, which acts as the input for the Mo-PVNS algorithm, as mentioned.

Algorithm 2 details the constructive method proposed to build initial efficient solutions for the *Bi-OpM*, which is generalized for any objective function, f_i. The input for the method is the set of candidate locations to host an obnoxious facility J, the parameter α which controls the greediness / randomness of the method, and the objective function under consideration f_i. The method starts by randomly selecting a candidate facility which is included in the solution under construction (steps 1-2). Then, a *Candidate List* (*CL*) is created with the remaining candidate facilities (step 3). The method iteratively selects other candidate facilities until p locations have been selected (step 4). In each iteration, the minimum and maximum values of the objective function among all the candidates are evaluated (steps 5–6). After that, the *Restricted Candidate List* (*RCL*) is created (step 8) with the most promising candidates, i.e., those whose objective function value is larger or equal than a threshold *th* (step 7). Notice that if $\alpha = 0$, the construction is totally greedy (it only considers the facilities that produce the greatest increase in the objective function value). On the contrary, if $\alpha = 1$, then all facilities are included in the *RCL* so the construction is totally random. The next vertex to be added to the solution is selected at random from the RCL (step 9), updating the solution under construction and the CL (steps 10–11). The method ends when the solution has exactly p locations selected.

Algorithm 2 Construct(J, α, f_i)

1: $v \leftarrow Random(J)$
2: $P \leftarrow \{v\}$
3: $CL \leftarrow J \setminus \{v\}$
4: **while** $|P| < p$ **do**
5: $\quad g_{\min} \leftarrow \min_{v \in CL} f_i(P \cup \{v\})$
6: $\quad g_{\max} \leftarrow \max_{v \in CL} f_i(P \cup \{v\})$
7: $\quad th \leftarrow g_{\max} - \alpha \cdot (g_{\max} - g_{\min})$
8: $\quad RCL \leftarrow \{v \in CL \ : \ f_i(P \cup \{v\}) \geq th\}$
9: $\quad v' \leftarrow Random(RCL)$
10: $\quad P \leftarrow P \cup \{v'\}$
11: $\quad CL \leftarrow CL \setminus \{v'\}$
12: **end while**
13: **return** P

2.2 Local search

Once the initial approximation of the Pareto front has been built using the greedy methodology, it is used to define the local search. The problem under consideration tries to optimize more than one objective function, so a reasonable and trivial approach would propose a different local search method for each objective function. However, this approach will likely intensify the search in the extreme parts of the Pareto front, but in multi-objective optimization it is desirable to obtain solutions uniformly spread along it. To overcome this issue, we propose a single local search method that aggregates both objective functions in a unique local search by means of a parameter $\beta \in [0, 1]$ that controls the influence of each objective function in the aggregated function f_a. This way, the aim is to explore a wider area in the search space. More formally, and for the particular case of a bi-objective optimization problem, $f_a(P) = \beta \cdot f_1(P) + (1 - \beta) \cdot f_2(P)$. Note that the function f_a is also known as the weighted-sum function, with β balancing the weight of each objective function.

Varying the value of β parameter will result in exploring different regions of the search space, potentially increasing the number of solutions included in the set of efficient solutions. Note that the previous expression can be easily generalized when $m > 2 : f_a(P) = \sum_{j=1}^{m} \beta_j \cdot f_j(P)$ with $\sum_{j=1}^{m} \beta_j = 1$.

For the *Bi-OpM*, the local search method traverses all the open facilities and tries to exchange them with every closed candidate facility. The search follows a first improvement approach in order to reduce the computational effort of the algorithm. In particular, every time an improvement move is found, it is performed and the search starts again.

3 Computational results

This section presents and discusses the results obtained in our experimental experience. In order to perform a fair comparison against the most competitive algorithm proposed in the literature, we have solved the same set of instances considered in [4]. Besides, we have also considered an additional number of instances and solved them in order to show a more exhaustive experimentation. Specifically, the previous paper presented 8 instances where the number of nodes (indicated as $|V| = |I \cup J|$) ranges from 400 to 900, the number of demand points and facilities ($|I|$ and $|J|$) varies from 200 to 450 and the number of open facilities (p) is between 25 and 225. In this paper, we have considered 64 additional instances under the same constraints, reaching a total number of 72 instances including the 8 instances considered in [4]. Table 1 shows the features for all the instances, which are available at http://grafo.etsii.urjc.es/optsicom/biopm/. Notice that they come from 24 different files which are processed with 3 different values of p. Our experiments were run on a computer provided with an Intel(R) Core(TM) i5-3470 processor running at 3.2 GHz, with 8 GB of RAM and Ubuntu 16.04. All the algorithms were implemented using Java 8.

Despite that there exists a large number of performance metrics, we have considered six different indicators: number of efficient solutions, hypervolume, coverage, epsilon, generational distance and generalized spread. As it is well-known, there is no single

Table 1 Features of the instances under study

Instance	$\|V\|$	$\|I\|$	$\|J\|$	p
pmed17	400	200	200	25, 50, 100
pmed18	400	200	200	25, 50, 100
pmed19	400	200	200	25, 50, 100
pmed20	400	200	200	25, 50, 100
pmed21	500	250	250	31, 62, 125
pmed22	500	250	250	31, 62, 125
pmed23	500	250	250	31, 62, 125
pmed24	500	250	250	31, 62, 125
pmed25	500	250	250	31, 62, 125
pmed26	600	300	300	37, 75, 150
pmed27	600	300	300	37, 75, 150
pmed28	600	300	300	37, 75, 150
pmed29	600	300	300	37, 75, 150
pmed30	600	300	300	37, 75, 150
pmed31	700	350	350	43, 87, 175
pmed32	700	350	350	43, 87, 175
pmed33	700	350	350	43, 87, 175
pmed34	700	350	350	43, 87, 175
pmed35	800	400	400	50, 100, 200
pmed36	800	400	400	50, 100, 200
pmed37	800	400	400	50, 100, 200
pmed38	900	450	450	56, 112, 225
pmed39	900	450	450	56, 112, 225
pmed40	900	450	450	56, 112, 225

metric that is capable of measuring all the aspects such as the cardinality (number of solutions in the set), the convergence (how close of the set to the Pareto front is) and the diversity (how are they spread, that is, the distribution and coverage of the set). This is the main reason why we have selected the most discriminant and used metrics.

The number of efficient solutions, #Sol, indicates the size of the approximation to the Pareto front obtained by the algorithm. Generally speaking, it is preferable a larger quantity of efficient solutions even if this metric ignores the quality of them. The hypervolume, HV, calculates the volume in the objective space covered by the points of an efficient set of solutions. To that end, a reference point is used and it can simply be found by constructing a vector of worst objective function values. In terms of quality, the larger the hypervolume value, the better the quality of the Pareto front analyzed. The coverage, $C(A, B)$, calculates the proportion of solutions of algorithm B that are weakly dominated by the efficient solutions found by the algorithm A. $C(A, B) = 1$ means that all the solutions in B are weakly dominated by A and $C(A, B) = 0$ means that no solution B is weakly dominated by A. Notice that it is not necessarily equal to $1 - C(B, A)$. The epsilon indicator, ϵ, measures the smallest distance to translate

Table 2 Preliminary results for different α values. Best values are depicted in bold font

α	#Sol.	HV	C(R,Mo-PVNS)	ϵ	GD	Δ	CPU Time (s)
Random	190.8750	**0.7496**	**0.6027**	**0.0151**	439.1371	0.9541	1642.0730
0.25	**201.6250**	0.7489	0.6228	0.0166	**431.9362**	**0.9512**	1660.8313
0.50	192.1250	0.7489	0.6034	0.0175	444.3233	0.9518	**1613.1695**
0.75	191.7500	0.7473	0.6730	0.0167	438.4848	0.9522	1650.1633

every point obtained in the approximation set so that it dominates the reference set (denoted as R) or the optimal Pareto if known. A lower indicator indicates a better approximation set. The generational distance, GD, measures how far are the efficient solutions in the computed approximation set obtained by the algorithm from those in the reference set or the optimal Pareto if known. If $GD = 0$, then all the solutions are in the reference set or the Pareto front. The generalized spread, Δ, measures the distance from a given efficient solution to its nearest neighbor. $\Delta = 0$ indicates a perfect spread of the efficient solutions in the reference set or the optimal Pareto if known, i.e., an ideal distribution. In addition to these indicators, we report the CPU time, which refers to the number of seconds required by the algorithm to obtain the set of efficient solutions. Obviously, a shorter computational time is preferable. We refer the reader to [12] for the definition and implementation of these multi-objective performance metrics.

The computational experience is divided into two parts: preliminary and final experiments. The first part allows us to select the best parameter setting for Mo-PVNS, and the second part compares the best performance of the Mo-PVNS against the state-of-art algorithms.

3.1 Preliminary experiments

Preliminary results of our Mo-PVNS are shown in order to select the best parameters for the algorithm. To that end, a representative subset of 8 instances out of 72 is selected. Specifically, the training set of instances is exactly the same set of instances used in [4]. The proposed algorithm requires two input parameters: α and $k_{\max}$. As suggested in [22], we will first determine the best value for one of the parameters, fix it and, then, determine the value of the second parameter. Hence, the first preliminary experiment aims to select the best value for the α parameter. To this end, we have obtained the Pareto front created after constructing and improving 100 solutions with each constructive procedure and the local search described in Sect. 2.2. Each experiment was run once. Column 1 of Table 2 includes the different α values for this parameter, specifically, random, 0.25, 0.50 and 0.75. The remaining columns of Table 2 show the average values of each performance metric (the number of efficient solutions, the hypervolume, the coverage, the epsilon indicator, the generational distance, the spread and the computational time respectively) obtained in this experiment.

On average, the best results are obtained when considering a random value for the α parameter in each construction. Specifically, this configuration reaches the best values

Table 3 Preliminary results for different $k_{\max}$ values. Best values are depicted in bold font

$k_{\max}$	#Sol.	HV	C(R,Mo-PVNS)	ϵ	GD	Δ	CPU Time (s)
0.1p	200.0000	**0.7380**	**0.6074**	**0.0145**	433.5662	**0.9495**	**1623.0769**
0.2p	191.6250	0.7359	0.7121	0.0208	432.6298	0.9524	1628.1353
0.3p	196.7500	0.7372	0.6670	0.0166	433.3020	0.9532	1638.7465
0.4p	193.2500	0.7368	0.6730	0.0167	438.2227	0.9530	1649.9543
0.5p	**200.7500**	0.7378	0.6123	0.0156	**430.4560**	0.9496	1660.2193

for the hypervolume, the coverage, and the epsilon indicators, while the differences in terms of spread values are insignificant. Therefore, we have selected a randomly generated value for α for the final version of the algorithm.

The second preliminary experiment is devoted to find the best value for the $k_{\max}$ parameter of the Mo-PVNS. Specifically, we have tried $k_{\max} = \{0.1p, 0.2p, 0.3p, 0.4p, 0.5p\}$, p being the number of open facilities. We do not consider larger values since such a large perturbation is equivalent to starting the search from a completely different solution, which is not in line with the VNS methodology. The values of the $k_{\max}$ parameter are shown in the first column of Table 3. The remaining columns of Table 3 show the average values of each performance metric derived from this experiment.

As it can be derived from Table 3, the best results are obtained when considering the smallest neighborhood size, $k_{\max} = 0.1p$. In particular, the value $k_{\max} = 0.1p$ is able to obtain the best results for all the metrics except for the generational distance and for the number of efficient solutions but being very similar to the best value of the experiment, $k_{\max} = 0.5p$. Therefore, we have selected $k_{\max} = 0.1p$ for the final version of the algorithm.

3.2 Final experiments

In the final experiment we perform a comparison against four different algorithms: the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [8], which is a classical multi-objective evolutionary algorithm, the Strength-Pareto Evolutionary Algorithm 2 (SPEA2) [24], and the two variants of the memetic algorithm that obtained the best results in the current state of the art [4]: Alternate Objective Local Search, AOLS and Dominance-Based Local Search, DBLS. The values of the parameters for NSGA-II and SPEA2 are those described in [4], obtained after running iRace [18] on the 8 instances from the representative subset. iRace is a software package that obtains the best combination of parameters after running an iterated race procedure (see [4] for details). We performed the same parameter tuning for AOLS and DBLS, stating a maximum number of evaluations of 10^6. Table 4 shows the ranges of the parameters analyzed by iRace and the corresponding results for the parameter values.

All the previous algorithms were run 30 times, as performed in [4]. In the case of our proposed Mo-PVNS, the results correspond to just one run. We have discarded the classical Genetic Algorithm presented in the previous paper since, as stated by

Table 4 Parameter optimization for DBLS and AOLS with iRace

Parameter	Range	AOLS	DBLS
Generations	(100, 5000)	3677	3912
Population	(50, 250)	214	212
Mutation probability	(0.100, 0.999)	0.6397	0.7011
Crossover probability	(0.100, 0.999)	0.5225	0.7083

Table 5 Average results for the final experiments

	#Sol.	HV	C(R,Alg.)	ϵ	GD	Δ	CPU Time (s)
Mo-PVNS	**248.0833**	**0.7422**	**0.0486**	**0.0028**	**431.6168**	**0.9479**	**1615.1111**
AOLS	107.6250	0.6794	0.8065	0.0802	583.7231	0.9591	1826.7894
DBLS	115.2222	0.6820	0.7938	0.0739	564.8547	0.9582	1881.0322
NSGA-II	30.3472	0.3856	0.8497	0.3837	1092.0925	0.9727	11794.3458
SPEA2	47.8472	0.5165	0.7918	0.2571	883.7648	0.9724	16543.8166

the authors, it consistently obtains worse quality solutions. Table 5 shows the average results obtained for the complete set of 72 instances by each algorithm.

Analyzing the results presented in Table 5, we can confirm the superiority of Mo-PVNS with respect to the previous algorithms, since it is able to obtain better results in all the considered metrics. Regarding the number of efficient points, Mo-PVNS is able to obtain, on average, more than twice the number of efficient points of the second best algorithm, which is DBLS. Furthermore, it is worth mentioning the coverage value obtained by Mo-PVNS, which is very close to zero. This result indicates that the reference set, R, contains most of the solutions given by the Mo-PVNS and meaning that our algorithm is able to better approximate the Pareto front. Notice that the small value of the epsilon indicator for the Mo-PVNS supports this statement. Similarly, Mo-PVNS obtains the smallest GD from the reference set and the best spread. In this latter case, Δ is similar in all the algorithms although the Mo-PVNS reaches a slightly smaller value. Details of the indicators for each problem separately can be found in the Appendix.

In relation to the execution time, we stated a stopping condition in Mo-PVNS. We let the algorithm run a maximum execution time of 1800 seconds, which is close to the average value of the fastest memetic approach of the state of the art. This way, we are able to compare the efficiency of the search spending a comparable amount of time. The analysis in terms of quality has been done above. Moreover, Table 5 also shows that the average execution time of Mo-PVNS is, aproximately, a 10% below the 1800 seconds time, which means that in many instances the algorithm ends the execution before reaching the deadline. The execution time of NSGA-II and SPEA2 is higher than the other algorithms, as it is explained in [4]. For future comparison, we also provide the detailed results of all instances in all the analyzed metrics in tables 9 to 14 in the Appendix section. Besides, Table 15 shows a breakdown of execution times for each instance and algorithm.

Table 6 Results of the nonparametric Kruskal-Wallis test

	#Sol.	HV	C(R,Alg.)	ϵ	GD	Δ
Mo-PVNS	**299.93**	**262.67**	**48.97**	**44.19**	**88.11**	**115.60**
AOLS	212.81	198.58	205.36	172.78	146.44	155.50
DBLS	226.18	201.13	202.31	166.74	138.83	154.11
NSGA-II	61.40	97.39	233.66	278.15	283.81	239.24
SPEA2	102.17	142.74	212.21	240.65	245.31	238.06
Chi-squared	250.75	105.29	172.88	213.57	174.87	81.75
p-value	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001

In order to statistically support that the performance among the five algorithms is different, for any of the considered indicators, nonparametric tests have been applied. To that end, the Kruskal-Wallis test checks if the five samples (solutions of the indicators obtained with the five algorithms) belong to five different populations. Specifically, Kruskal-Wallis test computes the averages of the ranks of each sample. Then, if the averages of the ranks are similar, it can be stated that they belong to the same population, otherwise, they do not belong to the same population. Rows 2 till 6 of Table 6 include the averages of the ranks of the Kruskal-Wallis test for each indicator (see columns 2–7). Row 7 includes the value of the statistic for future comparisons, and Row 8 includes the obtained p-values. For any of the indicators, the results of the p-values are close to zero (<0.0001), see Row 8 of Table 6. Then, this clearly indicates that there is enough statistical evidence to confirm that there are differences among the five algorithms. To get a visual idea of how superior is the Mo-PVNS with regard to the remaining algorithms, for each indicator, the best average of the ranks is depicted in bold font.

Nevertheless, the Wilcoxon test is also included to statistically prove that the Mo-PVNS performs always better than any of the other algorithms and perform a more comprehensive analysis. To that end, this test is carried out comparing every algorithm against the Mo-PVNS algorithm. In Table 7 the values of the statistic for the Wilcoxon test are shown, including the p-values in brackets. We have used one-tailed Wilcoxon tests that check if the results obtained by the Mo-PVNS are better than the other considered algorithms (AOLS, DBLS, NSGA-II and SPEA2). All p-values are equal to 1, except for two cases of the Δ indicator, being larger than 0.99, and confirming that the Mo-PVNS is indeed better than the AOLS, DBLS, NSGA-II or SPEA2 algorithms. The statistic of the Wilcoxon test is computed as the sum of the positive ranks of the absolute differences among the two samples (in our case, solutions of the indicators obtained with the two compared algorithms).

Finally, we conduct an experiment to evaluate the robustness of the proposed algorithm. In particular, we test the impact of the inherent stochastic nature of Mo-VNS which provides diversification. The initial hypothesis is that the randomness in Mo-PVNS does not produce drastically different solutions depending on the initial random seed on each execution. In order to confirm this hypothesis, we have performed 30 independent executions of the Mo-PVNS proposal. Then, we have created a reference

Table 7 Statistics of the nonparametric Wilcoxon test

Mo-PVNS vs	#Sol.	HV	C(R,Alg.)	ϵ	GD	Δ
AOLS	4456.5 (1)	3827.5 (1)	275 (1)	186 (1)	1452.5 (1)	1861.5 (0.9983)
DBLS	4377.5 (1)	3785.5 (1)	311 (1)	266.5 (1)	1525.5 (1)	1927.5 (0.9961)
NSGA-II	5120 (1)	4503 (1)	194 (1)	42 (1)	267 (1)	953 (1)
SPEA2	5013 (1)	4168 (1)	117.5 (1)	59 (1)	471 (1)	953 (1)

Table 8 Standard deviation of the considered metrics over 30 independent runs

Metric	SD
HV	0
C(R, Alg.)	0.18
ϵ	4.43
GD	23.51
Δ	0

front with the union of all the sets of non-dominated solutions, analyzing the standard deviation of the previously considered metrics in relation to the reference front. If the Mo-PVNS is robust against randomness, then the standard deviation will be small in all the metrics. Table 8 shows the standard deviation for all the multi-objective metrics previously analyzed.

One of the most important results to analyze is the deviation of the hypervolume. In this case, as it is equal to 0, it indicates that all the non-dominated fronts present the same hypervolume. The same result is obtained by the Δ metric. Hence, the robustness of the algorithm is confirmed in these metrics. In the case of coverage, the deviation of 0.18 indicates that all the non-dominated fronts are rather similar. The deviation of ϵ is slightly larger with respect to the average value (6.88). Finally, the deviation of GD is also small because it represents a 0.5% of the average value, 4743.56. Therefore, these results confirm the hypothesis: the Mo-PVNS algorithm is not affected by the inherent randomness of the process.

4 Conclusions and future research

This paper generalizes the Variable Neighborhood Search algorithm (VNS) to solve a bi-objective optimization problem known as the bi-objective obnoxious p-median problem, *Bi-OpM*. To that end, the VNS approach is designed to take into account two conflicting objectives: to maximize the sum of the distances between demand points and their nearest obnoxious facility and, to maximize the dispersion among obnoxious facilities. The interest of this problem appears because the *Bi-OpM* fits in many realistic situations where it is desired to locate facilities as far as possible from the demand points and among them.

We have coined the algorithm Multi-objective Parallel Variable Neighborhood Search (Mo-PVNS). The Mo-PVNS includes two new features: an adaptation of the

VNS, which has been mainly considered to deal single-objective optimization problems, to this multi-objective optimization problem, and an adaptation of the Replicated Shaking VNS design presented in [14] in order to increase the explored search space in a multi-objective scenario. Although VNS has been already adapted for tackling multi-objective problems [15], this work proposes a generalization of VNS for tackling multi-objective problems which mainly relies on the idea of considering the complete Pareto Front as the incumbent solution for VNS.

Computational results show the superiority of Mo-PVNS against the state-of-art algorithms to solve the *Bi-OpM*. To that end we have used the same set of instances but, furthermore, the set has been enlarged from 8 to 72 instances. Results obtained by the Mo-PVNS algorithm outperform, in most of the instances, the other considered algorithms: the two variants of the MOMA (Multi-Objective Memetic Algorithm), namely DBLS (Dominance-Based Local Search) and AOLS (Alternate Objectives Local Search), and the classical NSGA-II (the Non-dominated Sorting Genetic Algorithm II) and Strength-Pareto Evolutionary Algorithm 2 (SPEA2) algorithms in shorter computational times.

As future work, it would be interesting to solve an extension of the *Bi-OpM* which will include an additional objective function becoming in a multi-objective obnoxious p-median problem *Mo-OpM*. The idea is to optimize three objective functions: to maximize the sum of the minimum distances between each demand point and its nearest facility and maximize the sum of the minimum distances between two facilities as in the *Bi-OpM* but also to minimize the number of demand points affected (or covered) by the facilities. This is a natural generalization of the *Bi-OpM* since it is crucial to consider the number of citizen affected by the obnoxious facilities. Furthermore, we would like to discuss and compare a variant of the *Bi-OpM* in which the objectives will be to maximize the minimum distance between each demand point and its nearest obnoxious facility and maximize the minimum distance between two obnoxious facilities. In this way we are ensuring a minimum distance between each pair of demand points and obnoxious facilities and between each pairs of two obnoxious facilities.

Acknowledgements Ana D. López-Sánchez acknowledges support from the Spanish Ministerio de Economía, Industria y Competitividad through Grant Number ECO2016-76567-C4-1-R. Jesús Sánchez-Oro and J. Manuel Colmenar acknowledge support from the Spanish Ministerio de Ciencia, Innovación y Universidades (MCIU / AEI / FEDER, UE) under Grant Ref. PGC2018-095322-B-C22, and Comunidad de Madrid y Fondos Estructurales de la Unión Europea with Grant Ref. P2018/TCS-4566.

Appendix

In this section we provide a breakdown of results for each algorithm and instance on all the metrics we have shown in the paper (See Tables 9, 10, 11, 12, 13, 14, 15).

Table 9 Number of efficient solutions obtained for each algorithm on each instance

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed17.p25	96	87	87	81	70
pmed17.p50	207	156	168	29	126
pmed17.p100	317	128	158	34	43
pmed18.p25	43	43	43	42	41
pmed18.p50	241	150	167	43	101
pmed18.p100	311	131	154	31	26
pmed19.p25	109	105	103	99	91
pmed19.p50	206	138	162	28	102
pmed19.p100	299	134	147	30	37
pmed20.p25	74	75	74	71	64
pmed20.p50	219	113	117	50	103
pmed20.p100	373	176	195	33	43
pmed21.p31	58	58	58	54	49
pmed21.p62	166	76	84	13	37
pmed21.p125	305	135	124	29	32
pmed22.p31	114	100	112	91	89
pmed22.p62	259	127	151	17	52
pmed22.p125	347	173	182	23	34
pmed23.p31	95	91	91	88	84
pmed23.p62	251	123	130	18	52
pmed23.p125	302	139	114	32	33
pmed24.p31	111	105	110	100	93
pmed24.p62	277	125	151	19	48
pmed24.p125	338	166	159	21	32
pmed25.p31	37	39	39	36	38
pmed25.p62	179	90	97	16	35
pmed25.p125	347	154	146	25	30
pmed26.p37	80	75	75	61	72
pmed26.p75	270	97	109	19	29
pmed26.p150	344	114	115	23	24
pmed27.p37	72	70	72	57	70
pmed27.p75	220	93	96	18	19
pmed27.p150	365	117	142	22	30
pmed28.p37	54	46	48	42	42
pmed28.p75	262	99	111	13	33
pmed28.p150	334	142	135	15	22
pmed29.p37	126	106	105	55	106

Table 9 continued

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed29.p75	229	117	120	15	33
pmed29.p150	371	161	144	29	35
pmed30.p37	118	79	85	55	74
pmed30.p75	257	116	112	15	31
pmed30.p150	347	147	126	23	28
pmed31.p43	130	88	105	22	86
pmed31.p87	277	109	127	12	26
pmed31.p175	384	147	152	24	34
pmed32.p43	187	86	104	32	97
pmed32.p87	292	101	98	18	17
pmed32.p175	385	126	128	22	20
pmed33.p43	186	92	124	33	104
pmed33.p87	285	86	96	26	30
pmed33.p175	462	157	170	13	30
pmed34.p43	144	74	80	33	84
pmed34.p87	277	108	109	11	22
pmed34.p175	444	145	152	30	24
pmed35.p50	195	79	92	19	77
pmed35.p100	264	84	86	10	12
pmed35.p200	325	86	131	14	18
pmed36.p50	187	73	84	18	66
pmed36.p100	276	75	106	14	16
pmed36.p200	390	112	140	22	28
pmed37.p50	175	86	96	19	89
pmed37.p100	303	96	109	14	29
pmed37.p200	420	139	137	26	31
pmed38.p56	192	79	77	12	43
pmed38.p112	253	75	81	11	12
pmed38.p225	432	109	123	12	21
pmed39.p56	177	81	76	13	48
pmed39.p112	288	82	95	17	7
pmed39.p225	329	107	108	19	27
pmed40.p56	272	102	129	15	69
pmed40.p112	340	120	118	13	17
pmed40.p225	461	129	145	26	28

Table 10 Hypervolume value of the final set of non-dominated solutions for each algorithm on each instance

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed17.p25	0.7538	0.7779	0.7759	0.7741	0.7747
pmed17.p50	0.7543	0.7500	0.7524	0.6294	0.7315
pmed17.p100	0.7499	0.7214	0.7249	0.3402	0.4878
pmed18.p25	0.7648	0.7648	0.7648	0.7640	0.7643
pmed18.p50	0.7373	0.7177	0.7244	0.5830	0.7061
pmed18.p100	0.7620	0.7373	0.7413	0.3262	0.4813
pmed19.p25	0.7544	0.7542	0.7540	0.7533	0.7530
pmed19.p50	0.7559	0.7454	0.7490	0.6133	0.7251
pmed19.p100	0.6993	0.6691	0.6628	0.2974	0.4163
pmed20.p25	0.7138	0.7138	0.7138	0.7135	0.7119
pmed20.p50	0.7291	0.7214	0.7209	0.5792	0.7124
pmed20.p100	0.6744	0.6453	0.6481	0.3559	0.4563
pmed21.p31	0.7794	0.7795	0.7795	0.7756	0.7762
pmed21.p62	0.7606	0.7311	0.7401	0.3496	0.6590
pmed21.p125	0.7286	0.6676	0.6690	0.2005	0.3315
pmed22.p31	0.7854	0.7855	0.7853	0.7787	0.7840
pmed22.p62	0.7831	0.7611	0.7671	0.4926	0.6959
pmed22.p125	0.7161	0.6592	0.6616	0.2791	0.3841
pmed23.p31	0.7360	0.7365	0.7367	0.7347	0.7355
pmed23.p62	0.7416	0.7178	0.7262	0.4249	0.6585
pmed23.p125	0.7056	0.6543	0.6397	0.1942	0.3157
pmed24.p31	0.7697	0.7694	0.7697	0.7680	0.7690
pmed24.p62	0.7069	0.6737	0.6838	0.4138	0.6147
pmed24.p125	0.7097	0.6482	0.6570	0.2677	0.3666
pmed25.p31	0.8272	0.8277	0.8277	0.8272	0.8274
pmed25.p62	0.7393	0.7130	0.7128	0.3598	0.6427
pmed25.p125	0.6987	0.6334	0.6459	0.2417	0.3582
pmed26.p37	0.7156	0.7630	0.7626	0.7500	0.7627
pmed26.p75	0.7610	0.7106	0.7151	0.3149	0.5420
pmed26.p150	0.7606	0.6622	0.6778	0.2033	0.3129
pmed27.p37	0.7746	0.7744	0.7746	0.7705	0.7744
pmed27.p75	0.7753	0.7150	0.7158	0.2597	0.5279
pmed27.p150	0.7206	0.6212	0.6381	0.1591	0.2629
pmed28.p37	0.7979	0.7945	0.7969	0.7760	0.7940
pmed28.p75	0.7443	0.7133	0.7201	0.3341	0.5634
pmed28.p150	0.6785	0.5744	0.5768	0.1445	0.2488
pmed29.p37	0.7729	0.7678	0.7691	0.7398	0.7706
pmed29.p75	0.7767	0.7142	0.7157	0.3350	0.5628
pmed29.p150	0.6962	0.5967	0.6035	0.1928	0.2834

Table 10 continued

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed30.p37	0.7970	0.7896	0.7884	0.7521	0.7904
pmed30.p75	0.7090	0.6405	0.6378	0.2588	0.4928
pmed30.p150	0.6897	0.5961	0.5921	0.1488	0.2550
pmed31.p43	0.7607	0.7447	0.7516	0.6251	0.7499
pmed31.p87	0.7581	0.6821	0.6947	0.2176	0.4381
pmed31.p175	0.7001	0.5840	0.5901	0.1455	0.2292
pmed32.p43	0.8007	0.7769	0.7752	0.6703	0.7841
pmed32.p87	0.7675	0.6749	0.6810	0.2266	0.4382
pmed32.p175	0.7090	0.5742	0.5579	0.1308	0.2243
pmed33.p43	0.7522	0.7314	0.7360	0.6187	0.7244
pmed33.p87	0.7636	0.6617	0.6655	0.2741	0.4831
pmed33.p175	0.6845	0.5686	0.5886	0.1682	0.2446
pmed34.p43	0.7513	0.7262	0.7269	0.6177	0.7269
pmed34.p87	0.7245	0.6387	0.6248	0.2078	0.3986
pmed34.p175	0.7011	0.5859	0.5908	0.1809	0.2639
pmed35.p50	0.7344	0.6698	0.6744	0.4492	0.6673
pmed35.p100	0.7972	0.6602	0.6490	0.1250	0.3334
pmed35.p200	0.7410	0.5349	0.5467	0.0721	0.1500
pmed36.p50	0.7419	0.6904	0.6952	0.4872	0.7119
pmed36.p100	0.7542	0.6167	0.6270	0.1250	0.3184
pmed36.p200	0.6868	0.4936	0.5031	0.1081	0.1770
pmed37.p50	0.7276	0.7029	0.7047	0.4327	0.6781
pmed37.p100	0.7025	0.5649	0.5776	0.1215	0.3122
pmed37.p200	0.7175	0.5414	0.5584	0.1411	0.2135
pmed38.p56	0.7874	0.7051	0.7091	0.3823	0.6796
pmed38.p112	0.8038	0.6602	0.6387	0.0787	0.2888
pmed38.p225	0.7233	0.5226	0.5258	0.0655	0.1319
pmed39.p56	0.7618	0.7272	0.7202	0.3482	0.6751
pmed39.p112	0.7571	0.6213	0.6289	0.0505	0.2771
pmed39.p225	0.7127	0.4760	0.4708	0.0484	0.1070
pmed40.p56	0.7415	0.7101	0.7186	0.3897	0.6560
pmed40.p112	0.7563	0.6432	0.6442	0.1468	0.3225
pmed40.p225	0.7082	0.5136	0.5080	0.1309	0.1979

Table 11 Coverage measure of the final set of non-dominated solutions for each algorithm on each instance

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed17.p25	0.8333	0.0000	0.0230	0.0000	0.1571
pmed17.p50	0.4879	0.7115	0.5714	1.0000	0.3730
pmed17.p100	0.0032	0.9844	1.0000	1.0000	1.0000
pmed18.p25	0.0000	0.0000	0.0000	0.0000	0.0488
pmed18.p50	0.0000	0.9733	0.9162	1.0000	0.9307
pmed18.p100	0.0064	1.0000	0.9870	1.0000	1.0000
pmed19.p25	0.0000	0.0571	0.0583	0.0707	0.1978
pmed19.p50	0.0000	0.8478	0.7963	1.0000	0.8431
pmed19.p100	0.0000	1.0000	1.0000	1.0000	1.0000
pmed20.p25	0.0000	0.0267	0.0000	0.0141	0.1406
pmed20.p50	0.5342	0.8142	0.8205	1.0000	0.6796
pmed20.p100	0.0000	1.0000	1.0000	1.0000	1.0000
pmed21.p31	0.0000	0.0000	0.0172	0.0370	0.0612
pmed21.p62	0.0000	0.9737	0.9286	1.0000	1.0000
pmed21.p125	0.0000	1.0000	1.0000	1.0000	1.0000
pmed22.p31	0.0702	0.0100	0.0804	0.1758	0.2697
pmed22.p62	0.0000	0.9921	0.9669	1.0000	1.0000
pmed22.p125	0.0000	1.0000	1.0000	1.0000	1.0000
pmed23.p31	0.1053	0.0330	0.0000	0.1250	0.1429
pmed23.p62	0.1076	0.9675	0.8846	1.0000	1.0000
pmed23.p125	0.0000	1.0000	1.0000	1.0000	1.0000
pmed24.p31	0.0000	0.0286	0.0091	0.0300	0.1398
pmed24.p62	0.0000	1.0000	0.9868	1.0000	1.0000
pmed24.p125	0.0000	1.0000	1.0000	1.0000	1.0000
pmed25.p31	0.0541	0.0000	0.0000	0.0000	0.0263
pmed25.p62	0.0000	0.9778	0.9897	1.0000	1.0000
pmed25.p125	0.0000	1.0000	1.0000	1.0000	1.0000
pmed26.p37	0.9875	0.0133	0.0133	0.4754	0.0556
pmed26.p75	0.0000	1.0000	1.0000	1.0000	1.0000
pmed26.p150	0.0000	1.0000	1.0000	1.0000	1.0000
pmed27.p37	0.0000	0.0143	0.0000	0.2456	0.0286
pmed27.p75	0.0000	1.0000	1.0000	1.0000	1.0000
pmed27.p150	0.0000	1.0000	1.0000	1.0000	1.0000
pmed28.p37	0.0000	0.1087	0.0417	0.2619	0.0952
pmed28.p75	0.2023	0.9495	0.7748	1.0000	1.0000
pmed28.p150	0.0000	1.0000	1.0000	1.0000	1.0000
pmed29.p37	0.0000	0.4057	0.3143	0.8182	0.3774
pmed29.p75	0.0000	1.0000	1.0000	1.0000	1.0000
pmed29.p150	0.0000	1.0000	1.0000	1.0000	1.0000
pmed30.p37	0.0000	0.5696	0.5765	0.9273	0.5000

Table 11 continued

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed30.p75	0.0000	1.0000	1.0000	1.0000	1.0000
pmed30.p150	0.0000	1.0000	1.0000	1.0000	1.0000
pmed31.p43	0.0000	0.8295	0.5619	1.0000	0.3140
pmed31.p87	0.0000	1.0000	1.0000	1.0000	1.0000
pmed31.p175	0.0000	1.0000	1.0000	1.0000	1.0000
pmed32.p43	0.0000	0.9884	1.0000	1.0000	0.7216
pmed32.p87	0.0000	1.0000	1.0000	1.0000	1.0000
pmed32.p175	0.0000	1.0000	1.0000	1.0000	1.0000
pmed33.p43	0.0000	0.9565	0.9113	1.0000	0.7404
pmed33.p87	0.0000	1.0000	1.0000	1.0000	1.0000
pmed33.p175	0.0000	1.0000	1.0000	1.0000	1.0000
pmed34.p43	0.0000	0.9459	0.9750	1.0000	0.7143
pmed34.p87	0.0000	1.0000	1.0000	1.0000	1.0000
pmed34.p175	0.0000	1.0000	1.0000	1.0000	1.0000
pmed35.p50	0.0000	1.0000	1.0000	1.0000	0.9221
pmed35.p100	0.0000	1.0000	1.0000	1.0000	1.0000
pmed35.p200	0.0000	1.0000	1.0000	1.0000	1.0000
pmed36.p50	0.0000	1.0000	1.0000	1.0000	0.8333
pmed36.p100	0.0000	1.0000	1.0000	1.0000	1.0000
pmed36.p200	0.0000	1.0000	1.0000	1.0000	1.0000
pmed37.p50	0.0057	0.9767	0.9896	1.0000	0.7753
pmed37.p100	0.0000	1.0000	1.0000	1.0000	1.0000
pmed37.p200	0.0000	1.0000	1.0000	1.0000	1.0000
pmed38.p56	0.0000	1.0000	1.0000	1.0000	0.9767
pmed38.p112	0.0000	1.0000	1.0000	1.0000	1.0000
pmed38.p225	0.0000	1.0000	1.0000	1.0000	1.0000
pmed39.p56	0.0621	0.9136	0.9737	1.0000	1.0000
pmed39.p112	0.0000	1.0000	1.0000	1.0000	1.0000
pmed39.p225	0.0000	1.0000	1.0000	1.0000	1.0000
pmed40.p56	0.0404	1.0000	0.9845	1.0000	0.9420
pmed40.p112	0.0000	1.0000	1.0000	1.0000	1.0000
pmed40.p225	0.0000	1.0000	1.0000	1.0000	1.0000

Table 12 ϵ indicator of the final set of non-dominated solutions for each algorithm on each instance

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed17.p25	0.0402	0.0024	0.0182	0.0328	0.0279
pmed17.p50	0.0126	0.0166	0.0134	0.1636	0.0711
pmed17.p100	0.0009	0.0416	0.0324	0.3719	0.2303
pmed18.p25	0.0000	0.0000	0.0000	0.0129	0.0137
pmed18.p50	0.0000	0.0308	0.0178	0.1735	0.0630
pmed18.p100	0.0021	0.0337	0.0296	0.4030	0.2901
pmed19.p25	0.0000	0.0048	0.0048	0.0112	0.0112
pmed19.p50	0.0000	0.0181	0.0138	0.1828	0.0605
pmed19.p100	0.0000	0.0398	0.0389	0.3829	0.2757
pmed20.p25	0.0000	0.0018	0.0000	0.0076	0.0143
pmed20.p50	0.0118	0.0235	0.0170	0.1627	0.0386
pmed20.p100	0.0000	0.0439	0.0446	0.3411	0.2405
pmed21.p31	0.0098	0.0052	0.0052	0.0350	0.0182
pmed21.p62	0.0000	0.0402	0.0241	0.3876	0.1476
pmed21.p125	0.0000	0.0823	0.0746	0.5140	0.3969
pmed22.p31	0.0098	0.0107	0.0080	0.0402	0.0134
pmed22.p62	0.0000	0.0359	0.0221	0.3204	0.1436
pmed22.p125	0.0000	0.0956	0.1015	0.4474	0.3504
pmed23.p31	0.0113	0.0113	0.0049	0.0199	0.0119
pmed23.p62	0.0073	0.0272	0.0303	0.3243	0.1386
pmed23.p125	0.0000	0.0637	0.0834	0.5614	0.3928
pmed24.p31	0.0000	0.0069	0.0026	0.0220	0.0086
pmed24.p62	0.0000	0.0441	0.0278	0.3200	0.1453
pmed24.p125	0.0000	0.0943	0.0779	0.4451	0.3905
pmed25.p31	0.0048	0.0016	0.0027	0.0048	0.0110
pmed25.p62	0.0000	0.0267	0.0457	0.4332	0.1993
pmed25.p125	0.0000	0.0728	0.0842	0.4653	0.3612
pmed26.p37	0.0510	0.0127	0.0021	0.0494	0.0127
pmed26.p75	0.0000	0.0556	0.0483	0.4702	0.2508
pmed26.p150	0.0000	0.1122	0.1012	0.5208	0.4312
pmed27.p37	0.0000	0.0069	0.0000	0.0184	0.0083
pmed27.p75	0.0000	0.0750	0.0574	0.5306	0.3065
pmed27.p150	0.0000	0.1456	0.1084	0.6055	0.4739
pmed28.p37	0.0000	0.0218	0.0138	0.0509	0.0195
pmed28.p75	0.0125	0.0704	0.0687	0.4561	0.2285
pmed28.p150	0.0000	0.1419	0.1170	0.5469	0.4299
pmed29.p37	0.0000	0.0215	0.0139	0.0644	0.0119
pmed29.p75	0.0000	0.0701	0.0537	0.3528	0.2188
pmed29.p150	0.0000	0.1182	0.0998	0.5144	0.4129
pmed30.p37	0.0073	0.0156	0.0168	0.0728	0.0219

Table 12 continued

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed30.p75	0.0000	0.0498	0.0690	0.4931	0.2761
pmed30.p150	0.0000	0.1469	0.1659	0.6292	0.5143
pmed31.p43	0.0000	0.0433	0.0273	0.1565	0.0410
pmed31.p87	0.0000	0.0964	0.0719	0.6095	0.3938
pmed31.p175	0.0000	0.1365	0.1080	0.5519	0.4707
pmed32.p43	0.0015	0.0264	0.0322	0.1595	0.0337
pmed32.p87	0.0000	0.0762	0.0825	0.5542	0.3774
pmed32.p175	0.0000	0.1368	0.1774	0.6014	0.4654
pmed33.p43	0.0000	0.0471	0.0219	0.2159	0.0968
pmed33.p87	0.0000	0.1118	0.1005	0.4217	0.2572
pmed33.p175	0.0000	0.1685	0.1292	0.5969	0.4854
pmed34.p43	0.0000	0.0412	0.0373	0.1933	0.0824
pmed34.p87	0.0000	0.0761	0.1162	0.5827	0.3854
pmed34.p175	0.0000	0.1563	0.1675	0.5721	0.4752
pmed35.p50	0.0000	0.0729	0.0506	0.3105	0.1535
pmed35.p100	0.0000	0.1036	0.1085	0.6910	0.4640
pmed35.p200	0.0000	0.2380	0.2057	0.7307	0.6347
pmed36.p50	0.0000	0.0573	0.0507	0.2082	0.0683
pmed36.p100	0.0000	0.1644	0.1151	0.6727	0.4652
pmed36.p200	0.0000	0.2311	0.2011	0.6120	0.5339
pmed37.p50	0.0038	0.0328	0.0267	0.3559	0.1374
pmed37.p100	0.0000	0.2043	0.1656	0.6974	0.4783
pmed37.p200	0.0000	0.2331	0.1959	0.6273	0.5281
pmed38.p56	0.0000	0.0742	0.0913	0.4034	0.1663
pmed38.p112	0.0000	0.1448	0.1678	0.7692	0.5179
pmed38.p225	0.0000	0.2477	0.2326	0.6910	0.6272
pmed39.p56	0.0087	0.0436	0.0590	0.4242	0.1465
pmed39.p112	0.0000	0.1499	0.1231	0.7806	0.4757
pmed39.p225	0.0000	0.2748	0.2583	0.7151	0.6333
pmed40.p56	0.0046	0.0445	0.0320	0.4417	0.1887
pmed40.p112	0.0000	0.1468	0.1438	0.6890	0.4991
pmed40.p225	0.0000	0.2554	0.2610	0.6315	0.5478

Table 13 Generational Distance of the final set of non-dominated solutions for each algorithm on each instance

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed17.p25	682.6014	718.2675	717.6131	742.6109	799.3765
pmed17.p50	366.5348	420.5569	406.5507	970.7395	470.3925
pmed17.p100	222.9321	349.6170	315.3498	626.9825	578.5875
pmed18.p25	1126.2960	1126.2960	1126.2960	1141.1869	1153.2455
pmed18.p50	365.0371	459.4856	437.7471	842.9593	561.7705
pmed18.p100	246.7130	378.7096	349.9274	715.7064	815.3289
pmed19.p25	645.8529	660.2317	666.7024	679.3313	707.0364
pmed19.p50	373.1332	455.5424	422.2088	998.3336	531.4072
pmed19.p100	228.2869	336.9255	322.1547	667.2333	616.5681
pmed20.p25	818.3851	812.0056	818.3851	836.6844	877.2007
pmed20.p50	361.0605	498.8161	498.2598	739.2626	532.7560
pmed20.p100	208.8278	301.2730	286.8968	649.8956	585.0368
pmed21.p31	910.1904	911.5798	910.1940	945.4771	990.2287
pmed21.p62	441.0065	650.8282	617.4048	1506.6107	928.2819
pmed21.p125	236.8979	352.7487	368.5287	679.8536	676.4789
pmed22.p31	696.1699	742.4685	700.8017	776.7884	785.5626
pmed22.p62	368.4225	523.0520	483.4876	1370.7647	819.3678
pmed22.p125	234.7197	329.0590	320.7949	830.8207	704.1327
pmed23.p31	745.4123	762.3869	761.6229	773.3442	792.9854
pmed23.p62	360.5358	514.6290	500.3929	1267.0752	791.0246
pmed23.p125	244.6033	357.2316	393.8264	668.3395	682.6403
pmed24.p31	670.4326	689.1839	673.5149	709.2484	733.4901
pmed24.p62	317.8548	473.1110	430.9972	1155.6913	756.2919
pmed24.p125	227.9126	319.9483	326.6105	820.9291	692.1546
pmed25.p31	1207.2365	1177.0556	1179.2726	1225.6926	1192.8628
pmed25.p62	430.7704	608.9646	585.5257	1381.6618	967.7145
pmed25.p125	226.1971	336.4071	344.1785	755.6140	716.8575
pmed26.p37	879.7379	910.7808	910.5640	1008.3537	928.9677
pmed26.p75	349.9560	578.2535	547.1109	1218.8545	1032.5665
pmed26.p150	236.6365	403.0100	402.9871	792.8840	810.4364
pmed27.p37	876.5294	889.0487	876.5294	983.2731	888.4462
pmed27.p75	380.8768	580.5014	575.2429	1228.2739	1268.5891
pmed27.p150	223.9964	390.0619	353.6005	795.9591	704.5608
pmed28.p37	969.3420	1048.8186	1028.6926	1100.5853	1103.5470
pmed28.p75	332.6085	537.1208	509.7066	1411.6300	921.4635
pmed28.p150	226.3381	339.4269	347.3296	950.6045	807.6491
pmed29.p37	655.2992	720.2695	723.0062	994.4825	716.8821
pmed29.p75	363.4618	508.1099	498.1451	1331.4896	930.0311
pmed29.p150	213.1853	318.5955	337.5243	682.1500	636.0246
pmed30.p37	686.8881	841.7132	811.9352	1004.7000	866.0828

Table 13 continued

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed30.p75	363.5582	534.1136	544.5783	1406.0311	1019.5077
pmed30.p150	236.3526	357.5844	385.7675	807.6625	762.8214
pmed31.p43	642.3415	783.3927	714.8207	1548.9224	791.6145
pmed31.p87	330.4778	519.8974	482.2298	1436.9312	1019.8650
pmed31.p175	208.6309	329.0982	325.6057	731.2962	627.9198
pmed32.p43	538.8510	792.9545	719.1834	1276.6904	745.4626
pmed32.p87	342.2828	572.9157	584.8245	1226.1514	1341.0397
pmed32.p175	223.2873	381.2530	376.2404	810.4169	881.7499
pmed33.p43	513.5106	728.9604	629.7209	1205.0228	682.4682
pmed33.p87	328.4795	583.7081	558.3950	983.2009	975.7037
pmed33.p175	198.4986	331.8241	319.6863	1017.2745	695.1418
pmed34.p43	595.2837	826.3426	794.2950	1226.4950	776.8750
pmed34.p87	333.6813	527.5438	522.2113	1525.0196	1134.0517
pmed34.p175	201.2278	344.0923	334.1119	673.0675	786.2632
pmed35.p50	509.0748	790.5519	732.2403	1569.5109	798.1588
pmed35.p100	343.2090	596.6679	590.8014	1546.2968	1500.2466
pmed35.p200	231.8542	436.2805	354.0421	950.4703	874.0185
pmed36.p50	566.3094	905.4624	845.2145	1781.4894	959.7074
pmed36.p100	360.9054	679.9167	570.2162	1433.7374	1415.4989
pmed36.p200	227.1976	410.8491	367.3732	827.5135	759.8592
pmed37.p50	597.7990	850.1961	807.1015	1755.7718	832.6347
pmed37.p100	350.4797	607.9668	572.8995	1440.4825	1063.1787
pmed37.p200	218.6207	368.7275	371.5727	760.7885	727.7165
pmed38.p56	526.8598	811.7415	826.6376	1978.8671	1099.4753
pmed38.p112	361.0170	647.7181	624.1910	1498.6687	1531.6331
pmed38.p225	214.7684	413.2992	389.1924	1082.8317	852.9001
pmed39.p56	547.5834	805.7422	829.7982	1939.8672	1042.6890
pmed39.p112	344.2937	634.8333	592.1237	1227.5681	2058.1886
pmed39.p225	240.7103	408.5102	405.8770	846.0695	735.2185
pmed40.p56	464.3982	757.5564	672.3495	1871.3496	911.2670
pmed40.p112	335.5068	554.9588	561.0442	1471.1939	1363.9258
pmed40.p225	220.4799	401.3087	377.5764	791.9202	790.2374

Table 14 Δ Generalized Spread of the final set of non-dominated solutions for each algorithm on each instance

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed17.p25	0.9756	0.9743	0.9717	0.9721	0.9769
pmed17.p50	0.9508	0.9463	0.9485	0.9964	0.9604
pmed17.p100	0.9228	0.9399	0.9394	0.9409	0.9813
pmed18.p25	0.9729	0.9729	0.9729	0.9734	0.9714
pmed18.p50	0.9471	0.9460	0.9518	0.9663	0.9661
pmed18.p100	0.9291	0.9503	0.9383	0.9666	0.9599
pmed19.p25	0.9587	0.9625	0.9626	0.9675	0.9548
pmed19.p50	0.9407	0.9445	0.9479	0.9578	0.9753
pmed19.p100	0.9191	0.9341	0.9448	0.9679	0.9720
pmed20.p25	0.9694	0.9677	0.9694	0.9689	0.9687
pmed20.p50	0.9350	0.9371	0.9347	0.9652	0.9566
pmed20.p100	0.9058	0.9256	0.9216	0.9615	0.9479
pmed21.p31	0.9761	0.9771	0.9766	0.9797	0.9732
pmed21.p62	0.9669	0.9656	0.9691	0.9729	0.9751
pmed21.p125	0.9227	0.9293	0.9477	0.9698	0.9802
pmed22.p31	0.9687	0.9715	0.9709	0.9696	0.9701
pmed22.p62	0.9502	0.9633	0.9515	0.9923	0.9695
pmed22.p125	0.9188	0.9410	0.9444	0.9599	0.9439
pmed23.p31	0.9725	0.9706	0.9706	0.9701	0.9661
pmed23.p62	0.9451	0.9634	0.9416	0.9665	0.9689
pmed23.p125	0.9271	0.9398	0.9265	0.9534	0.9482
pmed24.p31	0.9799	0.9792	0.9794	0.9794	0.9776
pmed24.p62	0.9436	0.9570	0.9441	0.9720	0.9610
pmed24.p125	0.9164	0.9327	0.9370	0.9525	0.9649
pmed25.p31	0.9884	0.9890	0.9879	0.9889	0.9890
pmed25.p62	0.9604	0.9665	0.9663	0.9858	0.9749
pmed25.p125	0.9149	0.9398	0.9390	0.9520	0.9421
pmed26.p37	0.9792	0.9826	0.9880	0.9785	0.9802
pmed26.p75	0.9525	0.9557	0.9615	0.9831	0.9794
pmed26.p150	0.9248	0.9532	0.9466	0.9654	0.9900
pmed27.p37	0.9850	0.9847	0.9850	0.9889	0.9859
pmed27.p75	0.9581	0.9627	0.9645	0.9701	0.9792
pmed27.p150	0.9226	0.9463	0.9438	0.9589	0.9616
pmed28.p37	0.9862	0.9823	0.9878	0.9835	0.9877
pmed28.p75	0.9463	0.9655	0.9655	0.9719	1.0020
pmed28.p150	0.9239	0.9384	0.9332	0.9635	0.9496
pmed29.p37	0.9707	0.9689	0.9791	0.9742	0.9744
pmed29.p75	0.9513	0.9577	0.9629	0.9808	0.9717

Table 14 continued

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed29.p150	0.9166	0.9338	0.9348	0.9831	0.9648
pmed30.p37	0.9756	0.9740	0.9785	0.9852	0.9768
pmed30.p75	0.9536	0.9566	0.9608	0.9759	0.9668
pmed30.p150	0.9258	0.9395	0.9290	0.9477	0.9681
pmed31.p43	0.9781	0.9771	0.9761	0.9793	0.9766
pmed31.p87	0.9449	0.9568	0.9577	0.9818	0.9759
pmed31.p175	0.9182	0.9452	0.9212	0.9681	0.9658
pmed32.p43	0.9603	0.9775	0.9690	0.9756	0.9659
pmed32.p87	0.9406	0.9557	0.9557	0.9521	0.9791
pmed32.p175	0.9266	0.9374	0.9494	0.9589	0.9812
pmed33.p43	0.9611	0.9680	0.9659	0.9717	0.9741
pmed33.p87	0.9430	0.9549	0.9534	0.9709	0.9696
pmed33.p175	0.9060	0.9393	0.9249	0.9687	0.9676
pmed34.p43	0.9764	0.9808	0.9763	0.9830	0.9748
pmed34.p87	0.9477	0.9577	0.9655	0.9843	0.9648
pmed34.p175	0.9179	0.9482	0.9356	0.9487	0.9664
pmed35.p50	0.9684	0.9711	0.9716	0.9786	0.9782
pmed35.p100	0.9538	0.9596	0.9675	0.9747	0.9817
pmed35.p200	0.9315	0.9617	0.9473	0.9818	0.9802
pmed36.p50	0.9727	0.9776	0.9798	0.9803	0.9791
pmed36.p100	0.9486	0.9657	0.9751	0.9784	0.9782
pmed36.p200	0.9186	0.9493	0.9469	0.9499	0.9508
pmed37.p50	0.9722	0.9786	0.9770	0.9943	0.9857
pmed37.p100	0.9453	0.9643	0.9682	0.9735	0.9853
pmed37.p200	0.9196	0.9505	0.9524	0.9818	0.9854
pmed38.p56	0.9695	0.9764	0.9762	0.9730	0.9906
pmed38.p112	0.9593	0.9697	0.9726	1.0009	0.9737
pmed38.p225	0.9301	0.9599	0.9517	0.9589	0.9845
pmed39.p56	0.9698	0.9786	0.9774	0.9799	0.9853
pmed39.p112	0.9513	0.9617	0.9674	0.9897	0.9763
pmed39.p225	0.9416	0.9631	0.9566	0.9697	0.9762
pmed40.p56	0.9662	0.9759	0.9711	0.9891	0.9811
pmed40.p112	0.9396	0.9556	0.9591	0.9795	0.9742
pmed40.p225	0.9217	0.9478	0.9410	0.9762	0.9700

Table 15 Execution time (in sec) of the final set of non-dominated solutions for each algorithm on each instance

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed17.p25	559.983	369.612	379.108	1157.423	5144.575
pmed17.p50	1802.588	647.113	679.37	2648.074	6296.037
pmed17.p100	1811.224	1323.026	1403.544	7326.294	11880.241
pmed18.p25	92.487	347.191	350.027	1131.032	5013.282
pmed18.p50	1801.08	654.385	685.117	2666.113	6427.61
pmed18.p100	1809.333	1324.969	1386.195	7305.665	11844.258
pmed19.p25	448.756	368.043	368.555	1167.528	5328.069
pmed19.p50	1802.241	629.078	664.59	2658.088	6255.026
pmed19.p100	1810.26	1328.724	1381.216	7276.104	12127.677
pmed20.p25	205.215	364.279	349.17	1148.515	5320.73
pmed20.p50	1802.998	620.675	661.355	2708.265	6315.136
pmed20.p100	1800.735	1348.659	1399.609	7388.828	11791.64
pmed21.p31	348.614	502.4	511.384	1762.34	5767.847
pmed21.p62	1801.751	945.08	932.561	4421.521	8175.701
pmed21.p125	1802.054	1943.117	2008.801	12869.981	17970.712
pmed22.p31	1606.733	492.568	510.675	1732.194	5818.799
pmed22.p62	1801.13	943.297	990.107	4423.24	8235.734
pmed22.p125	1804.348	1905.451	1990.034	12611.729	17983.946
pmed23.p31	735.774	515.735	511.487	1759.389	5851.096
pmed23.p62	1800.561	933.461	968.829	4383.554	8234.126
pmed23.p125	1804.022	1995.179	2105.019	12668.718	18065.158
pmed24.p31	687.912	517.201	524.938	1759.696	5857.967
pmed24.p62	1800.748	960.589	995.015	4437.905	8345.57
pmed24.p125	1804.457	1965.128	2056.015	12647.184	18026.662
pmed25.p31	199.774	487.641	492.433	1750.982	5805.962
pmed25.p62	1802.596	899.387	946.281	4383.757	8360.203
pmed25.p125	1803.307	1931.682	1954.247	12609.062	17951.188
pmed26.p37	1118.214	665.78	737.047	2431.13	6041.432
pmed26.p75	1807.249	1326.218	1408.527	6723.191	10585.28
pmed26.p150	1807.973	2730.115	2773.622	19739.998	25929.23
pmed27.p37	754.313	713.257	742.613	2477.731	6061.52
pmed27.p75	1806.876	1333.999	1314.75	6748.584	10499.909
pmed27.p150	1806.931	2974.384	3074.018	19717.685	24901.292
pmed28.p37	896.867	638.057	671.094	2429.35	5998.978
pmed28.p75	1806.134	1254.454	1283.845	6726.993	10593.659
pmed28.p150	1805.361	2722.156	2750.252	19710.687	24931.718
pmed29.p37	1800.679	700.39	698.378	2512.337	6120.816
pmed29.p75	1806.135	1265.987	1328.631	6895.147	10988.397
pmed29.p150	1805.637	2717.277	2787.273	19841.529	24923.703

Table 15 continued

INSTANCE	Mo-PVNS	AOLS	DBLS	NSGA-II	SPEA2
pmed30.p37	1802.752	662.114	679.167	2364.12	6065.077
pmed30.p75	1806.544	1289.904	1292.473	6451.738	10502.388
pmed30.p150	1808.724	2715.869	2801.296	19835.739	25492.628
pmed31.p43	1803.061	884.717	921.729	3350.292	6891.541
pmed31.p87	1801.483	1702.271	1779.913	9232.899	13741.315
pmed31.p175	1813.708	3712.982	3868.433	28272.49	35200.497
pmed32.p43	1802.565	899.956	911.044	3285.824	6987.454
pmed32.p87	1802.363	1720.503	1735.69	9205.704	13678.591
pmed32.p175	1821.504	3644.142	3753.658	28263.174	35254.251
pmed33.p43	1800.874	888.341	955.148	3314.222	7014.248
pmed33.p87	1801.869	1667.102	1822.25	9227.759	13857.293
pmed33.p175	1818.281	3708.179	3869.516	28356.023	35259.477
pmed34.p43	1802.233	912.531	934.587	3565.131	6949.487
pmed34.p87	1802.89	1726.789	1757.17	9117.993	13689.109
pmed34.p175	1812.499	3863.333	3960.955	29173.071	35260.57
pmed35.p50	1807.055	1154.553	1217.844	4375.293	7961.46
pmed35.p100	1802.602	2133.513	2267.524	12730.265	17807.547
pmed35.p200	1819.452	4967.695	5095.762	39311.767	48200.588
pmed36.p50	1800.81	1202.682	1182.833	4431.054	8001.283
pmed36.p100	1805.465	2325.931	2329.336	12809.812	17802.27
pmed36.p200	1818.113	4998.88	5037.924	39346.173	48220.558
pmed37.p50	1807.431	1120.823	1145.495	4419.85	7742.646
pmed37.p100	1804.63	2140.454	2194.717	12788.378	17620.304
pmed37.p200	1829.425	4823.23	4950.464	39249.805	46698.738
pmed38.p56	1806.912	1392.809	1447.755	5973.063	9448.258
pmed38.p112	1803.072	2698.415	2803.455	17561.161	22492.692
pmed38.p225	1841.457	6244.761	6471.206	53841.901	62415.004
pmed39.p56	1803.702	1354.027	1498.687	5942.783	9440.823
pmed39.p112	1805.135	2907.285	2884.457	17545.61	22606.509
pmed39.p225	1837.069	6142.333	6331.688	53844.378	62390.012
pmed40.p56	1811.11	1506.756	1480.359	5964.141	9562.193
pmed40.p112	1808.26	2829.047	2994.584	17498.675	22642.241
pmed40.p225	2001.9	6281.163	6281.467	53785.068	62486.857

References

1. Carlsson, J.G., Jia, F.: Continuous facility location with backbone network costs. Transp. Sci. **49**(3), 433–451 (2014)
2. Coello, C.A.C., Lamont, G.B., Veldhuizen, D.A.V.: Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation). Springer, New York (2006)
3. Colmenar, J.M., Greistorfer, P., Martí, R., Duarte, A.: Advanced greedy randomized adaptive search procedure for the obnoxious p-median problem. Eur. J. Oper. Res. **252**(2), 432–442 (2016)
4. Colmenar, J.M., Martí, R., Duarte, A.: Multi-objective memetic optimization for the bi-objective obnoxious p-median problem. Knowl.-Based Syst. **144**, 88–101 (2018)
5. Cornuéjols, G., Nemhauser, G., Wolsey, L.: The uncapacitated facility location problem. In: Mirchandani, P.B., Francis, R.L. (eds.) Discrete Location Theory, pp. 119–171. Wiley, New York (1990)
6. Crainic, T.G., Gendreau, M., Hansen, P., Mladenović, N.: Cooperative parallel variable neighborhood search for the p-median. J. Heuristics **10**(3), 293–314 (2004)
7. Dantrakul, S., Likasiri, C., Pongvuthithum, R.: Applied p-median and p-center algorithms for facility location problems. Expert Syst. Appl. **41**(8), 3596–3604 (2014)
8. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
9. Drezner, Z., Brimberg, J., Mladenović, N., Salhi, S.: Solving the planar p-median problem by variable neighborhood and concentric searches. J. Global Optim. **63**(3), 501–514 (2015)
10. Duarte, A., Pantrigo, J., Pardo, E., Sánchez-Oro, J.: Parallel variable neighbourhood search strategies for the cutwidth minimization problem. IMA J. Manag. Math. **27**(1), 55 (2016)
11. Duarte, A., Pantrigo, J.J., Pardo, E.G., Mladenovic, N.: Multi-objective variable neighborhood search: an application to combinatorial optimization problems. J. Global Optim. **63**(3), 515–536 (2015)
12. Durillo, J., Nebro, A.: Jmetal: a java framework for multi-objective optimization. Adv. Eng. Softw. **42**, 760–771 (2011)
13. Feo, T.A., Resende, M.G.C.: Greedy randomized adaptive search procedures. J. Global Optim. **6**(2), 109–133 (1995)
14. García-López, F., Melian, B., Moreno-Pérez, J., Moreno-Vega, J.: The parallel variable neighborhood search for the p-median problem. J. Heuristics **8**, 375–388 (2002)
15. Geiger, M.J.: Randomised variable neighbourhood search for multi objective optimisation. arXiv preprint arXiv:0809.0271 (2008)
16. Hansen, P., Mladenović, N., Todosijević, R., Hanafi, S.: Variable neighborhood search: basics and variants. EURO J. Comput. Optim. **5**(3), 423–454 (2017)
17. Irawan, C.A., Salhi, S.: Solving large p-median problems by a multistage hybrid approach using demand points aggregation and variable neighbourhood search. J. Global Optim. **63**(3), 537–554 (2015)
18. López-Ibáñez, M., Dubois-Lacoste, J., Stützle, T., Birattari, M.: The irace package, iterated race for automatic algorithm configuration. Tech. Rep. TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium (2011)
19. Marín, A.: The discrete facility location problem with balanced allocation of customers. Eur. J. Oper. Res. **210**(1), 27–38 (2011)
20. Mladenović, N., Hansen, P.: Variable neighborhood search. Comput. Oper. Res. **24**(11), 1097–1100 (1997)
21. Sánchez-Oro, J., Sevaux, M., Rossi, A., Martí, R., Duarte, A.: Solving dynamic memory allocation problems in embedded systems with parallel variable neighborhood search strategies. Electron. Notes Dis. Math. **47**, 85–92 (2015)
22. Sánchez-Oro, J., Laguna, M., Duarte, A., Martí, R.: Scatter search for the profile minimization problem. Networks **65**(1), 10–21 (2015)
23. Wu, L.Y., Zhang, X.S., Zhang, J.L.: Capacitated facility location problem with general setup cost. Comput. Oper. Res. **33**(5), 1226–1241 (2006)

24. Zitzler, E., Laumanns, M., Thiele, L.: Spea2: Improving the strength pareto evolutionary algorithm. Technical report (2001)
25. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In: K. Giannakoglou, et al. (eds.) Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001), pp. 95–100. International Center for Numerical Methods in Engineering (CIMNE) (2002)