Computers & Operations Research 165 (2024) 106589

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

A parallel variable neighborhood search for α-neighbor facility location problems

Guilherme O. Chagas [a,*], Luiz A.N. Lorena [b,c], Rafael D.C. dos Santos [b], Jacques Renaud [a], Leandro C. Coelho [a,d]

[a] CIRRELT and Faculté des Sciences de l'Administration, Université Laval, Québec, Canada
[b] Instituto Nacional de Pesquisas Espaciais (INPE), Av. dos Astronautas, 1758, 12245-970, São José dos Campos, São Paulo, Brazil
[c] Universidade Federal de São Paulo (UNIFESP), Av. Cesare Mansueto Giulio Lattes, 1201, 12247-014, São José dos Campos, São Paulo, Brazil
[d] Canada Research Chair in Integrated Logistics, Université Laval, Québec, Canada

ARTICLE INFO

Keywords:
α-neighbor p-center
α-neighbor p-median
Basic VNS
Parallel VNS
LIMA

ABSTRACT

In this paper, we employ the *less is more approach* to develop a Parallel Variable Neighborhood Search (VNS) algorithm for the α-neighbor p-center problem (αNpCP) and the α-neighbor p-median problem (αNpMP). The αNpCP and the αNpMP are generalizations of the p-center (pCP) and p-median (pMP) problems, respectively. In the α-neighbor problems, one seeks to open p facilities and assign each of the n customers to their closest α ones. The objective is to minimize the maximum distance of a customer to its αth facility, in the case of the αNpCP, and the sum of the distances from each customer to their α nearest facilities, in the case of the αNpMP. Our VNS adapts simple but efficient algorithms and data structures from the pCP and pMP literature to the αNpCP and αNpMP context. We also introduce an updated objective function for the αNpCP, which adds more information to the solution cost and helps the VNS to escape from local optima. Several experimental tests show that our VNS outperforms more complex state-of-the-art algorithms. Regarding the αNpCP, on 120 instances derived from the OR-library set, our algorithm improved best-known solutions for 22, with an average improvement of 34.26%; the overall gap on the 120 instances is 6.18% in favor of our algorithm. Moreover, on 231 instances derived from the TSPLIB set, we improved the solutions for 115, with an average improvement of 5.30%, and an overall improvement gap of 2.47% for all 231 instances. Considering the αNpMP results, our heuristic obtained better results than a heuristic from literature in all 80 instances tested, finding optimal solutions in all these instances.

1. Introduction

Facility location problems are extensively studied and are an important topic in operations research (Daskin, 1995; Laporte et al., 2015). In such problems, one seeks to open facilities and assign each customer's demand to an opened one, optimizing an objective function typically composed of an assignment cost. These problems have several real-world applications, from logistics to data-mining (Ng and Han, 1994; Hansen et al., 2009; Laporte et al., 2015; Grangier et al., 2016; Contardo et al., 2019). Among many problems in this research topic, two of the most known facility location problems are the *p-center* (pCP) and the *p-median* (pMP) problems, both introduced by Hakimi (1964, 1965). Given a graph, the objective in the pCP is to select p vertices, also known as *centers*, so the maximum distance between the graph's vertices and their respective closest center is minimized, i.e., a min–max problem. In the pMP case, one also selects p vertices, here known as *medians*, but the objective is to minimize the sum of distances of every vertex to its nearest median. These problems were proven to be NP-hard (Garey and Johnson, 1979; Kariv and Hakimi, 1979), so one often relies on heuristics to solve large instances.

In the pCP and pMP, vertices are assigned to a single facility. However, in some applications, facilities may be prone to failure and become unavailable due to unpredictable reasons such as weather and electricity problems (Panteli et al., 2021). In such cases, it is important to ensure the continuity of service to customers assigned to the failed facility. This is common in critical services, such as hospitals, fire stations, and computer networks, where backup coverage is needed (Wang et al., 2009; Araújo et al., 2020; Panteli et al., 2021). For instance, during the COVID-19 pandemic, hospitals in highly dense urban areas that could handle the demand of a regular day were

* Corresponding author.
E-mail addresses: guilherme.oliveira-chagas.1@ulaval.ca (G.O. Chagas), luizlorena54@gmail.com (L.A.N. Lorena), rafael.santos@inpe.br (R.D.C. dos Santos), jacques.renaud@fsa.ulaval.ca (J. Renaud), leandro.coelho@fsa.ulaval.ca (L.C. Coelho).

https://doi.org/10.1016/j.cor.2024.106589
Received 11 September 2023; Received in revised form 14 December 2023; Accepted 14 February 2024
Available online 23 February 2024

facing an overwhelming demand (Miller et al., 2020). An alternative is assigning excess demand to a temporary healthcare structure or even to a backup hospital (Araújo et al., 2020). On the other hand, a hospital located in a less populated region might not be dealing with a burden on its system. Therefore, the best solution in this case is to reallocate the population to the hospital in a less populated area instead of opening an additional temporary facility.

Other examples arise in computer networks, where some critical systems must have higher redundancy than others, or more generally in any context where some entities being served are more critical than others (Wang et al., 2009). From the provider's perspective, such as in the hospital example, facilities may be elected to require extra coverage for their users. Thus, assigning clients to multiple facilities at a time becomes useful.

To handle these types of problems, Krumke (1995) generalized the pCP and introduced the α-neighbor pCP (αNpCP), where vertices are assigned not just to the nearest center but to their α nearest ones. This problem aims to minimize the maximum distance between a non-facility vertex to its αth closest center. Note that when $\alpha = 1$, the pCP is defined. Krumke (1995) also proposed an approximation algorithm for the αNpCP since it is a NP-hard problem as it generalizes the pCP. Since then, solution methods have been proposed, especially approximation and exact algorithms, for either the continuous and discrete versions of the αNpCP, e.g., the works of Chaudhuri et al. (1998), Khuller et al. (2000), Chen and Chen (2013), and Callaghan et al. (2019). In the continuous version of the αNpCP, facilities can be placed anywhere in the defined space. On the other hand, in the discrete version, facilities must be vertices in the graph. The latter is the topic of interest in this work.

We are aware of only two heuristics for the αNpCP, the works of Sánchez-Oro et al. (2022) and Mousavi (2023). Sánchez-Oro et al. (2022) proposed a Greedy Randomized Adaptive Search Procedure with Tabu Search and Strategic Oscillation methodology (GRASP-SO). The authors tested their algorithm in 37 instances derived from the TSPLIB (Reinelt, 1991) and compared their results to the exact method of Chen and Chen (2013). The heuristic of Sánchez-Oro et al. (2022) obtained the best results in all tested instances. Mousavi (2023) developed efficient local search algorithms for the pCP, the αNpCP and the p-next center problem (pNCP). They tested their αNpCP heuristic, using $\alpha = 2$, on the 40 pMP instances from the OR-library (Beasley, 1990), but did not compare their algorithm with the results of GRASP-SO of Sánchez-Oro et al. (2022). The author then ran the algorithm for all 40 OR-library instances for 10,000 s, showing that the heuristic can consistently find the same solutions in a much shorter execution time.

Exploring the multiple assignment feature in the pMP context is also important. However, this pMP variation has not been explored as much as the pCP one. Even though the literature related to pMP is vast (Barbaros et al., 1983; Reese, 2006; Mladenović et al., 2007; Daskin and Maass, 2015; Marín and Pelegrín, 2019), to the best of our knowledge, there are few works concerning variations of these problems where vertices can be assigned to more than one median. One of these studies is the work of Wang et al. (2009), who introduced the backup 2-center problem and the backup 2-median problem. In these problems, every vertex is served by two medians. Another study is that of Karatas et al. (2016), where the authors introduced the requirement of each vertex to be assigned to more than one facility and compared it under five different criteria. Also, Brimberg et al. (2021) introduced the distributed pMP, where, given a distribution function over customers' demands, multiple medians are used to fulfill the customers' demands. However, none of these definitions impose multiple assignments precisely as in the αNpCP.

To the best of our knowledge, the only work that generalizes the single assignment requirement to allow multiple assignments, as in the αNpCP, is the work of Panteli et al. (2021). These authors relaxed the single vertex-median assignment constraint of the pMP and imposed that each vertex is allocated to their nearest α medians. The objective is to minimize the total sum of vertices distances to their α facilities. Again, when $\alpha = 1$, the pMP is defined, and this problem is NP-hard as it generalizes the pMP. Panteli et al. (2021) denominated this pMP variation as the multiple p-median problem. For uniformity, here we refer to this problem as the α-neighbor pMP (αNpMP). These authors also proposed the *Biclustering Multiple Median algorithm* (BIMM) to solve the αNpMP and compared it with a commercial solver.

Since both αNpCP and αNpMP are NP-hard, in this work we propose a simple but effective Basic Parallel Variable Neighborhood Search (BP-VNS) algorithm; "basic" defines the VNS version originally proposed by Mladenović and Hansen (1997). This algorithm is used to produce high-quality solutions for these problems. This heuristic has been successfully applied to many facility location problems, e.g., pMP (Hansen and Mladenović, 1997), pCP (Mladenović et al., 2003), capacitated pMP (Fleszar and Hindi, 2008), probabilistic pCP (Martínez-Merino et al., 2017), obnoxious pMP (Herré et al., 2020; Mladenović et al., 2020), and pNCP problem (López-Sánchez et al., 2019; Ristić et al., 2023).

We have developed our heuristic using the Less is More Approach (LIMA) (Mladenović et al., 2016; Brimberg et al., 2023). The LIMA is a heuristic design methodology focused on simplicity and user-friendliness rather than developing complex algorithms just for the sake of proposing a new method, with no solid performance improvement (Mladenović et al., 2016). The idea is to use the minimum number of algorithm components to develop a heuristic as simple as possible and still be able to find solutions at a state-of-the-art level (Mladenović et al., 2020). Besides the method's simplicity, another advantage of using this approach is that it is easier to identify how and why the algorithm performs the way it does (Mladenović et al., 2020). As we will demonstrate, our method can be easily adapted to several classes of problems and performs very well thanks to the important components described next.

In our BP-VNS we adapted optimized and well-known algorithms and data structures from the literature to the αNpCP and αNpMP context. Also, to take advantage of modern multi-core CPUs, we parallelized our BP-VNS due to its simplicity. In addition, we couple to our heuristic an updated αNpCP objective function based on the idea of Torres-Jimenez et al. (2015), which adds more information about the solution quality and helps guide the VNS to escape from local optima. Then, the main contributions of our work are:

- We present a simple and effective BP-VNS for the αNpCP and αNpMP. Using the LIMA methodology, we adapt well-known algorithms and data structures from the literature;
- We use a new objective function for the αNpCP, which allows the heuristic to differentiate solutions with the same cost, improving the heuristic's convergence;
- We show that our simple heuristic can find high-quality solutions and outperform state-of-the-art methods.

This paper is organized as follows. The mathematical formulations are presented in Section 2. Our BP-VNS is detailed in Section 3. Section 4 shows the test results. Our concluding remarks and discussion about future works are presented in Section 5.

2. Mathematical notation and problems definitions

Let $G = (V, E)$ be an *undirected*, *weighted*, and *connected* graph, where V is the set of vertices and E is the set of edges, where $|V| = n$, $|E| = m$ and to each edge $(i, j) \in E$ is associated a weight $d_{ij} \in \mathbb{R}^+$. In facility location problems, d_{ij} is often the Euclidean distance or the shortest path length between vertices i and j, but dissimilarity values are also common. In all these cases, the triangular inequality is not violated. Even if an edge joining vertices i and j may not exist in the original graph, (i, j) can be added to E with d_{ij} equal to the length of the shortest path between these vertices since G is connected and the triangular inequality holds. In this way, $D = (d_{ij})$ is an $n \times n$ *distance matrix* of non-negative real values. Let S be the set of the p

open medians, where $1 \leq p \leq n$. Since it is required that all vertices be assigned to α facilities in the αNpCP and the αNpMP, it is implicitly assumed that each vertex is always assigned to its α closest medians among the p open ones, where $\alpha \leq p$.

The remainder of this section is organized as follows. In Section 2.1, the αNpCP formulation is presented. The integer linear program of the αNpMP is described in Section 2.2.

2.1. αNpCP formulation

In the αNpCP, a subset $S \subset V$ of vertices are selected as facilities and each vertex $i \in V \setminus S$ is assigned to the nearest α of them. The distance between a vertex i and its αth nearest facility $j \in S$ is know as the *α-center-distance* and is defined by $d^c_\alpha(i,S) = \min_{S' \subset S, |S'|=\alpha}\{\max_{j\in S'} d_{ij}\}$. Thus, in this problem, the objective is to minimize the maximum α-center-distance of vertices that are *not* facilities, that is, to find a set $S \subset V$, where $|S| = p$, such that $\max_{i\in V\setminus S} d^c_\alpha(i,S)$ is minimum. Observe that when a vertex is selected as a facility, it is not assigned to other facilities.

The mathematical formulation of the pCP (Daskin, 1995) can be adapted to allow each vertex to have multiple assignments. In this formulation, decision variables x_{ij} control whether client i is allocated at facility j or not, i.e.,

$$x_{ij} = \begin{cases} 1, & \text{if vertex } i \in V \text{ is assigned to facility vertex } j \in V,\\ 0, & \text{otherwise.}\end{cases}$$

It is worth mentioning that when $x_{ij} = 1$ and $i = j$, then vertex i is selected as a facility. The αNpCP can be formulated as the following mixed-integer linear program:

$$\min\ z \tag{1a}$$

subject to

$$\sum_{j\in V, j\neq i} x_{ij} = \alpha(1 - x_{ii}), \qquad i \in V, \tag{1b}$$

$$\sum_{j\in V} x_{jj} = p, \tag{1c}$$

$$x_{ij} \leq x_{jj}, \qquad i \in V,\ j \in V,\ i \neq j, \tag{1d}$$

$$d_{ij} x_{ij} \leq z, \qquad i \in V,\ j \in V,\ i \neq j, \tag{1e}$$

$$z \in \mathbb{R}^+,\ x_{ij} \in \{0,1\}, \qquad i \in V,\ j \in V. \tag{1f}$$

The value of the continuous variable z is minimized by the objective function (1a), whose lower bound is given by constraint (1e). In other words, the objective function (1a) minimizes the maximum distance between a vertex and its furthest (αth nearest) facility. Constraints (1b) assure that each vertex $i \in V \setminus S$ is assigned to α facilities. Note that if $x_{ii} = 1$, i.e., vertex i is a facility, then i is not assigned to any other facility since the right-hand-side of constraints (1b) is zero. Exactly p facilities are opened, which is guaranteed by constraint (1c). A vertex i can only be assigned to a facility j if j is open. This is ensured by constraints (1d). Variables x_{ij} are binary and z is a nonnegative continuous variable as in constraints (1f).

2.2. αNpMP formulation

The αNpMP requires p medians to be selected from V and that all vertices $v \in V$ are assigned to their closest α facilities. Let $d^m_\alpha(i,S) = \min_{S'\subset S, |S'|=\alpha} \sum_{j\in S'} d_{ij}$ be the *α-median-distance* of vertex i given a set of facilities S. In the αNpMP, the objective is to minimize the sum of the α-median-distances of all vertices. In other words, the objective is to find a set $S \subseteq V$, where $|S| = p$, such that $\sum_{i\in V} d^m_\alpha(i,S)$ is minimum. Unlike the αNpCP, in the αNpMP facilities are also assigned to α facilities.

The αNpMP can be formulated as an integer linear program (2a)–(2e). In this model, decision variables x_{ij} are the same as the ones defined in Section 2.1 and control whether client i is assigned to facility j. Again, when $x_{ij} = 1$ and $i = j$ then i is selected as a facility.

$$\min \sum_{i\in V}\sum_{j\in V} d_{ij} x_{ij} \tag{2a}$$

subject to

$$\sum_{j\in V} x_{ij} = \alpha, \qquad i \in V, \tag{2b}$$

$$\sum_{j\in V} x_{jj} = p, \tag{2c}$$

$$x_{ij} \leq x_{jj}, \qquad i \in V,\ j \in V, \tag{2d}$$

$$x_{ij} \in \{0,1\}, \qquad i \in V,\ j \in V. \tag{2e}$$

In the model above, the objective function (2a) minimizes the sum of the distances between every vertex i assigned to each facility j. Constraints (2b) are the multiple assignment constraints and impose that every vertex must be assigned to α facilities. Constraint (2c) guarantees that p vertices are open. A vertex i can only be assigned to a vertex j if j is an open facility, i.e, only if $x_{jj} = 1$, and this is ensured by inequalities (2d). Constraints (2e) define variables x_{ij} as binary. Note that the difference between the αNpMP model and the PMP model (Revelle and Swain, 1970) is in constraints (2b), which, in the αNpMP case, allow multiple assignments.

3. Basic parallel variable neighborhood search

The VNS is a well-known metaheuristic (Hansen and Mladenović, 2018), which consistently explores increasing neighborhoods if no improvement is detected. Whenever a better solution is found, the neighborhood range is reset to the minimum size, and the exploring process starts over, using the neighboring of the new solution. This metaheuristic also uses a local search procedure to polish newfound solutions, combining exploring and exploiting.

Since we employed the LIMA methodology for developing heuristics for the αNpCP and the αNpMP, we decided to implement the BP-VNS. This VNS is a parallel version of the original metaheuristic proposed in the seminal work of Mladenović and Hansen (1997), which is composed of finding a new neighbor solution using one shaking procedure, followed by one local search, which improves the found solution, and then deciding whether or not we move to the new neighborhood (Mladenović et al., 2020). These steps are repeated until a stop criterion is met, e.g., maximum execution time.

Remember that in both the αNpCP and the αNpMP, each client is assigned to its α nearest facilities. So, all the information we need to represent a solution to these problems is the p facilities. Let $S = \{v_{i_1}, \ldots, v_{i_p}\}$ denote a solution, i.e., $S \subset V$ is a set of p vertices (facilities). A metric to differentiate two solutions S and S' is $\rho(S,S') = p - |S \cap S'|$, the number of facilities they do not share. So we say a solution S' is at a distance of k from S if $\rho(S,S') = k$. Then, all solutions lying at a distance of k, with $k = 1, \ldots, k_{\max}$ and $k_{\max} \leq p$, are contained in the neighborhood set $\mathcal{N}_k(S)$.

The BP-VNS used in this work is depicted by Algorithm 1. This same structure is used in problems αNpCP and αNpMP. First, an initial solution is generated and becomes the current solution S. Then, the *shaker* procedure is applied, and a new solution $S' \in \mathcal{N}_k(S)$ is found within the neighborhood of size k of the solution S. The local search is then used to polish solution S'. If the cost of solution S' is less than that of the current best-known solution S, then S' becomes S, the neighborhood range k is reset, and the search continues from the new solution S. Otherwise, the neighborhood size increases, allowing it to explore $\mathcal{N}_k(S)$ even further. This step is repeated while the execution time limit is not reached.

To calculate the cost of a given solution s, for the αNpCP, we customized the *move evaluation* and *update* procedures as well as the

Algorithm 1: Basic Parallel Variable Neighborhood Search.

$S \leftarrow initialSolution()$; // generates a solution by Algorithm 2
while *time limit is not reached* **do**
  $k \leftarrow 1$;
  while $k < k_{max}$ **do**
    $S' \leftarrow shaker(S, k)$; // finds solution S' by using Algorithm 4
    $S' \leftarrow localSearch(S')$; // improves S' by applying Algorithm 5
    if $cost(S') < cost(S)$ **then**
      $S \leftarrow S'$;
      $k \leftarrow 1$;
    else
      $k \leftarrow k + 1$;
    end
  end
end
return S;

corresponding data structures from Mladenović et al. (2003), and from the work of Hansen and Mladenović (1997), for the αNpMP. Because these algorithms and data structures are well-known and easy to implement, another advantage is that one can compute a solution's cost in $O(n \log n)$.

We opted to use parallelization to enhance the performance of each component of our BP-VNS: the initial solution algorithm, the shaker procedure, and the local search. This decision was driven by the straightforward parallelization possibility each one of these methods offers, and we focused on keeping them simple. These components are explained in the following sections. The algorithm to generate an initial solution is described in Section 3.1. The shaker procedure is detailed in Section 3.2. The local search method is shown in Section 3.3. Further implementation details are presented in Section 3.4.

3.1. Initial solution

Algorithm 2 shows the parallel procedure used in this work for generating initial solutions for both αNpCP and αNpMP. In this algorithm, the best of r solutions, where r is a parameter of the number of threads, is selected as the initial solution. Each thread i starts from a solution S_i generated by Algorithm 3. This procedure returns a random solution for half of the threads and a solution generated by a constructive greedy algorithm for the other half. After generating a starting solution, a local search procedure improves it. We use the same local search detailed in Section 3.3 to keep the algorithm simple. Note that each thread calls the local search to improve its solution. So unlike the local search step of Section 3.3, here, each local search procedure runs in serial, unique to its thread. After all threads finish generating their solution, the best solution S among all S_i solutions is returned as the initial one. The initial algorithm can be viewed as multiple parallel calls of the serial local search starting from different solutions.

3.2. Shaker

Hansen and Mladenović (1997) and Mladenović et al. (2003) use a shaker procedure where the facility to be opened is selected randomly, and then they select the best open facility to be closed regarding the one to be opened. To find the best facility deletion, they use the move evaluation algorithm to identify the facility to be closed and to compute the new objective function value in $O(n)$. For the pCP, Mladenović et al.

Algorithm 2: *initialSolution* procedure.

Output: Initial solution S.
$S \leftarrow \emptyset$;
for $i \leftarrow 1$ **to** r **do in parallel** // r is the number of threads
  /* Each thread starts with a different solution */
  $S_i \leftarrow buildStartingSolution(i)$; // calls Algorithm 3
  $S_i \leftarrow localSearch(S_i)$; // calls single threaded Algorithm 5
  if $cost(S_i) < cost(S)$ **then**
    $S \leftarrow S_i$;
  end
end
return S;

Algorithm 3: *buildStartingSolution* procedure.

Input: Thread index i.
Output: Solution S.
if *i is even* **then**
  return *random solution S*; // opens p medians at random
end
/* Otherwise, build a greedy solution as follows */
$S \leftarrow v$; // where $v \in V$ is a vertex selected at random
while $|S| < p$ **do**
  select *a vertex $u \in V \setminus S$ which minimizes the $cost(S)$* **and do**
    $S \leftarrow S \cup u$;
  end
end
return S;

(2003) only considers opening the random facility if it is closer to the critical vertex than the critical vertex's current facility.

Since we design a parallel shaker algorithm, the approach of Mladenović et al. (2003) may trap the BP-VNS in local optima, as we select the best out of a number of candidates, which, in turn, were selected greedily. Then, in our shaker, we first decided to remove the requirement of only opening a facility if it improves the critical vertex assignment, avoiding making such greedy decisions. To improve the solution space exploration even further and simplify the heuristic, we opted to make the shaker completely random, i.e., to open and close facilities randomly. This very same shaker algorithm was successfully used by Mladenović et al. (2020) in the obnoxious pMP.

Algorithm 4 depicts the shaker procedure, which, given a solution S and the neighborhood size k, is used to find a new solution $S' \in \mathcal{N}_k(S)$. The role of this method, as the name implies, is to disturb the current solution and, thus, avoid being trapped in local optimum solutions. Like Algorithm 2, we explore r solutions in parallel and keep the best one. So from the input solution S, each thread i finds a solution $S'_i \in \mathcal{N}_k(S)$ by randomly swapping k facility vertices with k client vertices. In other words, each thread randomly selects a set of facilities $J = \{j_1, \dots, j_k\}$, such that $J \subseteq S$, and a set of non-facility vertices $L = \{l_1, \dots, l_k\}$, such that $L \subseteq V \setminus S$, and swap them.

We decided to close and open k facilities at random in our shaker instead of, for example, opening a facility at random and closing the best one as in Hansen and Mladenović (1997) and Mladenović et al. (2003) for two reasons. First, using swaps is well-aligned with the LIMA aspect of our algorithm; we have not observed any significant gains from using a more expensive approach during our preliminary tests.

Algorithm 4: *shaker* procedure.

Input: Current solution S; neighborhood size k.
Output: New solution $S' \in \mathcal{N}_k(S)$.

$S' \leftarrow S$;
for $i \leftarrow 1$ **to** r **do in parallel** // r is the number of threads
  select $S'_i \in \mathcal{N}_k(S)$ **and do** // randomly opens and closes k facilities
    if $cost(S'_i) < cost(S')$ **then**
      $S' \leftarrow S'_i$;
    end
  end
end
return S';

Second, this random approach helps the BP-VNS to escape local optima more effectively than a greedy alternative.

3.3. Local search

We implemented a best improvement local search presented by Algorithm 5. Given an input solution S', this algorithm evaluates the swap of every non-facility vertex with the best facility deletion concerning the opened one. Then, the best swap is selected and performed. This process systematically explores all solutions in $\mathcal{N}_1(S')$ since it opens every client vertex as a facility, one by one. If the best swap improves S', then this procedure continues refining solution S' as long as an improvement is found. If no improvement is detected, it stops and returns S'. This procedure runs in parallel but, unlike Algorithms 2 and 4 where threads run independently, here, each thread handles a subset of non-facility vertices. In other words, each thread explores a subset of $\mathcal{N}_1(S')$. Then, the best swap is the one selected to be performed.

To evaluate the swap between a facility vertex $j \in S'$ and a client vertex $l \in V \setminus S'$, we adapted the move evaluation algorithm from Mladenović et al. (2003), for the αNpCP, and from Hansen and Mladenović (1997), for the αNpMP. With these algorithms, one can compute, in $O(n)$ time complexity, the new objective function value if the swap between j and l would occur.

Algorithm 5: *localSearch* procedure.

Input: Candidate solution S'.
Output: Improved solution S', if any.

do
  $improved \leftarrow$ false;
  foreach $S'' \in \mathcal{N}_1(S')$ **do in parallel** // opens and closes one facility
    if $cost(S'') < cost(S')$ **then**
      $S' \leftarrow S''$;
      $improved \leftarrow$ true;
    end
  end
while $improved$;
return S';

3.4. Implementation details

We now present additional implementation details. First, we explain, in Section 3.4.1, how we adapted some data structures from the works of Hansen and Mladenović (1997) and Mladenović et al. (2003). Secondly, in Section 3.4.2, we show details regarding the parallelization of the initial solution, shaker, and local search procedures. Lastly, in Section 3.4.3, we present the αNpCP updated objective function used in this work to improve the BP-VNS convergence further.

3.4.1. Data structures

As mentioned earlier, we have adapted the *move evaluation* and the *update* algorithms of Mladenović et al. (2003) for the αNpCP, and of Hansen and Mladenović (1997) for the αNpMP, to evaluate facilities candidates efficiently and to compute the solution cost quickly. Besides other minor algorithmic details, the main difference between the original versions of these two algorithms and our adaptations lies in an auxiliary data structure denoted as $c1$ array. In the original move evaluation and update algorithms, each position i of array $c1$ holds the index of the nearest facility to each vertex i. However, in our case, where vertices are assigned to α facilities, the $c1$ array transforms into a $n \times \alpha$ matrix. Each row i of this matrix corresponds to the α facilities indices to which vertex i is assigned.

To efficiently update a facility in a row i of matrix $c1$, we keep each row's α facilities sorted into increasing order of distance from vertex i. This way, we can use binary search to remove or insert a facility. However, since α is a parameter and its values used in this work are small ($\alpha \leq 3$), as large values are not common in practice (Sánchez-Oro et al., 2022), we can consider it a constant. Then, there is no difference in the asymptotical time complexity between the original algorithms and our customizations. Thus, the time complexity of the move evaluation remains at $O(n)$, and the time complexity of the update algorithm of $O(n \log n)$ remains the same.

3.4.2. Parallelization

Three procedures of our heuristic are parallelized: the initial solution algorithm, the shaker, and the local search. The parallelization of each one was done independently. Fig. 1 depicts how multithreading is implemented in each of these algorithms.

In the initial solution algorithm, r threads are spawned and assigned a starting solution generated by Algorithm 3. Then, each thread tries to improve its solution by applying a single-threaded version of the local search, described in Algorithm 5. Finally, the best solution among the r ones is selected as the initial solution. A mutex controls the read and write operations to avoid data racing when selecting the best solution, a shared resource between the threads.

The parallelization of the shaker is similar to that of the initial solution algorithm. Here, r copies of the current solution are created and assigned to r threads. Then, each thread performs k random swap operations on its copy; that is, it selects k random vertices to become medians and k random medians to be closed. The best among the r solutions found by this method is returned. As in the initial solution algorithm, we use a mutex to avoid data racing on the best solution.

In the local search, the parallelization was done straightforwardly. Since we swap every non-median vertex with its respective best median to be closed, we can give each thread a subset of non-median vertices so they can compute the swaps. Then, the workload of calling the move evaluation method $|V \setminus S|$ times is evenly divided between the r threads. Again, we use a mutex to control read and write operations for finding the best swap.

3.4.3. αNpCP evaluation function

The αNpCP inherited an issue from the pCP: several solutions have the same cost. This problem is even worse in the αNpCP since we minimize the maximum distance between a vertex and its αth facility. However, this does not mean that solutions of the same cost are equal. Between two solutions of the same cost, we can consider one of them to be better than the other. For example, let S and S' be two solutions where $\rho(S, S') \geq 1$ and $cost(S) = cost(S') = 42$. Also, consider that S' has only one *critical vertex*, a vertex i in which the distance to its αth facility equals 42, whereas S has several critical vertices. It may be easier to reduce the cost of S' than to open new facilities and try to reduce the cost of S. So, adding more information to the cost of an αNpCP solution regarding the overall assignments is necessary instead of just considering the critical element.

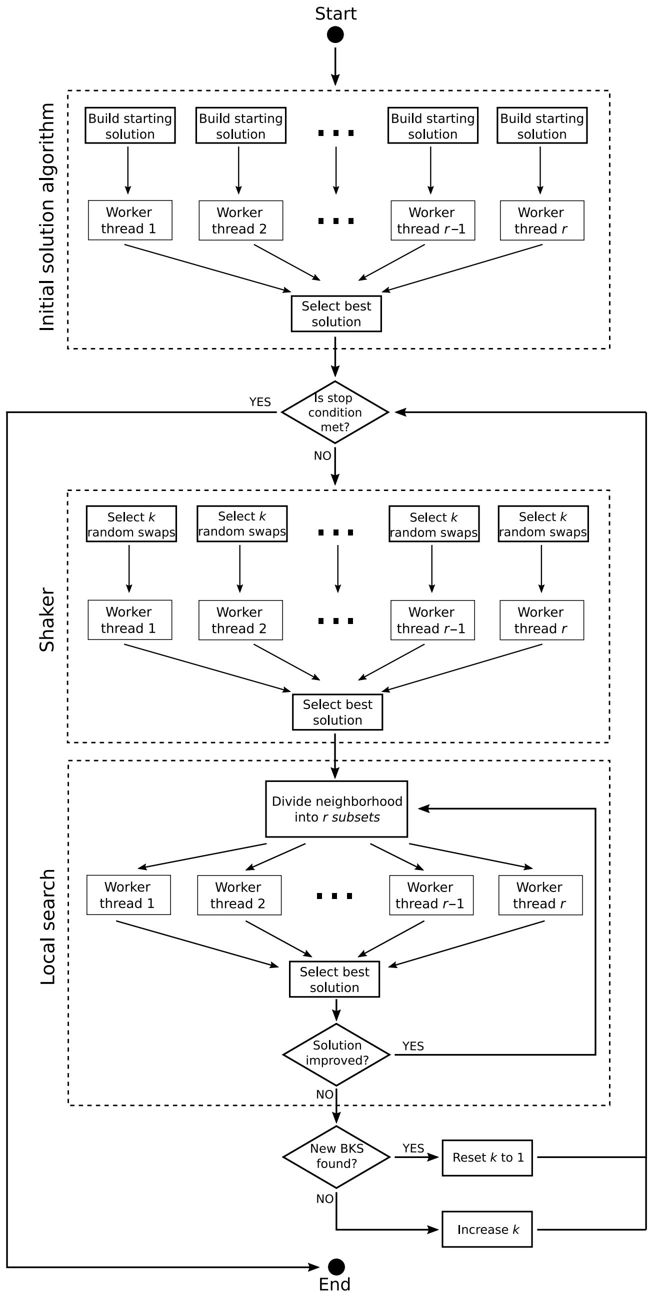


Fig. 1. BP-VNS flowchart.

Based on this observation, we adapted the method of Torres-Jimenez et al. (2015). These authors proposed a heuristic for the matrix bandwidth minimization problem (MBMP). The MBMP is also a min–max problem, where the objective is to minimize the maximum distance between a nonzero coefficient and the main diagonal of a square sparse symmetric matrix. There are also several solutions to this problem that cost the same. Then, the authors adapted the idea proposed by Rodriguez-Tello et al. (2008), which consists of counting the number of occurrences of distances between all the other nonzero coefficients and the main diagonal and then translating it to a value $\delta \in [0, 1)$. To compute the value of δ, they counted the number of distances of each value and the maximum number of each distance. They used these values to represent a number in a positional numbering system of variable base, also known as mixed radix. The value represented by this numerical system is then normalized to a value in the range $[0, 1)$ and added to the objective function value. This adds more meaning to the objective function value and helps to differentiate solutions with the same bandwidth. Please refer to Rodriguez-Tello et al. (2008) and Torres-Jimenez et al. (2015) for further details.

We can also use this method in the αNpCP since they are both problems where the objective is to minimize the maximum distance. To compute δ, the value to be added to the αNpCP objective function value, we use Algorithm 6 (Torres-Jimenez et al., 2015). In this algorithm, element $\mathtt{d}_i$ of array $\mathtt{d}$ represents the number of edges of length i used in solution S, i.e., the number of edges of length i connecting a client vertex to its αth facility. The ith value of array $\mathtt{v}$ is the maximum number of edges of length i that can be used in a solution plus one because no edge is a possible value. Then, with both arrays $\mathtt{d}$ and $\mathtt{v}$ one can represent the αNpCP solution as a number in a positional numbering system of variable base, where $\mathtt{d}$ values can be interpreted as digits and $\mathtt{v}$ as the base in this numerical system. To compute this value and then normalize it in the $[0, 1)$ range, Torres-Jimenez et al. (2015) proposed the Algorithm 6, where the normalized value is represented by δ.

Algorithm 6: αNpCP alternative objective function.

Input: Solution S; arrays $\mathtt{d}$ and $\mathtt{v}$.
Output: αNpCP cost of solution S.

```
δ ← 0;
for i ← 0 to cost(S) do
    if d_i > 0 then
        δ ← (δ + d_i) / v_i;
    end
end
return cost(S) + δ;
```

For example, consider two solutions S and S' depicted in Fig. 2. In this example, $n = 7$, $p = 2$, $\alpha = 2$, $cost(S) = cost(S') = 42$, and $\rho(S, S') = 1$. Note that solution S has two critical vertices (two vertices with a distance of 42 to their facilities), whereas solution S' has only one. So one could use Algorithm 6 to compute the value of δ of both solutions and compare them.

To compute the δ values for solutions S and S', every edge of the graph of the αNpCP is counted to define the array $\mathtt{v}$. Also, recall that the absence of the edge in the solution is counted, too, so we add one to every $\mathtt{v}_i$ value. As the array $\mathtt{v}$ is related to the graph and not to a particular solution, then, for solutions S and S', we have the same following $\mathtt{v}$ values: there is one edge of distance 21 in the graph, i.e., $\mathtt{v}_{21} = 2$ (the edge plus one to represent the absence of such edge in a solution); there is one edge of distance 22 in the graph, i.e., $\mathtt{v}_{22} = 2$; $\mathtt{v}_{24} = 2$, $\mathtt{v}_{30} = 2$, $\mathtt{v}_{35} = 2$, and $\mathtt{v}_{42} = 3$ (since there are two edges of length 42 in solution S, plus one to represent the absence of such edge in a solution). On the other hand, array $\mathtt{d}$ is specific to each solution, and for solution S we have: one edge of distance 21 used in the solution, i.e., $\mathtt{d}_{21} = 1$; one edge of distance 22 used in the solution, i.e., $\mathtt{d}_{22} = 1$; $\mathtt{d}_{30} = 1$, $\mathtt{d}_{42} = 2$. For solution S' we have the following: $\mathtt{d}_{21} = 1$, $\mathtt{d}_{22} = 1$, $\mathtt{d}_{24} = 1$, $\mathtt{d}_{35} = 1$, $\mathtt{d}_{42} = 1$. Then, using Algorithm 6, we get $\delta = 0.958$ for solution S and $\delta = 0.646$ for solution S'. Therefore, solution S' is better than S as $42.646 < 42.958$. Indeed, it is easier to improve solution S' cost because it has only one critical vertex.

Computing the value of δ for each solution can be done efficiently in $O(D_{\max})$, where $D_{\max} = \max_{i,j \in V} d_{ij}$, within the move evaluation and update algorithms, significantly improving the amount of information the algorithm considers. As demonstrated in the next section, this new updated objective function improves convergence and helps the algorithm achieve better solutions.

4. Computational experiments and analysis

We now describe the computational experiments performed to assess the performance of our methods. All algorithms described were

(a) Solution S. (b) Solution S'.

Fig. 2. Example of two αNpCP solutions of same cost (42).

implemented in C^{++} language and compiled with g^{++} compiler, version 11.3.0. We used Gurobi's C^{++} API, version 10.0.3 for solving the integer programs. The preliminary tests of Section 4.1 were executed on a computer equipped with an Intel® Core™ i9-13900K processor with 32 threads at 3.0 GHz and 128 GB of RAM. All the remaining experiments were conducted on a computing cluster based on AMD EPYC™ Rome 7532 processors running at 2.4 GHz using 24 threads and up to 96 GB of RAM. All instances described below and their detailed solutions are available at www.leandro-coelho.com/VNS-location-problems and in the appendices.

For the tests with both αNpCP and αNpMP, we used the well-known OR-library instances (Beasley, 1985, 1990). This set contains 40 instances with sizes ranging between 100 and 900 vertices. These instances are composed of connected weighted non-complete graphs. To transform these graphs into complete ones, we used the Floyd–Warshall algorithm to compute the shortest paths between every pair of vertices, as is done in the literature. Following Sánchez-Oro et al. (2022), we also used 77 instances derived from the TSPLIB (Reinelt, 1991) for the tests with the αNpCP and, for each of them, we tested using $\alpha \in \{1,2,3\}$. For the αNpMP tests, we used $\alpha \in \{10,20\}$ and compared our VNS with the BIMM heuristic (Panteli et al., 2021).

Since we use the LIMA methodology, the only parameters our heuristic uses are the $k_{\max}$, for which we used $k_{\max} = p$ (Mladenović et al., 2020), the number of threads r, which we set to $r = 24$ for the experimental tests, and the execution time limit, which we used 30 min for the experimental tests. Also, we set an additional stopping criterion for when our heuristic finds the best-known solution or improves it. In addition, we set 2 h as the execution time limit for the commercial solver to solve the models.

The remaining of the section is organized as follows. In Section 4.1 we show the preliminary tests carried out to evaluate the components of our BP-VNS. The αNpCP experimental results are presented in Section 4.2, and the αNpMP ones are presented in Section 4.3.

4.1. Preliminary tests

Preliminary tests to evaluate some key features of our BP-VNS are presented in this section. We tested our heuristic on the first 10 instances of the OR-library, which have $n \in \{100, 200\}$ and $p \in \{5,10,20,33,40,67\}$. In all tests of this section, we used $\alpha = 2$ and 60 s as the time limit of the BP-VNS. In addition, as the αNpCP and αNpMP versions of our heuristic share the same main structure, we performed the preliminary tests only for the αNpCP. This section is structured as follows. The impact of parallelism is analyzed in Section 4.1.1, tests to evaluate the *shaker* functions are shown in Section 4.1.2, and the ones to evaluate the usage of the updated αNpCP objective function are presented in Section 4.1.3.

4.1.1. Evaluation of parallelism

In this section, we assess the parallelism in our BP-VNS, described in Algorithm 1. First, we tested how the number of threads impacts the execution times of our heuristic by computing the speedup of the multithreading version against the serial one. To compute the speedups properly, we opted to only measure the execution time of the local search, as a single iteration of this algorithm always evaluates $|V| - p$ customer-median swaps. So, the workload of these swaps is evenly divided between the threads. On the other hand, this does not happen with the initial solution and shaker algorithms, as in these parallel procedures, the threads run independently and do not divide the workload but explore different regions of the solution space. Then, we could not directly compute a speedup.

Fig. 3 shows the execution times (on the left graph) and the speedups (on the right graph) of the local search algorithm with 1, 2, 4, 8, 12, 16, 20, 24, and 32 threads. To obtain these results, for each number of threads, we got the total execution time of 1000 runs of the local search algorithm on the pmed40 instance from the OR-library.

Parallelizing the local search significantly improves its execution time, as it is reduced from 13.75 s (single-threaded) to 1.02 s (32 threads). In terms of speedup, this indicates that our parallel algorithm can run up to 13.48 times faster than its serial version. As detailed at the beginning of Section 4, we have used machines from a computing cluster to run the tests with the α−neighbor problems. Since most machines of this cluster have 48 threads, we decided to use 24 threads to run two tests per machine, speeding up the time required to finish our tests and making a sensible use of variable resources. Also, as one can note in Fig. 3, the gain of using 32 threads over 24 threads is negligible (the execution time is reduced from 1.03 s to 1.02 s, and the speedup is improved from 13.40 to 13.48). Hence, 24 threads is a reasonable choice.

To evaluate how parallelism helps our heuristic's convergence, we tested two versions: the BP-VNS running on a single thread and the parallel BP-VNS where 24 threads are used. Table 1 shows these tests' results. In this table, the first three columns show the instances' names, the number of vertices, and the number of medians. The *optimum* (opt) of each instance is shown in the fourth column. Then, we present for both BP-VNS versions the best solution found, the iteration in which the best solution was found (iter$_{best}$), the total number of iterations (#iter), and the time, in seconds, in which the best solution was found (t$_{best}$ (s)).

As one can note, the parallel version obtained the optimum values in all instances. On the other hand, the single-thread BP-VNS did not find the optimum of instances *pmed3*, *pmed4*, *pmed9*, and *pmed10*. The parallel initial solution algorithm helps the heuristic convergence as it is a multistart procedure, so the BP-VNS starts from the best solution out of r candidates. The (parallel) BP-VNS found the best solution much earlier than the single-thread version, as one can see in columns iter$_{best}$ and t$_{best}$ (s). Indeed, in some instances, as for *pmed1* and *pmed6*, BP-VNS found the optimum at iteration 0, that is, in the initial algorithm step. Also, parallelism helps the heuristic explore the solution space faster as multiple solutions are visited in each call of the *shaker* procedure. Exploring different neighborhoods more efficiently helps BP-VNS escape from local optima, which can explain why the parallel version of BP-VNS found the optimum for all instances. In addition, the parallel local search algorithm is much faster than the single thread version, as each thread explores a subspace of $\mathcal{N}_1(S)$. Since the local search procedure is the most expensive step in our BP-VNS, parallelizing it helps decrease the computational burden. This can be noticed in the total number of iterations, where BP-VNS ran approximately eight to ten times more iterations than the single-thread version. Then, for these reasons, we decided to use the BP-VNS version for the remainder of the paper.

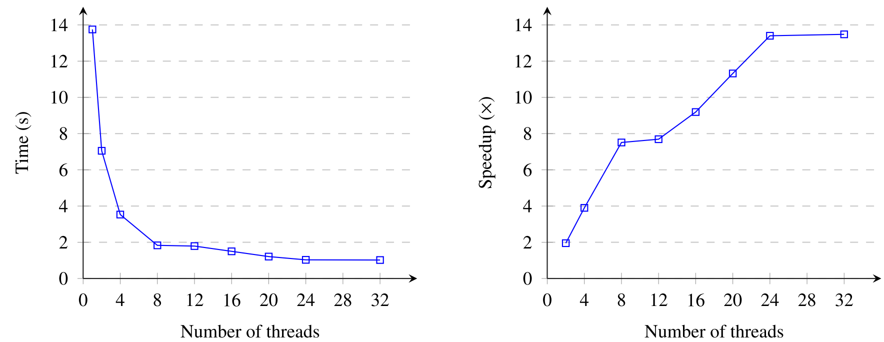


Fig. 3. Execution times and speedups of the parallel local search algorithm running with different numbers of threads.

Table 1
Comparison between the single-thread and the parallel versions of BP-VNS (αNpCP, with $\alpha = 2$).

Instance	n	p	opt	Single-thread BP-VNS				BP-VNS			
				Best	iter$_{best}$	#iter	t$_{best}$ (s)	Best	iter$_{best}$	#iter	t$_{best}$ (s)
pmed1	100	5	150	150	72	49 539	0.09	150	0	475 687	0.01
pmed2	100	10	121	121	2 965	56 666	3.24	121	674	469 980	0.11
pmed3	100	10	121	123	814	56 075	0.90	121	4210	447 565	0.65
pmed4	100	20	97	98	1 181	69 843	1.04	97	1067	483 936	0.18
pmed5	100	33	63	63	17 705	96 636	11.14	63	160	427 687	0.04
pmed6	200	5	99	99	6 954	12 429	33.57	99	0	159 187	0.04
pmed7	200	10	80	80	8 672	13 262	39.23	80	7	162 246	0.07
pmed8	200	20	70	70	839	14 520	3.60	70	79	157 030	0.12
pmed9	200	40	49	50	1 108	18 074	3.88	49	751	182 220	0.44
pmed10	200	67	28	29	11 503	25 421	27.29	28	2025	199 115	0.68

Table 2
Comparison between BP-VNS with the greedy and the random shakers (αNpCP, with $\alpha = 2$).

Instance	n	p	opt	Greedy shaker				Random shaker			
				Best	iter$_{best}$	#iter	t$_{best}$ (s)	Best	iter$_{best}$	#iter	t$_{best}$ (s)
pmed1	100	5	150	150	0	457 386	0.01	150	0	475 687	0.01
pmed2	100	10	121	121	810	430 671	0.17	121	674	469 980	0.11
pmed3	100	10	121	121	19 581	415 200	2.85	121	4210	447 565	0.65
pmed4	100	20	97	98	885	478 483	0.16	97	1067	483 936	0.18
pmed5	100	33	63	63	100	426 113	0.03	63	160	427 687	0.04
pmed6	200	5	99	99	0	150 220	0.05	99	0	159 187	0.04
pmed7	200	10	80	80	19	161 231	0.09	80	7	162 246	0.07
pmed8	200	20	70	70	161	142 285	0.21	70	79	157 030	0.12
pmed9	200	40	49	49	1 879	169 106	0.72	49	751	182 220	0.44
pmed10	200	67	28	28	5 757	176 213	1.86	28	2025	199 115	0.68

4.1.2. Evaluation of shaker functions

Recall that we designed a *random shaker* unlike the *greedy shaker* function initially proposed by Hansen and Mladenović (1997) and Mladenović et al. (2003). Table 2 shows the tests performed to compare BP-VNS using the greedy approach and with the random shaker. The table follows the same structure of Table 1. Note that the results of the random version are exactly the same as presented in Table 1, as the results presented earlier are related to our complete BP-VNS, which uses the random shaker.

Since our shaker procedure is parallelized, the BP-VNS with the greedy one might be trapped in local optima more often than the random shaker. Once again, our proposed random shaker performs best as the greedy shaker procedure cannot find all optimum. Moreover, the version with the greedy shaker took longer to find the optimum values in almost all instances, sometimes significantly (2.85 s vs. 0.65 s), as one can see from the iteration and time in which the best solutions were found in Table 2. Also, since in the random shaker there is no extra $O(n)$ computation of the move evaluation algorithm, BP-VNS could run faster and, therefore, the total number of iterations of the heuristic with random shaker is slightly larger than ones achieved by the BP-VNS with greedy shaker. We decided to employ the random shaker based on these results and its simplicity.

4.1.3. Evaluation of the updated αNpCP objective function

In this section, we evaluate our BP-VNS (Algorithm 1) with and without the updated αNpCP objective function, described in Section 3.4.3. Note that this new objective function is used only in the αNpCP, whereas the features tested in Sections 4.1.1 and 4.1.2 are used in the BP-VNS applied to both αNpCP and αNpMP.

In Table 3, the results from columns *updated OF* are the same as the ones from Tables 1 and 2, as our BP-VNS uses the updated αNpCP objective function. However, the results from columns *regular OF* refer to the BP-VNS version with the regular αNpCP objective function.

As one can note from the results of Table 3, the new objective function significantly helps BP-VNS achieve better results as the version with the regular objective function could not obtain optimum values in instances *pmed4*, *pmed9*, and *pmed10*, and some of these by a large gap. Also, note that the BP-VNS with the regular objective function took much longer to find the best solutions since the update objective function helps BP-VNS move to more promising neighborhoods as it adds more information to solutions costs. Even if the calculation of the new objective function adds a step of time complexity O($D_{\max}$), this time is clearly offset by the gains in terms of information embedded in the solution algorithm, allowing it to explore more promising neighborhoods and ultimately find better solutions faster. This development shows a huge potential for this problem and can help improve convergence and solution quality in other types of problems as well.

Table 3
Comparison between BP-VNS with the regular αNpCP objective function and the updated one (αNpCP, with $\alpha = 2$).

Instance	n	p	opt	Regular OF				Updated OF			
				Best	iter$_{best}$	#iter	t$_{best}$ (s)	Best	iter$_{best}$	#iter	t$_{best}$ (s)
pmed1	100	5	150	150	0	372 730	0.01	150	0	475 687	0.01
pmed2	100	10	121	121	524	354 236	0.11	121	674	469 980	0.11
pmed3	100	10	121	121	5 148	419 403	0.84	121	4210	447 565	0.65
pmed4	100	20	97	98	2 787	370 127	0.50	97	1067	483 936	0.18
pmed5	100	33	63	63	966	411 485	0.15	63	160	427 687	0.04
pmed6	200	5	99	99	0	123 946	0.03	99	0	159 187	0.04
pmed7	200	10	80	80	183	139 211	0.12	80	7	162 246	0.07
pmed8	200	20	70	70	44 606	131 411	23.36	70	79	157 030	0.12
pmed9	200	40	49	51	31 908	134 805	14.12	49	751	182 220	0.44
pmed10	200	67	28	34	4 561	191 058	1.44	28	2025	199 115	0.68

Table 4
αNpCP summary results on OR-library and TSPLIB instances.

Instance set	α	MIP solver			GRASP-SO (Sánchez-Oro et al., 2022)				Mousavi (2023)				BP-VNS			
		Best	#bks	t (s)	Best	#bks	Gap (%)	t$_{best}$ (s)	Best	#bks	Gap (%)	t$_{best}$ (s)	Best	#bks	Gap (%)	t$_{best}$ (s)
OR-lib	1	37.33	40	366.47	–	–	–	–	37.33	40	0.00	0.09	37.33	40	0.00	34.54
	2	54.95	22	4309.75	–	–	–	–	45.55	38	0.24	3.39	45.55	38	0.30	4.19
	3	60.98	18	4977.89	–	–	–	–	–	–	–	–	51.10	40	0.00	7.55
TSPLIB	1	2153.49	55	2920.51	505.43	11	5.63	653.16	–	–	–	–	481.70	60	0.29	397.43
	2	4515.61	33	4486.92	773.35	5	8.36	990.00	–	–	–	–	732.60	76	0.00	404.96
	3	4881.12	27	4687.35	997.18	0	7.46	1147.78	–	–	–	–	945.06	74	0.00	445.66

4.2. Performance evaluation on the αNpCP

In this section, we compare our method with the best-known αNpCP solution values from the literature. Specifically, for the 40 OR-library instances with $\alpha = \{1, 2\}$, we compare the results of our BP-VNS against the ones from the work of Mousavi (2023). For these instances, we also tested with $\alpha = 3$, and compared our results with the ones obtained by the commercial solver solving model (1a)–(1f). For the 77 TSPLIB instances, we used $\alpha = \{1, 2, 3\}$ and compared our results against the ones of the GRASP-SO heuristic (Sánchez-Oro et al., 2022). Since we extracted the results of the heuristic of Mousavi (2023) and of the GRASP-SO from their papers, and to provide a fair computational comparison, we have approximated their running times by dividing the reported values by 1.5 and 0.85 (PassMark Software Pty Ltd, 1998), respectively.

Table 4 summarizes the results of the tests on the OR-library and TSPLIB instances. This table shows the instance set names and the α values in the first two columns. Each row of the OR-library instances set corresponds to the average results of 40 instances, and each row of the TSPLIB to the average of 77 instances. We present for the MIP solver and the heuristics the average of the best solutions values (best), the number of best-known solutions (#bks) found, and the average of the running times (t (s)). Note that since the stopping criterion of our BP-VNS is the execution time limit of 30 min, we show for this heuristic the time when the best solution was found (t$_{best}$ (s)). In addition, we present the average percentage gap (gap (%)) related to the best-known solutions. The detailed results of these tests are presented in Appendix A.

Regarding the OR-library instances, MIP solver found optimal solutions in all instances with $\alpha = 1$, and so did our method and the heuristic of Mousavi (2023). Considering $\alpha = 2$, the commercial solver found 22 best solutions, all proven optimal. From these 22 solutions, our BP-VNS and the heuristic of Mousavi (2023) obtained 20 optimal ones. From the two solutions where both heuristics did not achieve the optima, one is the same instance (pmed24), and the other one is different for each heuristic (pmed25 for BP-VNS and pmed19 for the heuristic of Mousavi, 2023). In these cases, the difference to the optimal solutions was just one unit in all cases, but since the optimal solution value of pmed25 (15) is less than the pmed19 one (24), the relative gap of this one unit is greater in the pmed25 case. This is why the average gap of BP-VNS was slightly larger than the one of the algorithm of Mousavi (2023). Regarding $\alpha = 3$, BP-VNS found the best solutions in all 40 instances, and the commercial solver obtained 18 optimal solutions. Note that Mousavi (2023) did not test their algorithm with this configuration.

Considering all results in the OR-library instances with $\alpha = 1, 2, 3$, the BP-VNS presented an average improvement gap of 6.10% compared with results from the literature and the commercial solver. Also, our heuristic could find 22 new best-known solutions, where the average improvement in these cases was 34.26%.

Table 4 shows that our BP-VNS outperformed the MIP solver and the GRASP-SO on the TSPLIB instances with all α values, dominating that algorithm. With $\alpha = 1$, the commercial solver obtained 55 best solutions, of which 52 are optimal, the GRASP-SO obtained 11 best ones, and our heuristic found 60 best solutions out of the 77 instances. With $\alpha = 2$ and $\alpha = 3$, the difference in terms of solution quality between the proposed BP-VNS and the other two solution methods was even more pronounced. For $\alpha = 2$, BP-VNS achieved 76 best solutions out of the 77 instances, whereas the commercial solver and GRASP-SO found 33 and 11 best solutions, respectively. Similarly, for $\alpha = 3$, BP-VNS excelled by obtaining 74 best solutions, while the commercial solver achieved 27 best solutions, and the GRASP-SO found none. On the 231 instances of the TSPLIB set, our heuristic obtained an overall improvement gap of 2.47% compared with the results of the literature and the commercial solver. Moreover, the BP-VNS found 115 new best-known solutions with an average improvement of 5.30%.

Considering computational times, even though our heuristic ran for up to 30 min in all instances, finding the best solutions in both instances sets required much less time. In the OR-library, although the heuristic of Mousavi (2023) is fast, BP-VNS could find the same solutions found by this heuristic, and our method obtained all optimal solutions with $\alpha = 1$ in less runtime when compared to the MIP solver. Moreover, our method outperformed it with $\alpha = 2$ and $\alpha = 3$, finding the best solutions with significantly less execution time. Regarding the tests in the TSPLIB instances, BP-VNS could find better solutions in, on average, less than half of the GRASP-SO runtime and in much less execution time than the MIP solver.

To visualize these performances graphically, we show in Figs. 4 and 5 the box plot containing the distribution of the solution values for all instances and algorithms described. Fig. 4 shows the results for the OR-library instances where our algorithm is equivalent to that of Mousavi (2023) for $\alpha = 1$ and $\alpha = 2$, and both are significantly better than the

Table 5
αNpCP Pairwise Wilcoxon Test for the OR-library instances (p-values shown; significant difference between the performance of the algorithms if less than 0.05).

α	Method	Best solution values		Gap (%)		t (s)	
		Mousavi (2023)	BP-VNS	Mousavi (2023)	BP-VNS	Mousavi (2023)	BP-VNS
1	MIP solver	1.00	1.00	1.00	1.00	$1.82 \cdot 10^{-12}$	$7.14 \cdot 10^{-7}$
	Mousavi (2023)	–	1.00	–	1.00	–	$5.83 \cdot 10^{-8}$
2	MIP solver	$1.50 \cdot 10^{-4}$	$1.50 \cdot 10^{-4}$	$9.54 \cdot 10^{-6}$	$9.54 \cdot 10^{-6}$	$1.82 \cdot 10^{-12}$	$1.82 \cdot 10^{-12}$
	Mousavi (2023)	–	1.00	–	1.00	–	0.05
3	MIP solver	–	$4.30 \cdot 10^{-5}$	–	$4.77 \cdot 10^{-7}$	–	$1.82 \cdot 10^{-12}$

Table 6
αNpCP Pairwise Wilcoxon Test for the TSPLIB instances (p-values shown; significant difference between the performance of the algorithms if less than 0.05).

α	Method	Best solution values		Gap (%)		t (s)	
		GRASP-SO	BP-VNS	GRASP-SO	BP-VNS	GRASP-SO	BP-VNS
1	MIP solver	0.12	$3.90 \cdot 10^{-3}$	0.15	$1.71 \cdot 10^{-3}$	$4.87 \cdot 10^{-5}$	$1.43 \cdot 10^{-10}$
	GRASP-SO	–	$1.68 \cdot 10^{-12}$	–	$1.68 \cdot 10^{-12}$	–	$5.05 \cdot 10^{-3}$
2	MIP solver	$6.86 \cdot 10^{-5}$	$1.84 \cdot 10^{-8}$	$1.94 \cdot 10^{-5}$	$1.34 \cdot 10^{-8}$	$4.70 \cdot 10^{-14}$	$4.17 \cdot 10^{-14}$
	GRASP-SO	–	$1.69 \cdot 10^{-13}$	–	$1.69 \cdot 10^{-13}$	–	$2.43 \cdot 10^{-9}$
3	MIP solver	$1.01 \cdot 10^{-7}$	$5.12 \cdot 10^{-10}$	$6.21 \cdot 10^{-8}$	$7.36 \cdot 10^{-10}$	$7.18 \cdot 10^{-14}$	$4.88 \cdot 10^{-14}$
	GRASP-SO	–	$5.38 \cdot 10^{-14}$	–	$2.51 \cdot 10^{-14}$	–	$6.44 \cdot 10^{-10}$

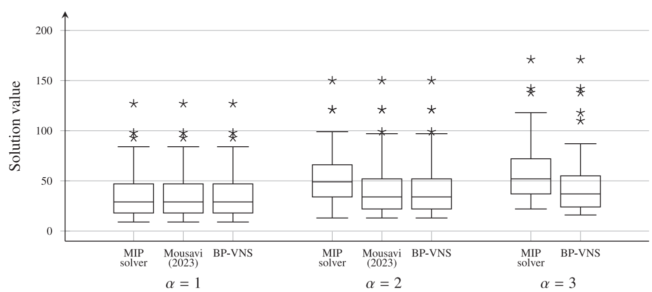


Fig. 4. Box plot of the αNpCP solution values of all methods on the OR-library instances.

MIP solver for $\alpha = 2$. When $\alpha = 3$, no results are reported by Mousavi (2023), and we can see a very large difference between the results of our method and the ones from the MIP solver. In Fig. 5, we show that for the TSPLIB instances, the differences are even larger and point to a better performance of our BP-VNS.

To verify if there is a significant difference between the average performance of these methods, we computed the Pairwise Wilcoxon Test, shown in Tables 5 and 6. These tables show the p-values related to the best solution costs, the gap (%), and the runtimes. A p-value less than 0.05 allows us to refute the null hypothesis, stating that there is a significant difference between the mean of the paired observations.

In Table 5, all p-values for the best solution and the gap for $\alpha = 1$ are greater than 0.05, implying that there is no significant difference between these results, as the MIP solver, the heuristic of Mousavi (2023), and our method obtained optimal solutions for all instances. The same can be concluded for $\alpha = 2$ between the results of the Mousavi (2023)'s heuristic and the BP-VNS. On the other hand, when these results are compared with those of the MIP solver, we confirm a significant difference as both methods outperformed the MIP solver. A similar conclusion can be drawn from the p-values for $\alpha = 3$.

Table 6 shows the results of the Pairwise Wilcoxon Test for the TSPLIB instances. As one can note, there is a statistically significant difference between the results of our heuristic and the results of the other methods in all scenarios, as the p-values are all less than 0.05.

4.3. Performance evaluation on the αNpMP

The results of the tests on the αNpMP are presented in this section. Here, we compare the results of our BP-VNS against the MIP solver and those of the BIMM heuristic (Panteli et al., 2021). Following Panteli et al. (2021), instead of using the original number of medians p from each instance, we used two values of p for every OR-library instance: $p = 10$ and $p = 20$. Moreover, to properly compare the results, we only used one value of α for each value of p. More specifically, when $p = 10$ we use $\alpha = 5$, and when $p = 20$ we set $\alpha = 10$. Then, we solved model (2a)–(2e) with a commercial solver and ran our BP-VNS on all 80 OR-library instances derived by using the values of p and α as just described.

Table 7 has a structure similar to the one presented in Section 4.2 and summarizes the results on the OR-library instances. In Table 7, each row corresponds to an average of 40 instances, where the first two columns show the p and α values used. Then, for each pair of p and α, we present the results for the commercial solver, the BIMM heuristic, and our BP-VNS. The table shows the average of the best solutions costs, the number of optimal solutions obtained by each method (#opt), and the running times in seconds. Again, we approximated the running times of the BIMM by dividing its reported runtime by 1.2 (PassMark Software Pty Ltd, 1998) as they were extracted from the work of Panteli et al. (2021). We also show the gap related to the best-known solutions, which in this case are the solutions from the solver, since the commercial solver could prove optimality for all instances. The detailed results of these tests are presented in Appendix B.

The results indicate that the proposed BP-VNS outperformed the BIMM heuristic regarding solution quality and computational performance in the two sets of 40 instances. Indeed, as the detailed results of Appendix B show, our heuristic found better solutions than the ones found by the BIMM in all instances, dominating that algorithm. Moreover, our method found an optimal solution for all of the 80 instances. On the other hand, the BIMM could not find any optimum, with a gap of more than 2.5% and an adjusted runtime more than 6.7 times that of our BP-VNS heuristic.

Although our heuristic ran for up to 30 min, the optimal solutions were obtained in much less time, as noted from the t_{best} (s) column. In fact, all BKS were found in less than 4 s. In addition, the average run time of BP-VNS for finding the best solutions was much faster than the ones from the commercial solver required to prove optimality.

Fig. 6 shows the box plot containing the distribution of the solution values of the methods compared for all OR-library instances. Fig. 6(a) presents the solutions value of the αNpMP with $p = 10$ and $\alpha = 5$ and those of $p = 20$ and $\alpha = 10$ are shown in Fig. 6(b). As one can note from these figures, our algorithm achieved better results than the BIMM heuristic and obtained the same optimal solutions as those from the MIP

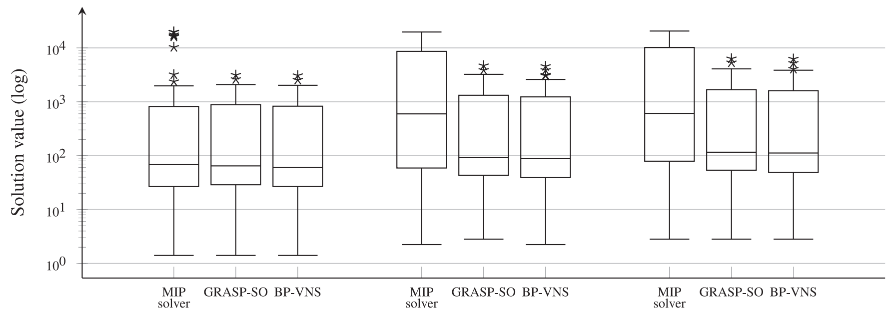
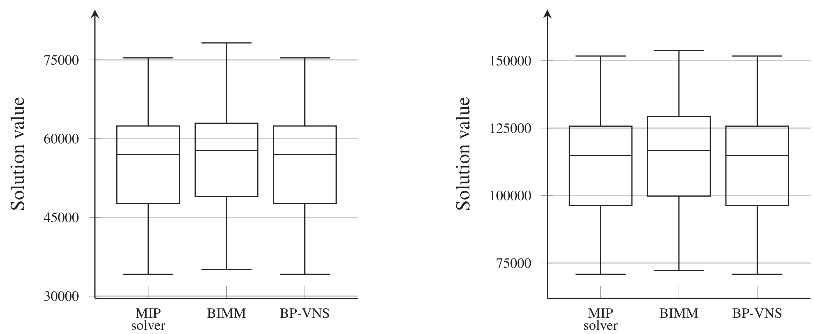


Fig. 5. Box plot of the αNpCP solution values of all methods on the TSPLIB instances.



(a) $p = 10$ and $\alpha = 5$. (b) $p = 20$ and $\alpha = 10$.

Fig. 6. Box plot of the αNpMP solution values of all methods on the OR-library instances.

Table 7
αNpMP summary results on OR-library instances.

p	α	MIP solver			BIMM (Panteli et al., 2021)				BP-VNS			
		Best	#opt	t (s)	Best	#opt	Gap (%)	t (s)[a]	Best	#opt	Gap (%)	t_{best} (s)
10	5	55 807.95	40	138.93	57 046.05	0	2.28	1.53	55 807.95	40	0.00	0.24
20	10	112 785.10	40	172.42	115 884.23	0	2.82	3.32	112 785.10	40	0.00	0.52
Average		84 296.53	40	155.67	86 465.14	0	2.55	2.43	84 296.53	40	0.00	0.38

[a] Original running times divided by 1.2, an approximation obtained from PassMark Software Pty Ltd (1998).

Table 8
αNpMP Pairwise Wilcoxon Test for the OR-library instances (p-values shown; significant difference between the performance of the algorithms if less than 0.05).

p	α	Method	Best solution values		Gap (%)		t (s)	
			BIMM	BP-VNS	BIMM	BP-VNS	BIMM	BP-VNS
10	5	MIP solver	$1.82 \cdot 10^{-12}$	1.00	$3.70 \cdot 10^{-8}$	1.00	$1.82 \cdot 10^{-12}$	$1.82 \cdot 10^{-12}$
		BIMM	–	$1.82 \cdot 10^{-12}$	–	$3.70 \cdot 10^{-8}$	–	$3.76 \cdot 10^{-6}$
20	10	MIP solver	$1.82 \cdot 10^{-12}$	1.00	$3.70 \cdot 10^{-8}$	1.00	$3.64 \cdot 10^{-12}$	$1.82 \cdot 10^{-12}$
		BIMM	–	$1.82 \cdot 10^{-12}$	–	$3.70 \cdot 10^{-8}$	–	$4.53 \cdot 10^{-6}$

solver in all 80 instances tested. However, the BP-VNS is significantly faster than both methods.

Table 8 shows the results of the Pairwise Wilcoxon Test for the OR-library instances. There is a statistically significant difference between the results of our heuristic and those of the BIMM in all scenarios, as the p-values are all less than 0.05. This corroborates the previous discussion since our BP-VNS outperformed the BIMM heuristic in all instances. On the other hand, there is no significant difference between the solution costs of our method and the ones from the MIP solver, as both achieved optimal solutions in all 80 instances. However, our heuristic is significantly faster than the commercial solver.

5. Conclusions

This paper presented an effective Basic Parallel VNS for the αNpCP and the αNpMP. Using the LIMA methodology, we have developed this heuristic using straightforward and user-friendly algorithmic components and adapting the robust and well-known algorithms of Hansen and Mladenović (1997) for the αNpMP, and Mladenović et al. (2003) for the αNpCP. Computational results indicate that the αNpC problem contains many symmetrical solutions, where only one edge determines the cost of the solution, and all remaining edges appearing in the solution are not considered. To overcome this problem and to give the algorithm more information about the whole solution, we have adapted

Table A.9
αNpCP results for the OR-library instances with $\alpha = 1$.

Instance	n	p	MIP solver			Mousavi (2023)			BP-VNS		
			Best	Gap_{opt} (%)	t (s)	Best	Gap (%)	t (s)[a]	Best	Gap (%)	t_{best} (s)
pmed1	100	5	127	0.00	4.93	127	0.00	0.00	127	0.00	0.02
pmed2	100	10	98	0.00	2.86	98	0.00	0.00	98	0.00	0.01
pmed3	100	10	93	0.00	2.85	93	0.00	0.01	93	0.00	0.46
pmed4	100	20	74	0.00	1.48	74	0.00	0.01	74	0.00	0.03
pmed5	100	33	48	0.00	1.15	48	0.00	0.00	48	0.00	0.01
pmed6	200	5	84	0.00	21.16	84	0.00	0.00	84	0.00	0.02
pmed7	200	10	64	0.00	15.46	64	0.00	0.00	64	0.00	0.05
pmed8	200	20	55	0.00	15.32	55	0.00	0.00	55	0.00	0.06
pmed9	200	40	37	0.00	6.09	37	0.00	0.00	37	0.00	0.08
pmed10	200	67	20	0.00	4.61	20	0.00	0.01	20	0.00	0.05
pmed11	300	5	59	0.00	45.49	59	0.00	0.05	59	0.00	0.03
pmed12	300	10	51	0.00	40.68	51	0.00	0.01	51	0.00	0.13
pmed13	300	30	36	0.00	23.83	36	0.00	0.02	36	0.00	0.18
pmed14	300	60	26	0.00	14.35	26	0.00	0.01	26	0.00	0.25
pmed15	300	100	18	0.00	11.17	18	0.00	0.00	18	0.00	0.12
pmed16	400	5	47	0.00	43.29	47	0.00	0.00	47	0.00	0.06
pmed17	400	10	39	0.00	102.43	39	0.00	0.00	39	0.00	0.17
pmed18	400	40	28	0.00	39.54	28	0.00	0.05	28	0.00	1.25
pmed19	400	80	18	0.00	33.28	18	0.00	0.41	18	0.00	109.08
pmed20	400	133	13	0.00	24.82	13	0.00	0.61	13	0.00	1.46
pmed21	500	5	40	0.00	128.51	40	0.00	0.00	40	0.00	0.12
pmed22	500	10	38	0.00	652.15	38	0.00	0.02	38	0.00	0.32
pmed23	500	50	22	0.00	92.78	22	0.00	0.27	22	0.00	43.18
pmed24	500	100	15	0.00	55.61	15	0.00	0.04	15	0.00	0.99
pmed25	500	167	11	0.00	48.95	11	0.00	0.05	11	0.00	0.62
pmed26	600	5	38	0.00	1414.87	38	0.00	0.00	38	0.00	0.17
pmed27	600	10	32	0.00	254.74	32	0.00	0.00	32	0.00	0.33
pmed28	600	60	18	0.00	115.44	18	0.00	0.09	18	0.00	14.64
pmed29	600	120	13	0.00	108.06	13	0.00	0.03	13	0.00	1.71
pmed30	600	200	9	0.00	93.71	9	0.00	0.63	9	0.00	182.49
pmed31	700	5	30	0.00	412.46	30	0.00	0.00	30	0.00	0.23
pmed32	700	10	29	0.00	1468.32	29	0.00	0.01	29	0.00	0.56
pmed33	700	70	15	0.00	321.06	15	0.00	0.53	15	0.00	429.87
pmed34	700	140	11	0.00	104.73	11	0.00	0.01	11	0.00	1.70
pmed35	800	5	30	0.00	882.36	30	0.00	0.01	30	0.00	0.26
pmed36	800	10	27	0.00	1178.22	27	0.00	0.03	27	0.00	0.57
pmed37	800	80	15	0.00	539.59	15	0.00	0.12	15	0.00	24.36
pmed38	900	5	29	0.00	535.79	29	0.00	0.01	29	0.00	0.40
pmed39	900	10	23	0.00	4943.95	23	0.00	0.17	23	0.00	8.95
pmed40	900	90	13	0.00	852.79	13	0.00	0.23	13	0.00	556.55
Average			37.33	0.00	366.47	37.33	0.00	0.09	37.33	0.00	34.54

[a] Original running times divided by 1.5, an approximation obtained from PassMark Software Pty Ltd (1998).

Table A.10
αNpCP results for the OR-library instances with $\alpha = 2$.

Instance	n	p	MIP solver			Mousavi (2023)			BP-VNS		
			Best	Gap_{opt} (%)	t (s)	Best	Gap (%)	t (s)[a]	Best	Gap (%)	t_{best} (s)
pmed1	100	5	150	0.00	36.53	150	0.00	0.01	150	0.00	0.02
pmed2	100	10	121	0.00	38.89	121	0.00	0.13	121	0.00	0.02
pmed3	100	10	121	0.00	116.35	121	0.00	0.17	121	0.00	0.05
pmed4	100	20	97	0.00	58.94	97	0.00	5.46	97	0.00	0.04
pmed5	100	33	63	0.00	27.94	63	0.00	0.01	63	0.00	0.07
pmed6	200	5	99	0.00	2109.87	99	0.00	0.02	99	0.00	0.06
pmed7	200	10	80	0.00	881.87	80	0.00	0.06	80	0.00	0.09
pmed8	200	20	70	0.00	654.04	70	0.00	0.02	70	0.00	0.15
pmed9	200	40	49	0.00	377.52	49	0.00	0.49	49	0.00	0.22
pmed10	200	67	28	0.00	113.37	28	0.00	0.41	28	0.00	0.24
pmed11	300	5	68	0.00	2418.26	68	0.00	0.00	68	0.00	0.11
pmed12	300	10	60	0.00	5043.11	60	0.00	0.18	60	0.00	0.71
pmed13	300	30	43	0.00	2504.04	43	0.00	1.38	43	0.00	0.49
pmed14	300	60	34	0.00	1147.22	34	0.00	0.62	34	0.00	0.52
pmed15	300	100	23	0.00	831.86	23	0.00	4.57	23	0.00	0.44
pmed16	400	5	66	96.97	7312.80	52	0.00	0.16	52	0.00	0.21
pmed17	400	10	45	0.00	3114.33	45	0.00	0.03	45	0.00	0.33
pmed18	400	40	34	0.00	3663.91	34	0.00	11.84	34	0.00	1.58
pmed19	400	80	24	0.00	3743.45	25	4.17	0.11	24	0.00	1.26
pmed20	400	133	19	0.00	642.21	19	0.00	0.83	19	0.00	5.87
pmed21	500	5	61	98.36	7200.10	45	0.00	0.80	45	0.00	32.20

(continued on next page)

Table A.10 (*continued*).

pmed22	500	10	59	98.31	7200.12	44	0.00	0.28	44	0.00	0.54
pmed23	500	50	36	88.89	7288.52	27	0.00	7.45	27	0.00	10.92
pmed24	500	100	19	0.00	7075.38	20	5.26	0.36	20	5.26	6.02
pmed25	500	167	15	0.00	2542.68	15	0.00	22.45	16	6.67	2.19
pmed26	600	5	53	98.11	7200.09	43	0.00	0.16	43	0.00	0.48
pmed27	600	10	41	97.56	7200.06	36	0.00	0.06	36	0.00	0.74
pmed28	600	60	50	98.00	7200.08	22	0.00	0.39	22	0.00	20.91
pmed29	600	120	59	98.31	7889.52	17	0.00	0.21	17	0.00	21.03
pmed30	600	200	13	0.00	4693.29	13	0.00	1.93	13	0.00	5.99
pmed31	700	5	44	97.73	7200.17	34	0.00	0.03	34	0.00	0.63
pmed32	700	10	46	97.83	7200.07	33	0.00	0.14	33	0.00	1.04
pmed33	700	70	34	97.06	7200.12	19	0.00	6.85	19	0.00	3.16
pmed34	700	140	75	98.67	7200.08	14	0.00	65.18	14	0.00	23.32
pmed35	800	5	43	97.67	7206.02	34	0.00	0.36	34	0.00	0.59
pmed36	800	10	46	97.83	7207.51	31	0.00	0.17	31	0.00	1.37
pmed37	800	80	68	98.53	7200.07	19	0.00	0.08	19	0.00	7.06
pmed38	900	5	52	98.08	7200.15	33	0.00	0.06	33	0.00	0.90
pmed39	900	10	40	97.50	7249.48	26	0.00	0.12	26	0.00	1.76
pmed40	900	90	50	100.00	7200.10	16	0.00	2.03	16	0.00	14.19
Average			54.95	43.88	4309.75	45.55	0.24	3.39	45.55	0.30	4.19

[a] Original running times divided by 1.5, an approximation obtained from PassMark Software Pty Ltd (1998).

Table A.11
αNpCP results for the OR-library instances with $\alpha = 3$.

Instance	n	p	MIP solver			BP-VNS		
			Best	Gap$_{opt}$ (%)	t (s)	Best	Gap (%)	t$_{best}$ (s)
pmed1	100	5	171	0.00	18.52	171	0.00	0.02
pmed2	100	10	138	0.00	105.23	138	0.00	0.08
pmed3	100	10	142	0.00	200.92	142	0.00	0.03
pmed4	100	20	118	0.00	304.62	118	0.00	0.59
pmed5	100	33	76	0.00	91.17	76	0.00	0.03
pmed6	200	5	110	0.00	1191.89	110	0.00	0.06
pmed7	200	10	87	0.00	1608.21	87	0.00	0.17
pmed8	200	20	75	0.00	1283.11	75	0.00	0.30
pmed9	200	40	55	0.00	955.37	55	0.00	0.92
pmed10	200	67	34	0.00	230.77	34	0.00	0.23
pmed11	300	5	72	0.00	1856.76	72	0.00	0.10
pmed12	300	10	84	96.43	7200.01	66	0.00	0.47
pmed13	300	30	48	0.00	4913.13	48	0.00	0.61
pmed14	300	60	38	0.00	3006.85	38	0.00	48.02
pmed15	300	100	27	0.00	1398.19	27	0.00	5.98
pmed16	400	5	55	0.00	4152.29	55	0.00	0.30
pmed17	400	10	52	82.69	7200.01	48	0.00	0.34
pmed18	400	40	38	0.00	6290.50	38	0.00	2.39
pmed19	400	80	28	0.00	5782.77	28	0.00	11.04
pmed20	400	133	22	0.00	3431.93	22	0.00	5.09
pmed21	500	5	58	96.55	7200.11	50	0.00	0.35
pmed22	500	10	61	98.36	9180.37	47	0.00	0.60
pmed23	500	50	34	70.59	7200.03	31	0.00	1.73
pmed24	500	100	53	98.11	7200.19	23	0.00	14.50
pmed25	500	167	34	97.06	7200.14	18	0.00	33.60
pmed26	600	5	59	96.61	7200.17	48	0.00	0.45
pmed27	600	10	52	98.08	7200.23	38	0.00	0.95
pmed28	600	60	34	97.06	8236.53	24	0.00	23.24
pmed29	600	120	28	96.43	8065.69	19	0.00	40.75
pmed30	600	200	72	98.61	7200.08	16	0.00	75.15
pmed31	700	5	47	100.00	7200.14	37	0.00	1.00
pmed32	700	10	49	97.96	7200.21	35	0.00	2.70
pmed33	700	70	61	100.00	7200.27	22	0.00	5.33
pmed34	700	140	74	98.65	7200.00	17	0.00	2.57
pmed35	800	5	45	97.78	7200.14	36	0.00	0.91
pmed36	800	10	46	97.83	7200.00	33	0.00	1.84
pmed37	800	80	37	100.00	7200.00	21	0.00	11.28
pmed38	900	5	52	98.08	7201.05	35	0.00	1.04
pmed39	900	10	40	97.50	7208.02	28	0.00	1.80
pmed40	900	90	33	96.97	7200.17	18	0.00	5.52
Average			60.98	48.28	4977.89	51.10	0.00	7.55

an evaluation function used in the bandwidth minimization problem, another a min–max optimization problem, and applied it for the first time to the αNpCP context. This updated objective function adds more information to a αNpCP solution, helping it to differentiate solutions of

Table A.12
αNpCP results for the TSPLIB instances with $\alpha = 1$.

Instance	n	p	MIP solver			GRASP-SO			BP-VNS		
			Best	Gap_{opt} (%)	t (s)	Best	Gap (%)	t (s)[a]	Best	Gap (%)	t_{best} (s)
att48	48	10	1 203.18	0.00	0.35	1203.18	0.00	2.18	1203.18	0.00	0.11
		20	710.72	0.00	0.22	710.77	0.01	0.76	710.72	0.00	0.06
		30	462.08	0.00	0.17	462.08	0.00	0.26	462.08	0.00	0.05
		40	319.85	0.00	0.31	319.85	0.00	0.07	319.85	0.00	0.04
eil101	101	10	14.14	0.00	9.46	14.32	1.27	30.64	14.14	0.00	0.03
		20	10.05	0.00	3.33	10.30	2.49	10.00	10.05	0.00	0.05
		30	8.06	0.00	1.59	8.25	2.36	5.59	8.06	0.00	2.83
		40	7.21	0.00	1.2	7.28	0.97	3.40	7.21	0.00	0.03
		50	6.70	0.00	0.88	7.07	5.52	2.12	6.70	0.00	0.03
		60	5.83	0.00	0.87	6.32	8.40	1.27	5.83	0.00	0.42
		70	5.00	0.00	0.73	5.00	0.00	0.65	5.00	0.00	0.01
		80	4.12	0.00	0.89	4.12	0.00	0.29	4.12	0.00	0.10
		90	3.16	0.00	0.8	3.16	0.00	0.09	3.16	0.00	0.02
		100	1.41	0.00	3.98	1.41	0.00	0.06	1.41	0.00	0.01
ch150	150	10	141.53	0.00	34.16	141.53	0.00	97.60	141.53	0.00	0.13
		20	94.93	0.00	12.84	97.13	2.32	47.02	94.93	0.00	0.36
		30	76.62	0.00	4.05	79.56	3.84	21.19	76.62	0.00	1.24
		40	64.45	0.00	3.72	68.23	5.87	14.38	64.45	0.00	1.06
		50	54.02	0.00	2.49	60.94	12.81	9.11	54.02	0.00	1.53
		60	46.27	0.00	2.65	49.64	7.28	7.40	46.27	0.00	0.15
		70	42.27	0.00	2.32	46.48	9.96	5.15	42.27	0.00	1.02
		80	39.10	0.00	2.51	41.46	6.04	3.86	39.10	0.00	0.06
		90	35.39	0.00	2.2	38.38	8.45	2.56	35.39	0.00	0.11
		100	32.30	0.00	2.19	33.47	3.62	1.76	32.30	0.00	0.05
		110	29.44	0.00	2.18	30.18	2.51	1.12	29.44	0.00	0.03
		120	26.61	0.00	2.16	27.36	2.82	0.65	26.61	0.00	0.05
		130	22.46	0.00	2.19	22.45	0.00	0.31	22.45	0.00	0.04
		140	17.58	0.00	2.21	17.58	0.00	0.13	17.58	0.00	0.04
pr439	439	10	1 971.83	0.00	369.77	1971.83	0.00	2118.65	1971.83	0.00	4.88
		20	1 185.59	0.00	130.6	1200.26	1.24	1842.95	1185.59	0.00	5.56
		30	883.53	0.00	162.47	886.71	0.36	895.65	883.53	0.00	4.01
		40	671.75	0.00	105.13	728.87	8.50	576.47	671.75	0.00	37.08
		50	564.03	0.00	71.1	600.00	6.38	346.49	564.03	0.00	10.78
		60	500.00	0.00	49.99	548.29	9.66	270.69	500.00	0.00	5.48
		70	474.34	0.00	80.87	500.00	5.41	206.22	475.66	0.28	46.27
		80	412.31	0.00	64.22	475.66	15.36	183.19	412.31	0.00	61.71
		90	395.28	0.00	83.27	416.08	5.26	154.33	395.28	0.00	192.35
rat575	575	10	72.67	45.73	3853.01	73.00	0.45	952.84	72.67	0.00	5.49
		20	49.65	51.58	7200.23	50.80	2.90	563.13	49.37	0.00	9.32
		30	41.04	24.74	7200.16	41.79	5.64	299.96	39.41	0.00	73.14
		40	33.42	33.03	7200.06	36.36	8.79	206.32	34.01	1.76	448.37
		50	29.43	2.87	4165.68	32.56	10.64	135.53	30.02	2.01	271.12
		60	27.00	0.00	4713.32	29.53	9.37	113.64	27.02	0.07	128.25
		70	24.76	0.00	2953.35	27.66	11.72	98.71	25.00	0.97	1312.73
		80	23.35	0.00	1601	25.50	9.23	83.20	23.35	0.02	1716.61
		90	21.93	0.00	472.15	24.19	10.30	67.68	22.09	0.72	167.66
		100	20.62	0.00	218.54	22.80	10.60	61.94	20.81	0.94	367.16
rat783	783	10	83.49	0.00	3811.9	85.23	2.08	2117.65	83.49	0.00	2.96
		20	329.20	22.21	7200.07	59.77	5.14	1486.40	56.85	0.00	86.02
		30	63.64	31.71	7200.08	49.04	4.70	896.69	46.32	0.00	1078.38
		40	64.38	1.85	7805.94	43.05	6.80	730.84	39.81	0.00	643.47
		50	44.60	0.00	7805.9	37.95	7.08	546.27	35.38	0.00	340.72
		60	34.67	0.00	7806.8	34.79	7.88	485.51	32.02	0.00	1568.18
		70	30.02	0.00	8433.45	32.20	10.24	403.09	29.07	0.00	398.34
		80	26.93	0.00	7200.52	30.08	11.71	354.31	27.31	1.43	1298.62
		90	25.94	0.00	7611.35	28.16	8.55	321.51	26.08	0.53	1404.50
		100	24.04	0.00	1906.5	27.02	12.39	258.48	24.74	2.90	21.99
pr1002	1002	10	3 200.00	0.00	7200.11	2610.08	2.75	2117.69	2540.18	0.00	24.30
		20	2 371.17	88.54	7200.08	1795.13	2.92	2117.66	1726.27	0.00	181.53
		30	1 403.57	53.10	7200.17	1439.62	4.10	1902.38	1350.93	0.00	774.82
		40	1 303.84	60.67	7200.07	1253.99	3.60	1517.40	1188.49	0.00	1516.24
		50	1 029.56	47.48	7200.11	1096.59	6.51	1168.76	1029.56	0.00	452.55
		60	912.41	38.35	3048.71	999.64	9.56	1160.79	943.40	3.40	1455.12
		70	850.00	29.06	4662.32	919.24	8.15	1053.49	851.47	0.17	355.77
		80	761.58	16.19	747.44	851.47	11.80	819.44	761.58	0.00	1038.36
		90	715.89	22.06	923.9	790.57	10.43	660.44	728.01	1.69	1002.65
		100	670.82	0.00	498.86	756.64	12.79	548.54	694.62	3.55	1693.61

(continued on next page)

Table A.12 (*continued*).

rl1323	1323	10	10 288.80	80.45	7200.17	3130.67	1.73	2117.71	3077.30	0.00	82.73
		20	19 687.52	93.34	7200.24	2088.39	3.29	2117.68	2020.35	0.00	726.64
		30	19 687.52	94.80	7200.15	1745.76	6.99	2117.67	1631.69	0.00	1316.55
		40	17 617.35	95.14	7200.13	1451.77	5.38	2117.66	1377.68	0.00	377.41
		50	19 687.52	96.22	7200.45	1290.32	6.11	2117.66	1206.07	0.00	589.28
		60	19 687.52	96.61	7200.16	1191.50	9.26	2117.66	1087.53	0.00	1449.15
		70	16 670.03	96.37	7200.13	1075.86	8.45	2117.67	992.00	0.00	899.69
		80	1 047.37	45.45	7200.12	987.47	5.77	2103.61	933.59	0.00	1758.99
		90	16 075.14	96.82	7200.16	926.77	7.09	1686.99	857.76	0.00	1477.00
		100	787.10	0.00	6615.58	880.00	11.80	1564.48	803.23	2.05	1708.71
Average			2153.49	17.72	2920.51	505.43	5.63	653.16	481.70	0.29	397.43

[a] Original running times divided by 0.85, approximation obtained from PassMark Software Pty Ltd (1998).

Table A.13
αNpCP results for the TSPLIB instances with $\alpha = 2$.

Instance	n	p	MIP solver			GRASP-SO			BP-VNS		
			Best	Gap_{opt} (%)	t (s)	Best	Gap (%)	t (s)[a]	Best	Gap (%)	t_{best} (s)
att48	48	10	1 592.12	0.00	7.25	1592.12	0.00	2.18	1592.12	0.00	0.19
		20	1 061.69	0.00	6.00	1130.85	6.51	0.76	1061.69	0.00	0.22
		30	729.90	0.00	1.21	936.38	28.29	0.26	729.90	0.00	0.09
		40	485.06	0.00	1.04	532.08	9.69	0.07	485.06	0.00	0.05
eil101	101	10	21.21	0.00	75.40	21.21	0.00	30.64	21.21	0.00	0.04
		20	13.60	0.00	48.22	14.14	3.97	10.00	13.60	0.00	0.05
		30	11.05	0.00	13.13	12.00	8.60	5.59	11.05	0.00	0.06
		40	9.06	0.00	12.76	9.43	4.08	3.40	9.06	0.00	64.68
		50	8.06	0.00	16.89	8.60	6.70	2.12	8.06	0.00	0.04
		60	7.07	0.00	15.45	8.25	16.69	1.27	7.07	0.00	0.06
		70	6.32	0.00	14.15	7.28	15.19	0.65	6.32	0.00	0.12
		80	5.10	0.00	13.20	6.32	23.92	0.29	5.10	0.00	0.19
		90	4.12	0.00	10.70	5.00	21.36	0.09	4.12	0.00	0.02
		100	2.24	0.00	5.04	2.83	26.34	0.06	2.24	0.00	0.02
ch150	150	10	205.66	0.00	250.77	205.66	0.00	97.60	205.66	0.00	0.32
		20	138.69	0.00	218.90	141.53	2.04	47.02	138.69	0.00	0.36
		30	108.03	0.00	126.20	112.51	4.15	21.19	108.03	0.00	2.29
		40	92.67	0.00	153.66	96.42	4.05	14.38	92.67	0.00	1.95
		50	82.11	0.00	145.98	87.69	6.80	9.11	82.11	0.00	1.88
		60	70.71	0.00	123.40	78.42	10.90	7.40	70.71	0.00	0.90
		70	64.45	0.00	83.45	68.23	5.87	5.15	64.45	0.00	2.09
		80	58.37	0.00	83.46	64.56	10.61	3.86	58.37	0.00	0.33
		90	51.50	0.00	87.67	62.04	20.46	2.56	51.50	0.00	0.21
		100	46.49	0.00	78.23	53.21	14.46	1.76	46.49	0.00	0.11
		110	43.77	0.00	71.87	51.65	18.01	1.12	43.77	0.00	0.05
		120	39.32	0.00	56.92	50.30	27.92	0.65	39.32	0.00	0.06
		130	36.02	0.00	56.51	46.63	29.46	0.31	36.02	0.00	0.04
		140	29.69	0.00	56.44	42.30	42.47	0.13	29.69	0.00	0.01
pr439	439	10	4 939.26	97.92	7200.04	3146.63	0.00	2118.65	3146.63	0.00	6.76
		20	2 177.44	0.00	6953.47	2226.26	2.24	1842.95	2177.44	0.00	7.05
		30	1 475.85	0.00	6198.28	1500.21	1.65	895.65	1475.85	0.00	7.92
		40	1 185.59	3.83	7200.03	1253.99	5.77	576.47	1185.59	0.00	17.65
		50	984.89	0.00	3411.00	1068.00	8.44	346.49	984.89	0.00	88.74
		60	886.71	14.25	7200.33	975.00	9.96	270.69	886.71	0.00	9.59
		70	726.72	0.00	4305.21	905.54	24.61	206.22	726.72	0.00	541.01
		80	637.38	0.00	5350.93	731.86	14.82	183.19	638.85	0.23	305.72
		90	583.10	0.00	6618.35	715.89	22.77	154.33	583.10	0.00	24.16
rat575	575	10	341.47	99.84	7200.04	116.87	0.66	952.84	116.10	0.00	42.33
		20	258.21	91.63	7213.12	74.25	1.71	563.13	72.40	0.00	9.32
		30	364.18	99.88	7200.11	60.67	4.15	299.96	57.78	0.00	73.14
		40	272.15	99.84	7200.12	51.40	4.81	206.32	49.04	0.00	1419.18
		50	390.75	99.90	7200.07	46.52	5.03	135.53	43.42	0.00	239.16
		60	62.63	99.37	7200.03	41.60	3.97	113.64	39.20	0.00	128.25
		70	59.68	99.36	7200.13	37.70	3.71	98.71	35.90	0.00	1312.73
		80	60.17	99.38	7201.95	35.90	6.85	83.20	33.24	0.00	1563.44
		90	53.34	97.38	7213.89	33.60	5.30	67.68	31.38	0.00	1243.46
		100	31.83	51.08	7200.03	31.39	7.46	61.94	29.21	0.00	1673.42

(*continued on next page*)

the same cost better. This improved objective function is demonstrated to not interfere with optimality and significantly helped the algorithm's convergence.

Despite its simplicity, our heuristic consistently achieves state-of-the-art solutions in almost all instances, and it could improve most of the best-known solutions from the literature. Specifically, in the αNpCP, the proposed VNS found the greater number of best solutions in both OR-library and TSPLIB instance sets, obtaining 328 best solutions out of 351 possible ones, including 22 new best solutions on the OR-library instances and 115 new best solutions on the TSPLIB set. Considering all these instances, the average gap obtained by our heuristic to the best-known solutions was 0.1%. Moreover, our VNS required only a

Table A.13 (*continued*).

Instance	n	p	MIP solver Best	Gap$_{opt}$ (%)	t (s)	GRASP-SO Best	Gap (%)	t (s)[a]	BP-VNS Best	Gap (%)	t_{best} (s)
rat783	783	10	544.47	99.91	7200.00	138.60	2.48	2117.65	135.25	0.00	5.69
		20	608.55	99.93	7200.00	86.38	2.74	1486.40	83.10	0.00	38.79
		30	608.55	99.94	7200.00	70.84	4.98	896.69	67.12	0.00	304.9
		40	608.55	99.94	7200.00	60.14	5.84	730.84	56.61	0.00	1044.09
		50	628.41	99.94	8230.08	52.80	2.29	546.27	51.62	0.00	358.91
		60	628.41	99.94	7200.00	48.75	5.25	485.51	45.62	0.00	1005.78
		70	628.41	99.95	7200.00	44.41	3.96	403.09	42.45	0.00	398.34
		80	628.41	99.95	7200.00	42.43	4.77	354.31	39.62	0.00	642.17
		90	74.09	100.00	7200.00	39.21	4.45	321.51	36.69	0.00	1317.58
		100	628.41	99.95	7200.00	37.48	5.76	258.48	34.71	0.00	21.99
pr1002	1002	10	15 502.02	99.93	7359.66	3853.89	0.00	2117.69	3853.89	0.00	23.59
		20	14 586.38	99.94	7200.00	2710.17	4.30	2117.66	2598.56	0.00	222.93
		30	14 297.73	99.91	7200.00	2150.58	4.32	1902.38	2057.30	0.00	313.41
		40	14 297.73	99.91	7200.00	1811.77	3.87	1517.40	1735.66	0.00	1483.48
		50	14 297.73	99.92	7200.00	1619.41	5.81	1168.76	1523.15	0.00	534.53
		60	17 479.42	99.95	7200.00	1431.78	4.28	1160.79	1353.70	0.00	1640.22
		70	17 479.42	99.95	7200.00	1346.29	5.05	1053.49	1258.97	0.00	430.92
		80	17 479.42	99.95	7200.00	1253.00	4.24	819.44	1167.26	0.00	995.9
		90	17 479.42	99.95	7200.00	1170.48	6.74	660.44	1077.03	0.00	1002.65
		100	17 479.42	99.95	7200.00	1079.35	4.84	548.54	1012.42	0.00	968.85
rl1323	1323	10	14 958.28	100.00	7200.23	4694.15	3.08	2117.71	4554.09	0.00	53.76
		20	13 332.43	99.93	7200.15	3227.00	5.65	2117.68	3036.90	0.00	493.29
		30	14 417.03	99.94	7200.16	2563.30	4.62	2117.67	2409.27	0.00	1382.01
		40	13 071.26	100.00	7200.25	2166.96	7.16	2117.66	2022.15	0.00	402.93
		50	14 417.53	100.00	7200.80	1907.69	5.48	2117.66	1808.50	0.00	790.25
		60	12 274.84	99.94	7200.14	1735.40	3.95	2117.66	1646.13	0.00	992.43
		70	19 687.52	99.96	7200.20	1595.20	6.85	2117.67	1493.00	0.00	1625.93
		80	19 687.52	99.97	7200.15	1440.89	4.82	2103.61	1374.60	0.00	1574.6
		90	19 687.52	99.97	7200.18	1374.72	6.23	1686.99	1294.08	0.00	923.85
		100	19 687.52	99.97	7200.15	1293.63	7.50	1564.48	1203.33	0.00	1398.54
Average			4515.61	52.63	4486.92	773.35	8.36	990.00	732.60	0.00	404.96

[a] Original running times divided by 0.85, approximation obtained from PassMark Software Pty Ltd (1998).

Table A.14
αNpCP results for the TSPLIB instances with $\alpha = 3$.

Instance	n	p	MIP solver Best	Gap$_{opt}$ (%)	t (s)	GRASP-SO Best	Gap (%)	t (s)[a]	BP-VNS Best	Gap (%)	t_{best} (s)
att48	48	10	2081.57	0.00	8.86	2186.31	5.03	7.91	2081.57	0.00	0.24
		20	1283.35	0.00	5.70	1374.48	7.10	1.89	1283.35	0.00	0.12
		30	949.29	0.00	1.35	1011.66	6.57	0.64	949.29	0.00	0.09
		40	645.88	0.00	1.18	675.00	4.51	0.09	645.88	0.00	0.05
eil101	101	10	29.43	0.00	88.35	29.43	0.01	108.75	29.43	0.00	0.07
		20	17.80	0.00	134.25	18.03	1.29	51.45	17.80	0.00	0.18
		30	13.15	0.00	145.80	14.14	7.50	22.79	13.15	0.00	0.10
		40	11.18	0.00	39.37	12.04	7.69	11.42	11.18	0.00	0.09
		50	9.43	0.00	14.69	10.63	12.73	5.58	9.43	0.00	0.17
		60	8.06	0.00	16.10	9.06	12.41	2.61	8.06	0.00	0.51
		70	7.28	0.00	18.92	8.54	17.31	1.25	7.28	0.00	0.03
		80	6.40	0.00	17.31	7.28	13.75	0.52	6.40	0.00	0.02
		90	5.00	0.00	16.98	6.08	21.60	0.13	5.00	0.00	0.02
		100	2.83	0.00	4.11	2.83	0.06	0.06	2.83	0.00	0.02
ch150	150	10	297.96	0.00	386.40	298.56	0.20	468.27	297.96	0.00	0.20
		20	176.47	0.00	913.48	179.71	1.84	177.58	176.48	0.01	2.55
		30	137.46	0.00	1332.72	146.41	6.51	91.86	137.46	0.00	6.52
		40	114.47	0.00	904.84	119.22	4.15	61.29	114.48	0.01	107.55
		50	100.34	0.00	1556.91	108.03	7.67	31.41	100.34	0.00	60.70
		60	90.58	0.00	604.79	97.46	7.60	20.92	90.58	0.00	71.95
		70	83.19	0.00	192.81	92.82	11.58	15.41	83.19	0.00	65.70
		80	74.93	0.01	159.37	83.38	11.28	9.81	74.94	0.02	247.78
		90	67.73	0.00	97.76	79.81	17.84	5.59	67.73	0.00	0.72
		100	63.42	0.00	91.36	69.35	9.35	3.80	63.42	0.00	2.33
		110	59.04	0.00	87.13	67.22	13.86	2.18	59.04	0.00	0.04
		120	52.97	0.00	77.66	61.29	15.71	1.12	52.97	0.00	0.11
		130	44.46	0.00	59.64	57.50	29.34	0.48	44.46	0.00	0.05
		140	38.56	0.00	54.57	52.20	35.37	0.19	38.56	0.00	0.03

(*continued on next page*)

fraction of a second or, at most, a few seconds to find these solutions. A similar performance could be seen in the αNpMP tests. In this case, our heuristic could find the optimal solutions in all 80 instances. Comparisons against the literature demonstrate that our VNS outperformed the BIMM heuristic in all instances. Again, the computational performance of the VNS was remarkable.

Our VNS algorithm's performance across various α values in solving many instances and its simplicity and user-friendliness make it an efficient choice for tackling the αNpCP and αNpMP optimization problems.

Table A.14 (*continued*).

		10	7385.88	99.55	7200.09	4076.23	0.64	2118.20	4050.31	0.00	8.58
		20	2725.46	77.44	7200.09	2726.03	1.59	2117.75	2683.28	0.00	182.96
		30	4907.27	99.47	7200.02	2231.73	8.05	2118.21	2065.49	0.00	34.29
		40	1637.83	73.66	7200.08	1644.88	2.75	2118.14	1600.78	0.00	19.62
pr439	439	50	3692.98	99.01	7635.19	1467.35	8.69	2117.71	1350.00	0.00	347.90
		60	1844.76	92.24	7200.02	1340.01	14.98	2117.68	1150.27	0.00	213.16
		70	1886.80	77.51	7200.02	1231.11	22.16	1548.82	1005.61	0.00	101.09
		80	1631.91	91.19	7200.02	1217.58	33.00	1124.40	915.49	0.00	182.67
		90	1566.25	91.81	7236.37	986.47	23.31	851.04	800.00	0.00	105.72
		10	462.77	99.82	7233.33	140.52	1.20	2117.66	138.85	0.00	4.29
		20	212.36	99.65	7296.31	94.64	0.29	2117.65	93.43	0.00	809.05
		30	139.90	99.52	7200.03	74.52	2.86	1295.68	72.09	0.00	523.44
		40	530.55	99.88	7527.81	64.88	2.97	1118.25	62.61	0.00	1596.92
rat575	575	50	411.02	99.85	7200.11	56.94	5.06	843.99	54.08	0.00	527.62
		60	396.13	99.85	7206.62	51.35	4.97	700.12	48.92	0.00	512.56
		70	400.70	99.86	7200.03	47.85	4.29	581.38	45.61	0.00	103.79
		80	417.90	99.87	7200.03	44.29	5.75	527.33	41.77	0.00	876.34
		90	62.30	99.12	7200.03	41.11	4.87	375.45	39.12	0.00	1569.39
		100	61.91	99.14	7200.03	38.63	5.06	291.72	36.62	0.00	1595.54
		10	550.718	99.87	7210.81	166.23	1.56	2117.71	163.68	0.00	5.01
		20	548.352	99.88	7200.00	112.70	2.79	2117.68	109.57	0.00	967.73
		30	608.547	99.90	7200.00	88.57	4.69	2117.66	83.55	0.00	1151.44
		40	608.547	99.91	7200.00	76.03	3.99	2117.66	72.45	0.00	1408.33
		50	608.547	99.91	7200.00	66.10	4.00	2117.65	63.53	0.00	461.70
rat783	783	60	628.405	100.00	7200.00	60.02	3.54	1903.00	56.72	0.00	1558.57
		70	628.405	99.92	7200.00	55.44	3.88	1931.82	53.16	0.00	265.23
		80	628.405	99.92	7200.00	51.66	4.05	1670.87	49.58	0.00	223.17
		90	628.405	99.92	7200.00	48.47	5.14	1425.35	45.88	0.00	1175.57
		100	628.405	99.92	7200.00	45.88	4.15	1199.53	43.42	0.00	496.48
		10	15 250.25	99.89	7296.08	5331.28	2.48	2117.76	5202.16	0.00	94.97
		20	14 205.02	99.90	7418.43	3290.14	3.77	2117.69	3170.57	0.00	36.92
		30	13 217.13	99.96	7200.00	2644.33	0.94	2117.68	2598.56	0.00	272.51
		40	14 297.73	99.91	7200.00	2304.89	4.52	2117.68	2191.46	0.00	1244.29
pr1002	1002	50	14 297.73	99.91	7200.08	2013.08	4.80	2117.67	1920.94	0.00	1582.57
		60	17 479.42	99.95	7200.00	1838.48	5.10	2117.68	1749.29	0.00	372.35
		70	17 479.42	99.95	7200.00	1710.26	5.86	2117.67	1607.02	0.00	610.14
		80	17 479.42	99.93	7200.08	1518.22	3.72	2117.66	1460.31	0.00	484.96
		90	17 479.42	99.95	7200.00	1442.22	5.68	2117.66	1360.15	0.00	1327.29
		100	17 479.42	99.95	7200.00	1353.70	3.82	2117.65	1274.75	0.00	1121.09
		10	17 207.72	0.00	7026.47	6313.82	1.35	2117.92	6229.60	0.00	56.51
		20	13 688.24	100.00	7200.31	4032.83	4.87	2117.75	3845.66	0.00	348.67
		30	15 039.71	99.92	7200.22	3204.16	4.04	2117.73	3054.32	0.00	1456.46
		40	16 174.19	100.00	7200.21	2774.72	6.25	2117.72	2575.08	0.00	1595.55
rl1323	1323	50	17 106.54	99.93	7200.14	2430.27	8.43	2117.69	2241.23	0.00	1062.41
		60	12 963.25	100.00	7200.21	2149.14	5.43	2117.69	2030.78	0.00	1285.23
		70	20 521.99	99.95	7200.16	1997.22	5.71	2117.69	1873.54	0.00	219.66
		80	20 521.99	99.95	7200.16	1842.10	5.53	2117.68	1744.86	0.00	1600.25
		90	20 521.99	99.95	7200.17	1745.58	6.55	2117.67	1637.63	0.00	1488.66
		100	20 521.99	99.95	7203.75	1620.92	5.02	2117.66	1533.75	0.00	458.90
Average			4881.12	60.99	4687.35	997.18	7.46	1147.78	945.06	0.00	445.66

[a] Original running times divided by 0.85, approximation obtained from PassMark Software Pty Ltd (1998).

The updated αNpCP objective function has shown to be a practical approach to help the heuristic escape from local optima, showing promising applications in other optimization problems where there are many solution symmetries, such as in other min–max problems.

For future works, one can apply this BP-VNS heuristic, for example, to the capacitated extensions of the αNpCP and αNpMP. Moreover, we expect the improved objective function to show promising results in other problems, given that it increases the amount of information used by the algorithm without much computational burden.

CRediT authorship contribution statement

Guilherme O. Chagas: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Luiz A.N. Lorena:** Conceptualization, Validation, Writing – review & editing. **Rafael D.C. dos Santos:** Conceptualization, Writing – review & editing, Validation. **Jacques Renaud:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Supervision, Validation, Writing – review & editing, Project administration. **Leandro C. Coelho:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Data availability

benchmark data from the literature, references and links provided.

Acknowledgments

This work was partly supported by the Canadian Natural Sciences and Engineering Research Council (NSERC) [grant numbers 2023-04167 and 2019-00094], by the CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Coordination for the Improvement of Higher Education Personnel), Brazil and by the CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico (National Council for Scientific and Technological Development), Brazil [grant number 301752/2019-2] in Brazil. We thank the Digital Research Alliance of Canada for providing high-performance parallel

Table B.15
αNpMP results with $p = 10$ and $\alpha = 5$.

Instance	n	MIP solver			BIMM			BP-VNS		
		Best	Gap_{opt} (%)	t (s)	Best	Gap (%)	t (s)[a]	Best	Gap (%)	t_{best} (s)
pmed1	100	40 592	0.00	0.41	41 462	2.14	0.09	40 592	0.00	0.01
pmed2	100	39 421	0.00	0.40	40 134	1.81	0.04	39 421	0.00	0.02
pmed3	100	43 345	0.00	0.56	44 000	1.51	0.10	43 345	0.00	0.01
pmed4	100	46 854	0.00	0.58	51 351	9.60	0.14	46 854	0.00	0.01
pmed5	100	34 167	0.00	0.42	35 054	2.60	0.11	34 167	0.00	0.01
pmed6	200	50 759	0.00	3.59	52 734	3.89	0.25	50 759	0.00	0.04
pmed7	200	44 978	0.00	2.86	46 621	3.65	0.40	44 978	0.00	0.09
pmed8	200	49 837	0.00	2.88	51 064	2.46	0.44	49 837	0.00	0.04
pmed9	200	47 636	0.00	3.14	48 638	2.10	0.05	47 636	0.00	0.13
pmed10	200	36 864	0.00	3.38	37 968	2.99	0.61	36 864	0.00	0.09
pmed11	300	46 297	0.00	19.16	47 657	2.94	0.08	46 297	0.00	1.05
pmed12	300	53 082	0.00	18.10	54 997	3.61	0.66	53 082	0.00	0.87
pmed13	300	48 257	0.00	18.78	49 012	1.56	0.56	48 257	0.00	0.08
pmed14	300	55 342	0.00	20.43	56 304	1.74	0.59	55 342	0.00	0.14
pmed15	300	47 426	0.00	17.12	47 581	0.33	0.04	47 426	0.00	0.08
pmed16	400	49 941	0.00	47.65	51 171	2.46	0.38	49 941	0.00	0.14
pmed17	400	53 403	0.00	49.11	55 475	3.88	0.35	53 403	0.00	0.20
pmed18	400	59 089	0.00	50.53	59 734	1.09	0.45	59 089	0.00	0.18
pmed19	400	56 234	0.00	49.40	57 270	1.84	0.13	56 234	0.00	0.42
pmed20	400	58 389	0.00	49.58	59 239	1.46	2.95	58 389	0.00	0.16
pmed21	500	56 961	0.00	93.45	57 735	1.36	0.09	56 961	0.00	0.04
pmed22	500	62 650	0.00	135.57	64 217	2.50	1.02	62 650	0.00	0.01
pmed23	500	60 660	0.00	107.13	62 488	3.01	0.21	60 660	0.00	0.01
pmed24	500	60 210	0.00	105.11	61 725	2.52	0.68	60 210	0.00	0.16
pmed25	500	54 793	0.00	90.52	56 284	2.72	0.46	54 793	0.00	0.03
pmed26	600	59 347	0.00	154.40	59 955	1.02	17.75	59 347	0.00	0.35
pmed27	600	57 705	0.00	143.48	58 046	0.59	1.72	57 705	0.00	0.03
pmed28	600	58 252	0.00	195.00	59 076	1.41	1.05	58 252	0.00	0.04
pmed29	600	60 745	0.00	160.02	61 661	1.51	0.65	60 745	0.00	0.13
pmed30	600	65 738	0.00	177.32	66 300	0.85	0.60	65 738	0.00	0.06
pmed31	700	61 463	0.00	244.27	62 571	1.80	7.03	61 463	0.00	0.48
pmed32	700	67 073	0.00	290.61	68 186	1.66	1.33	67 073	0.00	0.73
pmed33	700	66 024	0.00	239.31	67 924	2.88	2.21	66 024	0.00	0.05
pmed34	700	63 475	0.00	218.37	64 656	1.86	0.89	63 475	0.00	0.11
pmed35	800	62 408	0.00	432.30	62 937	0.85	5.14	62 408	0.00	0.19
pmed36	800	70 805	0.00	409.19	72 878	2.93	1.30	70 805	0.00	0.62
pmed37	800	74 125	0.00	381.64	74 661	0.72	2.04	74 125	0.00	0.19
pmed38	900	66 456	0.00	704.86	68 235	2.68	6.41	66 456	0.00	1.81
pmed39	900	66 129	0.00	456.37	66 604	0.72	2.13	66 129	0.00	0.69
pmed40	900	75 386	0.00	460.13	78 237	3.78	0.25	75 386	0.00	0.22
Average		55 807.95	0.00	138.93	57 046.05	2.28	1.53	55 807.95	0.00	0.24

[a] Original running times divided by 1.2, approximation obtained from PassMark Software Pty Ltd (1998).

Table B.16
αNpMP results with $p = 20$ and $\alpha = 10$.

Instance	n	MIP solver			BIMM			BP-VNS		
		Best	Gap_{opt} (%)	t (s)	Best	Gap (%)	t (s)[a]	Best	Gap (%)	t_{best} (s)
pmed1	100	84 027	0.00	0.39	88 745	5.61	0.23	84 027	0.00	0.04
pmed2	100	80 660	0.00	0.54	83 021	2.93	0.38	80 660	0.00	0.01
pmed3	100	88 180	0.00	0.36	91 166	3.39	0.23	88 180	0.00	0.05
pmed4	100	95 441	0.00	0.67	104 680	9.68	0.42	95 441	0.00	0.11
pmed5	100	70 836	0.00	0.29	72 192	1.91	0.30	70 836	0.00	0.04
pmed6	200	102 341	0.00	3.24	105 089	2.69	1.93	102 341	0.00	0.22
pmed7	200	91 465	0.00	2.68	95 486	4.40	0.99	91 465	0.00	0.22
pmed8	200	101 003	0.00	2.61	103 998	2.97	0.60	101 003	0.00	0.11
pmed9	200	96 365	0.00	3.10	99 371	3.12	0.20	96 365	0.00	0.16
pmed10	200	74 770	0.00	4.18	77 136	3.16	0.40	74 770	0.00	1.32
pmed11	300	93 903	0.00	13.30	94 851	1.01	1.40	93 903	0.00	0.21
pmed12	300	106 863	0.00	19.81	111 812	4.63	2.13	106 863	0.00	0.29

(continued on next page)

computing facilities. We also thank the editors and two anonymous referees for their valuable suggestions on an earlier version of this paper.

Appendix A. Results for the αNpCP

This appendix shows the detailed results for the αNpCP. Tables A.9, A.10 and A.11 and Tables A.12, A.13 and A.14 show the results of the tests in the OR-library instances and the TSPLIB instances, respectively. For the tests with the OR-library instances, we present the results of the commercial solver and our heuristic for $\alpha = \{1, 2, 3\}$ and the ones of the heuristic of Mousavi (2023) for $\alpha = \{1, 2\}$. For the tests with the TSPLIB instances, we show results of the commercial solver, the GRASP-SO heuristic (Sánchez-Oro et al., 2022) and our VNS for $\alpha = \{1, 2, 3\}$. Tables A.9–A.14 have the same structure where the instance name, the number of vertices, and the number of facilities are presented in the

Table B.16 (*continued*).

pmed13	300	97 837	0.00	14.21	99 802	2.01	0.29	97 837	0.00	0.31
pmed14	300	111 488	0.00	19.85	113 774	2.05	1.48	111 488	0.00	3.47
pmed15	300	96 190	0.00	19.53	98 231	2.12	0.93	96 190	0.00	0.34
pmed16	400	101 027	0.00	47.73	103 530	2.48	3.29	101 027	0.00	0.40
pmed17	400	107 608	0.00	70.44	111 679	3.78	0.92	107 608	0.00	2.70
pmed18	400	119 282	0.00	51.68	121 202	1.61	0.79	119 282	0.00	0.36
pmed19	400	113 107	0.00	50.92	115 688	2.28	2.53	113 107	0.00	1.33
pmed20	400	118 523	0.00	44.04	121 468	2.48	0.36	118 523	0.00	0.37
pmed21	500	114 895	0.00	87.38	116 754	1.62	0.89	114 895	0.00	0.00
pmed22	500	125 994	0.00	149.91	132 925	5.50	0.71	125 994	0.00	0.11
pmed23	500	122 437	0.00	100.05	127 093	3.80	0.34	122 437	0.00	0.15
pmed24	500	121 462	0.00	127.16	124 517	2.52	0.15	121 462	0.00	0.03
pmed25	500	111 435	0.00	83.16	114 231	2.51	5.67	111 435	0.00	0.07
pmed26	600	119 392	0.00	172.47	121 537	1.80	6.65	119 392	0.00	0.07
pmed27	600	116 498	0.00	135.63	117 508	0.87	4.51	116 498	0.00	0.08
pmed28	600	117 933	0.00	136.07	120 718	2.36	2.20	117 933	0.00	0.33
pmed29	600	122 339	0.00	150.88	125 649	2.71	0.99	122 339	0.00	0.17
pmed30	600	133 069	0.00	139.75	133 935	0.65	3.13	133 069	0.00	0.26
pmed31	700	123 848	0.00	240.92	129 303	4.40	21.73	123 848	0.00	0.18
pmed32	700	134 470	0.00	569.17	137 108	1.96	1.81	134 470	0.00	1.12
pmed33	700	132 822	0.00	228.99	136 182	2.53	13.63	132 822	0.00	0.40
pmed34	700	127 779	0.00	240.73	130 290	1.97	0.74	127 779	0.00	0.19
pmed35	800	125 727	0.00	427.53	127 188	1.16	10.57	125 727	0.00	0.41
pmed36	800	142 084	0.00	693.30	149 330	5.10	2.24	142 084	0.00	0.39
pmed37	800	149 976	0.00	265.97	152 607	1.75	1.95	149 976	0.00	0.53
pmed38	900	133 369	0.00	1091.66	135 485	1.59	13.59	133 369	0.00	0.76
pmed39	900	133 246	0.00	831.62	136 345	2.33	0.78	133 246	0.00	1.30
pmed40	900	151 713	0.00	654.96	153 743	1.34	20.68	151 713	0.00	2.04
Average		112 785.10	0.00	172.42	115 884.23	2.82	3.32	112 785.10	0.00	0.52

[a] Original running times divided by 1.2, approximation obtained from PassMark Software Pty Ltd (1998).

first three columns. Then, for the MIP solver results, we show the best solution found, the optimality gap (gap$_{opt}$ (%)), that is, the gap related to the branch-and-bound lower bound and the running time. For the heuristics, we also show the best solution found and the running times, but we show the gap related to the best known solution.

Appendix B. Results for the αNpMP

This appendix shows the detailed results for the αNpMP. Tables B.15 and B.16 show the results of the MIP solver, the BIMM and the VNS heuristic for the OR-library instances with $p = 10$ and $\alpha = 5$ and $p = 20$ and $\alpha = 10$, respectively. The results of the BIMM shown here were obtained from Panteli et al. (2021). These tables follow the same structure as the ones presented in Appendix A.

References

Araújo, E.J., Chaves, A.A., Lorena, L.A.N., 2020. A mathematical model for the coverage location problem with overlap control. Comput. Ind. Eng. 146, 106548.

Barbaros, C.T., Richard, L.F., Timothy, J.L., 1983. State of the art—Location on networks: A survey. Part I: The p-center and p-median problems. Manage. Sci. 29 (4), 482–497.

Beasley, J.E., 1985. A note on solving large p-median problems. European J. Oper. Res. 21, 270–273.

Beasley, J.E., 1990. OR-library: distributing test problems by electronic mail. J. Oper. Res. Soc. 41 (11), 1069–1072.

Brimberg, J., Maier, A., Schöbel, 2021. When closest is not always the best: The distributed p-median problem. J. Oper. Res. Soc. 72, 200–216.

Brimberg, J., Salhi, S., Todosijević, R., Urošević, D., 2023. Variable neighborhood search: The power of change and simplicity. Comput. Oper. Res. 155, 106221.

Callaghan, B., Salhi, S., Brimberg, J., 2019. Optimal solutions for the continuous p-centre problem and related *alpha*-neighbour and conditional problems: A relaxation-based algorithm. J. Oper. Res. Soc. 70 (2), 192–211.

Chaudhuri, S., Garg, N., Ravi, R., 1998. The p-neighbor k-center problem. Inform. Process. Lett. 65, 131–134.

Chen, D., Chen, R., 2013. Optimal algorithms for the *alpha*-neighbor p-center problem. European J. Oper. Res. 225, 36–43.

Contardo, C., Iori, M., Kramer, R., 2019. A scalable exact algorithm for the vertex p-center problem. Comput. Oper. Res. 103, 211–220.

Daskin, M., 1995. Network and Discrete Location: Models, Algorithms, and Applications. Wiley, New York.

Daskin, M.S., Maass, K.L., 2015. The p-median problem. In: Laporte, G., Nickel, S., Saldanha da Gama, F. (Eds.), Location Science. Springer International Publishing, Cham, pp. 21–45.

Fleszar, K., Hindi, K.S., 2008. An effective VNS for the capacitated p-median problem. European J. Oper. Res. 191, 612–622.

Garey, M.R., Johnson, D.S., 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, San Francisco, USA, p. 340.

Grangier, P., Gendreau, M., Lehuédéa, F., Rousseau, L.-M., 2016. An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. European J. Oper. Res. 254 (1), 80–91.

Hakimi, S.L., 1964. Optimum locations of switching centers and the absolute centers and medians of a graph. Oper. Res. 12 (3), 450–459.

Hakimi, S.L., 1965. Optimal distribution of switching centers in a communications network and some related graph-theoretic problems. Oper. Res. 13, 462–475.

Hansen, P., Brimberg, J., Urošević, D., Mladenović, N., 2009. Solving large p-median clustering problems by primal–dual variable neighborhood search. Data Min. Knowl. Discov. 19, 351–375.

Hansen, P., Mladenović, N., 1997. Variable neighborhood search for the p-median. Locat. Sci. 5 (4), 207–226.

Hansen, P., Mladenović, N., 2018. Variable neighborhood search. In: Martí, R., Pardalos, P.M., Resende, M.G.C. (Eds.), Handbook of Heuristics. Springer International Publishing, Cham, pp. 759–787.

Herré, A., Colmenar, J.M., Martí, R., Duarte, A., 2020. A parallel variable neighborhood search approach for the obnoxious p-median problem. Int. Trans. Oper. Res. 27, 336–360.

Karatas, M., Razi, N., Tozan, H., 2016. A comparison of p-median and maximal coverage location models with q-coverage requirement. Procedia Eng. 149, 169–176.

Kariv, O., Hakimi, S.L., 1979. An algorithmic approach to network location problems. I: The p-centers. SIAM J. Appl. Math. 37 (3), 513–538.

Khuller, S., Pless, R., Sussmann, Y.J., 2000. Fault tolerant k-center problems. Theoret. Comput. Sci. 242, 237–245.

Krumke, S., 1995. On a generalization of the p-center problem. Inform. Process. Lett. 56, 67–71.

Laporte, G., Nickel, S., Saldanha da Gama, F., 2015. Location Science. Springer, Switzerland, p. 644.

López-Sánchez, A.D., Sánchez-Oro, J., Herná, A.G., 2019. GRASP and VNS for solving the p-next center problem. Comput. Oper. Res. 104, 295–303.

Marín, A., Pelegrín, M., 2019. p-Median problems. In: Laporte, G., Nickel, S., Saldanha da Gama, F. (Eds.), Location Science. Springer International Publishing, Cham, pp. 25–50.

Martínez-Merino, L.I., Albareda-Sambola, M., Rodríguez-Chía, A.M., 2017. The probabilistic p-center problem: Planning service for potential customers. European J. Oper. Res. 262, 509–520.

Miller, I.F., Becker, A.D., Grenfell, B.T., Metcalf, C.J.E., 2020. Disease and healthcare burden of COVID-19 in the United States. Nat. Med. 26 (8), 1212–1217.

Mladenović, N., Alkandari, A., Pei, J., Todosijević, R., Pardalos, P.M., 2020. Less is more approach: basic variable neighborhood search for the obnoxious p-median problem. Int. Trans. Oper. Res. 27, 480–493.

Mladenović, N., Brimberg, J., Hansen, P., Moreno-Pérez, J.A., 2007. The p-median problem: A survey of metaheuristic approaches. European J. Oper. Res. 179 (3), 927–939.

Mladenović, N., Hansen, P., 1997. Variable neighborhood search. Comput. Oper. Res. 24 (11), 1097–1100.

Mladenović, N., Labbé, M., Hansen, P., 2003. Solving the p-center problem with tabu search and variable neighborhood search. Networks 42 (1), 48–64.

Mladenović, N., Todosijević, R., Urošević, D., 2016. Less is more: basic variable neighborhood search for minimum differential dispersion problem. Inform. Sci. 326, 160–171.

Mousavi, S.R., 2023. Exploiting flat subspaces in local search for p-center problem and two fault-tolerant variants. Comput. Oper. Res. 149, 106023.

Ng, R., Han, J., 1994. Efficient and effective clustering methods for spatial data mining. In: VLDB '94: Proceedings of 20th Conference Very Large Databases. ACM, San Francisco, USA, pp. 144–155.

Panteli, A., Basilis, B., Giannikos, I., 2021. On solving the multiple p-median problem based on biclustering. Oper. Res. 21, 775–799.

PassMark Software Pty Ltd, 1998. CPU benchmarks. https://www.cpubenchmark.net/. Last access: 3 September 2023.

Reese, J., 2006. Solution methods for the p-median problem: An annotated bibliography. Networks 48, 125–142.

Reinelt, G., 1991. TSPLIB - a traveling salesman problem library. ORSA J. Comput. 3 (4), 267–384.

Revelle, C., Swain, R., 1970. Central facilities location. Geogr. Anal. 2, 20–42.

Ristić, D., Mladenović, N., Ratli, M., Todosijević, R., Urošević, D., 2023. Auxiliary data structures and techniques to speed up solving of the p-next center problem: A VNS heuristic. Appl. Soft Comput. 140, 110276.

Rodriguez-Tello, E., Hao, J.-K., Torres-Jimenez, J., 2008. An improved simulated annealing algorithm for bandwidth minimization. European J. Oper. Res. 185, 1319–1335.

Sánchez-Oro, J., López-Sánchez, A.D., Herná, A.G., Duarte, A., 2022. GRASP with strategic oscillation for the α-neighbor p-center problem. European J. Oper. Res. 303, 143–158.

Torres-Jimenez, J., Izquierdo-Marquez, I., Garcia-Robledo, A., Gonzalez-Gomez, A., Bernal, J., Kacker, R.N., 2015. A dual representation simulated annealing algorithm for the bandwidth minimization problem on graphs. Inform. Sci. 303, 33–49.

Wang, H.-L., Wu, B.Y., Chao, K.-M., 2009. The backup 2-center and backup 2-median problems on trees. Networks 53, 39–49.