Optim Lett (2017) 11:1109–1121
DOI 10.1007/s11590-015-0867-6

ORIGINAL PAPER

A general variable neighborhood search for solving the uncapacitated r-allocation p-hub median problem

Raca Todosijević · Dragan Urošević · Nenad Mladenović · Saïd Hanafi

Received: 15 February 2014 / Accepted: 16 February 2015 / Published online: 4 March 2015
© Springer-Verlag Berlin Heidelberg 2015

Abstract The p-hub median problem consists of choosing p hub locations from a set of nodes with pairwise traffic demands in order to route the traffic between the origin-destination pairs at minimum cost. We accept general assumption that transportation between non-hub nodes is possible only via r-hub nodes, to which non-hub nodes are assigned. In this paper we propose a general variable neighborhood search heuristic to solve the problem in an efficient and effective way. Moreover, for the first time full nested variable neighborhood descent is applied as a local search within Variable neighborhood search. Computational results outperform the current state-of-the-art results obtained by GRASP based heuristic.

Keywords p-hub · Heuristics · Nested variable neighborhood descent · Variable neighborhood search

1 Introduction

Given a set of nodes with pairwise traffic demands, the p-hub median problem [27] consists of choosing p hub locations from the given set in order to route the traffic

R. Todosijević (✉) · N. Mladenović · S. Hanafi
LAMIH-UVHC, Le Mont Houy, Valenciennes Cedex 9, France
e-mail: racatodosijevic@gmail.com; raca.todosijevic@univ-valenciennes.fr

N. Mladenović
e-mail: nenad.mladenovic@univ-valenciennes.fr

S. Hanafi
e-mail: said.hanafi@univ-valenciennes.fr

D. Urošević
Mathematical Institute, Serbian Academy of Science and Arts, Belgrade, Serbia
e-mail: draganu@turing.mi.sanu.ac.rs

between origin–destination pairs at minimum cost. It is generally assumed that transportation between non-hub nodes i, j is possible only via hub nodes h_i, h_j, to which nodes i, j are assigned, respectively. Depending on the number of hubs that might be assigned to a node, several variants of the p-hub median problem can be distinguished:

– single allocation p-hub median problem [27]: each node is assigned to exactly one hub, which allows traffic to be sent and received only through this single hub;
– the r-allocation p-hub median problem [30]: each node can be connected to at most r (out of p) hubs, through which it sends (receives) traffic to (from) any other node;
– multiple allocation p-hub median problem [2,3]: each node can send and receive traffic through any of the p hubs.

The r-allocation p-hub median problem from [30] is a generalization of the single allocation p-hub median problem and the multiple allocation p-hub median problem. The motivation for this variant comes from the fact that the single allocation version is too restrictive, and the multiple allocation variant results in high fixed costs and complicated networks.

Mathematically, the uncapacitated r-allocation p-hub median (r-p hub median problem) problem may be stated as follows. Given n nodes and the distance matrix D, whose each entry d_{ij} represents the distance between nodes i and j. For every pair of nodes i and j, there is an amount of flow $t_{ij} \geq 0$ that needs to be transferred from i to j. Transferring t_{ij} units of flow through path $i \to h_i \to h_j \to j$ induces a cost $c_{ij}(h_i, h_j)$, which is computed as

$$c_{ij}(h_i, h_j) = t_{ij}\left(\gamma d_{ih_i} + \alpha d_{h_i h_j} + \delta d_{h_j j}\right).$$

Parameters γ, α and δ are unit rates for collection (origin–hub), transfer (hub–hub), and distribution (hub–destination), respectively. Generally, α is used as a discount factor to provide reduced unit costs on arcs between hubs, so $\alpha < \gamma$ and $\alpha < \delta$. Note that the hub nodes h_i and h_j on the path $i \to h_i \to h_j \to j$ may be equal (i.e., $h_i = h_j$). Then, the r-allocation p-hub median problem may be modelled in terms of the following variables (for details see [30]). Let z_{kk} be equal to 1 if the node k is a hub and 0 otherwise. Further, let z_{ik} be equal to 1 if non-hub node i is assigned or allocated to node k, and 0 otherwise. Finally, let f_{ijkl} denote the proportion of the traffic t_{ij} from node i to node j that travels along the path $i - k - l - j$, where k and l denote hubs. Then, the formulation of uncapacitated r-allocation p-hub median problem is as follows:

$$\min_{f_{ijkl}} \sum_{i\in N}\sum_{j\in N}\sum_{k\in N}\sum_{l\in N} t_{ij}(\gamma d_{ik} + \alpha d_{kl} + \delta d_{lj}) f_{ijkl} \tag{1}$$

subject to

$$\sum_{k\in N} z_{ik} \leq r \quad \forall i \in N \tag{2}$$

$$z_{ik} \leq z_{kk} \quad \forall i, k \in N \tag{3}$$

$$\sum_{k \in N} z_{kk} = p \tag{4}$$

$$\sum_{k \in N} \sum_{l \in N} f_{ijkl} = 1 \quad \forall i, j \in N \tag{5}$$

$$\sum_{l \in N} f_{ijkl} \leq z_{ik} \quad \forall i, j, k \in N \tag{6}$$

$$\sum_{k \in N} f_{ijkl} \leq z_{jl} \quad \forall i, j, l \in N \tag{7}$$

$$f_{ijkl} \geq 0 \quad \forall i, j, k, l \in N \tag{8}$$

$$z_{ik} \in \{0, 1\} \quad \forall i, k \in N \tag{9}$$

The set of constraints (2) ensures that each node is allocated to at most r hubs, while constraints (3) ensure that non-hub nodes can be allocated only to hub nodes. In addition, constraint (4) limits the number of chosen hubs to p. Finally, constraints (5) to (7) impose the requirement that the traffic between each pair of nodes i, j through their corresponding hubs k, l is routed entirely.

The p-hub median problem belongs to the class of NP hard problems. Furthermore, even if hubs are given, the sub-problem that allocates non-hub nodes to hubs is also NP hard [20]. The surveys of p-hub median problems may be found in Mladenović et al. [23], Alumur and Kara [1], and Campbell and O'Kelly [5]. It is well-known that the uncapacitated single and multiple allocation p-hub median problems are NP hard. Therefore, the uncapacitated r-allocation p-hub median problem is NP hard, as well.

While the single and multiple allocation p-hub median problems have been widely studied in the literature [4,7–10,15–17,21], there are only two methods proposed for solving the uncapacitated r-allocation p-hub median problem. Besides, the exact method proposed in [30], Peiro et al. [28] propose heuristic based on GRASP metaheuristic that employs three local search procedures and introduce a mechanism to eliminate low-quality solutions during the greedy phase. Therefore, they selectively apply local searches to promising solutions.

In this paper we propose a heuristic based on variable neighborhood search (VNS) [22] that uses nested variable neighborhood descent (nested VND) [15] as a local search. We continue the research direction established in [15,28]. In [28], three neighborhood structures for the r-p hub median problem, are proposed, but explored within GRASP methodology. Additionally, neighborhoods were not explored in the nested manner. In [15], a nested VND variant was suggested in its general form, but it was not applied for solving the uncapacitated single allocation p-hub median problem with three neighborhood structures since it would be time consuming. Therefore, the authors proposed and tested two Mixed-nested VND procedures. (Note that [11] is the first paper where mixed-nested VND is suggested, i.e., in each point of the j-means neighborhood, h-means and k-means are applied sequentially.) In this paper, we show that full nested VND, obtained by nesting two neighborhoods, is powerful enough, to enable us to efficiently and effectively solve r-p hub median problem (slightly different problem than the one considered in [15]). The merit of the proposed approach is disclosed on benchmark instances from the literature. It appears that the results obtained outperform those of the state-of-the-art heuristic based on GRASP. Moreover, the full Nested VND is used for the first time as a local search within VNS.

The rest of the paper is organized in the following way. In the next section we give rules of our heuristic. Section 3 gives computational results and comparative analysis, while Sect. 4 concludes paper and gives directions for future research.

2 General variable neighborhood Search for the uncapacitated r-allocation p-hub median problem

2.1 Building an initial solution

A solution of r-p hub median problem is determined by a set of hubs, the node-to-hubs assignments, and the travel paths for each pair of nodes. The choice of p hub locations as well as the node-to-hub assignments can be made at random or in a greedy way. As mentioned before, finding optimal allocations is NP hard problem even if the locations of hubs are known. If locations of hubs and node to hub allocations are known then, the optimal paths between each pair of nodes are determined as the possible routes with the lowest cost.

Now we will describe a greedy heuristic [28] so to get greedy solution. In order to construct a greedy solution, it is important to determine in advance the following functions: function $g(h)$—the attractiveness of node h to serve as a hub, and $aloc(i, h)$—the influence of allocating non-hub node i to hub h. For that purpose, we use the same functions as proposed in [28]. A more precise description of a greedy heuristic is given bellow.

The attractiveness of some node h to serve as a hub is measured using the following formula:

$$g(h) = \min_{j \in N} d_{jh} \sum_{i \in N} t_{ji}.$$

The rationale for such evaluation is the fact that if some node j is assigned to hub h, then all the traffic from j to any node i may be transferred via hub h. Once values of function g is determined for each node, the p hubs are chosen in a greedy way, i.e., as p nodes with the p smallest values of function g.

After choosing hub nodes, the node to hub allocations must be found. In order to estimate the influence of allocating the node i to a previously chosen hub h, we use the function $aloc(i, h)$, which is computed as:

$$aloc(i, h) = d_{ih} \sum_{j \in N} t_{ij} + \sum_{j \in N} t_{ij} d_{hj}.$$

The first term in $aloc(i, h)$ represents the traffic cost from i to h, while the second term measures the traffic cost associated with the arcs from h to all destinations j. Then, each node i is assigned to r hubs with the best allocation values $aloc(i, h)$.

When the locations of hubs are chosen and each node is assigned to r hubs, it is easy to find traffic routes for each pair i, j and to calculate the objective function value. Let

H_i and H_j be sets of r hubs assigned to nodes i and j, respectively. Then, for each pair i and j, hubs $h_i \in H_i$ and $h_j \in H_j$ through which the traffic is routed from i to j are chosen as those that minimize

$$c_{ij}(h_k, h_l) = t_{ij}(\gamma d_{ih_k} + \alpha d_{h_k h_l} + \delta d_{h_l j}),$$

i.e., $(h_i, h_j) = \arg\min\{c_{ij}(h_k, h_l) | h_k \in H_i, h_l \in H_j\}$. In that way the routing cost c_{ij} is obtained by searching hubs $h_i \in H_i$ and $h_j \in H_j$, which minimize $c_{ij}(h_k, h_l)$.

Note that even if sets H_i and H_j have a common hub, the traffic from i to j is not necessarily routed through that hub. Therefore, search for the best hubs h_i and h_j requires examining all r^2 possibilities for routing traffic from i to j.

2.2 Local search procedures

Solution representation As mentioned before, the construction of a solution for r-p hub median problem consists of three steps: location, assignment, and routing. However, as it was shown in the previous section, routing is uniquely determined by locating hubs and assigning hubs to nodes. Therefore, we represent a solution of r-p hub median problem by a set H containing p hubs and a matrix A, where each row i contains r hubs assigned to node i (i.e., i-th row coincides with the set H_i). Thus our solution is represented as $S = (H, A)$.

Nested variable neighborhood descent A solution of r-p hub median problem may be locally improved by changing a set of chosen hubs (set H) or by changing the assignments of hubs to nodes (i.e., changing the matrix A). Therefore, we introduce two neighborhood structures of a given solution $S = (H, A)$. The first neighborhood structure, denoted by N_H, is obtained by replacing one hub node from H by another non-hub node from $N \backslash H$. More formally,

$$N_H(S) = \left\{S' \mid S' = (H', A'), |H \cap H'| = p - 1\right\}.$$

The second neighborhood, denoted by N_A, is obtained by replacing one hub assigned to some node with another hub, while the set H remains unchanged:

$$N_A(S) = \left\{S' | S' = (H, A'), |A \backslash A'| = 1\right\}.$$

Let $H = \{h_1, h_2, \ldots h_i \ldots h_p\}$ be the set of hubs that corresponds to the solution S, and let $H' = \{h_1, h_2, \ldots h_i' \ldots h_p\}$ be a set of hubs that corresponds to the solution S' obtained by replacing hub node h_i with non-hub node h_i'. Obviously, node-to-hub assignments of solutions S and S' are not the same. In order to determine matrix A' of the solution S', it is necessary to re-evaluate the hub assignments of all vertices assigned to h_i, since those vertices cannot use the hub h_i anymore. Moreover, the routes that do not use the hub h_i should be also re-computed since the new hub h_i' could be a better choice than the one currently used.

Unfortunately, evaluating the value of each neighborhood solution from N_H requires solving allocation problem which is NP hard [20]. So, solving exactly the associated allocation problem will be quite time consuming. Therefore, we find near-optimal allocation A' of some solution $S'' \in N_H$ as follows. For each node, we firstly determine node-to-hub allocation using the greedy procedure described in Sect. 2.1 (see Algorithm 1).

Algorithm 1: Greedy allocation

```
   Function GreedyAllocation(H, A);
1  for $i \in N$ do
2      for $j = 1$ to $p$ do $value(j) = aloc(i, h_j)$;
3      Sort array $value$ in nondecreasing order i.e.,
       $value(\pi(1)) \leq value(\pi(2)) \leq \cdots \leq value(\pi(p))$;
4      for $j = 1$ to $r$ do $A[i][j] = h_{\pi(j)}$
   end
```

The solution $S'' = (H'', A'')$ obtained in this way is then improved by exploring the second neighborhood $N_A(S'')$. In that way, the so-called nested variable neighborhood descent (Nest-VND) is defined. Note that Nest-VND was suggested for the first time in Ilić et al. [15]. The reason why we use nested strategy for exploring both N_H and N_A neighborhood structures is the higher cardinality of the nested neighborhood $Nest(S)$:

$$|Nest(S)| = |N_H(S)| \cdot |N_A(S)|.$$

Such cardinality obviously increases chances to find an improvement in that nested neighborhood. The outline of Nest-VND procedure is provided below. The first improvement strategy, in which an improving move is executed as soon as it is detected, is used.

Algorithm 2: Nested VND.

```
   Function NestVND(S);
1  for each $S'' \in N_H(S)$ do
2      GreedyAllocation(H'', A'');
3      Select $S'$ as a best solution in $N_A(S'')$;
4      if $S'$ is better than $S$ then $S \leftarrow S'$;
   end
5  return S;
```

2.3 General variable neighborhood search for r-p-hub problem

General variable neighborhood search (GVNS) is a variant of Variable neighborhood search (VNS) metaheuristics [22]. VNS consists of an improvement phase, in which the current solution is possibly improved by examining one or more neighborhood structures, and of so-called shaking phase, in which the local minima trap problem is hopefully resolved. The most common VNS variants are: Basic VNS, where just one neighborhood structure is explored in the improvement phase; GVNS that explores deterministically several neighborhoods in the improvement phase, i.e., it uses VND as a local search. As it is well known, the VND search may be organized in sequential and nested way. For more variants of VNS and search strategies within VNS, we refer the reader to recent surveys on VNS [13,14]. It should be noted that VNS has been successfully applied for solving many optimization problems (see e.g. [6,18,19,24,26,29] for some recent successful applications).

The steps of our GVNS for solving r-p hub median problem is provided in Algorithm 4. Our `GVNS_RP` finds an initial solution by the greedy procedure described in Sect. 2.1. That solution is then improved by procedures `NestVND` (depicted at Algorithm 2) and `Shake` (depicted at Algorithm 3). These two subroutines, together with neighborhood change step, are run alternately until some stopping criterium is met. Here we used the maximum CPU time $t_{\max}$. The shaking procedure consists of replacing k hubs of a given solution with k non-hub nodes. The maximum number of replaced hubs is denoted by $k_{\max}$. It represents the VNS parameter.

Algorithm 3: Shaking procedure

```
   Function Shake(S,k);
 1 for $i = 1$ to $k$ do
 2     Select $S'$ in $N_H(S)$ at random;
 3     $S \leftarrow S'$;
   end
```

Algorithm 4: GVNS for solving r-p hub median problem.

```
   Function GVNS_RP($S$, $k_{max}$, $t_{max}$);
 1 $S \leftarrow$ greedy solution;
 2 repeat
 3     $k \leftarrow 1$;
 4     while $k \leq k_{max}$ do
 5         $S' \leftarrow$ Shake($S, k$) ;
 6         $S'' \leftarrow$ NestVND($S'$) ;
 7         $k \leftarrow k + 1$;
 8         if $S''$ is better then $S$ then
 9             $S \leftarrow S''$; $k \leftarrow 1$;
           end
       end
10     $t \leftarrow$ CpuTime();
   until $t > t_{max}$;
11 Return $S$;
```

3 Computational results

All experiments described in this section have been carried out on a computer with Intel i7 2.8 GHz CPU and 16 GB of RAM.

3.1 Test instances

The testing has been performed on benchmark instances available on the following address: http://www.optsicom.es. These instances are classified in two groups:

- *The AP* (*Australian Post*) *data set* It is based on real data from the Australian postal service, and was presented by Ernst and Krishnamoorthy in 1996 [7]. This set contains 231 instances whose number of nodes n takes the following values: 60, 65, 70, 75, 80, 85, 90, 95, 100, 150 and 200. For those instances p ranges from 3 to 8, while r takes values from 2 to $p - 1$. The cost parameters γ, α, and δ, for all instances, are set to 3, 0.75, and 2 respectively. These instances do not have a symmetric flows, i.e., for a given pair of nodes (i, j), t_{ij} is not necessarily equal to t_{ji}. Furthermore, flows from a node to itself can be positive (i.e., for a given node i, t_{ii} can be strictly positive).
- *The USA423 data set* This data set is proposed in [28]. It consists of a data file concerning 423 cities in the United States. Traffics are accumulated only for the three months period. From the original data, 30 instances have been extracted, setting p equal to 3,4,5,6,7 and r to $2, 3, \dots p - 1$. For each combination of parameters p and r, the two different sets of the cost parameter values γ, α and δ are used: 0.1, 0.07, 0.09, and 0.09, 0.075, 0.08.

3.2 Heuristics compared

We tested `GVNS_RP` presented in Algorithm 4, as well as its modification named `GVNS_RP_reduced` on both data sets. Namely, instead of performing complete exploration of the $N_H(S)$, `GVNS_RP_reduced` examines just $m \cdot p$ of its elements. These elements were generated randomly, replacing each hub node with one of m randomly chosen non-hub nodes. The reason why we decided to perform such reduction relies on the fact that a neighborhood $N_H(S)$ for instances in the second data set is much larger than the neighborhood $N_H(S)$ for instances in the first data set. Therefore, exploring entirely neighborhood $N_H(S)$ within GVNS applied on instances from the second data set might be time consuming. Note that m represents a parameter of `GVNS_RP_reduced` algorithm, which defines the size of the reduced neighborhood.

3.3 Parameter calibration of VNS based heuristics

Firstly, we examine the influence of the parameter $k_{\max}$ on `GVNS_RP` algorithm. We tested different values for $k_{\max}$ for instances of all sizes. However, to save the space, here, we present results on test instance AP200-82 ($n = 200$, $p = 8, r = 2$). This instance is of medium size and reveal the typical performance of `GVNS_RP` on most of instances. The testing is performed by varying the value of $k_{\max}$ from 1 to p (i.e., 8). The time limit (i.e., $t_{\max}$) for `GVNS_RP` was set to 300 seconds. The obtained results are presented in Table 1. For each choice of $k_{\max}$ value, we report the value of the best found solution, as well as the CPU time consumed until reaching that solution.

Table 1 Comparison of `GVNS_RP` with different values of $k_{\max}$ parameter

$k_{\max}$	Value	Time
1	118,699.63	91.16
2	119,043.72	58.07
3	118,207.08	62.31
4	119,155.42	107.85
5	118,604.50	80.26
6	118,604.50	82.86
7	118,604.50	78.66
8	118,604.50	82.49

Table 2 Comparison of `GVNS_RP_reduced` algorithms with different values of m

m	Value	Time
10	23,374,476,457.25	494.80
20	23,545,640,429.15	901.35
30	24,423,827,710.26	1923.83
40	24,423,827,710.26	1936.73
50	23,374,476,457.25	865.54

From the results presented in Table 1, it follows that `GVNS_RP` offers the best solution when the parameter $k_{\max}$ is set to 3. This result is interesting and expected. Indeed, nested neighborhood is relatively large, and big jumps from local minima are not necessary since the nested VND local search is able to find better solution in their vicinity. Also, the time needed to reach that solution is about 62 s, which is the second best time with respect to all tested $k_{\max}$ values. Therefore, for the rest of testing, we set $k_{\max}$ to 3.

The second part of experiments is devoted to determining the most suitable value for the parameter m of `GVNS_RP_reduced` algorithm, whose parameter $k_{\max}$ is set to 3. The testing is performed on test instance USA423-76, setting values of m to 10, 20, 30, 40 and 50. The reason why we decided to perform testing on that instance is the fact that USA423-76 instance is among the largest test instances, and thus suitable for examining influence of `GVNS_RP_reduced` parameter m on the solution process. The time limit (i.e., $t_{\max}$) for `GVNS_RP` was set to 3600 s. The obtained results are presented in Table 2. For each choice of m value, we report the value of the best found solution, as well as the CPU time consumed until reaching that solution.

From the results given in Table 2, it appears that `GVNS_RP_reduced` offers the best solution when its parameter m is set either to 10 or 50. However, the time consumed by `GVNS_RP_reduced` with $m = 10$ is about two times shorter than the time consumed by `GVNS_RP_reduced` with $m = 50$. Additionally, under all other settings of parameter m, `GVNS_RP_reduced` consumes much more CPU time, but does not succeed to find high quality solutions. So, in all further testings of `GVNS_RP_reduced` algorithm, its parameters m and $k_{\max}$ will be set to 10 and to 3, respectively.

3.4 Comparisons with the state-of-the-art heuristic

In this section we present comparisons of GVNS based heuristics with the GRASP heuristic. On AP data set `GVNS_RP` and `GVNS_RP_reduced` are tested adjusting their parameters in the following way. The running time $t_{\max}$ is set to 60 and 300 s for instances with $n < 80$ and with $n \geq 80$, respectively. On the second data set, for both GVNS variants time limit is set to 1 h (3600 s). The results obtained on both data sets are compared with those provided by the GRASP heuristic [28]. The GRASP heuristic were executed on a computer with characteristics similar to those of our computer (Intel i7 2.7 GHz CPU and 4 GB of RAM).

In Table 3 we provide comparison of solutions found by `GVNS_RP` and `GVNS_RP_reduced` with best solutions found by GRASP on instances from AP data set. The average value of best found solutions and average CPU times needed for finding these solutions over all instances with the same number of nodes are reported. The column headings are defined as follows. In the first column of Table 3, we report the number of nodes in the considered instances, whereas in the columns 'GRASP', '`GVNS_RP`' and '`GVNS_RP_reduced`', we provide the average of best solution values found by GRASP, `GVNS_RP` and `GVNS_RP_reduced`, respectively. In columns '`time`', the average time needed to reach best found solutions for instances with n nodes are given, while in Column '`impr.(%)`', we report the percentage improvement obtained by `GVNS_RP` and `GVNS_RP_reduced` compared with the current best known values. From the reported results, it follows that within each set of instances with the same number of nodes, there is at least one instance where the best known solution is improved by `GVNS_RP` and `GVNS_RP_reduced`. Moreover, the average improvement on AP data set achieved by GVNS variants is around 0.25 %. All new best known values may be found on the web site http://www.mi.sanu.ac.rs/~nenad/rphub/.

Table 3 Comparison of GRASP and GVNS on AP instances

n	GRASP		`GVNS_RP`			`GVNS_RP_reduced`		
	Aver. value	Aver. time	Aver. value	Aver. time	% impr.	Aver. value	Aver. time	% impr.
60	122,348.90	4.59	121,829.27	3.73	0.42	121,829.27	3.46	0.42
65	123,001.53	6.66	122,689.74	5.87	0.25	122,689.74	8.27	0.25
70	123,931.76	10.51	123,604.38	5.75	0.26	123,604.38	8.64	0.26
75	124,776.42	11.11	124,650.73	5.93	0.10	124,650.73	7.52	0.10
80	125,148.22	14.40	124,844.76	9.36	0.24	124,844.76	10.23	0.24
85	125,566.58	19.48	125,378.23	13.10	0.15	125,378.23	15.79	0.15
90	124,934.99	22.95	124,734.55	12.32	0.16	124,744.00	11.52	0.15
95	125,121.18	24.27	124,926.55	25.45	0.16	124,931.36	19.35	0.15
100	125,805.04	4.81	125,588.19	10.39	0.17	125,588.19	9.73	0.17
150	126,728.85	21.42	126,307.10	24.70	0.33	126,310.54	29.08	0.33
200	129,144.44	58.86	128,788.66	98.67	0.28	128,913.20	41.83	0.18
Avg.	125,137.08	18.10	124,849.29	19.57	0.23	124,862.22	15.04	0.22

Table 4 Comparison of GRASP and GVNS variants on USA423 instances

Parameters			GRASP		`GVNS_RP`			`GVNS_RP_reduced`		
γ	α	δ	Aver. value	Aver. time	Aver. value	Aver. time	% impr.	Aver. av. value	Aver. time	% impr.
0.1	0.07	0.09	28,673,188,808	2880	28,541,499,618	1115	0.46	28,474,980,301	682	0.70
0.09	0.075	0.08	25,767,289,897	2880	25,552,243,305	1288	0.84	25,610,842,214	690	0.61

Also, it should be emphasized that just on one instance, the solution obtained by GRASP was better than the one reported by GVNS variants. Comparing average CPU times needed to solve instances, we conclude that `GVNS_RP_reduced` outperforms both `GVNS_RP` and GRASP.

In Table 4 we report summarized results on USA423 data set. On these instances, `GVNS_RP` and `GVNS_RP_reduced` have been tested setting their parameters in the previously described way. The obtained results are compared with the results obtained by GRASP [28]. For each choice of parameters γ, α, δ, and for each method, we report the average of best found solution values (columns '`av.value`'), and average CPU times (columns '`av.time`') consumed to reach best found solutions for all instances with the same parameter values. In columns '`impr.(%)`', we report the average percentage improvement of solution values achieved by GVNS variants relatively to GRASP. From the reported results, we conclude that both GVNS variants outperform GRASP heuristic regarding both solution quality and CPU time. Also, it is noted that `GVNS_RP_reduced` is significantly faster than `GVNS_RP`. That could be explained by the fact that nested VND employed within `GVNS_RP_reduced` has smaller complexity than the one used inside `GVNS_RP`. Further, regarding average solution improvement, `GVNS_RP_reduced` performs better than `GVNS_RP` on the instances whose parameters γ, α, δ are set to 0.1, 0.07,0.09 respectively. On the other hand, `GVNS_RP` achieves greater average improvement on the instances whose parameters γ, α, δ take values 0.09, 0.075 and 0.08, respectively. Moreover, for each tested instance, either `GVNS_RP` or `GVNS_RP_reduced` provided new best known solution. These new best known solutions are available at http://www.mi.sanu.ac.rs/~nenad/rphub/. Note that the average results obtained by `GVNS_RP_reduced` outperform `GVNS_RP` on the instances where parameters γ, α, δ are set to 0.1, 0.07, and 0.09 respectively, despite the fact that `GVNS_RP_reduced` does not perform complete exploration of the $N_H(S)$, which `GVNS_RP` does. This can be explained by the fact that the stopping condition is represented by maximum CPU time allowed for the search. Therefore, sometimes it is more beneficial to explore the subset of the neighborhood than the whole neighborhood, reducing the CPU time for the intensification but increasing the time for diversification (see e.g. [12]).

Finally, we use two well-known nonparametric tests for pairwise comparisons: Wilcoxon test and the Sign test to compare the proposed GVNS variants against the GRASP heuristic, taking into account all considered test instances. Wilcoxon test enables us to get the answer on the question whether two samples represent two different populations or not, while the Sign test computes the number of instances on which an algorithm improves upon the other algorithm. p-values of 0.000, returned by both tests, when used to compare `GVNS_RP` with the GRASP, and `GVNS_RP_reduced` with GRASP, confirm the superiority of GVNS variants over the GRASP heuristic.

4 Concluding remarks

In this paper we presented new General Variable Neighborhood Search (GVNS) approaches for solving the uncapacitated r-allocation p-hub median problem. The full nested Variable Neighborhood Descent (VND) is implemented for the first time as a local search routine. The merit of these approaches has been disclosed by testing them on benchmark instances, and comparing obtained results with those offered by GRASP heuristic. It appears that GVNS approaches are very efficient, providing a large number of new best known solutions.

It is worth mentioning that our algorithm may be easily adapted for solving other versions of p-hub location problems. Also, the proposed algorithm may be extended to two level GVNS approach [25] by introducing variable neighborhood search within nested VND. Thus, future research may include development of new GVNS based heuristics for p-hub location problems.

Acknowledgements The present research work has been partly supported by International Campus on Safety and Intermodality in Transportation, and by an international Chaire from "Nord-Pas-de-Calais" Region et the University of Valenciennes (France).

References

1. Alumur, S., Kara, B.Y.: Network hub location problems: the state of the art. Eur. J. Oper. Res. **190**, 1–21 (2008)
2. Campbell, J.F.: Location and allocation for distribution systems with transshipments and transportation economies of scale. Ann. Oper. Res. **40**, 77–99 (1992)
3. Campbell, J.F.: Integer programming formulations of discrete hub location problems. Eur. J. Oper. Res. **72**, 387–405 (1994)
4. Campbell, J.F.: Hub location and the p-hub median problem. Oper. Res. **44**, 923–935 (1996)
5. Campbell, J.F., O'Kelly, M.E.: Twenty-five years of hub location research. Transp. Sci. **46**(2), 153–169 (2012)
6. Carrizosa, E., Mladenović, N., Todosijević, R.: Variable neighborhood search for minimum sum-of-squares clustering on networks. Eur. J. Oper. Res. **230**(2), 356–363 (2013)
7. Ernst, A., Krishnamoorthy, M.: Efficient algorithms for the uncapacitated single allocation p-hub median problem. Location Sci. **4**(3), 139–154 (1996)
8. Ernst, A., Krishnamoorthy, M.: An exact solution approach based on shortest paths for p-hub median problems. INFORMS J. Comput. **10**, 149–162 (1998)
9. Ernst, A., Krishnamoorthy, M.: Exact and heuristic algorithms for the uncapacitated multiple allocation p-hub median problems. Eur. J. Oper. Res. **4**, 100–112 (1998)
10. Filipović, V.: An electromagnetism metaheuristic for the uncapacitated multiple allocation hub location problem. Serdica J. Comput. **5**(3), 261–272 (2011)
11. Hansen, P., Mladenović, N.: J-Means: A new local search heuristic for minimum sum-of-squares clustering. Pattern Recogn. **34**, 405–413 (2001)
12. Hansen, P., Mladenović, N.: First vs. best improvement: an empirical study. Discrete Appl. Math. **154**, 802–817 (2006)
13. Hansen, P., Mladenović, N., Moreno-Pérez, J.A.: Variable neighbourhood search: methods and applications. 4OR **6**, 319–360 (2008)
14. Hansen, P., Mladenović, N., Moreno-Pérez, J.A.: Variable neighbourhood search: methods and applications. Ann. Oper. Res. **175**, 367–407 (2010)

15. Ilić, A., Urosević, D., Brimberg, J., Mladenović, N.: A general variable neighborhood search for solving the uncapacitated single allocation p-hub median problem. Eur. J. Oper. Res. **206**(2), 289–300 (2010)
16. Kratica, J.: An electromagnetism-like metaheuristic for the uncapacitated multiple allocation p-hub median problem. Comput. Ind. Eng. **66**(4), 1015–1024 (2013)
17. Kratica, J., Stanimirović, Z., Tosić, D., Filipović, V.: Two genetic algorithms for solving the uncapacitated single allocation p-hub median problem. Eur. J. Oper. Res. **182**(1), 15–28 (2007)
18. Lazić, J., Todosijević, R., Hanafi, S., Mladenović, N.: Variable and single neighbourhood diving for MIP feasibility. Yugosl. J. Oper. Res. (2014). doi:10.2298/YJOR140417027L
19. Lopez, C.O., Beasley, J.E.: A note on solving MINLPs using formulation space search. Optim. Lett. **8**(3), 1167–1182 (2014)
20. Love, R.F., Morris, J.G., Wesolowski, G.O.: Facilities Location: Models and Methods. Elsevier Science Publishing Co., New York (1988)
21. Milanović, M.: A new evolutionary based approach for solving the uncapacitated multiple allocation p-hub median problem. In: Gao XZ, et al., editors. Soft Computing in Industrial Applications, AISC75. Berlin: Springer-Verlag; 2010.p. 81–88
22. Mladenović, N., Hansen, P.: Variable neighborhood search. Comput. Oper. Res. **24**, 1097–1100 (1997)
23. Mladenović, N., Brimberg, J., Hansen, P., Moreno-Pérez, J.A.: The p-median problem: a survey of metaheuristic approaches. Eur. J. Oper. Res. **179**(3), 927–939 (2007)
24. Mladenović, N., Todosijević, R., Urošević, D.: An efficient General variable neighborhood search for large TSP problem with time windows. Yugosl. J. Oper. Res. **22**, 141–151 (2012)
25. Mladenović, N., Todosijević, R., Urošević, D.: Two level General variable neighborhood search for attractive traveling salesman problem. Comput. Oper. Res. **52**, 341–348 (2014)
26. Mladenović, N., Urošević, D., Perez-Brito, D.: Variable neighborhood search for Minimum Linear Arrangement Problem. Yugosl. J. Oper. Res. (2014). doi:10.2298/YJOR140928038M
27. O'Kelly, M.E.: A quadratic integer program for the location of interacting hub facilities. Eur. J. Oper. Res. **32**(3), 393–404 (1987)
28. Peiró, J., Corberan, A., Marti, R.: GRASP for the uncapacitated r-allocation p-hub median problem. Comput. Oper. Res. **43**, 50–60 (2014)
29. Urošević, D.: Variable Neighborhood search for maximum diverse grouping problem. Yugosl. J. Oper. Res. **24**, 21–33 (2014)
30. Yaman, H.: Allocation strategies in hub networks. Eur. J. Oper. Res. **211**(3), 442–451 (2011)