Transportation Research Part E 106 (2017) 45–59

Contents lists available at ScienceDirect

Transportation Research Part E

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

Manpower allocation and vehicle routing problem in non-emergency ambulance transfer service

Zizhen Zhang [a], Hu Qin [b], Kai Wang [c,*], Huang He [a], Tian Liu [d]

[a] School of Data and Computer Science, Sun Yat-sen University, Guangzhou, Guangdong, China
[b] School of Management, Huazhong University of Science and Technology, No. 1037, Luoyu Road, Wuhan, China
[c] Department of Management Science and Engineering, Economics and Management School, Wuhan University, Wuhan, China
[d] School of Economics and Management, Dongguan University of Technology, Dongguan 523000, China

ARTICLE INFO

Article history:
Received 17 February 2017
Received in revised form 2 July 2017
Accepted 2 August 2017

Keywords:
Manpower allocation
Vehicle routing
Healthcare
Variable neighborhood search

ABSTRACT

We present a manpower allocation and vehicle routing problem (MAVRP), which is a real-life healthcare problem derived from the non-emergency ambulance transfer service in Hong Kong public hospitals. Both manpower and vehicles are critical resources for the hospitals in their daily operations. The service provider needs to make an effective schedule to dispatch drivers, assistants and ambulances to transport patients scattered in different locations. We formulate the MAVRP into a mathematical programming model and propose several variable neighborhood search (VNS) algorithms to solve it. We tested the VNS with steepest descent, first descent and a mixed of two descent strategies on the MAVRP instances. The computational results demonstrate the effectiveness and efficiency of the VNS algorithms. Moreover, we also conducted additional experiments to analyze the impact of the number of vehicles on the solutions of the MAVRP instances.

1. Introduction

This research studies a manpower allocation and vehicle routing problem (MAVRP), which is originated from the hospital environments. Manpower and vehicles are critical resources for the hospitals in maintaining their daily operations, especially in transporting patients everywhere. It is of great importance to well organize these resources so as to increase the effectiveness and reduce the operational costs of the hospitals.

As a typical example, a public hospital in Hong Kong provides the non-emergency ambulance transportation service. The service provider arranges drivers, assistants and ambulances to pick up elderly or disabled patients from home to clinics. The patients make requests at different places for the ambulance transportation. The patients could have different types of disabilities. Some patients who cannot complete the treatment process independently may be accompanied by their family members, so one or more seats will be reserved for one request. Moreover, some injured patients who cannot move without wheelchairs or stretchers may require special assistance. The ambulance vehicle provides adaptive seats that can be converted to beds and wheel chairs (see Fig. 1). In this case, besides drivers, additional assistants have to be staffed on the ambulance to handle wheelchairs or carry stretchers.

* Corresponding author.
E-mail addresses: zhangzizhen@gmail.com (Z. Zhang), tigerqin1980@qq.com (H. Qin), kai.wang@whu.edu.cn (K. Wang), 1020383827@qq.com (H. He), liutian@dgut.edu.cn (T. Liu).

http://dx.doi.org/10.1016/j.tre.2017.08.002
1366-5545/

Fig. 1. Adaptive seats in an ambulance.

To fulfill the patients' requests, the planner of the service provider needs to consider the following two issues simultaneously: (1) deploying a number of staff members on each ambulance; (2) designing a route for each ambulance to transport patients. The goal of the plan is to optimize the financial expenditure, which roughly consists of three parts: (1) the penalty costs of the unfulfilled requests (such requests are outsourced to external communities with additional costs); (2) the costs of deploying manpower on ambulances; and (3) the total travelling costs of each ambulance.

Fig. 2 illustrates an example of the MAVRP, where two ambulances are situated at the hospital and 8 requests are scattered in different locations. The ambulance has a limited capacity C of seats, e.g., $C = 12$. Patient request 1 reserves for 2 seats and requires 2 staff members. Other patient requests can be found in this figure. If the first ambulance serves patients 1, 2, 3 and 4, at least 3 staff members must be deployed, which is the maximum number of staff demands among 4 patient requests. Therefore, patients occupy 8 seats, staff members occupy 3 seats and 11 seats are occupied in total. Note that patient request 8 is unfulfilled and has to be outsourced.

The MAVRP extends the traditional vehicle routing problems (VRPs) by incorporating with the allocation of manpower for each vehicle. The vehicle load is constituted by both staff members and so-called customers (patients with their family members). The contributions of this paper are threefold. First, we introduce a new and practical planning and routing problem that simultaneously optimizes the resources of both manpower and vehicles. Second, we formulate the problem as an integer programming model and propose effective heuristic algorithms for solving the problem. Third, we modify the datasets from traditional capacitated VRP (CVRP) data sets to generate the MAVRP instances. The comprehensive experimental results can serve as a baseline for future research on this work or other related problems.

The remainder of this paper is organized as follows. In Section 2, we briefly describe the relevant literature on the MAVRP. We then provide a formal definition of the MAVRP in Section 3. In Section 4, we introduce variable neighborhood search (VNS) algorithms for solving this problem. To evaluate our approaches, Section 5 reports results of the experiments that

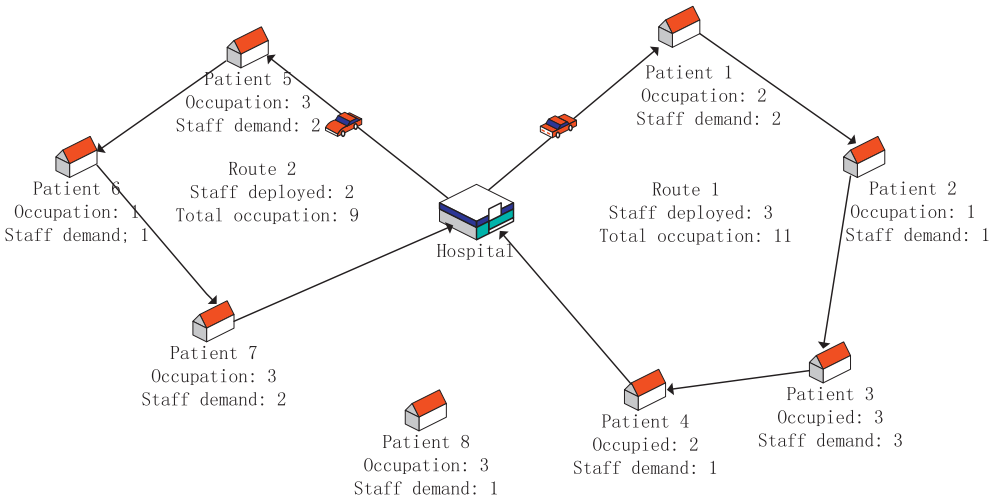


Fig. 2. An example of the manpower allocation and vehicle routing problem.

we conducted based on the classical CVRP instances and the MAVRP instances. Section 6 gives some closing remarks and some suggestions to extend this work.

2. Literature review

Vehicle routing problems (VRPs) (Laporte, 1992) are important topics in the area of transportation services. We refer the reader to Laporte (2009), Toth and Vigo (2014), Lin et al. (2014), Lai et al. (2016), Xiao and Konak (2016) and Dayarian et al. (2016) for a comprehensive overview on the VRPs. Manpower allocation problem is a kind of manpower scheduling problems, which are well-studied research topics. Generally speaking, manpower allocation problems (MAPs) deal with the assignment of manpower to duties subject to a series of constraints. Various methods and applications have been presented in the relevant literature. For more details, we refer the reader to De Bruecker et al. (2015), Ernst et al. (2004), Brucker et al. (2011) and Van den Bergh et al. (2013).

The MAVRP combines the characteristics of both VRPs and MAPs. It takes into consideration the assignment of manpower to vehicles and the design of vehicle routes. In literature, there are several studies relevant to the MAVRP. Lim et al. (2004) investigated a real-life port manpower allocation problem and gave a manpower allocation model. They proposed a tabu-embedded simulated annealing algorithm and developed a squeaky wheel optimization method for solving the problem. Li et al. (2005) studied an extension of the research of Lim et al. (2004) that considers an additional job-teaming constraint, i.e., each job requires different types of workers and must be carried out in a predetermined time window. The authors proposed construction heuristics to solve the problem. Günlük et al. (2006) introduced a decision support system for a sedan service provider that assists to schedule driver shifts and route the fleet to satisfy the customer demands. The authors formulated the problem into an integer programming model and then used column generation to solve it. Eveborn et al. (2006) considered a staff scheduling problem originating from Swedish healthcare systems. They formulated the problem into a set partitioning model and devised a repeated matching algorithm to solve it. Gutiérrez et al. (2009) focused on the combined routing and staff scheduling problem in healthcare logistics, in which the staff was scheduled and sequenced to visit patients. Then the authors formulated the problem into a mixed integer linear programming model and proved its high complexity. Kim et al. (2010) solved a combined problem of vehicle routing and manpower scheduling to carry out multi-staged tasks, in which each customer demands several tasks and these tasks are completed by a lot of teams. The authors built a mixed integer programming model for the problem and used a particle swarm optimization-based heuristic algorithm to solve the problem. Lim et al. (2017) studied a multi-trip pickup and delivery problem with time windows and manpower scheduling. The authors developed an iterated local search metaheuristic to solve the problem, which applies a variable neighborhood descent method in the local search stage. Finally, the experimental results demonstrated the performance of their proposed algorithm.

In this paper, we propose VNS algorithms (Hansen and Mladenović, 2014; Hansen and Mladenović, 2001) for solving the MAVRP. The effectiveness of the VNS algorithms has been proven in solving VRPs and other related topics (Hertz and Mittaz, 2001; Bräysy, 2003; Escobar et al., 2014; Yang et al., 2015). The results produced by our VNS algorithms can serve as a baseline of meta-heuristics for the MAVRP.

3. Problem description

The MAVRP is defined on a complete graph $G = (N, E)$, where $N = \{0, 1, \ldots, n\}$ is the set of nodes and $E = \{(i,j)|i,j \in N\}$ is the set of edges. Node 0 is the depot at which the ambulances are available. Node $i \in N \setminus \{0\}$ corresponds to a patient request. Each edge (i,j) is associated with a non-negative cost $c_{i,j}$ that indicates the travelling cost between nodes i and j. The seat reservation and staff demand for node i are denoted by r_i and d_i, respectively. Let $V = \{1, \ldots, m\}$ be the set of vehicles or ambulances. Assume that all vehicles are homogeneous with the same capacity C. If the request at node i is unfulfilled, the outsourcing cost is p_i. The cost of deploying a staff member on vehicle is supposed to be identical, denoted by q.

Let M denote a sufficiently large positive number (e.g., we can set $M \geqslant \max d_i$). The MAVRP can be formulated into the following arc-flow model:

Decision variables:

$x_{i,j}^k$: equals to 1 if edge (i,j) is in the route of vehicle k, and 0 otherwise;
y^k: the number of staff members equipped on vehicle k;
z_i^k: equals to 1 if node i is served by vehicle k, and 0 otherwise;
u_i^k: additional variable representing the load of vehicle k after visiting patient i.

Model:

$$\min \sum_{i \in N} p_i \left(1 - \sum_{k \in V} z_i^k\right) + q \sum_{k \in V} y^k + \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{i,j} x_{i,j}^k \tag{1}$$

subject to

$$\sum_{i\in N, i\neq j} x_{i,j}^k = z_j^k, \forall k \in V, j \in N \setminus \{0\} \tag{2}$$

$$\sum_{j\in N, i\neq j} x_{i,j}^k = z_i^k, \forall k \in V, i \in N \setminus \{0\} \tag{3}$$

$$\sum_{j\in N, j\neq 0} x_{0j}^k = z_0^k, \forall k \in V \tag{4}$$

$$\sum_{i\in N, i\neq 0} r_i z_i^k + y^k \leqslant C z_0^k, \forall k \in V \tag{5}$$

$$y^k + M(1 - z_i^k) \geqslant d_i, \forall i \in N \setminus \{0\}, k \in V \tag{6}$$

$$\sum_{k\in V} z_i^k \leqslant 1, \forall i \in N \setminus \{0\} \tag{7}$$

$$u_i^k - u_j^k + C x_{i,j}^k \leqslant C - r_j, \forall i, j \in N \setminus \{0\}, i \neq j, k \in V, r_i + r_j \leqslant C \tag{8}$$

$$r_i \leqslant u_i^k \leqslant C, \forall i \in N \setminus \{0\}, k \in V \tag{9}$$

$$x_{i,j}^k, z_i^k \in \{0,1\}, \forall i, j \in N, i \neq j, k \in V \tag{10}$$

$$y^k \in Z^+, \forall k \in V \tag{11}$$

In the above model, the objective (1) is to minimize the summation of outsourcing costs, manpower costs and total traveling costs. Constraints (2)–(4) are flow conservation constraints. Constraints (2) state that for each node j (not the depot), if it is visited, then only one vehicle, coming from some node $i \neq j$, visits it. Constraints (3) mean that for each node i (not the depot), if it is visited, then only one vehicle outgoing from it, visits the next node $j \neq i$. Constraints (4) ensure that if vehicle k is employed, it must visit a node (not the depot). Constraints (5) guarantee that the capacity of each vehicle should not be exceeded, where the load of a vehicle comprises staff occupations and patient reservations. Constraints (6) require that the staff on vehicle k must satisfy the staff demands of the served patients. Constraints (7) assure that each node $i \neq 0$ is served at most once. Constraints (8) and (9) are subtour elimination constraints, which were originally proposed for the travelling salesman problem by Miller et al. (1960) and later extended for the VRPs.

The MAVRP is an NP-hard problem, since there exists a polynomial time reduction from the MAVRP to the classical VRP by setting $p_i = 0, r_i = d_i = 1$ for $i \in N \setminus \{0\}$ and $q = 0$. The above mixed integer linear programming model of the MAVRP is very difficult to be solved using general commercial solvers, e.g., Ilog Cplex. Preliminary experiments showed that Cplex can only solve the MAVRP instance with $n \leqslant 20$ in a reasonable computation time. This motivates us to develop efficient metaheuristics to obtain high quality solutions of the MAVRP.

4. Solution approaches

We devise variable neighborhood search (VNS) approaches for solving the MAVRP. The VNS was first introduced by Mladenović and Hansen (1997) for complex combinatorial optimization problems. The main idea of VNS is to repeatedly apply local search methods to obtain the local optimum and then perform perturbation to escape from it. The reasons that we apply VNS in solving the MAVRP are as follows. First, VNS and its extensions have been proven their effectiveness in solving VRPs as mentioned in the literature. Second, we devised five groups of neighborhood structures for the MAVRP. It is very natural to adopt VNS approach to well organize all these neighborhood structures. Third, the MAVRP is a new problem and we would like to use relatively standard metaheuristic framework to obtain the results. Unlike many other metaheuristics, the basic scheme of VNS is quite straightforward and requires few user-defined parameters. In the following context, we first present the basic components of our VNS algorithms. Then, we discuss a basic VNS framework and extend it to a mixed VNS framework.

4.1. Solution representation and evaluation

A solution to the MAVRP consists of m vehicle routes, each of which starts from the depot, visits a sequence of patients and finally ends at the depot. The vehicle associated with the corresponding route must be assigned enough staff members to satisfy the patient requests, and its load can not exceed its capacity.

We take Fig. 2 as an example, where the vehicle capacity is assumed to be $C = 12$. There are two *vehicle routes* in this solution. The first route is $(0, 1, 2, 3, 4, 0)$. The reservations of patient requests 1 – 4 are 2, 1, 3 and 2, respectively. The corresponding staff demands are 2, 1, 3 and 1, respectively. Thus, the first route requires $max\{2, 1, 3, 1\} = 3$ staff members and the total occupation is $2 + 1 + 3 + 2 + max\{2, 1, 3, 1\} = 11$. Similarly, the total occupation of the second route $(0, 5, 6, 7, 0)$ is $3 + 1 + 3 + max\{2, 1, 2\} = 9$. Those outsourced patient requests (i.e., patient request 8 in the example) are collected into an *outsourcing route*, which has an infinite capacity. In sum, we can use a set of routes $\{(0, 1, 2, 3, 4), (0, 5, 6, 7, 0), (8)\}$ to represent the solution in Fig. 2, where the last route is an outsourcing route.

To evaluate a solution, we respectively calculate the cost of each route. For a vehicle route $R = (0, x_1, x_2, \ldots, x_k, 0)$, the *number of patient occupations*, denoted by $PO(R)$, is $\sum_{i=1}^{k} r_{x_i}$. The *number of staff members*, denoted by $NS(R)$, is $max\{d_{x_1}, d_{x_2}, \ldots, d_{x_k}\}$. Thus, the *vehicle load* is $VL(R) = PO(R) + NS(R)$. The *manpower cost* $MC(R)$ is equal to $NS(R) \cdot q$. The incurred *travelling cost* is $TC(R) = c_{0,x_1} + \sum_{i=1}^{k-1} c_{x_i,x_{i+1}} + c_{x_k,0}$. If the route is an outsourcing route, the *penalty cost* of the outsourcing patient requests is $PC(R) = \sum_{i=1}^{k} p_{x_i}$.

4.2. Initial solution generation

We modify a well-known saving algorithm proposed by Clarke and Wright (1964) to generate an initial solution of the MAVRP. This saving algorithm works as follows. First, a simple route $(0, i, 0)$ is constructed for each patient request i. Second, for each pair of routes $(0, \ldots, i, 0)$ and $(0, j, \ldots, 0)$, if they can be feasibly merged into a single route $(0, \ldots, i, j, \ldots, 0)$, then calculate the cost saving $c_{i,0} + c_{0,j} - c_{i,j}$. Note that the feasibility of merging only depends on whether the vehicle capacity limitation is violated or not. Third, merge the pair of routes with the largest saving into a single route. The algorithm repeats the second and the third steps until no further pair of routes can be feasibly merged.

The initial solution constructed by the saving algorithm may use excessive vehicles. We invoke a greedy post-processing phase to form a MAVRP solution. Suppose that the saving algorithm generates $\tilde{m}$ vehicle routes ($\tilde{m} > m$). For each route R, if it is recognized as an outsourcing route, the penalty cost is $PC(R)$. If it is not, the incurred cost is $MC(R) + TC(R)$. We calculate the difference between them: $\Delta(R) = MC(R) + TC(R) - PC(R)$, and then sort all the $\Delta(R)$ in the ascending order. The first m routes are chosen as vehicle routes and the remaining $\tilde{m} - m$ routes are concatenated into a single outsourcing route.

4.3. Local search

The local search is one of the most important factors in designing an effective and efficient meta-heuristic. In our VNS approaches, the local neighbors of a solution are defined by seven neighborhood operators in five groups. For ease of exposition, if the operator is applied to a single route R_1, we denote by $R_1 = (0, x_1, \ldots, x_k, 0)$, where the node x_i corresponds to a patient request and the route consists of k nodes. If the operator is applied to two different routes R_1 and R_2, then $R_1 = (0, x_1, \ldots, x_k, 0)$ and $R_2 = (0, y_1, \ldots, y_l, 0)$. Note that the route that is not specified can be either a vehicle route or an outsourcing route.

(1) *Exchange*. The *Exchange* operator can be either *internal-exchange* (N1) or *external-exchange* (N2). The *internal-exchange* operator is applied to only one route. The procedure is as follows: select a route R_1, choose two nodes x_i and x_j on route R_1 and change the positions of these two nodes to generate a new route R_1'. The *external-exchange* operator affects two routes. First, pick out two routes R_1 and R_2. Second, select one node x_i on route R_1 and one node y_j on route R_2. Finally, change the positions of nodes x_i and y_j, thereby generating two new routes. The *external-exchange* operator should guarantee that the vehicle loads of two new routes do not exceed the vehicle capacity.

(2) *Reverse*. The *Reverse* (N3) operator is applied to one route. First, a route R_1 is selected. Next, choose two nodes x_i and x_j with $i < j$. Finally, reverse all nodes between nodes x_i and x_j to generate a new route R_1'.

(3) *Insertion*. The *Insertion* operator tries to move a node to the next position of another node. If the two nodes are on the same route, we refer to this situation as the *Internal-insertion* (N4). Otherwise, it is referred to as the *External-insertion* (N5). For the *External-insertion*, node x_i is inserted into the vehicle route R_2. We need to check the vehicle load of the new route R_2' and ensure that it does not exceed the vehicle capacity.

(4) *2-opt*. The basic idea of the *2-opt* (N6) operator is to exchange edges in two routes. Two nodes x_i and y_j on two different routes are first selected. Next, two edges (x_i, x_{i+1}) and (y_j, y_{j+1}) are replaced by edges (x_i, y_{j+1}) and (y_j, x_{i+1}), respectively. The loads of the resultant routes must not exceed the vehicle capacity.

(5) *Reassignment*. The above operators can be regarded as the traditional VRP operators (Bräysy and Gendreau, 2005). The *Reassignment* (N7) operator is designed based on the characteristics of the MAVRP. It consists of two phases. The first phase is called conditional selection, where the condition depends on similar patient requests. For the MAVRP, each node i is associated with a patient seat reservation r_i and a staff demand d_i. The ranges of the values r_i and d_i are relatively small in practice ($r_i, d_i \in \{1, 2, 3\}$ in our real cases). The operator first determines a combination (r, d), and then finds all the nodes having the same combination. Subsequently, some of these nodes are selected according to a probability ρ, namely, each node has ρ percent chances of being selected. The second phase is to reassign the selected nodes. Specifically, all the selected node are removed from their routes and then reinserted into the routes at their best positions. Note that all the selected nodes have the same (r, d) combination. If a route has previously removed k nodes, then exactly k nodes must be reinserted into the route.

Let us take Fig. 3 as an example. Assume that the condition is set to $(r, d) = (2, 1)$. We find that the nodes $\{x_1, x_2, x_3\}$ have $(r, d) = (2, 1)$ and they are selected. Next, remove these nodes from their routes. Finally, reinsert these nodes into the routes at their best positions.

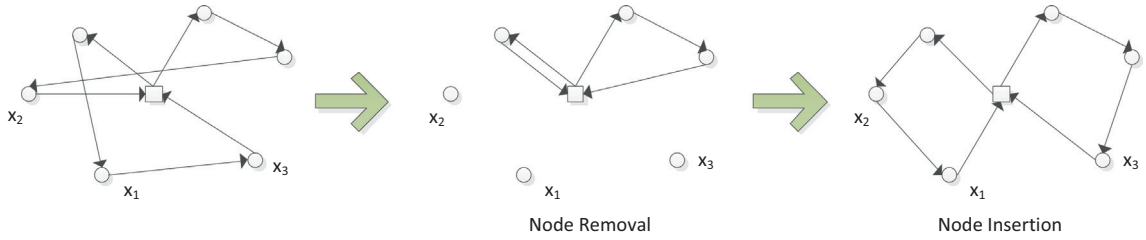


Fig. 3. The *Reassignment* operator.

4.4. Shaking process

The shaking (or perturbation) process is also essential in designing an effective VNS algorithm. It helps to escape the local optimum and thus the search has more chances to reach the global optimum. Our shaking process makes use of large neighborhood search process (Shaw, 1998). It consists of a removal phase and an insertion phase. The removal phase extracts a percentage of nodes from their corresponding routes and puts them into a *public pool*. The percentage is controlled by a prescribed parameter δ, then $q = \delta N$ nodes are to be removed. Following the discussions by (Ropke and Pisinger, 2006; Hemmelmayr et al., 2012; Mancini, 2016), we study the following three types of removal strategies.

(1) Random removal. It simply selects q nodes in the solution in a purely random manner.
(2) Worst removal. It selects q nodes with the highest removal gain. The removal gain is defined by the difference between the cost when node exists in the solution and the cost when the node is removed.
(3) Related removal. The related removal first randomly removes a node called the *seed* node, and then removes $q-1$ nodes which are most related to the seed node. The relatedness of two nodes i and j is denoted as $R(i,j)$ and is given by

$$R(i,j) = w_1 \frac{c_{i,j} - \min\limits_{i,j\in N, i\neq j} c_{i,j}}{\max\limits_{i,j\in N, i\neq j} c_{i,j} - \min\limits_{i,j\in N, i\neq j} c_{i,j}} + w_2 \frac{|r_i - r_j|}{\max\limits_{i\in N} r_i - \min\limits_{i\in N} r_i} + w_3 \frac{|d_i - d_j|}{\max\limits_{i\in N} d_i - \min\limits_{i\in N} d_i} \tag{12}$$

This equation is modified from a Ropke and Pisinger (2006). The first term in the RHS considers the normalized distance between two nodes, while the second term and third term measure the similarity based on the normalized differences between seat reservation and staff demand, respectively. w_1, w_2 and w_3 are the weights and we simply set them to the same value. A smaller $R(i,j)$ indicates that two patient requests are more related to each other.

For the insertion phase, all the removed nodes in the public pool together with the nodes of unfulfilled requests are reinserted into the vehicle routes at their best positions. The insertion should guarantee the feasibility of the routes, which implies that a node that is unable to be inserted into vehicle routes must be put into the outsourcing route.

4.5. Implementation

The order in which we arranged the local search operators basically follows the running time complexity. The operators are ranked in the increasing order of their complexities so that it is more likely to obtain a promising neighbor in the early stage. We further detail some implementation issues of these operators. The first six operators (N1–N6) are applied to either a single route R_1 or two routes R_1, R_2. The neighborhood size is $O(n^2)$ for each of them. We summarize the traveling cost and the vehicle load after applying each of them in Table 1. Remind that we have defined $TC(R), VL(R), PO(R)$ and $NS(R)$ to represent the travelling cost, vehicle load, patient occupation and the number of staff members on route R, respectively.

In this table, the column "Route" indicates the resultant route of the corresponding operator. We can see from this table that the operators N1, N3 and N4 affect the traveling cost of route R_1 but do not change its vehicle load. The time complexity of each of these operators is O(1).

For the operators N2 and N5, the traveling costs of route R_1' and R_2' can be obtained in O(1) time. However, the vehicle load is related to the number of staff members and thus cannot be directly calculated in a constant time. To accelerate the calculation, before applying these operators, we pre-process $PO(\cdot)$ and $NS(\cdot)$ for every partial routes as follows. Denote by $\overrightarrow{R^i}$ the forward partial route of $R = (0, x_1, \ldots, x_k, 0)$ obtained by visiting the first i nodes of R, i.e., $\overrightarrow{R^i} = (0, x_1, \ldots, x_i)$, and denote by $\overleftarrow{R^i} = (x_i, \ldots, x_k, 0)$ the backward partial route of R obtained by visiting the last $k-i+1$ nodes of R. Then, route R contains k forward partial routes and k backward partial routes. Once $PO(\cdot)$ and $NS(\cdot)$ are pre-computed for every forward and backward partial routes in the solution, the operators N2 and N5 only take O(1) time. Similarly, for the operator N6, the traveling costs and vehicle loads of the resultant routes R_1' and R_2' can also be obtained in O(1) using the pre-computing process.

For the *Reassignment* (N7) operator, a condition (r,d) is first made. Suppose that there are k nodes satisfying this combination. Then the time complexity is $O(\rho k n)$. This is because there are on average ρk nodes selected, and the reassignment of

Table 1
The travelling cost and the vehicle load after applying each neighborhood operator

Operator	Route	Traveling Cost & Vehicle Load
(N1) Internal-exchange(x_i, x_j)	$R_1'(0, x_1, \ldots, x_{i-1}, x_j, \ldots, x_i, x_{j+1} \ldots, x_k, 0)$	$TC(R_1') = TC(R_1) - c_{x_{i-1}x_i} - c_{x_j x_{j+1}} + c_{x_{i-1}x_j} + c_{x_i x_{j+1}}$ $VL(R_1') = VL(R_1)$
(N3) Reverse(x_i, x_j)	$R_1'(0, x_1, \ldots, x_{i-1}, x_j, x_{j-1}, \ldots, x_i, x_{j+1}, \ldots, x_k, 0)$	$TC(R_1') = TC(R_1) - c_{x_{i-1}x_i} - c_{x_j x_{j+1}} + c_{x_{i-1}x_j} + c_{x_i x_{j+1}}$ $VL(R_1') = VL(R_1)$
(N4) Internal-insertion(x_i, x_j)	$R_1'(0, x_1, \ldots, x_i, x_j, x_{i+1}, \ldots, x_{j-1}, x_{j+1}, \ldots, x_k, 0)$	$TC(R_1') = TC(R_1) - c_{x_i x_{i+1}} - c_{x_{j-1}x_j} - c_{x_j x_{j+1}} + c_{x_i x_j} + c_{x_j x_{i+1}} + c_{x_{j-1}x_{j+1}}$ $VL(R_1') = VL(R_1)$
(N2) External-exchange(x_i, y_j)	$R_1'(0, x_1, \ldots, x_{i-1}, y_j, x_{i+1}, \ldots, x_k, 0)$	$TC(R_1') = TC(R_1) - c_{x_{i-1}x_i} - c_{x_i x_{i+1}} + c_{x_{i-1}y_j} + c_{y_j x_{i+1}}$ $VL(R_1') = PO(\overrightarrow{R_1}^{i-1}) + r_{y_j} + PO(\overleftarrow{R_1}^{i+1}) + \max\{NS(\overrightarrow{R_1}^{i-1}), d_{y_j}, NS(\overleftarrow{R_1}^{i+1})\}$
	$R_2'(0, y_1, \ldots, y_{j-1}, x_i, y_{j+1}, \ldots, y_l, 0)$	$TC(R_2') = TC(R_2) - c_{y_{j-1}y_j} - c_{y_j y_{j+1}} + c_{y_{j-1}x_i} + c_{x_i y_{j+1}}$ $VL(R_2') = PO(\overrightarrow{R_2}^{j-1}) + r_{x_i} + PO(\overleftarrow{R_2}^{j+1}) + \max\{NS(\overrightarrow{R_2}^{j-1}), d_{x_i}, NS(\overleftarrow{R_2}^{j+1})\}$
(N5) External-insertion(x_i, y_j)	$R_1'(0, x_1, \ldots, x_i, y_j, x_{i+1}, \ldots, x_k, 0)$	$TC(R_1') = TC(R_1) - c_{x_i x_{i+1}} + c_{x_i y_j} + c_{y_j x_{i+1}}$ $VL(R_1') = PO(\overrightarrow{R_1}^{i}) + o_{y_j} + PO(\overleftarrow{R_1}^{i+1}) + \max\{NS(\overrightarrow{R_1}^{i}), d_{y_j}, NS(\overleftarrow{R_1}^{i+1})\}$
	$R_2'(0, y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_l, 0)$	$TC(R_2') = TC(R_2) - c_{y_{j-1}y_j} - c_{y_j y_{j+1}} + c_{y_{j-1}y_{j+1}}$ $VL(R_2') = PO(\overrightarrow{R_2}^{j-1}) + PO(\overleftarrow{R_2}^{j+1}) + \max\{NS(\overrightarrow{R_2}^{j-1}), NS(\overleftarrow{R_2}^{j+1})\}$
(N6) 2-opt(x_i, y_j)	$R_1'(0, \ldots, x_{i-1}, x_i, y_{j+1}, y_{j+2}, \ldots, y_l, 0)$	$TC(R_1') = TC(\overrightarrow{R_1}^{i}) + c_{x_i y_{j+1}} + TC(\overleftarrow{R_2}^{j+1})$ $VL(R_1') = PO(\overrightarrow{R_1}^{i}) + PO(\overleftarrow{R_2}^{j+1}) + \max\{NS(\overrightarrow{R_1}^{i}), NS(\overleftarrow{R_2}^{j+1})\}$
	$R_2'(0, \ldots, y_{j-1}, y_j, x_{i+1}, x_{i+2}, \ldots, x_k, 0)$	$TC(R_2') = TC(\overrightarrow{R_2}^{j}) + c_{y_j x_{i+1}} + TC(\overleftarrow{R_1}^{i+1})$ $VL(R_2') = PO(\overrightarrow{R_2}^{j}) + PO(\overleftarrow{R_1}^{i+1}) + \max\{NS(\overrightarrow{R_2}^{j}), NS(\overleftarrow{R_1}^{i+1})\}$

one node needs to examine at most n positions. Note that we do not need to consider the capacity constraints for this operator, since all the nodes have the same (r, d) pair and the final vehicle load will not change.

The above discussions show that implementing all the local search operators is quite efficient. We next turn to discuss the implementation of the shaking process. The process can be decomposed into two following elementary operations, namely, removing a node from some position of route R and inserting a node into some position of route R. Either of the elementary operations affects both $TC(R)$ and $VL(R)$. For each elementary operation, $TC(R)$ can be easily computed in a constant time by its reduced or increased traveling cost. However, $VL(R)$ is related to $NS(R)$, which can only be obtained in linear time by the naive implementation. In order to accelerate the calculation of $VL(R)$, we introduce an efficient data structure called *red–black tree* (Cormen et al., 2001). A red–black tree is a kind of self-balancing binary search tree. We use it to keep track of d_{x_i} when a node x_i is added into or removed from the red–black tree. Note that $NS(R)$ equals the maximum d-value in the tree, and finding the maximum value in the red–black tree requires $O(\log k)$ time, where k is the total number of nodes in the tree. Therefore, if a solution consists of m routes with an average of k nodes in a route, then the time complexity is $O(\log k)$ for a node removal and $O(m \log k + n)$ for a node insertion. To sum up, the time complexity of the shaking process is $O(\delta n(m \log k + n))$.

4.6. Variable neighborhood search

4.6.1. Basic VNS (BVNS)

The BVNS framework is given in Algorithm 1. It begins with an *initialization* phase and comprises an outer loop and an inner loop. The outer loop iterates from 1 to t_{max}, where t_{max} is a parameter for termination. The inner loop involves all the shaking neighborhood structures (N_k, for $k = 1, 2, \ldots, k_{max}$). At each inner iteration, a solution x' is first generated by perturbing the incumbent solution x using the k-th neighbor $N_k(x)$ (this corresponds to the *shaking process* in line 6). Note that different shaking neighbors control different removal probability δ (see Section 4.4). It is suggested by Lourenço et al. (2010) that several routing problems require large perturbation strength. In our implementation, δ increases in proportion to the number k. When k reaches to k_{max}, δ is equal to 1. In line 7, the *local search* procedure is applied to the solution x', yielding an improved solution x''. Lines 8–13 are known as the *neighborhood change* component. If x'' is better than the incumbent solution x, then x is updated and the search goes back to explore the first neighborhood N_1. Otherwise, the search continues with the neighborhood N_{k+1}. The final local optimal solution x is returned when the termination criterion is reached.

The local search procedure is one of the most important components in the VNS algorithm. In the BVNS, several descent heuristics are applied in the local search. Steepest descent (or called best improvement) and first descent (or called first improvement) heuristics are two commonly used methods. The steepest descent heuristic finds the local minimum with respect to all the current neighborhood $N(x)$, so the neighborhood of x is completely explored. The first descent heuristic returns the first neighbor solution as soon as a descent is encountered. Based on different descent strategies, we name the BVNS with the *steepest descent* as BVNS-S, and name the BVNS with the *first descent* as BVNS-F.

To sum up, the BVNS consists of a shaking phase, a local search phase and a neighborhood change phase. For more details of the BVNS, we refer the reader to Hansen and Mladenović (2001) and Hansen and Mladenović (2014).

Algorithm 1. The basic variable neighborhood search algorithm: *BVNS* (x, k_{max}, t_{max}).

```
1: Generate an initial solution x;
2: t ← 1;
3: while t ⩽ t_max do
4:    k ← 1;
5:    while k ⩽ k_max do
6:       x′ ← Shake(x, k);
7:       x″ ← Local_Search(x′); // steepest descent or first descent heuristic is applied
8:       if f(x″) < f(x) then
9:          x ← x″;
10:         k ← 1;
11:      else
12:         k ← k + 1;
13:      end if
14:   end while
15:   t ← t + 1;
16: end while
17: return x;
```

4.6.2. Mixed VNS (MVNS)

In the BVNS, different local search strategies may have different effects. By preliminary experiments, we find that the BVNS with the steepest descent can attain better solutions, however, it is time-consuming in general. The BVNS with the first descent needs less computation time, while the solution quality is relatively lower. Considering both the computation time and the solution quality, we propose a mixed VNS algorithm for excellent solutions in short computational times.

Algorithm 2 presents the MVNS approach in which both steepest descent and first descent strategies are applied. We set a parameter $Iter_{max}$, the maximum number of iterations, in the local search procedure. In lines 9–18, a series of neighbors of x' are explored and compared with the incumbent solution x_{local}. If there exists an improvement neighbor and the current iteration ($iter$) is greater than $Iter_{max}$, the local search is broken out. Thus, if $Iter_{max}$ is set to a small value, less computation time is needed and the solution quality may be poor. On the contrary, if $Iter_{max}$ is set to a large value, more computation time needs to be consumed and a better solution is more likely to be obtained. Specially, when $Iter_{max} = 0$ and $Iter_{max} \to \infty$, the algorithm is equivalent to BVNS-F and BVNS-S, respectively.

Algorithm 2. The mixed variable neighborhood search algorithm: *MVNS* $(x, k_{max}, t_{max}, Iter_{max})$.

```
1: Generate an initial solution x;
2: t ← 1;
3: while t ⩽ t_max do
4:    k ← 1;
5:    while k ⩽ k_max do
6:       x′ ← Shake(x, k);
7:       x_local ← x;
8:       iter ← 1;
9:       while iter ⩽ |N(x′)|
10:         Find a solution x″ ∈ N(x′);
11:         if f(x″) < f(x_local) then
12:            x_local ← x″;
13:            if iter > Iter_max then
14:               break;
15:            end if
16:         end if
17:         iter ← iter + 1;
18:      end while
19:      if f(x_local) < f(x) then
20:         x ← x_local;
21:         k ← 1;
22:      else
23:         k ← k + 1;
```

```
    end if
  end while
  t ← t + 1;
end while
return x;
```

5. Computational experiments

In this section, we present the experimental results of BVNS (including BVNS-S and BVNS-F) and MVNS algorithms on both CVRP and MAVRP instances. Obviously, the CVRP is a special case of our MAVRP. The MAVRP is a new problem and no existing algorithm can be found in literate to compare with our algorithms. Therefore, we use the CVRP instances to evaluate the performance of our algorithms. All experiments were conducted on a personal computer with an Intel i7-4790 4.00 GHz CPU, 8 GB RAM, and Windows 10 operating system. Each instance was executed 10 times with different random seeds.

5.1. Test instances

The test instances used in our experiments contain a class of CVRP instances and a class of MAVRP instances. The CVRP instances can be found at http://neo.lcc.uma.es/vrp. We used 20 instances in *Augerat* part and 7 instances in *Christofides* and *Eilon* parts. This is because the optimal solution is known for each of these 27 instances. In order to adapt our algorithms to solve the CVRP instances, we set the seat reservation r to the customer demand of the corresponding CVRP instance and fix the staff demand d to 0.

The MAVRP instances are generated by modifying the CVRP instances as follows. First, the locations of the hospital and patients are given by the coordinates of the depot and customers in the CVRP instance, respectively. Second, each seat reservation r is randomly selected from $\{1, 2, 3\}$. Third, the staff demand d is also randomly drawn from $\{1, 2, 3\}$. The rationale of such setting is given in the following. If a patient can go to the hospital independently, only one driver is required. If a patient uses the wheelchair, an additional assistant is assigned to manage the wheelchair. If a patient needs the stretcher, two assistants must be deployed to carry the stretcher. Finally, the capacity of the vehicle is set to $C = 12$. The unit manpower cost q is set to 0 for simplicity. The penalty cost p_i of each unfulfilled patient request i is set to a large number and is proportional to the product of seat reservation and staff demand, e.g., $p_i = M \cdot r_i \cdot d_i$, where M is a very large positive number. Here, we use $r_i \cdot d_i$ to represent the *outsourcing intensity* of patient request i. Such setting implies that the patient request with larger seat reservation and staff demand is more likely to be carried out by in-house service than outsourcing service.

5.2. Parameter settings

Our VNS algorithms do not require many user-defined parameters. After some preliminary experiments, the maximum iteration t_{max} was set to 1000 and the neighborhood size k_{max} was set to 30. Such number ensures that every single execution can terminate within 10 min in our experiments.

In order to fix other parameters, we selected those instances with the sizes greater than 70 for parameter tuning. The removal probability ρ used in the *Reassignment* operator (N7) was tuned first. We used the BVNS-F framework and the random shaking strategy. The results are reported in Table 2. $\rho = 0$ means that the operator N7 was excluded. The corresponding solution was recorded and served as a baseline. When ρ was set to 0.1, 0.2 and 0.3, the solution quality improvement was 0.40%, 0.52% and 0.41%, respectively. The running time was not affected significantly for different ρ, so we do not report it in the table. Finally, we fixed ρ to 0.2.

The MVNS algorithm also requires a maximum number of iterations ($Iter_{max}$) in the local search procedure. $Iter_{max}$ is related to $|N(x')|$, which is the neighbor size of x' in the local search. We introduce a parameter λ here and let $Iter_{max} = |N(x')|^{\lambda}$. We then set λ to different values and observed the results shown in Table 3. When $\lambda = 0$ and $\lambda = 1$, the algo-

Table 2
Parameter tuning for ρ.

$\rho = 0$	$\rho = 0.1$	$\rho = 0.2$	$\rho = 0.3$
0%	0.40%	0.52%	0.41%

Table 3
Parameter tuning for λ.

$\lambda = 0$		$\lambda = 1/2$		$\lambda = 2/3$		$\lambda = 3/4$		$\lambda = 1$	
Quality impr.	Time incr.	Quality impr.	Time incr.	Quality impr.	Time incr.	Quality impr.	Time incr.	Quality impr.	Time incr.
0%	0%	0.27%	30.50%	0.76%	40.90%	0.98%	67.60%	1.19%	440%

rithm corresponds to BVNS-F and BVNS-S, respectively. Suppose that BVNS-F is the benchmark algorithm, the quality improvements and time increments for different λ values are presented in the table. We finally chose $\lambda = 3/4$ so that the corresponding algorithm can achieve a nice balance between solution quality and running time.

5.3. Results of BVNS and MVNS on the CVRP instances

Table 4 shows the results of BVNS-F, BVNS-S and MVNS algorithms on the CVRP instances. Random removal, worst removal and related removal strategies for the shaking process are also studied. Column "Instance" gives the name of each instance. The first character of the instance name indicates the contributor. The middle term denotes the number of nodes and the last term corresponds to the number of vehicles. Column "Opt." shows the optimal solution found by the existing approaches. Columns "Best" and "Avg." present the best and the average solution values found by 10 executions with respect to the corresponding approach. Column "Time" provides the average running time (in seconds).

From this table, we can see that all the five VNS algorithms can solve 27 CVRP instances to optimality. Under the random removal strategy, BVNS-S produced slightly better average solution values than BVNS-F and three MVNS algorithms did. However, the average running time of BVNS-S is more than three times greater than those of BVNS-F and MVNS algorithms. MVNS with worst removal and related removal perform slightly better than MVNS with random removal. Overall speaking, the performances of BVNS-F and three MVNS algorithms on these CVRP instances are comparable in terms of the solution quality and the running time. Clearly, this table demonstrates the effectiveness of VNS algorithms in solving the CVRP instances.

5.4. Results of BVNS and MVNS on the MAVRP instances

The MAVRP instances do not include the number of vehicles. In our experiments, we consider two scenarios. The first scenario is that the number of vehicles is sufficient and thus no patient request is outsourced. This can be done by setting the number of vehicles to a sufficiently large positive integer, e.g., $\lceil \sum_{i \in N}(r_i + d_i)/C \rceil$. By executing our VNS algorithms, we can obtain the actual number of vehicles used for each instance. After this number is decreased, it is very likely that some patient requests has to be outsourced. Therefore, in the second scenario, we decrease the actual number of vehicles used by one to observe the outsourcing situation of the MAVRP.

Table 5 presents the results of the MAVRP instances under the first scenario. In this table, 27 CVRP instances were modified to create the MAVRP instances, each with a prefix "MS-" in the corresponding instance name. The column "NV" shows the actual number of vehicles used in the best found solution. Note that this number is different from the number of vehicles given in the original CVRP instance. For each instance and each algorithm, the best found value is reported. It is further marked in bold if it is the smallest one among five algorithms. From this table, we can discover that all these five algorithms obtained the solutions with the same number of vehicles. On average, the MVNS with related removal algorithm produced the most promising solutions compared with the other algorithms. The running times of MVNS algorithms are slightly greater than BVNS-F and far less than BVNS-S.

For the second scenario, the objective is constituted by the outsourcing cost and the total traveling cost. By our settings, the outsourcing cost is the primary objective determined by the outsourcing intensity of the unfulfilled patient requests, while the traveling cost is the secondary objective. Table 6 shows the results. In the columns "Best" and "Avg.", a number pair instead of a single number in a row is reported, where the former number indicates the outsourcing intensity and the latter number denotes the traveling cost. As can be seen from this table, BVNS-S outperforms BVNS-F and three MVNS algorithms in terms of the average objective value. BVNS-F consumed the smallest running time. Three MVNS algorithms executed slightly slower than BVNS-F but much faster than BVNS-S. The MVNS with related removal algorithm outperforms the MVNS with random removal and worst removal algorithms. To sum up, we can conclude from Tables 5 and 6 that the MVNS with related removal algorithm achieved a good balance between solution quality and the computation time.

We further conducted experiments on the instances "MS-A-n80-k10" and "MS-E-n101-k14" to analyze how different numbers of vehicles affect the outsourcing behavior and the traveling cost. For the instance "MS-A-n80-k10", 14 vehicles are adequate for accomplishing all patient requests. We set the number of vehicles from 1 to 14 and executed our proposed MVNS algorithm. The corresponding outsourcing intensity and traveling cost are recorded and plotted in Fig. 4(a). When the number of vehicles is 14, the outsourcing intensity and the routing cost is 0 and 2492, respectively. This corresponds to the MVNS best result shown in Table 5. Similarly, when the number of vehicles decreases to 13, the number pair of the outsourcing intensity and the traveling cost is (9,2324). We can find the corresponding pair in Table 6. When the number of vehicles is 1, the result is (200,152). Fig. 4(b) illustrates another case. Both figures demonstrate that the objective outsourcing intensity is in conflict with the objective traveling cost. Reducing the in-house vehicles definitely increases the outsourcing intensity.

5.5. Results of Cplex on the MAVRP instances

We used the Ilog Cplex 12, a commercial MILP solver, to deal with the MAVRP instances. The time limit was set to 7200 s for each instance. After some runs, we found that Cplex is hardly to solve any MAVRP instance to optimality. Therefore, we decide to only report the results of the instances with $n \leqslant 35$.

Table 4
The results of the BVNS-F, BVNS-S and different MVNS algorithms on the CVRP instances.

Instance	Opt	BVNS-F (Random removal)			BVNS-S (Random removal)			MVNS (Random removal)			MVNS (Worst removal)			MVNS (Related removal)		
		Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time
A-n32-k5	784	784	784	6.7	784	784	16.1	784	784	8.4	784	784	10.9	784	784	11.3
A-n33-k5	661	661	661	6.3	661	661	15.7	661	661	8.1	661	661	9.9	661	661	11.3
A-n33-k6	742	742	742	6.4	742	742	15.4	742	742	7.9	742	742	8.9	742	742	10.9
A-n34-k5	778	778	778	6.4	778	778	16.2	778	778	8.3	778	778	9.8	778	778	11.3
A-n36-k5	799	799	799	9.4	799	799	25.3	799	799	11.7	799	799	14.2	799	799	15.6
A-n37-k5	669	669	669	9.3	669	669	22.9	669	669	11.1	669	669	15.2	669	669	15.2
A-n38-k5	730	730	730	9.8	730	730	27.0	730	730	11.8	730	730	15.1	730	730	16.1
A-n39-k5	822	822	822	11.6	822	822	29.9	822	822	13.7	822	822	18.0	822	822	17.7
A-n39-k6	831	831	831	11.7	831	831	31.4	831	831	14.1	831	831	17.8	831	831	19.4
A-n44-k6	937	937	937	15.8	937	937	45.6	937	937	19.3	937	937	24.0	937	937	23.9
A-n45-k7	1146	1146	1146	16.2	1146	1146	51.9	1146	1146	20.2	1146	1146	24.4	1146	1146	26.2
A-n46-k7	914	914	914	16.9	914	914	53.2	914	914	20.8	914	914	26.7	914	914	26.9
A-n48-k7	1073	1073	1073	19.9	1073	1073	63.4	1073	1073	25.0	1073	1073	33.0	1073	1073	33.4
A-n53-k7	1010	1010	1010	24.5	1010	1010	80.2	1010	1010	30.1	1010	1010	35.6	1010	1010	39.2
A-n54-k7	1167	1167	1167	30.3	1167	1167	94.4	1167	1167	35.4	1167	1167	44.7	1167	1167	45.7
A-n55-k9	1073	1073	1073	25.9	1073	1073	88.8	1073	1073	30.3	1073	1073	36.4	1073	1073	41.7
A-n61-k9	1034	1034	1034.8	32.8	1034	1034.7	124.9	1034	1034.9	39.1	1034	1034.7	46.4	1034	1034.8	54.0
A-n65-k9	1174	1174	1176.4	39.7	1174	1176.4	146.5	1174	1176.4	46.4	1174	1177	55.2	1174	1176.7	63.8
A-n69-k9	1159	1159	1159.8	52.3	1159	1161.4	195.2	1159	1160.6	61.4	1159	1160.2	77.3	1159	1159.8	83.1
A-n80-k10	1763	1763	1763.2	96.9	1763	1763	392.9	1763	1763	110.6	1763	1763.2	133.0	1763	1763	132.9
E-n22-k4	375	375	375	2.0	375	375	3.9	375	375	2.6	375	375	3.4	375	375	3.6
E-n23-k3	569	569	569	2.4	569	569	3.5	569	569	3.0	569	569	3.9	569	569	4.5
E-n30-k3	534	534	534	5.5	534	534	10.8	534	534	6.6	534	534	8.6	534	534	9.1
E-n33-k4	835	835	835	7.3	835	835	17.4	835	835	9.0	835	835	10.9	835	835	12.4
E-n51-k5	521	521	521	22.8	521	521	64.9	521	521	26.7	521	521	27.8	521	521	37.9
E-n76-k7	682	682	682	90.1	682	682	321.7	682	682	102.6	682	682	139.3	682	682	142.4
E-n101-k14	1067	1067	1076.1	129.1	1067	1074	589.0	1067	1075.4	145.8	1067	1074	184.3	1067	1074	198.0
Average	883.3	883.30	883.79	26.2	883.30	883.76	94.4	883.30	883.79	30.7	883.30	883.74	37.9	883.30	883.71	41.0

Table 5
Computational results of the MAVRP instances under the first scenario.

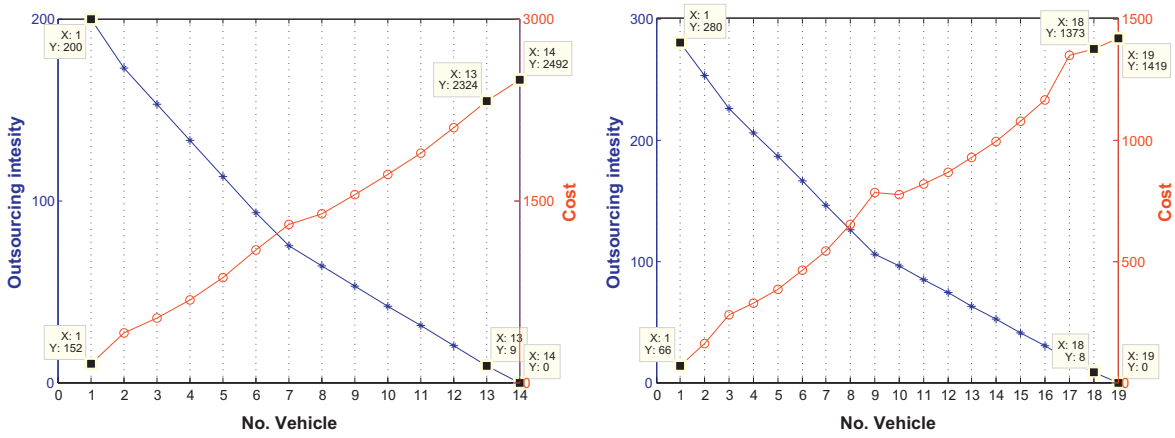
Instance	NV	BVNS-F (Random removal)			BVNS-S (Random removal)			MVNS (Random removal)			MVNS (Worst removal)			MVNS (Related removal)		
		Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time
MS-A-n32-k5	6	**937**	937	6.3	**937**	937	16.0	**937**	937	8.4	**937**	937	11.2	**937**	937	10.4
MS-A-n33-k5	7	**825**	825	6.5	**825**	825	17.0	**825**	825	8.6	**825**	825	9.9	**825**	825	11.0
MS-A-n33-k6	7	**909**	909	6.0	**909**	909	16.8	**909**	909	7.8	**909**	909	9.7	**909**	909	10.0
MS-A-n34-k5	7	**954**	954	6.7	**954**	954	19.3	**954**	954	9.6	**954**	954	10.2	**954**	954	11.8
MS-A-n36-k5	7	**1026**	1026	7.9	**1026**	1026	22.6	**1026**	1026	10.7	**1026**	1026	12.2	**1026**	1026	13.5
MS-A-n37-k5	6	**820**	820	8.5	**820**	820	23.3	**820**	820	10.7	**820**	820	13.0	**820**	820	13.2
MS-A-n38-k5	7	**879**	879	8.4	**879**	879	25.9	**879**	879	11.7	**879**	879	14.8	**879**	879	14.6
MS-A-n39-k5	7	**1045**	1045	10.1	**1045**	1045	30.8	**1045**	1045	12.5	**1045**	1045	15.7	**1045**	1045	15.7
MS-A-n39-k6	7	**1039**	1039.2	9.6	**1039**	1039	29.9	**1039**	1039	12.0	**1039**	1039	14.4	**1039**	1039	15.9
MS-A-n44-k6	8	**1151**	1151	13.8	**1151**	1151	47.2	**1151**	1151	18.2	**1151**	1151	21.7	**1151**	1151	21.9
MS-A-n45-k7	9	**1527**	1529	13.7	**1527**	1529	49.1	**1527**	1530.2	17.3	1531	1531	20.4	**1527**	1529.8	22.9
MS-A-n46-k7	9	**1234**	1236	14.0	**1234**	1234	51.1	**1234**	1234	18.8	**1234**	1234	22.0	**1234**	1234	24.5
MS-A-n48-k7	9	**1444**	1444	15.8	**1444**	1444	58.8	**1444**	1444	19.4	**1444**	1447.1	26.6	**1444**	1444	26.9
MS-A-n53-k7	10	**1379**	1381.8	20.9	**1379**	1380.5	78.8	**1379**	1380.7	26.0	**1379**	1380.7	34.9	**1379**	1381.9	33.6
MS-A-n54-k7	10	**1473**	1473	23.4	**1473**	1473	85.0	**1473**	1473	28.3	**1473**	1473	37.0	**1473**	1473	37.5
MS-A-n55-k9	12	**1316**	1316	19.8	**1316**	1316	76.0	**1316**	1316	25.5	**1316**	1316	29.4	**1316**	1316	35.8
MS-A-n61-k9	12	**1243**	1243	26.9	**1243**	1243	109.2	**1243**	1243	33.3	**1243**	1243	42.5	**1243**	1243	47.7
MS-A-n65-k9	13	**1515**	1515	31.3	**1515**	1515	131.7	**1515**	1515	39.8	**1515**	1515	47.3	**1515**	1515	55.4
MS-A-n69-k9	12	**1502**	1506.4	42.0	**1502**	1505.3	187.6	**1502**	1502.7	49.5	1503	1507.4	62.2	**1502**	1508.6	66.3
MS-A-n80-k10	14	2503	2520	67.7	2493	2523.6	335.8	**2492**	2513.2	75.9	2494	2515.9	91.5	**2492**	2504.6	99.6
MS-E-n22-k4	5	**437**	437	2.1	**437**	437	4.0	**437**	437	3.0	**437**	437	3.7	**437**	437	4.0
MS-E-n23-k3	4	**712**	712	2.7	**712**	712	5.1	**712**	712	3.5	**712**	712	4.0	**712**	712	4.4
MS-E-n30-k3	7	**804**	804	4.4	**804**	804	10.3	**804**	804	5.9	**804**	804	7.1	**804**	804	7.8
MS-E-n33-k4	7	**1243**	1243	6.3	**1243**	1243	16.5	**1243**	1243	8.2	**1243**	1243	9.3	**1243**	1243	10.7
MS-E-n51-k5	10	**842**	850.2	17.3	**842**	842.9	68.1	**842**	843.2	22.9	**842**	867.7	28.8	**842**	842.3	28.5
MS-E-n76-k7	14	**1000**	1002.4	48.1	**1000**	1002.3	213.6	**1000**	1002.6	60.1	**1000**	1002.4	72.3	**1000**	1001.9	75.5
MS-E-n101-k14	19	1427	1446.8	105.1	**1420**	1444.4	568.4	**1420**	1438.3	125.1	1428	1456.9	156.1	**1420**	1436.4	157.4
Average	9.1	1155.0	1157.21	20.2	1154.4	1156.81	85.1	**1154.4**	1156.2	24.9	1154.9	1158.2	30.2	**1154.4**	1156.0	32.5

The best solution values are marked in bold.

Table 6
Computational results of the MAVRP instances under the second scenario.

Instance	NV	BVNS-F (Random removal)			BVNS-S (Random removal)			MVNS (Random removal)			MVNS (Worst removal)			MVNS (Related removal)		
		Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time
MS-A-n32-k5	5	**(6, 965)**	(6, 965)	6.6	**(6, 965)**	(6, 965)	14.6	**(6,965)**	(6,965)	7.8	**(6,965)**	(6,965.2)	8.7	**(6,965)**	(6,965)	9.3
MS-A-n33-k5	6	**(1, 917)**	(1.8, 836.2)	7.7	**(1, 917)**	(1, 917)	17.9	**(1,917)**	(1,917)	9.5	**(1,917)**	(1,917)	11.2	**(1,917)**	(1,917)	12.0
MS-A-n33-k6	6	**(10, 765)**	(10, 767.5)	6.0	**(10, 765)**	(10, 765)	13.2	**(10,765)**	(10,765)	7.7	**(10,765)**	(10,765)	8.9	**(10,765)**	(10,765)	9.8
MS-A-n34-k5	6	**(4, 1004)**	(4, 1007.2)	6.8	**(4, 1004)**	(4, 1004)	18.7	**(4,1004)**	(4,1004.4)	9.8	**(4,1004)**	(4,1006.5)	11.1	**(4,1004)**	(4,1004.6)	12.4
MS-A-n36-k5	6	**(4, 963)**	(4, 963)	8.4	**(4, 963)**	(4, 963)	20.4	**(4,963)**	(4,963)	10.7	**(4,963)**	(4,963)	12.3	**(4,963)**	(4,963)	14.4
MS-A-n37-k5	5	**(9, 703)**	(9, 706.5)	8.8	**(9, 703)**	(9, 703)	21.4	**(9,703)**	(9,704.5)	10.7	**(9,703)**	(9,704.7)	13.0	**(9,703)**	(9,704.8)	12.3
MS-A-n38-k5	6	**(6, 999)**	(6, 999)	8.6	**(6, 999)**	(6, 999)	24.0	**(6,999)**	(6,1000.4)	12.1	**(6,999)**	(6,1007.2)	14.2	**(6,999)**	(6,999)	15.1
MS-A-n39-k5	6	**(10, 860)**	(10, 861.1)	9.9	**(10, 860)**	(10, 861.4)	24.3	**(10,860)**	(10,860.8)	11.7	**(10,860)**	(10,862.8)	14.2	**(10,860)**	(10,861.6)	13.3
MS-A-n39-k6	6	**(9, 953)**	(9, 961.1)	10.1	**(9, 953)**	(9, 953)	26.0	**(9,953)**	(9,958.4)	12.2	**(9,953)**	(9,957.7)	14.9	**(9,953)**	(9,953)	14.9
MS-A-n44-k6	7	**(5, 1123)**	(5, 1125.7)	14.5	**(5, 1123)**	(5, 1123.4)	42.3	**(5,1123)**	(5,1125.4)	18.0	(5,1125)	(5,1130.2)	21.5	**(5,1123)**	(5,1124.1)	22.4
MS-A-n45-k7	8	(10, 1354)	(10, 1359.4)	13.9	**(9, 1480)**	(9, 1482.7)	44.1	**(9,1480)**	(9,1482.8)	18.2	(9,1483)	(9,1488.6)	19.8	**(9,1480)**	(9,1482.3)	22.0
MS-A-n46-k7	8	(9, 1126)	(9, 1143.5)	13.8	**(9, 1119)**	(9, 1126.2)	47.0	(9,1126)	(9,1136.5)	19.8	**(9,1119)**	(9,1129.2)	22.2	**(9,1119)**	(9,1129.5)	24.1
MS-A-n48-k7	8	**(10, 1271)**	(10, 1271)	16.8	**(10, 1271)**	(10, 1271)	48.0	**(10,1271)**	(10,1271)	20.4	**(10,1271)**	(10,1271)	22.4	**(10,1271)**	(10,1271)	24.6
MS-A-n53-k7	9	**(7, 1382)**	(7.4, 1363.3)	21.2	**(7, 1382)**	(7, 1386.3)	78.5	**(7,1382)**	(7,1386.1)	28.8	(7,1413)	(7,1428.2)	32.8	**(7,1382)**	(7,1382)	34.2
MS-A-n54-k7	9	**(2, 1484)**	(2, 1485.6)	24.4	**(2, 1484)**	(2, 1485.6)	89.7	**(2,1484)**	(2,1485.1)	33.0	(2,1487)	(2,1492.9)	37.6	**(2,1484)**	(2,1484.2)	41.4
MS-A-n55-k9	11	**(5, 1330)**	(5, 1342.5)	22.1	**(5, 1330)**	(5, 1334.7)	84.3	**(5,1330)**	(5,1343.9)	29.9	**(5,1330)**	(5,1345.6)	33.2	**(5,1330)**	(5,1340.1)	40.6
MS-A-n61-k9	11	**(5, 1320)**	(5.1, 1328.9)	27.7	**(5, 1320)**	(5, 1326.5)	117.5	**(5,1320)**	(5,1330.8)	38.5	**(5,1320)**	(5,1336.8)	42.5	**(5,1320)**	(5,1327.1)	52.2
MS-A-n65-k9	12	(3, 1596)	(3.1, 1599.4)	33.3	**(2, 1665)**	(2, 1665)	142.2	**(2,1665)**	(2.6,1628.5)	47.2	**(2,1665)**	(2.2,1652.2)	53.8	**(2,1665)**	(2,1665)	64.4
MS-A-n69-k9	11	(10, 1336)	(10, 1358.8)	43.5	**(9, 1398)**	(9, 1408.1)	188.5	**(9,1398)**	(9,1419.1)	57.2	(9,1411)	(9,1442.6)	64.5	**(9,1398)**	(9,1401)	69.3
MS-A-n80-k10	13	(9, 2352)	(10, 2286.8)	67.6	(9, 2333)	(9, 2350.5)	334.4	**(9,2324)**	(9,2351.1)	88.6	(9,2380)	(9,2418)	95.0	**(9,2324)**	(9,2335.3)	110.8
MS-E-n22-k4	4	**(7, 346)**	(7, 346)	2.9	**(7, 346)**	(7, 346)	3.5	**(7,346)**	(7,346)	2.7	**(7,346)**	(7,346)	3.3	**(7,346)**	(7,346)	3.8
MS-E-n23-k3	3	**(10, 446)**	(10, 446)	2.8	**(10, 446)**	(10, 446)	3.6	**(10,446)**	(10,446)	2.7	**(10,446)**	(10,446)	3.1	**(10,446)**	(10,446)	3.5
MS-E-n30-k3	6	**(2, 798)**	(2, 798.4)	4.6	**(2, 798)**	(2, 798)	12.5	**(2,798)**	(2,798)	7.2	**(2,798)**	(2,798)	8.6	**(2,798)**	(2,798)	9.8
MS-E-n33-k4	6	**(2, 1183)**	(2, 1183.4)	6.1	**(2, 1183)**	(2, 1183)	17.4	**(2,1183)**	(2,1183)	9.3	**(2,1183)**	(2,1183)	10.7	**(2,1183)**	(2,1183)	12.3
MS-E-n51-k5	9	**(10, 786)**	(10, 795.7)	18.4	**(10, 786)**	(10, 788.6)	54.4	(10,787)	(10,797)	22.4	(10,787)	(10,800.2)	25.3	**(10,786)**	(10,787.2)	28.8
MS-E-n76-k7	13	(4, 990)	(4, 1015.9)	49.9	**(3, 1084)**	(3, 1086.6)	252.3	(3,1089)	(3.9,1023.9)	71.6	(3,1112)	(3.8,1049.1)	80.7	**(3,1084)**	(3.8,1019.8)	94.2
MS-E-n101-k14	18	(8, 1387)	(8.8, 1379.2)	113.3	**(7, 1457)**	(7.9, 1388.2)	571.3	(8,1381)	(8,1389.8)	141.8	(8,1404)	(8,1418.5)	153.3	(7,1473)	(7.9,1386.7)	186.7
Average	8.1	(6.56, 1101.4)	(6.67, 1099.9)	21.1	**(6.37, 1116.1)**	(6.40, 1115.8)	85.6	(6.41,1113.4)	(6.46,1114.2)	27.8	(6.41,1118.9)	(6.44,1123.2)	31.1	(6.37,1116.3)	(6.43,1112.6)	35.5

The best solution values are marked in bold.



(a) MS-A-n80-k10 (b) MS-E-n101-k14

Fig. 4. The impact of different numbers of vehicles on the outsourcing intensity and the traveling cost.

Table 7
The comparison results of Cplex and VNS on the MAVRP instances with $n \leqslant 35$.

Instance	First scenario					Second scenario					
	NV	UB	LB	VNS	Gap	NV	UB	LB	Cplex	VNS	Gap
MS-A-n32-k5	6	952	620.58	937	1.58%	5	60989	50570.80	(6,989)	(6,965)	2.43%
MS-A-n33-k5	7	828	474.61	825	0.36%	6	10917	10638.79	(1,917)	(1,917)	0.00%
MS-A-n33-k6	7	934	565.88	909	2.68%	6	100817	90575.83	(10,817)	(10,765)	6.36%
MS-A-n34-k5	7	964	583.36	954	1.04%	6	41015	20711.07	(4,1015)	(4,1004)	1.08%
MS-E-n22-k4	5	437	366.91	437	0.00%	4	70346	70346.00	(7,346)	(7,346)	0.00%
MS-E-n23-k3	4	712	610.71	712	0.00%	3	100446	100446.00	(10,446)	(10,446)	0.00%
MS-E-n30-k3	7	823	427.52	804	2.31%	6	20800	20431.07	(2,800)	(2,798)	0.25%
MS-E-n33-k4	7	1318	833.81	1243	5.69%	6	21203	851.13	(2,1203)	(2,1183)	1.66%
Average	–	–	–	–	1.71%	–	–	–	–	–	1.47%

As shown in Table 7, the results of 8 MAVRP instances under two scenarios are presented. For the first scenario, the vehicles are adequate for fulfilling all patient requests. The columns "UB" and "LB" give the upper bound and lower bound of traveling costs found by Cplex. The column "VNS" presents the best solution obtained by the VNS algorithms. The values in the column "Gap" was calculated by $(UB - VNS)/UB$. We can see that Cplex can only solve the instances "MS-E-n22-k4" and "MS-E-n23-k3" to optimality, but their gaps to "LB" are still not closed.

Considering the second scenario, vehicles are inadequate and some patient requests are outsourced. By our setting, the penalty cost p_i of each unfulfilled patient request i is set to $M \cdot r_i \cdot d_i$ ($M = 10,000$ in our experiments). So once we obtained a result z, the outsourcing intensity is $\lfloor z/M \rfloor$ and the traveling cost is $z - \lfloor z/M \rfloor$. The pairs in the column "Cplex" are $(\lfloor UB/M \rfloor, UB - \lfloor UB/M \rfloor)$, which are feasible solutions obtained by Cplex. We can observe that the outsourcing intensity obtained by Cplex is equal to that obtained by VNS for each instance. The column "Gap" reports the relative difference between the traveling costs of the best solution found by Cplex and VNS. This column tells us that the results obtained by Cplex are worse than VNS.

6. Conclusions

In this paper, we study a manpower allocation and vehicle routing problem (MAVRP), which simultaneously considers the allocation of manpower and the routing of vehicles. We formulate this problem into a mathematical programming model, and then devise variable neighborhood search (VNS) algorithms to solve it, where the steepest descent, the first descent and a mixed strategy are used. We propose five groups of operators, i.e., *Exchange*, *Reverse*, *Insertion*, *2-opt* and *Reassignment*, to define the local neighborhood structure of VNS. To test the performance of our proposed algorithms, we use two classes of instances, namely, the CVRP and the MAVRP instances. From the computational results, we find that all the VNS algorithms can achieve optimal solutions for the CVRP instances. For the MAVRP instances, two scenarios (adequate and inadequate numbers of vehicles) are considered. The results demonstrate that the mixed VNS algorithm can achieve a good balance between the solution quality and the running time.

To extend our work on the MAVRP, exact approaches such as branch-and-cut algorithm and branch-and-price algorithm, can be developed to optimally solve the MAVRP instances. Other meta-heuristics can also be designed for the MAVRP

instances with even larger size. In addition, more practical constraints can be taken into consideration to make the problem closer to the real environment. These constraints include, for example, the time windows of services, working time of assistants, multi-period transportation, etc.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (71601191, 71571077, 71390335 and 71671131), Guangdong Natural Science Funds (2016A030313264), Science and Technology Planning Project of Guangdong (2014B010118002) and the fund from Huazhong University of Science and Technology (5001300001).

References

Bräysy, O., 2003. A reactive variable neighborhood search for the vehicle-routing problem with time windows. INFORMS J. Comput. 15 (4), 347–368.
Bräysy, O., Gendreau, M., 2005. Vehicle routing problem with time windows, Part i: Route construction and local search algorithms. Transport. Sci. 39 (1), 104–118.
Brucker, P., Qu, R., Burke, E., 2011. Personnel scheduling: Models and complexity. Eur. J. Oper. Res. 210 (3), 467–473.
Clarke, G., Wright, J.W., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Operat. Res. 12 (4), 568–581.
Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C., 2001. Introduction to algorithms second edition.
Dayarian, I., Crainic, T.G., Gendreau, M., Rei, W., 2016. An adaptive large-neighborhood search heuristic for a multi-period vehicle routing problem. Transport. Res. Part E: Logist. Transport. Rev. 95, 95–123.
De Bruecker, P., Van den Bergh, J., Beliën, J., Demeulemeester, E., 2015. Workforce planning incorporating skills: State of the art. Eur. J. Oper. Res. 243 (1), 1–16.
Ernst, A.T., Jiang, H., Krishnamoorthy, M., Sier, D., 2004. Staff scheduling and rostering: A review of applications, methods and models. Eur. J. Oper. Res. 153 (1), 3–27.
Escobar, J.W., Linfati, R., Baldoquin, M.G., Toth, P., 2014. A granular variable tabu neighborhood search for the capacitated location-routing problem. Transport. Res. Part B: Methodolog. 67, 344–356.
Eveborn, P., Flisberg, P., Rönnqvist, M., 2006. LAPS CARE – an operational system for staff planning of home care. Eur. J. Oper. Res. 171 (3), 962–976.
Günlük, O., Kimbrel, T., Ladanyi, L., Schieber, B., Sorkin, G.B., 2006. Vehicle routing and staffing for sedan service. Transport. Sci. 40 (3), 313–326.
Gutiérrez, E.V., Amaya, C.A., Guéret, C., Péton, O., Velasco, N., 2009. Combined routing and staff scheduling model for home health care. In: ORAHS 2009.
Hansen, P., Mladenović, N., 2001. Variable neighborhood search: Principles and applications. Eur. J. Oper. Res. 130 (3), 449–467.
Hansen, P., Mladenović, N., 2014. Variable Neighborhood Search. Springer.
Hemmelmayr, V.C., Cordeau, J.-F., Crainic, T.G., 2012. An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. Comput. Operat. Res. 39 (12), 3215–3228.
Hertz, A., Mittaz, M., 2001. A variable neighborhood descent algorithm for the undirected capacitated arc routing problem. Transport. Sci. 35 (4), 425–434.
Kim, B.-I., Koo, J., Park, J., 2010. The combined manpower-vehicle routing problem for multi-staged services. Expert Syst. Appl. 37 (12), 8424–8431.
Lai, D.S.W., Caliskan Demirag, O., Leung, J.M.Y., 2016. A tabu search heuristic for the heterogeneous vehicle routing problem on a multigraph. Transport. Res. Part E: Logist. Transport. Rev. 86, 32–52.
Laporte, G., 1992. The vehicle routing problem: An overview of exact and approximate algorithms. Eur. J. Oper. Res. 59 (3), 345–358.
Laporte, G., 2009. Fifty years of vehicle routing. Transport. Sci. 43 (4), 408–416.
Li, Y., Lim, A., Rodrigues, B., 2005. Manpower allocation with time windows and job-teaming constraints. Naval Res. Logist. 52 (4), 302–311.
Lim, A., Rodrigues, B., Song, L., 2004. Manpower allocation with time windows. J. Operat. Res. Soc., 1178–1186
Lim, A., Zhang, Z., Qin, H., 2017. Pickup and delivery service with manpower planning in Hong Kong public hospitals. Transport. Sci. 51 (2), 688–705.
Lin, C., Choy, K.L., Ho, G.T., Chung, S., Lam, H., 2014. Survey of green vehicle routing problem: past and future trends. Expert Syst. Appl. 41 (4), 1118–1138.
Lourenço, H.R., Martin, O.C., Stützle, T., 2010. Iterated local search: Framework and applications. In: Handbook of Metaheuristics. Springer, pp. 363–397.
Mancini, S., 2016. A real-life multi depot multi period vehicle routing problem with a heterogeneous fleet: Formulation and adaptive large neighborhood search based matheuristic. Transport. Res. Part C: Emerging Technol. 70, 100–112.
Miller, C.E., Tucker, A.W., Zemlin, R.A., 1960. Integer programming formulation of traveling salesman problems. J. ACM 7 (4), 326–329.
Mladenović, N., Hansen, P., 1997. Variable neighborhood search. Comput. Operat. Res. 24 (11), 1097–1100.
Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transport. Sci. 40 (4), 455–472.
Shaw, P., 1998. Using constraint programming and local search methods to solve vehicle routing problems. In: International Conference on Principles and Practice of Constraint Programming. Springer, pp. 417–431.
Toth, P., Vigo, D., 2014. Vehicle Routing: Problems, Methods, and Applications, vol. 18. Siam.
Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., De Boeck, L., 2013. Personnel scheduling: A literature review. Eur. J. Oper. Res. 226 (3), 367–385.
Xiao, Y., Konak, A., 2016. The heterogeneous green vehicle routing and scheduling problem with time-varying traffic congestion. Transport. Res. Part E: Logist. Transport. Rev. 88, 146–166.
Yang, P., Miao, L., Xue, Z., Ye, B., 2015. Variable neighborhood search heuristic for storage location assignment and storage/retrieval scheduling under shared storage in multi-shuttle automated storage/retrieval systems. Transport. Res. Part E: Logist. Transport. Rev. 79, 164–177.