The VNS Approach for a Consistent Capacitated Vehicle Routing Problem Under the Shift Length Constraints

Igor Kulachenko[1] and Polina Kononova[1,2(✉)]

[1] Novosibirsk State University, Novosibirsk, Russia
soge.ink@gmail.com
[2] Sobolev Institute of Mathematics SB RAS, Novosibirsk, Russia
pkononova@math.nsc.ru

Abstract. We consider a new real-world application of vehicle routing planning in a finite time horizon. A company has a set of capacitated vehicles in some depots and must serve a set of clients. There is a frequency for each client stating how often this client must be visited. Time intervals between two consecutive visits must be the same but the visiting schedule is flexible. To get some competitive advantage, the company tries to increase its service quality. To this end, each client should be visited by one driver only. The goal is to minimize the total length of vehicles' paths over the planning horizon under the frequency constraints and driver shift length constraints. We present an integer linear programming model for this new consistent capacitated vehicle routing problem. To find near-optimal solutions, we design the Variable Neighborhood Search metaheuristic with eleven neighborhood structures. The driver shift length and capacity constraints are penalized and included into the objective function. Empirical results for real test instances from Orenburg region in Russia with up to 900 clients and four weeks in the planning horizon are discussed.

Keywords: Operations research · Mathematical models · Optimization problems · Time scheduling · Search methods · Routing algorithms · Computer experiments

1 Introduction

The literature on vehicle routing problems has become very rich and covers nowadays a variety of applications, modeling approaches, and solution methods [16]. Due to their huge importance in practice, these problems have attracted attention of many researchers and motivate a large number of collaborations between

The reported study was funded by RFBR and Novosibirsk region according to the research project N 19-47-540005.

© Springer Nature Switzerland AG 2019
I. Bykadorov et al. (Eds.): MOTOR 2019, CCIS 1090, pp. 51–67, 2019.
https://doi.org/10.1007/978-3-030-33394-2_5

companies and academia. In addition, vehicle routing problems lead to challenging formulations that require the development of sophisticated solution strategies and motivates the design of clever heuristics and meta-heuristics.

Earlier [25] we considered the uncapacitated variant of our problem, and now we consider a more general case. We have a capacitated heterogeneous fleet of vehicles and a finite set of clients with their demand. Our goal is to find a set of routes for the vehicles to service all clients with minimal total distance. This optimization problem and its variants have been extensively studied for nearly 60 years (see the early work of [6]). It is the consistent vehicle routing problem (ConVRP) where the companies focus on client satisfaction to get some competitive advantage [23,24]. Over a given time horizon we need to construct a set of routes for the vehicles such that to service all clients. The consistency is modeled as follows:

- a client can be visited by one driver only, and split deliveries are not allowed;
- a client should be visited at about the same time of a specific day selected by the client in advance.

Thus, a company can increase client satisfaction by providing consistent service [12,23]. In this paper, we consider a new ConVRP assuming that each client is served by the same vehicle, and a frequency is given for each client indicating how often this client should be visited. Each client is visited on the same day of the week one, two or four times a month. These consistency requirements were suggested by a Russian logistics company interested in results. Each vehicle has a maximum capacity that limits the number of clients it can visit before returning to the depot. All vehicles start from and return to their depot in the given working interval. Our goal is to find a visiting schedule for each client and a set of routes for each vehicle that jointly service all clients under the frequency constraints and driver shift length constraints. The objective is to minimize the total traveling distance for all vehicles over the planning horizon.

In [5], a similar periodic VRP was studied without consistency requirements. In [12,24], the ConVRP was studied with fixed visiting scheduling for clients and unlimited fleet. In our problem, we consider ConVRP with a flexible schedule and limited fleet. To solve this real-world routing problem, we design the Variable Neighborhood Search heuristic (VNS) [26,27]. We use eleven neighborhood structures for local search including four large neighborhoods of Kernighan–Lin [17,22]. To enlarge the search space, we relax the shift length and capacity constraints and include them into the objective function with non-negative penalties that are modified during the search [5,10]. Intensification and diversification strategies are applied in the VNS framework as well.

The rest of this paper is structured as follows. We first introduce the mathematical model in Sect. 2. Neighborhood structures are presented in Sect. 3. The framework of the VNS heuristic is described in Sect. 4. Computational results for real-world instances are discussed in Sect. 5. The last Sect. 6 concludes the paper.

2 Mathematical Model

Let us consider a complete directed graph $G = (V, A)$ with the set of nodes V and the set of arcs A. The set V is the union of the set of depots M and the set of clients I. Each depot $m \in M$ has a heterogeneous fleet of vehicles. The set K defines the total vehicle park. For each vehicle $k \in K$, we know its depot $m(k)$ and its capacity v_k. For each arc $(i, j) \in A$, we have two parameters: the length of arc d_{ij} and traveling time t_{ij}. We denote the length of a driver's shift by T. Each client $i \in I$ has a given frequency of visits μ_i in the planning horizon D. Time intervals between two consecutive visits of client i should be the same and equal to $\tau_i = \lfloor |D|/\mu_i \rfloor$. A demand q_i for each client i is given. By s_i we denote the service time which is positive for each client and 0 for each depot.

We introduce the following binary decision variables:

$$x_{ijkd} = \begin{cases} 1, \text{ if vehicle } k \text{ on day } d \text{ traverses arc } (i,j), \\ 0, \text{ otherwise,} \end{cases}$$

$$y_{ikd} = \begin{cases} 1, \text{ if vehicle } k \text{ on day } d \text{ visits client } i, \\ 0, \text{ otherwise,} \end{cases}$$

$$w_{id} = \begin{cases} 1, \text{ if client } i \text{ is visited on day } d, \\ 0, \text{ otherwise.} \end{cases}$$

The auxiliary non-negative variables u_{ikd} will be used for subtour elimination.

Now we can present the consistent capacitated vehicle routing problem under the shift length constraints as the mixed integer linear program:

$$\min \sum_{d \in D} \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} d_{ij} x_{ijkd} \tag{1}$$

subject to

$$\sum_{i \in I} q_i y_{ikd} \leq v_k, \quad k \in K, d \in D, \tag{2}$$

$$y_{mkd} = \begin{cases} 1, m = m(k), \\ 0, m \neq m(k), \end{cases} \quad m \in M, k \in K, d \in D, \tag{3}$$

$$\sum_{k \in K} y_{ikd} = w_{id}, \quad i \in I, d \in D, \tag{4}$$

$$\sum_{d \in D} w_{id} = \mu_i, \quad i \in I, \tag{5}$$

$$\sum_{t=0}^{\tau_i - 1} w_{i(d+t)} = 1, \quad i \in I, d \in \{0, \ldots, (\mu_i - 1)\tau_i\}, \tag{6}$$

$$w_{i\alpha} + w_{i\beta} - 2 \leq y_{ik\alpha} - y_{ik\beta}, \quad i \in I, k \in K, \alpha, \beta \in D, \alpha \neq \beta, \tag{7}$$

$$\sum_{i \in V} x_{ijkd} = \sum_{i \in V} x_{jikd} = y_{jkd}, \quad j \in V, k \in K, d \in D, \tag{8}$$

$$u_{ikd} - u_{jkd} + n x_{ijkd} \leq n - 1, \quad i, j \in I, \ k \in K, \ d \in D, \tag{9}$$

$$\sum_{i\in V}\sum_{j\in V} x_{ijkd}(t_{ij}+s_j) \le T, \qquad k\in K, d\in D, \tag{10}$$

$$u_{ikd} \ge 0, \;\; i\in I, \;\; k\in K, \; d\in D, \tag{11}$$

$$w_{id}, x_{ijkd}, y_{ikd} \in \{0,1\}, \;\; i,j\in V, \;\; k\in K, d\in D. \tag{12}$$

The objective function (1) minimizes the total traveling distance for all vehicles and all days of the planning horizon. In constraint (2), the total load of vehicle k should not exceed its capacity. Equalities (3) show the distribution of vehicles by depots. Equations (4) and (5) ensure that each client is visited according to its frequency. Constraints (6) guarantee that time intervals between two consecutive visits of each client are the same. Driver consistency is guaranteed in (7). Constraints (8) make sure that each client has exactly one predecessor and one successor and each vehicle returns to its own depot. Inequalities (9) prevent subtours on the set of clients, $n = |I|$. The completion of the routes within the driver shift is enforced by inequalities (10). The last two constraints define the types of variables.

It is easy to see that variables u_{ikd} can be replaced by new variables u_{id} without loss of generality and dimension of the program can be reduced. Note that the problem (1)–(12) can be infeasible because of the limited fleet of vehicles in each depot and the driver shift constraints. To overcome this, we relax the constraints (2) and (10) and include them into the objective function with penalties $\gamma_{kd} \ge 0, \lambda_{kd} \ge 0, \; k\in K, d\in D$. As a result, we have got a relaxation of the original problem (1)–(12) as follows:

$$L(x,\gamma,\lambda) = \min \sum_{d\in D}\sum_{k\in K}\sum_{i\in V}\sum_{j\in V} d_{ij}x_{ijkd} + \sum_{d\in D}\sum_{k\in K}(\gamma_{kd}\kappa_{kd} + \lambda_{kd}\varepsilon_{kd}) \tag{13}$$

subject to (3)–(9), (11), (12) and additional constraints for new variables $\kappa_{kd}, \varepsilon_{kd} \ge 0$ which indicate the excess capacity in kilograms and the over-hours in minutes for each pair (k,d):

$$\kappa_{kd} \ge \sum_{i\in V} q_i y_{ikd} - v_k, \;\; k\in K, d\in D, \tag{14}$$

$$\varepsilon_{kd} \ge \sum_{i\in V}\sum_{j\in V} x_{ijkd}(t_{ij}+s_j) - T, \;\; k\in K, d\in D. \tag{15}$$

Now the relaxed problem (3)–(9), (11)–(15) is feasible even if there is just one vehicle at any depot, and we can solve it by local search metaheuristics [28]. The penalties $\gamma_{kd}, \lambda_{kd}$ will be modified during the search in order to get a feasible solution.

3 Neighborhoods

In the past four decades, local search has grown from a simple heuristic idea into a mature field of research in combinatorial optimization [1]. Local search is often used to solve NP-complete problems since it provides a reliable approach for obtaining high-quality solutions for realistic-size problems in a reasonable time. For partition and permutation problems, many small and large neighborhoods are introduced and studied from a theoretical and an empirical point of views [2, 3, 11, 13, 14, 16, 21]. Below we present eleven neighborhoods for the problem which is a special case of partition and permutation problems. We already considered all these neighborhoods in [25].

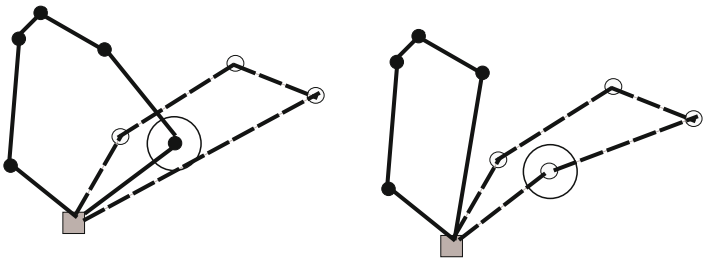

Fig. 1. Moving the client to another route

Let us denote by σ a feasible solution to the problem. For each vehicle $k \in K$ and each day $d \in D$ we have a route (the order of clients). We say that a driver of vehicle k is *happy* on day d if $\kappa_{kd} = \varepsilon_{kd} = 0$ and *unhappy* if these constraints are violated. We want to move clients from unhappy pairs (k, d) to happy ones.

Now we define the following neighborhood structures for solution σ.

The move neighborhood $N_{\text{move}}(\sigma)$ consists of some feasible solutions resulting from σ by *moving* a client to another vehicle or the same vehicle but another day (Fig. 1). If the client must be visited several times, we move all his visits respectively. Moreover, we move an unhappy pair to a happy one only. In order to find the best permutation for new schedules, we select the best positions of new visits in previous schedules. The cardinality of this neighborhood is $O(|I||D||K|)$. It is a large set. Thus, we will use a randomized neighborhood $N^q_{\text{move}}(\sigma)$, $0 < q < 1$, which is a random part of the neighborhood $N_{\text{move}}(\sigma)$. Each element of the set $N_{\text{move}}(\sigma)$ is included in the set $N^q_{\text{move}}(\sigma)$ with probability q independently of other elements.

The neighborhood $\tilde{N}^q_{\text{move}}(\sigma)$ has the same structure but includes the solutions for all moves, except those from happy pairs to unhappy.

The swap neighborhood $N_{\text{swap}}(\sigma)$ consists of some feasible solutions resulting from σ by *swapping* two clients with the same frequency for the same or different vehicles (Fig. 2). We consider only the clients which are close enough to each other, the mutual distance between them is at most R, where R is a parameter

of the neighborhood. The cardinality of the neighborhood is $O(|I|^2)$. Thus, we apply the same randomization trick and use $N^q_{\text{swap}}(\sigma)$ neighborhood instead of the deterministic case.

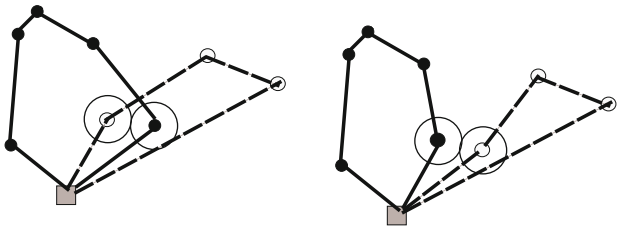

Fig. 2. Swapping the two clients

The neighborhood $\tilde{N}^q_{\text{swap}}(\sigma)$ has the same structure but includes the solutions for swapping clients with different frequency. Thus, we can swap client with 4 visits with two clients with 2 visits or four clients with 1 visits or another client with 4 visits and so on.

Now we are ready to define four large Kernighan–Lin neighborhoods for a feasible solution σ. The main idea of these neighborhood structures is similar to the truncated Tabu Search method by a small neighborhood, say $N(\sigma)$. The neighborhood $KL(\sigma)$ consists of l solutions resulting from σ by the following rule [17,19]:

1. Find the best feasible solution σ' in the neighborhood $N(\sigma)$.
2. Set $\sigma := \sigma'$, even if σ' is worse than σ.
3. Repeat steps 1 and 2 l times, if a move or swap is used at step 1 or 2 of previous iterations, it can not be used anymore.

The sequence of $\sigma_1, \ldots, \sigma_l$ defines l neighbors of the solution σ. We say that σ_b is a local minimum with respect to the KL–neighborhood if σ_b is the best solution of $\sigma_1, \ldots, \sigma_l$.

Using the six basic neighborhoods $N_{\text{move}}(\sigma)$, $N^q_{\text{move}}(\sigma)$, $\tilde{N}^q_{\text{move}}(\sigma)$, $N_{\text{swap}}(\sigma)$, $N^q_{\text{swap}}(\sigma)$, and $\tilde{N}^q_{\text{swap}}(\sigma)$ instead of the neighborhood $N(\sigma)$, we may have four Kernighan–Lin neighborhoods $KL_{\text{move}}(\sigma)$, $\tilde{KL}_{\text{move}}(\sigma)$, $KL_{\text{swap}}(\sigma)$, and $\tilde{KL}_{\text{swap}}(\sigma)$ respectively. We illustrate the idea of the Kernighan–Lin neighborhoods in Fig. 3.

As we have mentioned above, the position of a new client in scheduling is selected without reordering other clients for the same pair (k, d). For improving the final scheduling, we apply local descent algorithm by the well-known *2-opt* neighborhood for each pair (k, d). The idea of this neighborhood is to choose two non-adjacent arcs and replace them by two other arcs for creating a new tour. The main goal is removing intersections of arcs (see Fig. 4). In fact, we divide the problem into $|K||D|$ the traveling salesman subproblems, and a local optimum by the *2-opt* neighborhood is obtained for each subproblem independently [15].

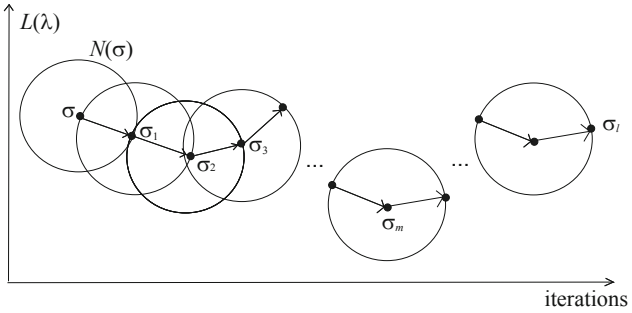

Fig. 3. The Kernighan–Lin neighborhood

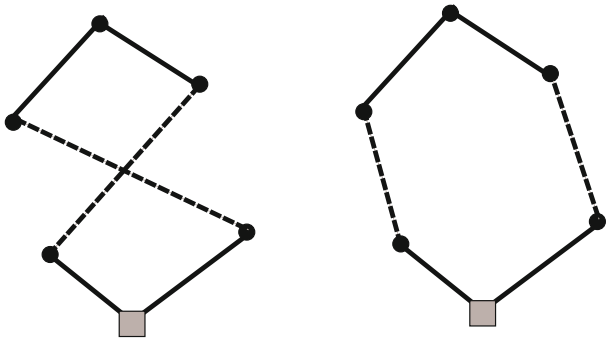

Fig. 4. Neighborhood *2-opt*. Removing the intersection

4 Optimization Method

Variable Neighborhood Search is an efficient framework of local search invented about 20 years ago by Pierre Hansen and Nenad Mladenovich [26]. It is based upon a simple, but a strong principle: a systematic change of a neighborhood within the search. Its development has been rapid and successful in many real-world applications [27], including hard routing problems [18,20] and games [7–9]. The main idea is to focus on local optima and change the landscape of search assuming that the local optimum for one neighborhood may not be the local optimum for another neighborhood. Below we apply this method to the relaxed problem. We used this method for a simpler problem in [25].

To start the method, we need to create an initial solution σ and define the penalties. The VNS method can start from an arbitrary solution, but we use a greedy solution to get a uniform distribution of clients through all pairs (k, d), $k \in K, d \in D$. We start from clients with high frequency and wish to minimize the maximal number of clients per day and per vehicle [4]. We put identical initial values of all penalties, $\lambda_{kd} = 2.5$, $\gamma_{kd} = 3$ in such a way to get approximately the same values of the items in the objective function $L(x, \gamma, \lambda)$.

In each iteration of the local search, we select a neighborhood and move from the current solution to the best neighboring solution. For the Kernighan–Lin neighborhoods, we generate l solutions and select the best one. The pseudo-code of the VNS algorithm is presented below.

Algorithm 1. VNS

Require: initial solution σ, neighborhoods $N_1, \ldots, N_8$, stopping criterion, shaking and intensification rules

1: Define parameters λ, γ, $q_1, \ldots, q_4$, l, R
2: **while** stopping criterion is not reached **do**
3: $k = 1$
4: **while** $k \leq 8$ **do**
5: Apply local search by neighborhood N_k
6: $k = k + 1$
7: Apply local descent by *2-opt* neighborhood for each pair (k, d)
8: Update the penalties
9: Intensification
10: **if** shaking condition is true **then**
11: Shaking
12: Apply local descent by two swap neighborhoods with $q = 1$
13: Apply local descent by *2-opt* neighborhood for each pair (k, d)
14: **return** the best found feasible solution

At the initialization step, we generate an initial solution by a greedy algorithm and define the parameters of the method (line 1). Note that a randomization of the first four neighborhoods may be different and $q_i \neq q_j$, $1 \leq i \neq j \leq 4$. We define these values in such a way that the cardinality of each randomized neighborhood is the same and equal to 200 on average. Thus, we accelerate the search, reduce the running time per iteration, and add a diversification aspect into the search process. The shaking procedure (line 11) is an additional diversification rule. We use some random steps by the swap or move neighborhoods in this procedure if the best found solution does not change for a long time.

The stopping criterion (line 2) is the total number of iterations which depends on the number of clients and their frequency. We use up to $O(n_1^2)$ iterations in our experiments, where $n_1 = \sum_{i \in I} \mu_i$.

Local search (lines 5, 6) is applied by the move and swap neighborhoods and then by the Kernighan–Lin neighborhoods. In the latter case, we use local descents only and terminate the process in a local optimum. For the four basic neighborhoods, we terminate the process after a prescribed number of iterations. Further (line 7), we get a local optimum by *2-opt* neighborhood for each pair (k, d) of vehicle and day. As a rule, we discover a new best solution at this stage. If we find a solution with $\kappa_{kd} > 0$ for some pair (k, d) then we increase the penalties $\gamma_{kd} := 1.05\gamma_{kd}$. If a new solution has $\varepsilon_{kd} = 0$ for all pairs (k, d), then we decrease the penalties λ. Otherwise, we increase them. In general case, we modify the penalties by the following rule (line 8):

$$\gamma_{kd} = \begin{cases} 1.05\gamma_{kd}, & \text{if } \kappa_{kd} > 0, \\ \gamma_{kd}, & \text{if } \kappa_{kd} = 0, \end{cases}$$

$$\lambda_{kd} = \begin{cases} 1.03\lambda_{kd}, & \text{if } \varepsilon_{kd} > 0, \\ 0.97\lambda_{kd}, & \text{if } \varepsilon_{kd} = 0. \end{cases}$$

In the intensification procedure (line 9), we return to the best found solution and increase all randomization parameters $q_1, \ldots, q_4$ and the value of penalties λ_{kd} again to check the most promising area more carefully. If we discover a new best solution, we return to the previous values of these parameters. In the shaking procedure (line 11), we do the same for $q_1, \ldots, q_4$ and λ_{kd} to start the search in a new area of the feasible domain. Finally, we apply deterministic local descent (lines 12, 13) by N_{swap} and $\tilde{N}_{\text{swap}}$ neighborhoods ($q_3 = q_4 = 1$) and get local optima by *2-opt* neighborhood for each pair (k, d).

5 Computational Results

The described VNS algorithm was implemented in `C++` with `MSVC++ 14.16` compiler using standard release options. All experiments were conducted on a computer with an `AMD Ryzen 5 2600` 3.4 GHz processor and 16 GB of RAM running under `Microsoft Windows 10` (64-bit).

The data set used to test the algorithm is proposed by a Russian logistics company with 892 clients from Orenburg region. Among them, one third are clients of frequency 1, and slightly more than half are clients of frequency 2. There are three depots located at a distance of 250 km from each other. We randomly select a part of the large instance to get small ones. For this purpose, we varied certain parameters used during client selection. These parameters include a number necessary for localization of the depots (radius), selection probabilities different for different depots, and a number specifying the random shift of the vehicles relative to their initial position. For the client selection and further in the algorithm, a 32-bit Mersenne Twister pseudo-random number generator was used. As a result, we generated 10 various instances with 600–700 clients. This range of the number of clients allows obtaining diverse large instances with the same number of vehicles in each depot. Besides, this range is close to the actual number of clients served by the company in one region. We assigned two vehicles in each depot for these test instances.

Also, to compare the algorithm with an optimization solver, data set with 672 clients from Orenburg region was used. Slightly less than three-quarters of these clients have a frequency of 1, while the numbers of clients of frequency 2 and 4 are approximately equal. There are the same three depots for this set. Using the same method as for the larger data set, we generated instances with 320–350 and 130–150 clients. We assigned one vehicle in each depot for the former instances and one vehicle in a single depot for the latter ones.

Client attributes include name, GPS coordinates (latitude and longitude), the frequency of visits, service time, and demand. The shift length is 8 h, including 40 min for a break. The time for a break is not fixed in drivers' schedules, and

they can spend it at any free from client service time of the working day. The problem we are investigating does not include time windows. Hence, we can just adjust the shift length to $T = 7\,\text{h}\ 20\,\text{min}$.

Vehicles can leave the depot starting at 8:30, but must arrive at the first client no earlier than 9:00. The last client must be serviced before 17:00, but the vehicle must return to the depot no later than 18:00. To include these additional requirements into the model, we modify the matrix (t_{ij}) by the following rule:

$$t_{ij} = \begin{cases} t_{ij}, \text{ if } i, j \in I, \\ \max\{0, t_{ij} - 30'\}, \text{ if } i \in M, \quad j \in I, \\ \max\{0, t_{ij} - 60'\}, \text{ if } i \in I, \quad j \in M. \end{cases}$$

The planning horizon is 20 days. The speed of each vehicle is 50 km/h. For Kernighan–Lin neighborhoods we generate $l = 25$ neighboring solutions. The threshold R for the swap neighborhoods is defined as 20 km. Local search by the all basic neighborhoods of the VNS algorithm (line 5) is set as 1300 iterations. Results are obtained by running the VNS in 10 min 10 times per instance.

Figure 5 illustrates the typical behavior of the method. The initial value of the objective function $L(x, \gamma, \lambda)$ is huge with excess capacity $\kappa = \sum_{kd} \kappa_{kd}$ and total over-hours $\varepsilon = \sum_{kd} \varepsilon_{kd}$. And after 2000 iterations we found a solution with $\kappa = \sum_{kd} \kappa_{kd} = 0$. Note that the total over-hours and overload decrease as iterations grow. We denoted by $\times$ new record values of the objective function for a feasible solution. We see that the value decreases for a feasible solution too.

In Tables 1, 2, 3 and 4, we show computational results for these 10 small instances. The purpose is to study the impact of the capacity constraint on the solution. Each instance was run 10 times and the minimal, average, and

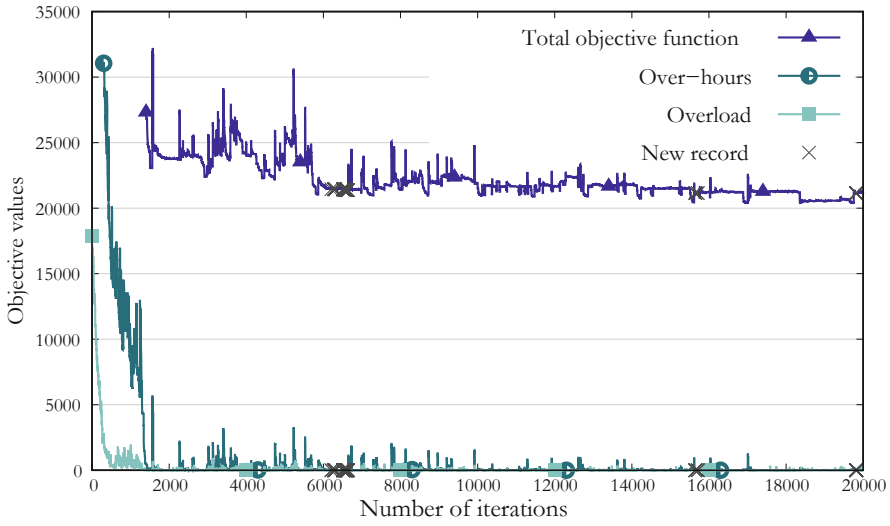

Fig. 5. Easy instance with three vehicles in each depot

maximum values for these runs are presented in the tables. To compare the obtained solutions the objective function (13) with penalties $\gamma_{kd} = 50, \lambda_{kd} = 40$ was used.

Table 1. Capacity $v_k \in [800, 1000]$

Instance	Min		Avg		Max	
	ε	Distance	ε	Distance	ε	Distance
1	0	15360	0	15867.6	0	16195
2	0	16060	0	16298.4	0	16586
3	4	16282	9	16357.6	28	16260
4	0	16961	0	17203	0	17370
5	0	15434	3.5	15803.1	0	16796
6	0	14334	0	14639.4	0	15204
7	0	15521	0	16118.5	0	16704
8	0	16324	0	16695.3	0	16993
9	0	16381	5	16886.3	10	17519
10	4	17158	6.4	17572.5	0	18443

At first (Tables 1, 2 and 3), we used the model which does not allow violation of the capacity constraint at any step of the algorithm (including shakes). Table 1 show results for the case when the vehicles have enough capacity for all clients they may need to serve. By doing this, we were able to get an average daily load for the vehicles from different depots. For most of the instances, this number turned out to be 400–500 kg. From Tables 2 and 3 we see that the results become worse with the tightening of the capacity. Also, Table 3 shows that the search space can become so narrow that for some vehicles there will be no other options but to serve clients intended for another depot (instances 2–4).

Next, the relaxed formulation of the problem with capacity penalties was used (Table 4). This allowed us to find much better solutions in most of the instances. It confirms the need for the penalties.

It should be taken into account that despite the presence of penalties in some solutions, in most of the instances they are small enough to be neglected.

In order to study the efficiency of the algorithm, we compared it with the results obtained by metaheuristic solver LocalSolver. We chose it instead of such classical MILP solvers as CPLEX and Gurobi, since the latter cannot find the exact solution in a reasonable time for even quite small instances of the problem. The results are presented in Tables 5, 6, 7 and 8.

Table 2. Capacity $v_k \in [500, 600]$

Instance	Min		Avg		Max	
	ε	Distance	ε	Distance	ε	Distance
1	0	15467	0	16125.4	0	16686
2	0	17354	17.3	17446.5	20	18271
3	13	16319	29.9	17295.2	75	18631
4	0	17308	0	17546.5	0	17905
5	0	16207	4.8	16489	0	17241
6	0	14599	0	14986.9	0	15324
7	0	15896	0.6	16513.6	0	16900
8	0	16798	6.5	17347.1	48	18106
9	33	16180	45	16362	101	16201
10	1	17667	1.9	18149.1	1	18464

Table 3. Capacity $v_k \in [450, 500]$

Instance	Min		Avg		Max	
	ε	Distance	ε	Distance	ε	Distance
1	0	16083	0	16373.6	0	16700
2	326	17968	1236.7	19365.9	1693	19335
3	1023	18604	1651.5	18985.9	1992	18722
4	389	18886	1400.2	20205.4	2309	20979
5	0	16704	2.1	17420.2	1	18091
6	0	15893	22.8	16584.8	162	16623
7	18	16529	14.5	17306.4	51	18197
8	2	17822	28	17718.9	126	18523
9	104	16073	111.7	16314.3	101	17389
10	36	19520	87.8	19101.7	111	19416

Since LocalSolver did not allow us to obtain satisfactory solutions to the problem with all the hard constraints, we decided to use the minimized objective function (16) for it. The main challenges for the solver were caused by the constraints (5)–(7) of the problem. We include constraints (6)–(7) into the objective function (16) with penalty ψ, while constraints (5) remained hard. Variable ζ in (16) denotes the total number of constraints (6)–(7) violations. We used penalties $\gamma_{kd} = 100, \lambda_{kd} = 200, \psi = 10^5$ for this objective function. The constant ϕ necessary for determining the variable χ was set to $3 \cdot 10^5$ for Table 6

Table 4. Capacity $v_k \in [450, 500]$ with the penalties

Instance	Min			Avg			Max		
	ε	κ	Distance	ε	κ	Distance	ε	κ	Distance
1	4	0	15388	6.1	0	15896.2	36	0	16411
2	178	0	17449	270.5	11	18472.9	468	0	18909
3	280	0	17978	276.5	88	18606.8	595	0	19204
4	0	0	19710	24.7	0	19920.5	34	0	20324
5	0	0	16335	9.3	0	16560.7	31	0	16563
6	22	0	14638	26.5	0	15131.6	79	0	14767
7	7	0	16092	2.5	0	16700	0	0	17212
8	0	0	16883	0.3	0	17200.3	1	0	17930
9	114	0	16208	115.7	0	16338.7	130	0	16084
10	11	20	18837	21.5	18	19037.7	18	20	19503

and $8 \cdot 10^4$ for Table 8. To provide non-deterministic results for LocalSolver runs, we added to the model one excessive constraint repeating already contained one.

$$\chi = \begin{cases} 1000, L(x, \gamma, \lambda) > \phi, \\ 1, \text{ otherwise.} \end{cases}$$

$$L_{\mathrm{s}}(x, \gamma, \lambda, \psi) = \min \left(\sum_{d \in D} \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} d_{ij} x_{ijkd} + \sum_{d \in D} \sum_{k \in K} (\gamma_{kd} \kappa_{kd} + \lambda_{kd} \varepsilon_{kd}) \right) \chi + \psi \zeta \tag{16}$$

In Tables 5, 6, 7 and 8, we show computational results obtained by our VNS algorithm and by LocalSolver. There were 10 runs for each instance. For Tables 5 and 8, computational time was set to 5 min, and for Tables 6 and 7, it was set to 10 and 2 min, respectively. To compare the obtained solutions, penalties $\lambda_{kd} = 40$ for Tables 5 and 7 and $\gamma_{kd} = 25, \lambda_{kd} = 20, \psi = 300$ for Tables 6 and 8 were used. All the solutions obtained for the former tables have $\kappa = 0$.

It is clearly visible that the results obtained by our algorithm are better than those of LocalSolver in all instances. It is also worth noting the decrease in the difference between the maximum and minimum objective values of the results with a decrease in the dimension of the problem.

Table 5. The results for instances with 320–350 clients for VNS algorithm

Instance	Min		Avg		Max	
	ε	Distance	ε	Distance	ε	Distance
1	0	10473	0.1	10486	1	10483
2	0	10543	2.1	10643	0	10874
3	0	10821	0	10853.3	0	10892
4	2	9950	3	10018.5	3	10141
5	0	8834	0.7	9042.2	0	9399
6	0	8830	0	8913.4	0	9301
7	0	9748	1.4	9867.3	11	9870
8	0	9204	0.6	9266.5	0	9383
9	0	10904	0.8	11014.9	6	11134
10	0	10217	0	10243.7	0	10277

Table 6. The results for instances with 320–350 clients for LocalSolver

Instance	Min				Avg				Max			
	ε	κ	ζ	Distance	ε	κ	ζ	Distance	ε	κ	ζ	Distance
1	48	0	0	10673	307.2	2	16.9	11713.1	312	0	73	12336
2	316	0	0	11094	404.7	3	46.7	11490.6	236	0	243	11874
3	120	0	0	10659	304.3	1	67.3	11525.5	377	0	219	11586
4	163	0	0	10790	326	0	17	11708.9	294	0	96	12603
5	12	0	0	11309	223.2	2	2.1	10722.3	474	20	3	10730
6	189	0	0	9980	329.9	10	3.7	10120	567	0	14	10240
7	145	0	0	11016	381	4	4.2	11163.1	585	20	11	10929
8	296	0	0	9827	482.1	7	29.5	10531.7	502	0	78	10452
9	374	0	0	11202	405.1	1	47.6	11993	240	10	124	12520
10	67	0	0	10974	216.2	0	4.3	11258.5	361	0	14	11891

The large instance with 892 clients can be effectively solved using developed VNS algorithm as well. However, to achieve an acceptable value of the variance for the results of applying the algorithm for an instance with such a number of clients, more computational time is required. Although smaller instances are usually examined in the literature, large-scale ones are also of interest to study, as they are often applicable in the real world.

Table 7. The results for instances with 130–150 clients for VNS algorithm

Instance	Min		Avg		Max	
	ε	Distance	ε	Distance	ε	Distance
1	0	3199	0	3205.5	0	3217
2	0	3388	0	3394.8	0	3405
3	0	2747	0	2759.4	0	2769
4	0	1657	0	1666.7	0	1687
5	0	3040	0.2	3053.6	2	2992
6	0	2110	0	2115.5	0	2126
7	0	2818	1.1	2819.3	11	2818
8	0	2960	0	2972.8	0	2987
9	0	2879	0	2883.5	0	2894
10	0	3061	3	2994.1	5	2958

Table 8. The results for instances with 130–150 clients for LocalSolver

Instance	Min				Avg				Max			
	ε	κ	ζ	distance	ε	κ	ζ	Distance	ε	κ	ζ	Distance
1	0	0	0	3430	20.4	4	0.4	3626.4	98	40	0	3650
2	65	0	0	3382	123.9	3	1	3683	241	0	6	3668
3	0	0	0	3130	40	0	0.7	3309.9	236	0	7	3357
4	0	0	0	1784	1.6	0	0.7	2481.8	16	0	3	3074
5	2	0	0	3207	129.4	8	1.3	3440.3	305	0	9	3502
6	0	0	0	2273	2.3	0	0.9	2637	23	0	4	2281
7	0	0	0	3212	89.6	1	0	3238.8	277	0	0	3223
8	0	0	0	3219	87.5	2	3.2	3258.5	321	0	32	3475
9	0	0	0	3128	17.9	0	0.5	3413.8	62	0	5	3451
10	61	0	0	3373	150.5	17	2	3286.5	265	0	13	3515

6 Conclusion

In this paper, we have studied a new consistent capacitated vehicle routing problem and designed the VNS algorithm for real-world instances. This algorithm is able to solve large-scale instances and reduce the total traveling distance.

Companies today are increasingly focused on customer satisfaction to achieve a competitive advantage. One of the components of these customer-first strategies is service consistency [12,23]. Thus, it is important to study problems with consistency requirements addressing real-world challenges. In our version of ConVRP, it is required that the same driver visit the same clients on the same day of the week according to their frequency of visits. We presented our program to the logistics company, and they were satisfied with it.

One of the new research directions is the control of time for client visits. As we have mentioned before, a logistics company can get an additional competitive advantage if each client is visited at about the same time. Such type of constraints can be incorporated into the model to improve the service of clients. Sure, new constraints will increase the total traveling distance and may require additional vehicles. The optimal balance here is an important line for research as well.

References

1. Aarts, E., Lenstra, J.: Local Search in Combinatorial Optimization. Wiley, New York (1997)
2. Ahuja, R.K., Ergun, Ö., Orlin, J.B., Punnen, A.P.: A survey of very large-scale neighborhood search techniques. Discrete Appl. Math. **123**(1–3), 75–102 (2002)
3. Alekseeva, E., Kochetov, Y., Plyasunov, A.: Complexity of local search for the p-median problem. Eur. J. Oper. Res. **191**(3), 736–752 (2008)
4. Coene, S., Arnout, A., Spieksma, F.: On a periodic vehicle routing problem. J. Oper. Res. Soc. **61**(12), 1719–1728 (2010)
5. Cordeau, J.F., Gendreau, M., Laporte, G.: A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks **30**(2), 105–119 (1997)
6. Dantzig, G.B., Ramser, J.H.: The truck dispatching problem. Manage. Sci. **6**(1), 80–91 (1959)
7. Davydov, I., Kochetov, Y., Carrizosa, E.: VNS heuristic for the $(r|p)$-centroid problem on the plane. Electron. Notes Discrete Math. **39**, 5–12 (2012)
8. Davydov, I., Kochetov, Y., Carrizosa, E.: A local search heuristic for the $(r|p)$-centroid problem in the plane. Comput. OR **52**, 334–340 (2014)
9. Diakova, Z., Kochetov, Y.: A double VNS heuristic for the facility location and pricing problem. Electron. Notes Discrete Math. **39**, 29–34 (2012)
10. Gendreau, M., Hertz, A., Laporte, G.: A tabu search heuristic for the vehicle routing problem. Manage. Sci. **40**(10), 1276–1290 (1994)
11. Golden, B.L., Raghavan, S., Wasil, E.A.: The Vehicle Routing Problem: Latest Advances and New Challenges. Springer, US (2008). https://doi.org/10.1007/978-0-387-77778-8
12. Groër, C., Golden, B., Wasil, E.: The consistent vehicle routing problem. Manuf. Serv. Oper. Manag. **11**(4), 630–643 (2009)
13. Grover, L.K.: Local search and the local structure of NP-complete problems. Oper. Res. Lett. **12**(4), 235–243 (1992)
14. Gutin, G.Z., Yeo, A.: Small diameter neighbourhood graphs for the traveling salesman problem: at most four moves from tour to tour. Comput. Oper. Res. **26**(4), 321–327 (1999)
15. Hemmelmayr, V.C., Doerner, K.F., Hartl, R.F.: A variable neighborhood search heuristic for periodic routing problems. Eur. J. Oper. Res. **195**(3), 791–802 (2009)
16. Irnich S., Toth, P., Vigo, D.: The family of vehicle routing problems.In: Vehicle Routing: Problems, Methods, and Applications, pp. 1–33. Society for Industrial and Applied Mathematics, Philadelphia (2014)
17. Kernighan, B., Lin, S.: An efficient heuristic procedure for partitioning graphs. Bell Syst. Tech. J. **49**(2), 291–307 (1970)
18. Khmelev, A., Kochetov, Y.: A hybrid VND method for the split delivery vehicle routing problem. Electron. Notes Discrete Math. **47**, 5–12 (2015)

19. Kochetov, Y., Kononova, P., Paschenko, M.: Formulation space search approach for the teacher/class timetabling problem. Yugoslav J. Oper. Res. **18**(1), 1–11 (2008)
20. Kochetov, Y., Khmelev, A.: A hybrid algorithm of local search for the heterogeneous fixed fleet vehicle routing problem. J. Appl. Ind. Math. **9**(4), 503–518 (2015)
21. Kochetov, Y.: Computational bounds for local search in combinatorial optimization. Comput. Math. Math. Phys. **48**(5), 747–763 (2008)
22. Kononova, P., Kochetov, Y.: The variable neighborhood search for the two machine flow shop problem with a passive prefetch. J. Appl. Ind. Math. **7**(1), 54–67 (2013)
23. Kovacs, A.A., Golden, B.L., Hartl, R.F., Parragh, S.N.: Vehicle routing problems in which consistency considerations are important: a survey. Networks **63**(3), 192–213 (2014)
24. Kovacs, A.A., Parragh, S.N., Hartl, R.F.: A template-based adaptive large neighborhood search for the consistent vehicle routing problem. Networks **63**(1), 60–81 (2014)
25. Kulachenko, I., Kononova, P., Kochetov, Y., Kurochkin, A.: The variable neighborhood search for a consistent vehicle routing problem under the shift length constraints. In: Proceedings of the 9-th IFAC Conference Manufacturing Modelling, Management and Control MIM 2019 (2019, Accepted)
26. Mladenovic, N., Hansen, P.: Variable neighborhood search. Comput. Oper. Res. **24**, 1097–1100 (1997)
27. Mladenovic, N., Hansen, P.: Developments of variable neighborhood search. In: Ribeiro, C., Hansen, P. (eds.) Essays and Surveys in Metaheuristics, vol. 2, 1st edn, pp. 415–439. Springer, Boston (2002). https://doi.org/10.1007/978-1-4615-1507-4_19
28. Talbi, E.G.: Metaheuristics: From Design to Implementation. Wiley, USA (2009)