A Reinforcement Learning Based Variable Neighborhood Search Algorithm for Open Periodic Vehicle Routing Problem with Time Windows

Binhui Chen
University of Nottingham, Nottingham, UK, Binhui.Chen@nottingham.ac.uk

Rong Qu
University of Nottingham, Nottingham, UK, Rong.Qu@nottingham.ac.uk

Ruibin Bai
University of Nottingham Ningbo, Ningbo, China, Ruibin.Bai@nottingham.edu.cn

Wasakorn Laesanklang
Mahidol University, Bangkok, Thailand, Wasakorn.lae@mahidol.ac.th

Based on a real-life container transport problem, a model of Open Periodic Vehicle Routing Problem with Time Windows (OPVRPTW) is proposed in this paper. In a wide planning horizon, which is divided into a number of shifts, a fixed number of trucks are scheduled to complete container transportation tasks between terminals subject to time constraints. In this problem, the routes traveled by trucks are open, as returning to the starting depot is not required in every single shift but every two shifts.

Our study shows that it is unrealistic to address this large scale and nonlinearly constrained problem with exact search methods. A Reinforcement Learning Based Variable Neighbourhood Search algorithm (VNS-RLS) is developed for OPVRPTW. The initial solution is constructed with an urgency level-based insertion heuristic, while different insertion selection strategies are compared. In the local search phase of VNS-RLS, reinforcement learning is used to guide the search, adjusting the probabilities of operators being invoked adaptively according to the change of generated solutions' feasibility and quality. In addition, the impact of sampling neighbourhood space in single solution-based algorithms is also investigated. Three indicators are designed in the proposed Sampling module to set the starting configuration of local search.

Experiment results on different sizes of real and artificial benchmark instances show that, the proposed Sampling scheme and feasibility indicator decrease the infeasible rate during the search. However, Sampling's contribution to solution quality improvement is not significant in this single solution-based algorithm. Comparing to the exact search and two state-of-the-art algorithms, VNS-RLS produces promising results.

Key words: Open Periodic Vehicle Routing Problem with Time Windows, Adaptive Operator Selection, Metaheuristics, Variable Neighbourhood Search

1. Introduction

Research on the Vehicle Routing Problem (VRP) can be tracked back to the *truck dispatching problem* proposed by Dantzig and Ramser (1959). It is defined as, starting from a depot, a number of vehicles with capacity constraints are to be routed to service a set of

customers with demands and return to the depot after servicing the last customer in their scheduled routes. Each customer is visited only once. In the scheduling graph of VRP, all the routes are Hamiltonian Cycles (close routes). The most common objectives in VRPs are minimizing the the number of vehicles used (or routes) and minimizing the total travel cost (distance/time). After decades of study, VRP has become one of the biggest successful stories in operational research and derives a large number of variants with different features (Golden et al. 2008), e.g. Vehicle Routing Problem with Time Windows (VRPTW), Vehicle Routing Problem with Pickups and Deliveries (VRPPD), Periodic Vehicle Routing Problem (PVRP), Open Vehicle Routing Problem (OVRP) and so on (Toth and Vigo 2001, Eksioglu et al. 2009).

1.1. Basic VRP Variants

Based on the basic definition of VRP, in VRPTW, customers' demands are associated with time constraints. The time of servicing a customer must be within a specific time interval given by the customer, and all vehicles must return to the depot before the end of the planning horizon given (Solomon 1987). VRPTW is the basic model for many other more complicated VRP variants. In this section, we review the variants which are most relevant to our study.

In VRPPD, customers' demands are divided into two categories: pick up shipments from a source and deliver shipments to a destination. Various constraints on pickup and deliver points lead to diverse VRPPD variants (Golden et al. 2008). If the depot is the only one pickup point while the customers are delivery points, or all shipments are picked up from customers and delivered to the depot, the problem would be classified as a *One-to-Many-to-One* problem. Whilst customers can be both pickup and delivery points, it is a *Many-to-Many* problem. In *One-to-One* problems, one customer's pickup demand is another customer's delivery demand. Furthermore, if consolidation is allowed when picking up, it is called a *Less-than Truckload Transportation* problem, otherwise it is a *Full Truckload Transportation (FTL)* problem (Wieberneit 2008).

In some real-life problems, the planning horizon is long and divided into multiple periods/shifts, where they are notated as Multi-Period Vehicle Routing Problems (MPVRP) (Mourgaya and Vanderbeck 2007). The vehicles must return to the depot every shift in MPVRP. Especially, if each customer has a specific visiting frequency within the planning horizon, this type of MPVRP is called Periodic Vehicle Routing Problem (PVRP). Each

customer can be visited more than once in PVRP, and its solution is usually represented as a set of visiting day (shift) combinations for customers. PVRP may occur in grocery distribution, soft drink industry, waste collection and so on (Hemmelmayr et al. 2009).

The earliest OVRP is proposed by Eppen and Schrage (1981), where a fleet collect goods from the central depot and deliver the goods to a number of geographically scattered customers. The main characteristic of OVRP is that its routes are Hamiltonian paths (open routes) rather than cycles (Tarantilis et al. 2005). This characteristic reflects the reality that many companies in real-world do not own a fleet for some reasons, and they hire external carriers, e.g. third party logistic providers and private vehicles, to service customers. Those hired vehicles do not need to return to the collection depot after servicing all customers assigned. The routes in OVRP terminate at the last customer serviced (Li et al. 2007).

1.2. Extended VRP Variants and Solution Methods

After decades of study, both exact and approximate algorithms have been intensively investigated for diverse VRP variants. However, due to the NP-hard feature of VRPs (Lenstra and Kan 1981), exact methods are more suitable to small or medium size of VRPs (Toth and Vigo 2001). On the other hand, in approximate approaches (or heuristics), metaheuristics have shown powerful performance in solving big-size and complex VRPs.

The first exact method for OVRP is proposed in (Letchford et al. 2007), but only small and medium size of instances (less than 151 customers and 14 vehicles) are solved with it. Tarantilis et al. (2004, 2005) propose two local search metaheuristics with annealing based threshold accepting scheme and list based threshold accepting scheme respectively. Both algorithms outperform previous approaches for OVRP. Two years later, a Record-to-Record Travel algorithm is adopted as the acceptance criterion, and the associated algorithm generates better solutions than 11 previous algorithms of OVRP (Li et al. 2007). Two metaheuristics algorithms based on Tabu Search for OVRP can be found in (Brandão 2004, Fu et al. 2005). A Static Move Descriptor is proposed in (Zachariadis and Kiranoudis 2010) to speed up the evaluation in best improvement search. The associated algorithm produces the best results on benchmarks of OVRP, however, it is inapplicable to OVRP with Time Windows (OVRPTW) where time constraints are considered.

The first introduction of the OVRPTW is in (Repoussis et al. 2007). In the proposed solution methodology, an insertion-based construction heuristic employs an improved *IMPACT*

criterion (Ioannou et al. 2001) to select insertion customer. Since then, a large number of metaheuristics are developed for OVRPTW. A Variable Neighbourhood Search (VNS) algorithm for OVRPTW can be found in (Perwira Redi et al. 2013), which outperforms the *IMPACT* approach. An Ant Colony Optimization (ACO) algorithm and a Genetic Algorithm are also applied to the problems of relatively small size (Guiyun 2009a,b).

In a special OVRPTW scenario, the routes start from geographically scattered customers to pick up goods. After visiting the customers assigned, vehicles return to the central depot to unload the goods collected. The routes in this scenario are open at the starting point, which is opposite to the standard OVRP. An Adaptive Large Neighbourhood search algorithm is proposed for this Reverse-OVRPTW in (Schopka and Kopfer 2016). School Bus Routing Problem is another special case of OVRP, where its morning and afternoon routing problems can be addressed as the same problem. Every morning, school buses send students to the school from pickup stops, while in afternoon students are sent back to the stops in the reverse order of the morning routes (Park and Kim 2010). In addition, Liu and Jiang (2012) and Brito et al. (2015) study the VRP with both open and close routes.

The first research on Periodic Vehicle Problem with Time Windows (PVRPTW) is in (Cordeau et al. 2001). Considering travel time, capacity, duration and time windows, a construction heuristic followed by a Tabu Search (TS) is proposed. An improved TS method adopting the *Forward Time Slack* (Savelsbergh 1992) is later proposed to further reduce the route (Cordeau et al. 2004). An ACO model for PVRPTW can be found in (Yu and Yang 2011) where a multiple pheromone information matrix is designed. The flexibility of customer visiting date in PVRPTW further increases the number of parameters in this ACO algorithm and the complexity of this methodology. More solutions for PVRPTW can be found in (Pirkwieser and Raidl 2008, Rahimi-Vahed et al. 2015).

An Open Periodic Vehicle Rouging Problem (OPVRP) model is introduced in (Danandeh et al. 2010). The vehicles in it are not obliged to return to the depot at the end of each day in a planning horizon of multiple days. In the proposed solution construction heuristic, a k-means clustering algorithm is used to assign customers to routes, without considering the capacity limit. A feasibility procedure would fix the infeasible assignments latter. This approach can swiftly generate a feasible solution on small size problems. However, it would be inefficient in bigger problems or when the depot is not in the geographic center. Time windows are not considered in this model.

Vidal et al. (2014) propose a unified hybrid genetic search framework (UHGS) aims to provide a general-purpose solver for diverse VRP variants. It produces the results better than or close to the state-of-the-art algorithms on benchmarks. However, as a genetic algorithm, the experiments show the computation time sharply increases when meeting MPVRP. This is because it has to face the problem of positioning the shift and route delimiters in chromosome/genotype (solution representation), which is hard to handle. This solution partition problem is tackled as a shortest path problem in UHGS. The long computation time impedes the application of UHGS in big-size MPVRP. More metaheuristics for VRPs see (Bräysy and Gendreau 2005, Gendreau et al. 2008)

Most researches model VRPs with connected network, where the connected nodes can be customers, demands, services and so on. The weight of a edge connecting two nodes represents the cost of vehicle traveling from a node to the other one. This method is easy to apply to the problems whose service activities are simple. However, in some real-life problems, the service activities are complex and hard to simplified or combined. Bai et al. (2015) use the loading and unloading nodes pair to represent a transportation task, converting a container transportation problem into a Set-Covering problem. Then the problem is solved with an exact algorithm. This method is not suitable to those problems with many container terminals. Wang and Regan (2002) integrate loading, traveling and unloading activities into a *task node* for a pickup and delivery problem. This task node-based model simplifies the model of that real-life complex problem, making the large number of classic node-based algorithms applicable. This way has been used in various VRPs research (Zhang et al. 2010, Chen et al. 2013).

This paper addresses a real world container transportation problem, which shares some common features with the classic OVRP and PVRPTW. The mathematical problem model and an introduction example are presented in *Section 2*. The proposed solution methodology is introduced in *Section 3*. Apart from an urgency level-based constructive heuristic, an improvement metaheuristic with reinforcement learning is also developed. *Section 4* presents the experiment results and analysis on benchmark instances. The conclusions of this paper are presented in *Section 5*.

2. Problem Definition & Mathematical Model

The problem concerned in this paper is a real-world inter-dock container transportation problem at Ningbo Port, which is the fifth largest port in the world. Everyday a fleet of 100

trucks transport commodities (containers) between nine container terminals (see *Figure 1*). Each commodity consists of a number of containers. The containers are picked up from source terminals and delivered to destination terminals, satisfying the time constraints on commodities and drivers. Since the size and operational cost of the fleet is relatively fixed in the Ningbo Port, the port manager expect to decrease the empty-load travel distance of the fleet, optimizing the utility of trucks. This problem is a One-to-One VRPPD with FTL. The trucks used are identical and each truck can carry only one container at a time due to the capacity, thus consolidation is impracticable.

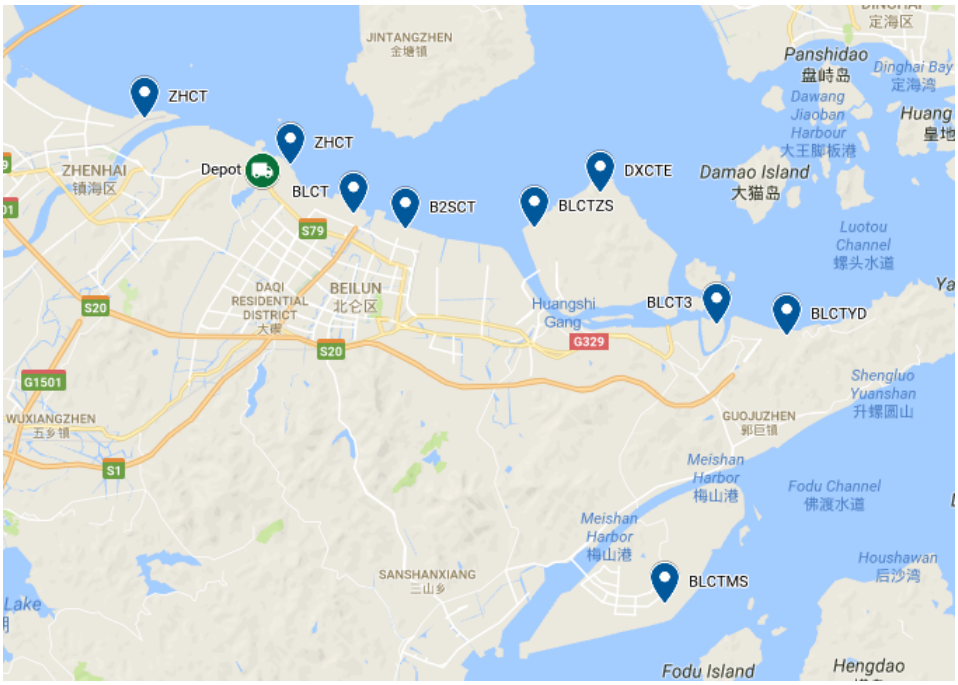

Figure 1 **Nine container terminals' locations in the Ningbo Port China.**

The commodities to be transshipped are usually issued several days before their shipment deadlines, which brings a long scheduling horizon. Each working day in the scheduling horizon is divided into two shifts (i.e. day and night shifts with 12 hours per shift) obliging the related regulations on drivers' working hours in Labour Law. In the first shift (day shift) everyday, trucks depart the depot with a list of tasks to complete. When all tasks assigned are completed, the trucks would park at the last task destinations, or at the first task source terminals of the second shift (night shift) as long as the trucks can arrive there before the end of the day shift. Then, at the beginning of the night shift, a new group of drivers get their trucks (shift change) at the terminals appointed respectively. In the

night shift, after all assigned tasks being completed trucks must go back to the depot for maintenance and preparation of the next day. The travel of trucks heading to the next task source is empty-load, with no container being carried.

Drivers take shift change in the middle of a working day at scattered terminals, instead of one specific depot. It can be found that, here the routes in each shift are open as the routes in the OVRP. More precisely, the scheduled routes of day shifts are open as they are not ended at the depot, while the routes of night shifts are reverse open routes. The difference between this problem and the School Bus Routing Problem is that, in our problem, the routes in night shifts are not the same as the routes in day shifts in reversed order.

The typical scheduling horizon in Ningbo Port spans from 2 to 4 days (4-8 shifts). In such a multi-shift problem, the number of containers in each commodity can be treated as the visiting frequency of customers in PVRP. However, the model of PVRP cannot be directly applied due to the distinct practical constraints in this problem. In this paper, the Ningbo Port container transportation problem is modeled as an Open Periodic Vehicle Routing Problem with Time Windows (OPVRPTW). To the best of our knowledge, this is the first time OPVRPTW being introduced in the literature.

In VRPPD research, the shipment loading and unloading time is usually ignored or uniformed as they are small and/or identical. However, because of the limitation of cranes at terminals, the time of loading and unloading account for a considerable proportion of the total service time in OPVRPTW. Besides, the service time are different at different terminals in OPVRPTW, thus the service time cannot be simply ignored or uniformed. In our mathematical OPVRPTW model, the method of Wang and Regan (2002) is adopted. The time of loading and unloading activities are combined with the traveling time, defining the *task nodes* with different service time. In the proposed model, a *task node* is defined by the loading at the source terminal, unloading at the destination terminal, loaded travel from the source to the destination, and the associated time window of the task.

The notations used in the model are introduced in Table 1. Every scheduled route starts a *starting depot*, connects a number of task nodes (tasks to be completed) and ends at a *termination depot*. It is notable that, some of these routes are empty, which means no task is completed on them. The first shift of every working day is denoted as an *odd-indexed* shift, while the second shift is *even-indexed*.

Table 1 The list of notations

Input Parameters:	
K	Fleet size.
S	The set of time-continuous working shifts, which can be divided into odd-indexed shifts (S_{odd}) and even-indexed shifts (S_{even}).
$[Y_s,Z_s]$	Time window of shift s
$N=\{0,1,2,...,n\}$	Set of $n+1$ nodes. Each node represents a task except node 0 is the physical depot.
$[a_i,b_i]$	Time window for node i. The time window for node 0 is zero at the boundary of a shift. If a truck arrives at the source of i early, it has to wait until a_i.
W	Set of *Artificial Depots*. This set of nodes are introduced to represent the termination depots in S_{odd} or starting depots in S_{even}. Which physical terminal would be an artificial depot is decided by if the associated truck in S_{odd} can arrive at that terminal before the end of shift. W varies with solution changing, i.e. a physical terminal may not appear or may appear more than once in W.
A	Set of arcs. Each arc(i,j) represents that node j is immediately serviced/visited after servicing/visiting node i.
c_{ij}	The cost of empty load travel from node i to node j. When both i and j are task nodes, c_{ij} is the distance from the destination of i to the source of j. Otherwise, it is the distance between a terminal and the depot.
t_{ij}	The travel time from node i to node j. When both i and j are task nodes, t_{ij} is the travel time from the destination of i to the source of j. Otherwise, it is the travel time between a terminal and the depot.
T_i	The arrival time at node i.
B_i	The time to begin the service of node i.
l_i	The time for servicing node i, which includes the loading time, transportation time (from pick-up source to delivery destination) and unloading time. The service time of a depot is zero.
Decision Variables:	
x_{ij}^s	A binary decision variable for nodes $i,j \in N \cup W$. Its value is 1 if arc(i,j) is included in the solution in shift s, otherwise is 0. $i \in W$ AND $j \in W$ is not allowed

Artificial depots (W), are introduced to connect an odd-indexed shift (S_{odd}) to the following even-indexed shift (S_{even}). In S_{odd}, artificial nodes are termination depots, while they become the starting depots in the following S_{even}. When a truck is transferred from S_{odd} to S_{even}, the first source terminal of a route in S_{even} serves as an artificial depot, if it can be reached before the end of S_{odd}. Otherwise, the last destination terminal of the route in S_{odd} will be the artificial depot.

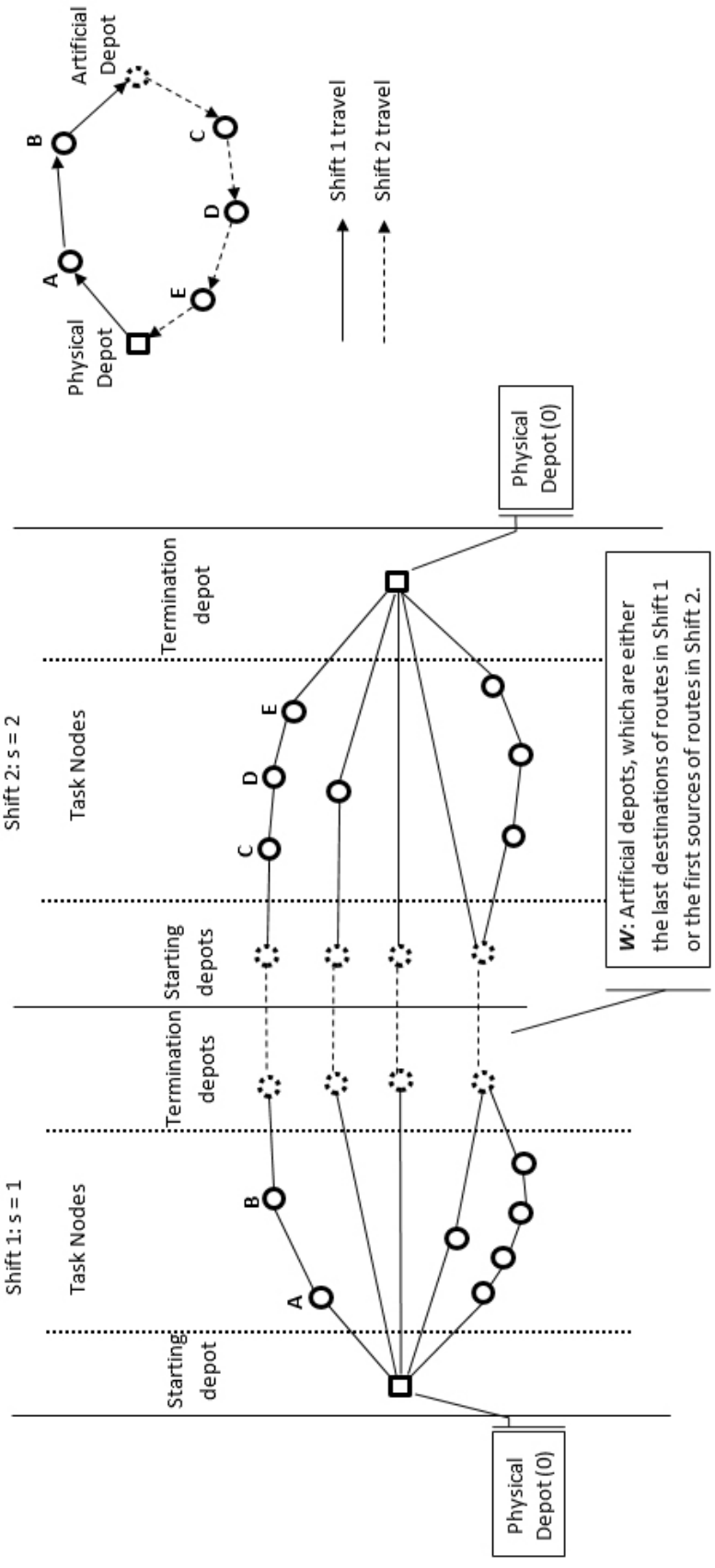

Figure 2 A scheduling example of two consequent shifts with five trucks. The solid line circles represent the transportation tasks to be completed. The right sub-figure indicates the first route in the schedule.

An example of one day schedule (two consecutive shifts of an odd-indexed and an even-indexed) is presented in *Figure 2*. The fleet size is five (i.e. $K = 5$), corresponding to five routes. Take the first route as an example, two and three tasks are completed in the odd-indexed and the even-indexed shifts, respectively. The lines directly connect starting depots and termination depots are empty routes, which means no task is completed on them. In this example, each shift has two empty routes. The artificial depots in W are either the last destinations on Shift 1 routes or the first sources on Shift 2 routes. The number of artificial depots is decided by the number of terminals at where shift-changes happen. In this example, two trucks change shift at the fourth artificial depot.

The problem can be formally defined as follows:

$$Minimise \qquad TD = \sum_{s\in S}\sum_{i\in N\cup W}\sum_{j\in N\cup W} c_{ij}\cdot x_{ij}^{s} \tag{1}$$

Subject to:

$$\sum_{s\in S}\sum_{i\in N\setminus\{0\}} x_{ij}^{s} = 1, \qquad \forall j \in N\setminus\{0\} \tag{2}$$

$$\sum_{s\in S}\sum_{j\in N\setminus\{0\}} x_{ij}^{s} = 1, \qquad \forall i \in N\setminus\{0\} \tag{3}$$

$$\sum_{i\in N\cup W} x_{ij}^{s} = \sum_{f\in N\cup W} x_{jf}^{s}, \qquad \forall j \in N\setminus\{0\}, s \in S \tag{4}$$

$$T_j = \sum_{i\in N\setminus\{0\}} (B_i + l_i + t_{ij})\cdot x_{ij}^{s} + \sum_{i=\{0\}\cup W} (Y_s + t_{ij})\cdot x_{ij}^{s}, \qquad \forall j \in N\setminus\{0\}, s \in S \tag{5}$$

$$B_j = T_j + max\{a_j - T_j, 0\}, \qquad \forall j \in N\setminus\{0\} \tag{6}$$

$$x_{ij}^{s}\cdot Y_s \quad \leq \quad x_{ij}^{s}\cdot T_j, \qquad \forall i \in \{0\}\cup W, j \in N\cup W, s \in S \tag{7}$$

$$x_{ij}^{s}\cdot (B_i + l_i) \quad \leq \quad x_{ij}^{s}\cdot Z_s, \qquad \forall i \in N\cup W, j \in \{0\}\cup W, s \in S \tag{8}$$

$$a_i \leq B_i \leq b_i - l_i, \qquad \forall i \in N\setminus\{0\} \tag{9}$$

$$x_{ij}^{s} \in \{0,1\}, \qquad \forall i,j \in N\cup W, s \in S \tag{10}$$

$$x_{vw}^{s} = 0, \qquad \forall v \in W, w \in W, s \in S \tag{11}$$

In odd-indexed shifts ($\forall s \in S_{odd}$):

$$\sum_{j\in N\setminus\{0\}\cup W} x_{0j}^{s} = K, \qquad \forall s \in S_{odd} \tag{12}$$

$$x_{i0}^{s} = 0, \qquad \forall i \in N\setminus\{0\}\cup W, s \in S_{odd} \tag{13}$$

$$\sum_{i\in N}\sum_{w\in W} x_{iw}^{s} = K, \qquad \forall s \in S_{odd} \tag{14}$$

In even-indexed shifts ($\forall s \in S_{even}$):

$$\sum_{j \in N} x_{jw}^{s-1} = \sum_{e \in N} x_{we}^{s} \qquad \forall w \in W, s \in S_{even} \tag{15}$$

$$x_{0j}^{s} = 0, \qquad \forall j \in N \backslash \{0\} \cup W, s \in S_{even} \tag{16}$$

$$\sum_{w \in W} \sum_{j \in N} x_{wj}^{s} = K, \qquad \forall s \in S_{even} \tag{17}$$

$$\sum_{i \in N \backslash \{0\} \cup W} x_{i0}^{s} = K, \qquad \forall s \in S_{even} \tag{18}$$

The objective of this problem is to minimize the total travel distance (TD), eq. (1). In classic VRPs, this objective is the secondary objective usually. However, the operational cost of the fleet in this real-life problem is fixed basically, thus, minimizing the travel distance become the only one objective. Since the loaded travel distance is fixed in each instance, the objective actually is minimizing the empty-load travel distance.

Constraints (2) and (3) denote that every task node can be visited exactly once and all the tasks must be visited. Constraint (4) specifies that a task may only be serviced after the previous task is completed. Constraints (2)-(4) together make sure arcs of over more than one shift are unacceptable. Constraint (5) is the arrival time at a task node. Constraint (6) defines the beginning time of servicing a task node. This time is calculated by the arrival time plus the waiting time at the source of a task. Constraints (5) and (6) enforce the correct successive relationship between consecutive nodes.

Constraints (7) and (8) are the time window constraints of each shift, while constraint (9) represents the time constraint on each task. The domain of the respective decision variable is defined by constraints (10) and (11). Especially, constraints (11) prohibits the travel between two artificial depots.

The starting and termination depots in odd-indexed shifts and even-indexed shifts are different. Constraints (12) and (14) represent that K trucks leave the physical depot (0) at the beginning of an odd-indexed shift, and they would stop at artificial depots at the end of the shift. Constraint (13) represents that no truck returns to the physical depot in odd-indexed shifts. Constraints (16) - (18) place the reverse restraints in even-indexed shifts. Constraint (15) defines the shift change from an odd-indexed shift to the following even-indexed shift on artificial depots, where the in-degree of each artificial terminal in S_{odd} equals its out-degree in the following S_{even}.

From this integer programming model, we can find that this problem is constrained nonlinearly with a huge solution space. The size of the solution space of OPVRPTW is

decided by the length of the scheduling horizon ($|S|$), the fleet size (K) and the number of tasks (n). Since the number of total routes in a solution could be up to $|S| \cdot K$, while the number of permutations of tasks is $n!$, the size of the search space is $|S| \cdot K \cdot n!$. In real-life scenarios, n in the whole planning horizon could be larger than 1,000. It is worth noting that, empty routes in S_{even} do not mean their travel distances are zero. The cost of empty route can be zero, only when the connected artificial node represents the physical depot exactly.

3. Solution Methodology

3.1. Motivations & Algorithm Framework

3.1.1. Motivitions

We use the CPLEX solver on both real-life and artificial benchmark instances, and the results (see *Section 4.2*)show that it is unrealistic to address the OPVRPTW with exact methods. Heuristics and metaheuristics are thus investigated in this study for OPVRPTW. A large number of metaheuristic algorithms have obtained promising results in VRPs, including Population-Based and Single Solution-Based metaheuristics, e.g. Evolutionary Algorithms (EA) (Baker and Ayechew 2003), Population-Based Incremental Learning (Lourens 2005), Simulated Annealing (Kaji and Ohuchi 1999), TS (Potvin et al. 1996) and VNS (Chen et al. 2016).

Population-based approaches evolve a population of solutions during the search, which show powerful performance in tightly constrained and multi-objective VRPs (Jozefowiez et al. 2008). However, OPVRPTW is challenging to population-based approaches due to its high dimensional solution structure and large problem scale. The tasks in OPVRPTW must be indexed from three dimensions in the solution (index of the shift, index of route and the position in the route). This 3D structure increases the difficulty of operations between solution individuals, e.g. the above-mentioned solution partition problem in Genetic Algorithms. In addition, when the problem size and population size are big, the computation time for the huge population would be very long. Thus, population-based methods are not suitable to this large scale high-dimensional problem.

Single solution-based metaheuristics (or Local Search) use different strategies and neighbourhood operators to explore the solution space iteratively, while only one solution is updated in each iteration. Apart from straightforward operators, e.g. Swap and Insertion,

many delicately designed neighbourhood operators are developed and applied in various metaheuristics, such as λ-opt (Lin 1965), or-opt (Or 1976), Cyclic Transfers (Thompson et al. 1989), 2-opt* (Potvin and Rousseau 1995), CROSS-exchange (Taillard et al. 1997) and so on. More successful single solution-based approaches can be found in (Bräysy and Gendreau 2001).

VNS systematically changes neighbourhood operators to extensively explore search space and shows excellent performance in VRPs (Hansen et al. 2010). However, in real-life big-size and tight-constraint instances, the classic VNS structure often shows low efficiency. To avoid this deficiency, *Reinforcement Learning* is introduced to VNS in our study. Reinforcement Learning (RL) is an adaptive learning and decision-making scheme, based on a probability distribution sampled over the candidate set. The probability distribution is continuously updated according to the reinforcement feedback from the learning environment (Thathachar and Sastry 2002).

RL scheme can be used in *Adaptive Operator Selection* (AOS), guiding the search process (Veerapen et al. 2012). Credit Assignment mechanism and Selection Rule are the two essential components in AOS. According to the historical performance of operators, credit assignment mechanism updates the invoked probability (or weight) of candidate operators during the search, e.g. increasing or reducing the probability with elaborately designed reward function (Pisinger and Ropke 2007). Then, the operators to be executed are chosen with a Selection Rule. Roulette Wheel scheme and Pursuit Algorithm (Thierens 2005) are the two commonly used selection rules. The former chooses operators on the basis of their probability distribution and the latter always selects the operator with the largest probability/weight.

AOS has been widely used in metaheuristics. In population-based metaheuristics, credit assignment mechanism calculates the feedback values based on the quality of solution population (Smith and Fogarty 1997, Lin et al. 2016). To single solution-based approaches, RL and AOS has also shown outstanding performance in VRPPD and other VRP variants, e.g. in Adaptive Large Neighbourhood Search (Hemmelmayr et al. 2012, Schopka and Kopfer 2016). Some associated methodologies produce the best results on a the benchmarks of classic VRP variants (Ropke and Pisinger 2006a,b). More applications of adaptive learning in single solution-based heuristics can be found in (Burke et al. 2011, Veerapen et al. 2012).

In metaheuristics, Sampling can be used to measure the fitness landscape surrounding a solution, providing guidance for the search. It shows powerful performance in Evolutionary Computation (Jin 2005). Soria Alcaraz et al. (2014) propose an Evolvability Metric of Sampling, where two scalars are proposed as the indicators in the credit assignment mechanism during evolution. One indicator is the probability of the offspring having higher or equal fitness comparing to their parents, while the other indicator is the mean fitness of all sampling offspring. The proposed EA approach with Sampling produces promising results in three classic optimization problems. Whether Sampling also works in single solution-based heuristics and OPVRPTW is worthy to be investigated further.

3.1.2. Algorithm Framework

A VNS algorithm with Reinforcement Learning and Sampling (VNS-RLS) is proposed in this paper, see the framework in *Algorithm 1*. The initial feasible solution S is constructed with a heuristic in *Step 1*. Then, in each VNS iteration, *Shaking* generates a new feasible solution S' as the starting solution by perturbing S. In *Step 2.2*, with S', *Sampling* initializes a set of weights (WS) for all the operators in the neighbourhood operator set (NS). With the initial invoked probability distribution of operators, which is generated according to WS, *Local Search* pursues better solutions iteratively (*Step 2.3*). These steps are repeated until the objective value has not been improved by 0.01% after a predefined number of iterative times.

Algorithm 1 The VNS-RLS framework

Input: The set of tasks to be assigned and the scheduling horizon.

Start

Step 1: Produce an initial feasible solution S with a constructive heuristic. // Section 3.2.

Step 2:

while Terminate condition is not met **do**

Step 2.1: $S' \leftarrow$ **Shaking**(S). // Section 3.3.

Step 2.2: $WS \leftarrow$ **Sampling**(S', NS). // Section 3.4.

Step 2.3: $S \leftarrow$ **Local Search**(S, S', WS). // Section 3.5.

end while

Stop

Output: S.

Eight improved classical neighbourhood operators, which require relatively less computation time, are adopted in VNS-RLS. These operators are modified by considering the specific solution structure of the OPVRPTW. The solutions of classic VRPs are 2-dimensional. However, as mentioned before, the solution of OPVRPTW has one more dimension of shift. To systematically increase the diversification of the search in this tightly constrained problem, we specify the level of perturbation for each operator. Previous research have shown that properly using operators with diverse degrees of perturbation can significantly improve search performance (Chen et al. 2016). The eight improved operators work at different levels (intra-route, intra-shift and inter-shift) with diverse perturbations, introduced in Table 2. Considering the time constraints in OPVRPTW, the directions of the operated strings are kept in all the eight neighbourhood structures.

Table 2 Operations of the eight operators within *NS* in VNS-RLS.

Operator	Description
Intra-Shift 2-opt*	Execute 2-opt* exchange, where the two chosen routes are randomly selected from the same shift.
Inter-Shift 2-opt*	Execute 2-opt* exchange, where the two chosen routes are randomly selected from different shifts.
Intra-Route Or-opt	A string of task nodes is repositioned in the original route.
Intra-Shift Or-opt	A string of task nodes is repositioned from one route to another route in the same shift.
Inter-Shift Or-opt	A string of task nodes is repositioned from one route to another route in a different shift.
Intra-Route CROSS	Swap two strings of task nodes in the same route.
Intra-Shift CROSS	Swap two strings of task nodes from different routes in the same shift.
Inter-Shift CROSS	Swap two strings of task nodes from two routes which are from different shifts.

3.2. Initial Solution Construction

A large number of constructive heuristics have been developed for VRPs. Saving Algorithm (Clarke and Wright 1964) always selects the operation which brings the largest travel distance saving. However, it shows worse performance on asymmetrical Capacitated Vehicle Routing Problem and cannot control the number of vehicles used (Vigo 1996). The later constructive algorithms, e.g. Cluster First-Route Second (Gillett and Miller 1974), Route First-Cluster Second, Sweep Algorithm (Golden et al. 1984) and so on, perform better on instances where customers are geographically clustered around the depot. More constructive heuristics for VRPs can be found in (Laporte et al. 2000).

Based on the above-mentioned methods, Solomon (1987) proposes four constructive heuristics for VRPTW, while the Insertion-Based heuristic outperforms the other three.

Insertion-based heuristics can be integrated with diverse insertion selection strategies easily. It can control the number of vehicles used when constructing the solution. At Ningbo Port, the nine container terminals are neither clustered nor uniformly locating around the depot, and the triangle rules do not work in its transportation network due to the real traffics. Besides, the number of trucks in this problem is limited (100). Based on these realities, an Urgency Level-based Insertion constructive Heuristic (ULIH) is thus developed for OPVRPTW.

The number of containers to be transshipped every day is large in Ningbo Port. To complete all the tasks before their deadlines, tasks are classified into two categories to specific shifts: *mandatory* and *optional*. When scheduling, the mandatory tasks will be assigned first. This urgency level-based schedule tactic is often used in solving real-life problems (Chen et al. 2013). Another strategy often used in practice is that tasks should be assigned as early as possible to avoid leaving too many unassigned tasks to later shifts. In real-life, there may be new tasks submitted to the scheduling system in real-time, so leaving more free time slots and trucks for later shifts can also improve the system reliability.

The urgency level of a task varies as the shift changing. In brief, in ULIH, when task i is available to shift s, if i must be completed no later than shift s, then i is *mandatory* to shift s; Otherwise, it is *optional*. Tasks are inserted into the scheduled routes, which means being assigned to trucks. If a task cannot be inserted into any one existing route, a new route will be created, which means the number of trucks used (k) is increased by one. Considering the *big* tasks (with the large service time) are harder to assign in scheduling, the biggest task is selected as the first task in the newly created route. After all mandatory tasks being assigned, optional tasks will be inserted until no feasible insertion and free truck available ($k \leq K$).

The insertion selection tactic determines the solution quality and constructive speed of insertion-based heuristics. Some criteria have been developed to select the best candidate and insertion position for routing problems (Ioannou et al. 2001, Repoussis et al. 2007), however, their greedy selection dramatically increase the computation time in large problems. To reduce the computation time, some other researchers randomly select the next insertion or narrow the comparison range in each iteration (Ropke and Pisinger 2006a). Three insertion selection tactics are proposed in ULIH, which are introduced below.

- *Greedy Tactic.* All unassigned tasks and insertion positions on all routes are evaluated, and the insertion which brings the lowest empty-load travel distance increase would be executed. This greedy tactic generates *tighter* routes while more evaluation time is required.
- *First-Insertion Tactic.* All tasks are sorted according to their closeness to their *deadlines*. This tactic always insert the first task in the unassigned task set (closest to deadline) into the partial solution. This strategy can construct routes more quickly, but sacrificing the solution quality as more trucks are required, and some mandatory tasks may not be assigned due to lack of trucks. When K is small, this tactic should be used cautiously.
- *One-Route Tactic.* Only consider the insertions for one route in one iteration to strike a balance between construction speed and solution quality. It may cost more computation time than *First-Insertion* tactic, but bringing better solution quality.

These three tactics are tested and compared on benchmark instances, results presented in *Section 4.3*.

3.3. Shaking

Shaking is often used in VNS algorithms to jump out the local optimum and diversify the search. The commonly used Shaking schemes include: randomly invoking neighbourhood operators, designing specific Shaking Operators and so on. To jump to farther areas from the current search region, in VNS-RLS, four neighbourhood operators which bring larger changes are employed in Shaking. Inter-Shift 2-opt*, Intra-Shift 2-opt*, Inter-Shift Or-opt and Inter-Shift CROSS are sequentially used to generate the starting solution of *Local Search*. Each operator is executed once, and the solution generated is passed to the following operator as the seed solution. Note that the new solution generated will be accepted in Shaking as long as it is feasible, even if its quality is worse. The aim of Shaking is increasing diversification, rather than myopic solution improvement.

3.4. Sampling of Neighbourhood

Sampling of surrounding environment can provide guidance for the search and has improved the search performance in many population-based algorithms. The impact of sampling on the search trajectory of single solution-based algorithms is investigated in our study. A sampling phase is applied before the *Local Search* phase in VNS-RLS.

Based on the perturbed solution S' generated by *Shaking*, every candidate neighbourhood operator NS_i is sampled, producing d sampling solutions (S_i^s). Empirically, each

operator is sampled 10 times ($d = 10$) to balance the evaluation time and the metric approximation. Those sampling solutions are then measured, while the feedback will be used to generate the initial weights of operators *WS*, providing an initial search direction to *Local Search*.

Three scalars (eqs. 19 - 21) are defined to measure the sampling solutions. To operator NS_i, the first scalar E_i^a indicates the probability of finding a solution which is not worse than the current solution. E_i^b concerns the average solution quality of the solutions generated. In the preliminary experiments, it is observed that more than half of the neighbourhood moves produce infeasible solutions. To reduce the infeasible moves in the search, a third indicator E_i^c is proposed in VNS-RLS, which indicates the probability of obtaining a feasible solution with operator NS_i.

$$E_i^a = \frac{\{|S_i^s| \mid TD(S_{ij}^s) \leq TD(S')\}}{d} \tag{19}$$

$$E_i^b = 10 \cdot \left(\frac{TD(S)}{\sum_{j=1}^{d} TD(S_{ij}^s) \ / \ d} - 1 \right) \tag{20}$$

$$E_i^c = \frac{\{|S_i^s| \mid S_{ij}^s \ is \ feasible\}}{d} \tag{21}$$

For each operator NS_i, the overall evaluation value (E_i) is a combination of the above three metrics. Based on the preliminary experiment results, the weights of indicators are set as $\alpha = 0.5$, $\beta = 0.2$ and $\gamma = 0.3$.

$$E_i = \alpha \cdot E_i^a + \beta \cdot E_i^b + \gamma \cdot E_i^c \qquad (s.t.\ \alpha + \beta + \gamma = 1) \tag{22}$$

The weight (W_i) determines the probability of a neighbourhood operator NS_i being invoked in the Local Search, which is stored in *WS*. To each operator, the initial W_i is calculated out based on E_i, see eq. (23). The sum of the initial W_i of the P operators is 1. Let E_{min} be the minimal E_i in the P operators, a operator with larger E_i would have a larger initial W_i. To avoid some of the weights being too small and the associated operators will never be invoked, a minimal initial weight W_{min} is adopted, which is empirically set to 5% in VNS-RLS.

$$W_i = W_{min} + (1 - P \cdot W_{min})(E_i - E_{min}) / \sum_{i=1}^{P} (E_i - E_{min}) \tag{23}$$

3.5. Local Search with Reinforcement Learning

In VNS-RLS, the *Local Search* is an iterative search module with reinforcement learning, see *Algorithm 2*. Firstly, the initial invoked probabilities are distributed to operators according to W_i. Using the *Roulette Wheel scheme* as the selection rule, in Step 2, one neighbourhood operator is selected and applied once to the current solution (S'), generating a new solution (S''). In *Move or Not*, S'' will be evaluated to decide whether it is accepted as the new current solution. W_i is then updated according to the evaluation feedback. These steps are repeated until no improvement is obtained after a predefined number (L_{max}) of evaluation times. Based on the preliminary experiments, L_{max} is set to 150.

Algorithm 2 Local Search(S, S',WS)

Input: Current best solution S, starting solution S' and the initial weight set WS (initial W_i).

Start

Set $L \leftarrow 0$.

while $(L < L_{max})$ **do**

Step 1: Generate the invoked probability (Pr_i) of each operator based on its weight (W_i).

$Pr_i \leftarrow W_i / \sum_{i=1}^{P} W_i$

Step 2: Neighbourhood Search

Select a neighbourhood operator NS_i with *Roulette Wheel Scheme* with the probability Pr_i.

$S'' \leftarrow NS_i(S'),\ L \leftarrow L+1$.

Step 3: Move or Not

if S'' is infeasible **then** //Feasibility Indicator (FI)

$W_i \leftarrow W_i * 0.9$

else if $Evaluation(S'') < Evaluation(S)$ **then** //Solution Quality Indicator: TD

$S \leftarrow S'', S' \leftarrow S'', L \leftarrow 0$.

$W_i \leftarrow W_i * 1.1$

else if $Evaluation(S'') - Evaluation(S') < DEVIATION$ **then**

$S' \leftarrow S''$.

else

$W_i \leftarrow W_i * 0.9$

end if

end while

Stop

Output: The best found solution S.

In the *Move or Not* phase, a *Record-to-Record Travel* scheme (Dueck 1993) is used as the acceptance criterion. When S'' has better solution quality than S, or the difference of the solution quality between S'' and S' is less than a predefined value ($DEVIATION$), S'' will be accepted as new S'. In our algorithm, $DEVIATION$ is set to 2 based on preliminary experiments. In practice, less than 2 km increase in total travel distance is acceptable.

The weight of NS_i is updated during the search. In the literature, most learning indicators focus on the solution quality (Ropke and Pisinger 2006a), while the feasibility of search attracts less attention. When updating the weight, a feasibility indicator is employed in VNS-RLS, which is just the feasibility of the newly generated solution. To decrease the computing time in calculating the updated weights, a simple Credit Assignment Mechanism is employed. If S'' is infeasible or cannot be accepted due to its low solution quality, a penalty of 10% reduction of weight will be applied to NS_i. Otherwise, if the quality of S'' is better than the current best solution S, the weight will be increased 10% as a reward. The penalty and reward rates are set based on the preliminary experiments. More discussion about adaptive weight adjustment can be found in (Thathachar and Sastry 2002).

In our study, the contribution of the feasibility indicator and different components in VNS-RLS are analysed and discussed, results presented in *Section 4.5*.

4. Benchmark & Computational Experiments

4.1. Benchmark Instances

Bai et al. (2015) extract 15 instances from the real-life Ningbo Port dataset. In different instances, the planning horizons are 4, 6 or 8 shifts, respectively, of 384 up to 1073 tasks. An artificial instance set including 17 instances is created as well, with different feature combinations on time window tightness (Tight/Loose), workload balance at terminals (Balanced/Unbalanced) and planning horizons of 4 and 8 shifts. Especially, a very large instance with more than 2000 tasks is created as well. The detailed features of instances are presented in Table 3 and Table 4. The time window of a task, defined by the time it becomes available and the deadline it must be delivered to its destination terminal, varies from 1-2 hours, up to 6 shifts.

Comparing to classic VRP benchmarks (Solomon 1987, Gehring and Homberger 1999), it is easy to notice that the number of tasks in the Ningbo Port benchmark is quite big. To systematically test the performance of VNS-RLS on diverse instances, we have generated two more datasets scaled down to 25% and 50% respectively by extracting tasks

Table 3 The list of real-life instances of the Ningbo Port.

Instance	No. of shifts	No. of tasks
NP4-1	4	465
NP4-2	4	405
NP4-3	4	526
NP4-4	4	565
NP4-5	4	765
NP6-1	6	1073
NP6-2	6	920
NP6-3	6	384
NP6-4	6	746
NP6-5	6	557
NP8-1	8	913
NP8-2	8	827
NP8-3	8	786
NP8-4	8	1008
NP8-5	8	798

Table 4 The list of artificial instances.

Instance	Configuration	No. of Shifts	No. of tasks
LB4-1	Loose, Balanced	4	484
LB4-2	Loose, Balanced	4	396
TB4-3	Tight, Balanced	4	282
TB4-4	Tight, Balanced	4	368
LU4-5	Tight, Unbalanced	4	448
LU4-6	Tight, Unbalanced	4	479
TU4-7	Loose, Unbalanced	4	217
TU4-8	Loose, Unbalanced	4	354
LB8-1	Loose, Balanced	8	592
LB8-2	Loose, Balanced	8	657
TB8-3	Tight, Balanced	8	497
TB8-4	Tight, Balanced	8	621
LU8-5	Tight, Unbalanced	8	551
LU8-6	Tight, Unbalanced	8	559
TU8-7	Loose, Unbalanced	8	607
TU8-8	Loose, Unbalanced	8	525
Large	Mixed, Unbalanced	8	2614

from the complete dataset, keeping the original features (available at http://www.cs.nott.ac.uk/˜psxbc2/OPVRPTW.rar).

4.2. Exact Approach

In *Section 2*, the new mathematical model of OPVRPTW is established. We use CPLEX to solve it, adopting the default parameter setting. In the preliminary experiments, the solver runs on a machine of 16 cores (2.6 GHz), 16 GB memory and 24 hours running time limit. The solver found feasible solutions for 7 out of 16 the 25% scaled down artificial instances and 2 out of 15 real-life instances, while on the other instances ran out of memory. Therefore, we increase the memory resource to 100 GB in the later experiments. The results are reported in Table 5 and Table 6. In these tables, the objective value of a solution is converted into *Heavy-Loaded Distance Rate* (HLDR), which is widely used by logistic companies in practice, see equation (24). Note that the objectives (1) is equivalent to (24).

$$HLDR = Loaded\ Distance\ /\ (Loaded\ Distance + Unloaded\ Distance) \qquad (24)$$

By using the HLDR, the problem becomes a maximization problem. In Tables 5 and 6, **NF** means no solution is found. **Bound** is the upper bound of HLDR obtained by CPLEX, while **Time** is the actual runtime. + indicates that the memory is insufficient to generate a feasible solution, and * represents that the search reached the memory limit while a feasible solution is obtained.

Table 5 CPLEX solutions on the 25% scaled down real-life instances.

	NP4-1	NP4-2	NP4-3	NP4-4	NP4-5
HLDR	78.36%	65.14%	64.83%	54.39%	NF
Bound	92.36%	97.04%	100%	97.72%	100%
Time (hr)	24	24	24	24	24
	NP6-1	**NP6-2**	**NP6-3**	**NP6-4**	**NP6-5**
HLDR	NF	NF	54.30%	NF	66.11%
Bound	NF	NF	95.20%	NF	98.39%
Time (hr)	24	24	24	+	24
	NP8-1	**NP8-2**	**NP8-3**	**NP8-4**	**NP8-5**
HLDR	NF	NF	NF	NF	NF
Bound	98.98%	100%	100%	NF	100%
Time (hr)	24	24	24	+	24

Table 6 CPLEX solutions on the 25% scaled down artificial instances.

	LB4-1	LB4-2	TB4-3	TB4-4	LU4-5	LU4-6	TU4-7	TU4-8
HLDR	66.62%	76.41%	69.91%	69.30%	NF	58.65%	50.37%	55.36%
Bound	100%	94.87%	86.31%	83.51%	79.94%	73.90%	52.17%	66.38%
Time (hr)	24	24	13*	19*	24	24	1*	24
	LB8-1	**LB8-2**	**TB8-3**	**TB8-4**	**LU8-5**	**LU8-6**	**TU8-7**	**TU8-8**
HLDR	NF	NF	56.85%	52.40%	57.42%	NF	47.65%	50.74%
Bound	100%	100%	82.33%	88.75%	78.33%	86.84%	71.59%	70.43%
Time (hr)	24	24	24	24	24	24	24	24

It can be found that, given a large amount of computing resources, it is still hard to obtain optimal solutions on all the 25% scaled down instances. Except TU4-7, on other instances, the gaps between the obtained solutions and the upper bounds are still large. It is no doubt that there must be other exact methods may find better solution than CPLEX on this benchmark. However, it still can be concluded that, exact search would be time and resource-consuming in this large scale and nonlinearly constrained problem. The results obtained on the 25% instances provide the upper bounds of the optimal solutions, which can be used to estimate the performance of our proposed heuristic algorithms.

4.3. Construction Scheme Selection

Completing all tasks on time with the limited vehicle resource is a major issue to logistic companies. In the Ningbo Port, the number of trucks is relatively sufficient. But when the number of tasks is large, efficiently using the limited vehicles becomes critical. In

Section 3.2, three insertion selection tactics are proposed for the solution construction. Nine different combinations of tactics are tested on the 100% benchmark, results presented in Table 7.

Table 7 **Comparison of nine different combinations of insertion selection tactics.**

Constructive Heuristic	Mandatory Task Tactic	Optional Task Tactic	Average HLDR	Average Time (s)
1	Greedy	Greedy	59.16%	3,576
2	Greedy	First-Insertion	56.40%	445
3	Greedy	One-Route	58.54%	710
4	First-Insertion	Greedy	58.60%	2,302
5	First-Insertion	First-Insertion	56.16%	705
6	First-Insertion	One-Route	57.99%	753
7	One-Route	Greedy	59.16%	3,402
8	One-Route	First-Insertion	56.64%	599
9	One-Route	One-Route	58.54%	620

Among these nine heuristics in Table 7, the three tactics are applied to mandatory tasks and optional tasks respectively. The runtime is obtained on a PC with i7 CPU 3.2GHz and 6 GB memory. It can be found that, heuristics 1 and 7 obtain the best solution quality (the highest average HLDR), while the runtime are significantly higher than the other heuristics. Besides, when Greedy is used on mandatory tasks and First-Insertion is applied to optional tasks (heuristic 2), the fastest constructive speed is obtained, while the average HLDR is not the worst. Applying First-Insertion to both mandatory and optional tasks (heuristic 5) does not bring the fastest heuristic.

The details of constructed solution quality is present in Tables 8 and 9. We can find that, to optional tasks, the heuristics of Greedy (heuristics 1, 4 and 7) obtain significantly better objective value than those of First-Insertion (heuristics 2, 5 and 8). The HLDR of tactic One-Route group (heuristics 3, 6 and 9) is between the above-mentioned two groups. However, to mandatory tasks, no tactic shows obvious better performance than the others. It can be concluded that, the tactic used on optional tasks is the key issue to a constructive heuristic's performance. This is because, in practice, optional tasks always account for a major proportion of total unassigned tasks in one shift, and also mandatory tasks have less available insertion options due to their tight time constraints. Therefore, optional tasks have greater influence on solution construction.

These constructive heuristics have not shown bias on any specific features of instances. Overall, the obtained HLDR values span from 46.79% to 69.93%, higher than 50% on most instances. It should be noted that, two pairs of heuristics (1, 7) and (3, 9) have constructed the same solutions while heuristics 7 and 9 spend less computation time. When

Table 8 **HLDR of initial solutions constructed with nine construction heuristics (on 100% real-life instances). Best obtained HLDR in bold.**

Constructive Heuristic	1	2	3	4	5	6	7	8	9
NP4-1	60.14%	**64.10%**	58.64%	60.11%	61.69%	57.63%	60.14%	61.88%	58.64%
NP4-2	**58.42%**	54.13%	58.36%	**58.42%**	54.60%	58.36%	**58.42%**	54.60%	58.36%
NP4-3	**60.51%**	55.88%	59.22%	59.48%	54.73%	57.96%	**60.51%**	55.46%	59.22%
NP4-4	**56.37%**	50.86%	**56.37%**	**56.37%**	50.29%	**56.37%**	**56.37%**	50.29%	**56.37%**
NP4-5	**69.93%**	57.45%	69.68%	**69.93%**	59.98%	69.68%	**69.93%**	59.98%	69.68%
NP6-1	**62.28%**	54.90%	62.18%	62.03%	56.64%	62.06%	**62.28%**	56.74%	62.18%
NP6-2	60.10%	54.94%	**60.18%**	60.06%	54.55%	60.14%	60.10%	54.58%	**60.18%**
NP6-3	**49.66%**	46.79%	49.60%	**49.66%**	47.17%	49.60%	**49.66%**	47.17%	49.60%
NP6-4	**66.99%**	59.16%	63.20%	64.59%	56.92%	62.78%	**66.99%**	57.52%	63.20%
NP6-5	**56.90%**	54.82%	56.48%	56.39%	55.58%	55.52%	**56.90%**	56.22%	56.48%
NP8-1	64.05%	**64.56%**	63.19%	61.07%	62.08%	60.99%	64.05%	64.38%	63.19%
NP8-2	**64.83%**	60.59%	64.74%	63.42%	59.10%	63.21%	**64.83%**	61.01%	64.74%
NP8-3	68.56%	58.45%	68.22%	**68.98%**	58.39%	68.63%	68.56%	58.22%	68.22%
NP8-4	**57.27%**	55.53%	55.74%	57.16%	**57.27%**	55.88%	**57.27%**	56.83%	55.74%
NP8-5	55.42%	55.57%	55.35%	54.88%	54.96%	54.90%	55.42%	**55.58%**	55.35%

Table 9 **HLDR of initial solutions constructed with nine construction heuristics (on 100% artificial instances). Best obtained HLDR in bold.**

Constructive Heuristic	1	2	3	4	5	6	7	8	9
LB4-1	**59.79%**	54.41%	58.76%	59.42%	53.78%	58.75%	**59.79%**	54.39%	58.76%
LB4-2	**67.30%**	65.07%	66.29%	67.28%	66.92%	66.83%	**67.30%**	65.93%	66.29%
TB4-3	63.14%	63.06%	60.57%	**63.32%**	62.13%	58.95%	63.14%	63.06%	60.57%
TB4-4	**58.19%**	57.67%	57.95%	57.89%	57.55%	57.74%	**58.19%**	57.84%	57.95%
LU4-5	53.84%	52.66%	**53.89%**	51.88%	51.36%	52.68%	53.84%	52.69%	**53.89%**
LU4-6	**61.35%**	59.67%	60.69%	53.79%	52.95%	53.57%	**61.35%**	60.66%	60.69%
TU4-7	48.12%	**48.14%**	47.93%	48.10%	48.11%	47.66%	48.12%	**48.14%**	47.93%
TU4-8	**53.20%**	**53.20%**	52.97%	52.79%	52.66%	52.49%	**53.20%**	**53.20%**	52.97%
LB8-1	**58.32%**	54.64%	**58.32%**	**58.32%**	54.81%	**58.32%**	**58.32%**	54.81%	**58.32%**
LB8-2	65.88%	61.10%	**65.88%**	**65.88%**	61.08%	**65.88%**	65.88%	61.08%	**65.88%**
TB8-3	**59.93%**	59.67%	59.69%	58.74%	58.35%	58.43%	**59.93%**	59.67%	59.69%
TB8-4	54.73%	54.48%	53.59%	54.83%	**55.42%**	54.11%	54.73%	55.07%	53.59%
LU8-5	59.67%	59.70%	57.81%	**62.90%**	62.89%	60.59%	59.67%	59.70%	57.81%
LU8-6	49.63%	49.39%	48.87%	50.77%	**51.00%**	48.66%	49.63%	50.10%	48.87%
TU8-7	56.64%	56.28%	**57.65%**	56.59%	56.29%	57.63%	56.64%	56.28%	**57.65%**
TU8-8	**52.65%**	**52.65%**	**52.65%**	51.70%	51.70%	51.56%	**52.65%**	**52.65%**	**52.65%**

the candidate tasks are few, if the existing routes are too tight to insert other tasks, tactic Greedy can only assign one task to one route one by one. This may also happen to tactic One-Route, but its computation time is less since it only evaluate one existing route. As there are few mandatory tasks which are always assigned first in each shift, and there is no randomization in this heuristic, applying tactic Greedy or One-Route to mandatory tasks may generate the same partial routes. In this case, if the tactics applied to optional tasks are the same, their final routes (solutions) generated would be the same.

In metaheuristics, the quality of initial solutions does not always directly determine the quality of final improved solutions. Our study also find that better initial solutions cannot guarantee better final solutions in VNS-RLS. Considering all the nine heuristics can produce feasible solutions on the benchmark instances, in normal cases, the construction heuristic 2 is recommended, as it has the fastest construction speed generally.

The two main constraints of OPVRPTW are time window constraints and the available trucks. To find out the recommended heuristic in tight constrain scenario, we compare the performances of the nine constructive heuristics on the big instance of 2,614 tasks, results presented in Table 10. It can be found that, when tactic First-Insertion is applied to optional tasks (heuristics 2, 5 and 8), the quality of solutions is poor (lower HLDR and more violations). Heuristics 1 and 7 produce the same best solutions, while heuristic 7 takes less computation time. Heuristic 7 is thus recommended when facing this kind of very large instances. The other solutions with acceptable quality are generated when tactic Greedy or One-Route is applied to optional tasks (heuristics 3, 4, 6 and 9).

Table 10 Quality of solutions on large instances.

Constructive Heuristic	HLDR	Time (hr)	Additional Trucks Requirement	No. of Failed Tasks
1	73.38%	35.7	0	56
2	65.49%	23.1	116	98
3	73.24%	30.0	4	59
4	73.18%	30.3	0	67
5	65.01%	22.1	100	118
6	73.31%	22.8	1	57
7	73.38%	28.5	0	56
8	65.00%	22.7	98	118
9	73.24%	23.2	4	59

4.4. Impact of the Sampling

To evaluate the contribution of *Sampling* introduced in *Section 3.4*, a variant of removing the *Sampling* scheme from VNS-RLS and assigning all candidate operators an equal initial weights is implemented. This variant of standard VNS algorithm with Reinforcement Learning (VNS-RL) is tested on different sizes of benchmark instances. The results of 30 runs on 25% scaled down instances are presented in Appendix Tables 14 and 15. It can be summarized that VNS-RLS performs better than VNS-RL in both the best found and average HLDR in 19/31 of all instances, but more evaluations were conducted. In 26/31 instances, both VNS-RLS and VNS-RL obtain better results than CPLEX with previously mentioned configuration.

Results of large 100% instances are presented in Appendix Tables 16 and 17. In 16/31 instances, solutions with better best found and average HLDR are found by VNS-RLS. In (Bai et al. 2015), the HLDR for the relax problem removing the travels from and to the depot are calculated as the upper bounds of the benchmark. These upper bounds are used as references in these two tables. It can be found that, the gaps between the obtained solutions and upper bounds are less than 5% on 14 instances. Especially, in the instances

of NP4-2, NP4-5, NP6-4, NP8-4 and TU8-8, the objective values of solutions generated are very close to the upper bounds.

However, the contribution of *Sampling* on improving the objective value is not significant. The t-test results on VNS-RLS and VNS-RL (see Tables 11 and 12) show that the improvement is significant on only 25% instances. This observation indicates that, by sampling on the search trajectory of single solution-based algorithm, the guidance collected may cannot properly reflect the neighbourhood environment as the search going. Correspondingly, the guidance information provided cannot help the search when the search region changed. In other words, limited *Sampling* points in Single Solution-Based approaches is not so helpful in improving objective value comparing to its impact in Population-Based approaches. The further discussion on the impact of Sampling indicators is presented in *Section 4.5*.

Table 11 T-test between VNS-RLS and VNS-RL on 25% scaled down and 100% real-life instances. (Y means significantly different, while N means not.)

	NP4-1		NP4-2		NP4-3		NP4-4		NP4-5	
Instance Scale	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**
T-test result	N	Y	N	N	N	Y	N	N	N	N
	NP6-1		**NP6-2**		**NP6-3**		**NP6-4**		**NP6-5**	
Instance Scale	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**
T-test result	N	N	N	N	N	Y	Y	N	N	N
	NP8-1		**NP8-2**		**NP8-3**		**NP8-4**		**NP8-5**	
Instance Scale	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**
T-test result	Y	Y	N	N	N	N	N	N	Y	N

Table 12 T-test between VNS-RLS and VNS-RL on 25% scaled down and 100% artificial instances.

	LB4-1		LB4-2		TB4-3		TB4-4		LU4-5		LU4-6		TU4-7		TU4-8	
Instance Scale	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**
T-test result	N	N	N	N	Y	N	N	Y	N	N	Y	N	N	N	N	N
	LB8-1		**LB8-2**		**TB8-3**		**TB8-4**		**LU8-5**		**LU8-6**		**TU8-7**		**TU8-8**	
Instance Scale	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**	**25%**	**100%**
T-test Result	Y	N	Y	Y	N	N	N	N	Y	N	N	N	N	N	N	Y

The same observation is found on the medium size 50% scaled down instances (see Appendix Tables 18 and 19). VNS-RLS significantly outperforms VNS-RL on 16 out of 31 instances, while higher stability (smaller standard deviation) is obtained on 17 instances. In this single solution-based algorithm, no significant objective value improvement is achieved by adopting *Sampling*.

4.5. Feasibility Indicators & Feasible Solution Space Exploration

In *Sampling* and *Local Search*, feasibility measure is employed aiming to improve the search efficiency. The hypothesis is that, with using the feasibility indicators, the infeasible solution rate during the search will decrease. To validate this hypothesis, the feasibility indicators in *Sampling* (E^c) and *LocalSearch* (FI) are removed respectively leading to three new algorithm variants (VNS-RLS-NoEc, VNS-RLS-NoFI, VNS-RL-NoFI), detailed results presented in Appendix Tables 14 - 19. Comparing the best and average solutions, VNS-RLS-NoFI produces the most best results on medium and large instances, while VNS-RLS-NoEc outperforms the other variants on small instances.

We record the amount of infeasible solutions over every 10,000 evaluations during the search and present a typical infeasible rate changing trend chart of the five algorithm variants, see *Figure 3*. It is easy to find that, without *Sampling*, both VNS-RL and VNS-RL-NoFI have higher infeasible rates than the other three variants throughout of the search. The infeasible rates of the three VNS-RLS variants decrease at the beginning of search and stay at a lower level afterwards, which shows the strong power of Sampling on reducing infeasible neighbourhood moves. In each VNS iteration, by estimating the surrounding environment of the starting solution, Sampling leads Local Search to the search regions with high feasibility. Using feasibility indicators in both Sampling and Local Search, VNS-RLS achieves the lowest infeasible rate.

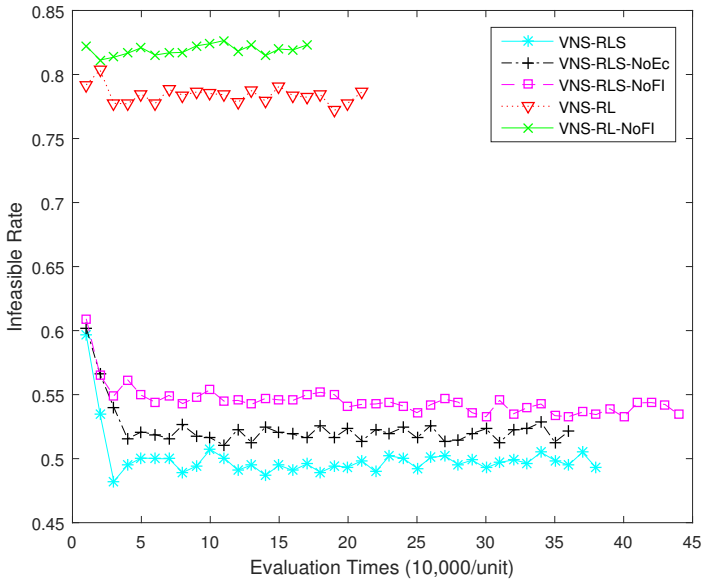

Figure 3 **Comparison of infeasible rates on the 50% NP4-3 instance.**

In addition, in *Figure 3*, the two lines of VNS-RL and VNS-RL-NoFI are shorter than the other three VNS-RLS variants. It indicates the two variants converge earlier and less evaluation times. Without the guidance to from Sampling, it is harder to find better feasible solutions in the search space with lower feasible rate, and the two VNS-RL variants terminate earlier thus.

It can be concluded that, both *Sampling* and *Feasibility indicator* can reduce the infeasible rate of neighbourhood moves, while *Sampling* brings a greater impact. *Sampling*, even if without the feasibility indicator E^c in it, can significantly decrease the infeasible rate, which indicates that the other two indicators (E^a and E^b) apt to guide the search to feasible solution regions as well. The use of feasibility indicator in *Sampling* and *Local Search* increases the probability feasible neighbourhood moves.

It should be noted that, the lowest infeasible rate does not represent the best objective function value. The removal of feasibility indicators may also increase the search flexibility and increase the diversification of search, thus improves the search performance, e.g. on the LB8-1 100% instance, VNS-RLS-NoFI is the only one variant which finds the solution with objective value 87.49%. It is almost 2% higher than the other four variants. Overall, VNS-RLS-NoEc and VNS-RLS-NoFI produce the most best found solutions and average solutions in all the five variants.

4.6. Comparison with State-of-the-art Algorithms

OPVRPTW is a new model in the community, thus there is no existing algorithms can be applied directly. Considering the similarity between PVRP and OPVRPTW, in our research, two state-of-the-art methodologies for PVRP, RVNS (Pirkwieser and Raidl 2008) and FVNS (Hemmelmayr et al. 2009), are modified and applied to OPVRPTW for comparison. Both RVNS and FVNS use the VNS framework as well. The main difference between them is that, apart from the neighbourhood structures used are different, the shaking operators are randomly selected in RVNS while they are applied in a fixed sequence in FVNS. Because of the time constraints and multiple period feature in OPVRPTW, those algorithms for OVRP and OVRPTW are either inapplicable or not suitable to OPVRPTW, thus they are not compared in this study.

Generally, RVNS and FVNS generate similar results on the benchmark. Comparing with VNS-RLS-NoEc, their results are worse on all categories of instances, while the deterioration is presented in Table 13. VNS-RLS-NoEc outperforms the two algorithms on both the

Table 13 Result deterioration comparing with VNS-RLS.

		NP4	NP6	NP8	LB4	LB8	TB4	TB8	LU4	LU8	TU4	TU8
RVNS	Best	-1.77%	-2.36%	-1.73%	-3.42%	-5.66%	-1.00%	-2.58%	-1.58%	-1.39%	-0.47%	-0.55%
	Average	-2.23%	-3.43%	-2.11%	-4.21%	-6.62%	-0.74%	-2.19%	-2.52%	-2.02%	-0.42%	-0.80%
	S.D.	-0.33%	-0.50%	-0.34%	-0.49%	-0.51%	0.06%	0.20%	-0.20%	-0.21%	0.15%	0.04%
FVNS	Best	-1.86%	-2.26%	-1.64%	-2.67%	-5.59%	-1.55%	-3.16%	-2.07%	-1.99%	-0.47%	-0.73%
	Average	-2.53%	-3.94%	-2.58%	-4.65%	-7.21%	-0.94%	-2.27%	-2.83%	-2.50%	-0.48%	-0.83%
	S.D.	-0.35%	-0.60%	-0.50%	-0.71%	-0.71%	0.20%	0.24%	-0.15%	-0.11%	0.16%	0.07%

quality of solutions generated and the algorithm stability. The reasons for this may include lacking of neighbourhood structures with diverse degree of perturbation and efficient search guiding scheme. The operators in VNS-RLS bring the perturbations at different levels, cooperating with the proposed reinforcement learning scheme and the proper indicators, thus exploration and exploitation are better balanced in the search space. The solution deterioration is smaller on tight and unbalance instances. This indicates that, when facing the larger search space caused by loose time windows and balanced workload, VNS-RLS has stronger search ability. However, RVN and FVNS show better stability on tight instances (TB4, TB8, TU4 and TU8) than VNS-RLS.

5. Conclusions

An Open Period Vehicle Routing Problem with Time Windows (OPVRPTW) arising from real-life container transportation problem is modeled in this paper. The node-based mathematical problem model is established by combining service activities into task nodes. This is a problem with nonlinear constraints and huge solution space. The experiments on both real-life and artificial benchmarks show that, it is unrealistic to address this problem with exact methods, even if large numbers of computation resources are given. An approximate method Variable Neighbourhood Search algorithm with Reinforcement Learning and Sampling (VNS-RLS) is developed for OPVRPTW in this study.

In VNS-RLS, an urgency level-based insertion heuristic is devised to construct the initial feasible solutions. After classifying tasks according to their urgency levels (mandatory and optional), nine different insertion selection tactic combinations are investigated and compared. Experiment results show that the tactic applied to optional tasks mainly determines the heuristic's performance. When the resource of trucks is sufficient, applying the *First-Insertion* tactic to optional tasks can obtain the fastest constructing speed. When the amount of tasks is extremely large, using *One-Route* tactic on mandatory tasks and *Greedy* tactic on optional tasks is recommended to produce the better solution within less computation time.

A *Sampling* scheme is proposed to investigate the neighbourhood surrounding the starting solution of Local Search, providing an initial search direction for the Local Search with Reinforcement Learning. Experiment results show that its contribution to objective value improvement is not significant. This indicates that, to single solution-based metaheuristics, the neighbourhood environment of the search trajectory might change dramatically during the search, thus the validity period of the guidance provided by Sampling would be short. In this case, Sampling is not so powerful on objective value improvement like it is in population-based approaches.

To reduce the infeasible solutions generated in the search, feasibility indicator is used in both phases of Sampling and Local Search. The study shows that, use of Sampling greatly reduces the infeasible rate of solutions generated, while the feasibility indicators make further contribution as well. Comparing to results of exact method and problem upper bounds, the proposed algorithms generate promising results and show strong search ability on instances with large search space.

On several instances, the gaps between the obtained objective values and upper bounds are still large. Considering the operators used in VNS-RLS bring relatively small changes to solutions, techniques with larger perturbation and higher diversification (e.g. Large Neighbourhood Search) are worthy to try and investigate in our future work. In addition, more practical objectives and constraints in real-life problems, such as the balance of workload and carbon emission, may also be considered in this new OPVRPTW model.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (Grant No. 71471092), Zhejiang Natural Science Foundation (Grant No. LR17G010001), Ningbo Science & Technology Bureau (Grandt No. 2014A35006) and School of Computer Science, The University of Nottingham. We are grateful for access to the University of Nottingham High Performance Computing Facility.

References

Bai R, Xue N, Chen J, Roberts GW (2015) A set-covering model for a bidirectional multi-shift full truckload vehicle routing problem. *Transportation Research Part B: Methodological* 79:134–148, ISSN 0191-2615.

Baker BM, Ayechew MA (2003) A genetic algorithm for the vehicle routing problem. *Computers & Operations Research* 30(5):787–800, ISSN 0305-0548.

Brandão J (2004) A tabu search algorithm for the open vehicle routing problem. *European Journal of Operational Research* 157(3):552–564.

Bräysy O, Gendreau M (2001) Metaheuristics for the vehicle routing problem with time windows. *Report STF42 A* 1025.

Bräysy O, Gendreau M (2005) Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation science* 39(1):119–139.

Brito J, Martínez FJ, Moreno J, Verdegay JL (2015) An aco hybrid metaheuristic for close–open vehicle routing problems with time windows and fuzzy constraints. *Applied Soft Computing* 32:154–163.

Burke EK, Gendreau M, Ochoa G, Walker JD (2011) Adaptive iterated local search for cross-domain optimisation. *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, 1987–1994 (ACM), ISBN 1450305571.

Chen B, Qu R, Bai R, Ishibuchi H (2016) A variable neighbourhood search algorithm with compound neighbourhoods for vrptw. *Proceedings of the 5th International Conference on Operations Research and Enterprise Systems (ICORES 2016), Rome, Italy*, 25–35 (SCITEPRESS).

Chen J, Bai R, Qu R, Kendall G (2013) A task based approach for a real-world commodity routing problem. *Computational Intelligence In Production And Logistics Systems (CIPLS), 2013 IEEE Workshop on*, 1–8 (IEEE).

Clarke Gu, Wright JW (1964) Scheduling of vehicles from a central depot to a number of delivery points. *Operations research* 12(4):568–581, ISSN 0030-364X.

Cordeau JF, Laporte G, Mercier A (2001) A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational research society* 52(8):928–936, ISSN 0160-5682.

Cordeau JF, Laporte G, Mercier A (2004) Improved tabu search algorithm for the handling of route duration constraints in vehicle routing problems with time windows. *Journal of the Operational Research Society* 55(5):542–546, ISSN 0160-5682.

Danandeh A, Ghazanfari M, Tavakoli-Moghaddam R, Alinaghian M (2010) A swift heuristic algorithm based on capacitated clustering for the open periodic vehicle routing problem. *Proceedings of the 9th WSEAS international conference on Artificial intelligence, knowledge engineering and data bases, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA*, 208–214.

Dantzig GB, Ramser JH (1959) The truck dispatching problem. *Management science* 6(1):80–91.

Dueck G (1993) New optimization heuristics: The great deluge algorithm and the record-to-record travel. *Journal of Computational Physics* 104(1):86–92, ISSN 0021-9991.

Eksioglu B, Vural AV, Reisman A (2009) The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering* 57(4):1472–1483, ISSN 0360-8352.

Eppen G, Schrage L (1981) Centralized ordering policies in a multi-warehouse system with lead times and random demand. *Multi-level production/inventory control systems: Theory and practice* 16:51–67.

Fu Z, Eglese R, Li LY (2005) A new tabu search heuristic for the open vehicle routing problem. *Journal of the operational Research Society* 56(3):267–274.

Gehring H, Homberger J (1999) A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. *Proceedings of EUROGEN99*, volume 2, 57–64 (Citeseer).

Gendreau M, Potvin JY, Bräumlaysy O, Hasle G, Løkketangen A (2008) Metaheuristics for the vehicle routing problem and its extensions: A categorized bibliography. *The vehicle routing problem: latest advances and new challenges* 143–169.

Gillett BE, Miller LR (1974) A heuristic algorithm for the vehicle-dispatch problem. *Operations research* 22(2):340–349, ISSN 0030-364X.

Golden B, Assad A, Levy L, Gheysens F (1984) The fleet size and mix vehicle routing problem. *Computers & Operations Research* 11(1):49–66, ISSN 0305-0548.

Golden BL, Raghavan S, Wasil EA (2008) *The Vehicle Routing Problem: Latest Advances and New Challenges: latest advances and new challenges*, volume 43 (Springer Science & Business Media), ISBN 0387777784.

Guiyun L (2009a) An improved ant colony algorithm for open vehicle routing problem with time windows. *Information Management, Innovation Management and Industrial Engineering, 2009 International Conference on*, volume 2, 616–619 (IEEE), ISBN 0769538762.

Guiyun L (2009b) Research on open vehicle routing problem with time windows based on improved genetic algorithm. *Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on*, 1–5 (IEEE), ISBN 1424445078.

Hansen P, Mladenovi N, Prez JAM (2010) Variable neighbourhood search: methods and applications. *Annals of Operations Research* 175(1):367–407, ISSN 0254-5330.

Hemmelmayr VC, Cordeau JF, Crainic TG (2012) An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Computers & Operations Research* 39(12):3215–3228, ISSN 0305-0548.

Hemmelmayr VC, Doerner KF, Hartl RF (2009) A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research* 195(3):791–802, ISSN 0377-2217.

Ioannou G, Kritikos M, Prastacos G (2001) A greedy look-ahead heuristic for the vehicle routing problem with time windows. *Journal of the Operational Research Society* 52(5):523–537, ISSN 0160-5682.

Jin Y (2005) A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing-A Fusion of Foundations, Methodologies and Applications* 9(1):3–12.

Jozefowiez N, Semet F, Talbi EG (2008) Multi-objective vehicle routing problems. *European journal of operational research* 189(2):293–309.

Kaji T, Ohuchi A (1999) A simulated annealing algorithm with the random compound move for the sequential partitioning problem of directed acyclic graphs. *European Journal of Operational Research* 112(1):147–157, ISSN 0377-2217.

Laporte G, Gendreau M, Potvin JY, Semet F (2000) Classical and modern heuristics for the vehicle routing problem. *International transactions in operational research* 7(4-5):285–300, ISSN 1475-3995.

Lenstra JK, Kan A (1981) Complexity of vehicle routing and scheduling problems. *Networks* 11(2):221–227.

Letchford AN, Lysgaard J, Eglese RW (2007) A branch-and-cut algorithm for the capacitated open vehicle routing problem. *Journal of the Operational Research Society* 58(12):1642–1651.

Li F, Golden B, Wasil E (2007) The open vehicle routing problem: Algorithms, large-scale test problems, and computational results. *Computers & Operations Research* 34(10):2918–2930, ISSN 0305-0548.

Lin Q, Liu Z, Yan Q, Du Z, Coello CAC, Liang Z, Wang W, Chen J (2016) Adaptive composite operator selection and parameter control for multiobjective evolutionary algorithm. *Information Sciences* 339:332–352, ISSN 0020-0255.

Lin S (1965) Computer solutions of the traveling salesman problem. *Bell System Technical Journal, The* 44(10):2245–2269, ISSN 0005-8580.

Liu R, Jiang Z (2012) The close–open mixed vehicle routing problem. *European Journal of Operational Research* 220(2):349–360.

Lourens T (2005) *Using population-based incremental learning to optimize feasible distribution logistic solutions*. Thesis.

Mourgaya M, Vanderbeck F (2007) Column generation based heuristic for tactical planning in multi-period vehicle routing. *European Journal of Operational Research* 183(3):1028–1041, ISSN 0377-2217.

Or I (1976) *Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking* (Xerox University Microfilms).

Park J, Kim BI (2010) The school bus routing problem: A review. *European Journal of operational research* 202(2):311–319, ISSN 0377-2217.

Perwira Redi A, Maghfiroh MF, Yu VF (2013) An improved variable neighborhood search for the open vehicle routing problem with time windows. *Industrial Engineering and Engineering Management (IEEM), 2013 IEEE International Conference on*, 1641–1645 (IEEE).

Pirkwieser S, Raidl GR (2008) A variable neighborhood search for the periodic vehicle routing problem with time windows. *Proceedings of the 9th EU/meeting on metaheuristics for logistics and vehicle routing, Troyes, France*, 23–24.

Pisinger D, Ropke S (2007) A general heuristic for vehicle routing problems. *Computers & operations research* 34(8):2403–2435, ISSN 0305-0548.

Potvin JY, Kervahut T, Garcia BL, Rousseau JM (1996) The vehicle routing problem with time windows part i: tabu search. *INFORMS Journal on Computing* 8(2):158–164, ISSN 1091-9856.

Potvin JY, Rousseau JM (1995) An exchange heuristic for routeing problems with time windows. *Journal of the Operational Research Society* 46(12):1433–1446, ISSN 0160-5682.

Rahimi-Vahed A, Gabriel Crainic T, Gendreau M, Rei W (2015) Fleet-sizing for multi-depot and periodic vehicle routing problems using a modular heuristic algorithm. *Computers & Operations Research* 53:9–23, ISSN 0305-0548.

Repoussis PP, Tarantilis CD, Ioannou G (2007) The open vehicle routing problem with time windows. *Journal of the Operational Research Society* 58(3):355–367, ISSN 0160-5682.

Ropke S, Pisinger D (2006a) An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science* 40(4):455–472, ISSN 0041-1655.

Ropke S, Pisinger D (2006b) A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research* 171(3):750–775.

Savelsbergh MW (1992) The vehicle routing problem with time windows: Minimizing route duration. *ORSA journal on computing* 4(2):146–154.

Schopka K, Kopfer H (2016) *An Adaptive Large Neighborhood Search for the Reverse Open Vehicle Routing Problem with Time Windows*, 243–257 (Springer), ISBN 3319208624.

Smith JE, Fogarty TC (1997) Operator and parameter adaptation in genetic algorithms. *Soft computing* 1(2):81–87, ISSN 1432-7643.

Solomon MM (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research* 35(2):254–265, ISSN 0030-364X.

Soria Alcaraz JA, Ochoa G, Carpio M, Puga H (2014) Evolvability metrics in adaptive operator selection. *Proceedings of the 2014 conference on Genetic and evolutionary computation*, 1327–1334 (ACM), ISBN 1450326625.

Taillard , Badeau P, Gendreau M, Guertin F, Potvin JY (1997) A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation science* 31(2):170–186, ISSN 0041-1655.

Tarantilis CD, Ioannou G, Kiranoudis CT, Prastacos GP (2004) A threshold accepting approach to the open vehicle routing problem. *RAIRO-Operations Research* 38(04):345–360, ISSN 1290-3868.

Tarantilis CD, Ioannou G, Kiranoudis CT, Prastacos GP (2005) Solving the open vehicle routeing problem via a single parameter metaheuristic algorithm. *Journal of the Operational Research Society* 56(5):588–596, ISSN 0160-5682.

Thathachar M, Sastry PS (2002) Varieties of learning automata: an overview. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 32(6):711–722, ISSN 1083-4419.

Thierens D (2005) An adaptive pursuit strategy for allocating operator probabilities. *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, 1539–1546 (ACM), ISBN 1595930108.

Thompson PM, Orlin JB, et al. (1989) The theory of cyclic transfers .

Toth P, Vigo D (2001) *The vehicle routing problem* (Siam), ISBN 0898718511.

Veerapen N, Maturana J, Saubion F (2012) An exploration-exploitation compromise-based adaptive operator selection for local search. *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, 1277–1284 (ACM), ISBN 1450311776.

Vidal T, Crainic TG, Gendreau M, Prins C (2014) A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research* 234(3):658–673.

Vigo D (1996) A heuristic algorithm for the asymmetric capacitated vehicle routing problem. *European Journal of Operational Research* 89(1):108–126.

Wang X, Regan AC (2002) Local truckload pickup and delivery with hard time window constraints. *Transportation Research Part B: Methodological* 36(2):97–112, ISSN 0191-2615.

Wieberneit N (2008) Service network design for freight transportation: a review. *OR spectrum* 30(1):77–112, ISSN 0171-6468.

Yu B, Yang ZZ (2011) An ant colony optimization model: The period vehicle routing problem with time windows. *Transportation Research Part E: Logistics and Transportation Review* 47(2):166–181, ISSN 1366-5545.

Zachariadis EE, Kiranoudis CT (2010) An open vehicle routing problem metaheuristic for examining wide solution neighborhoods. *Computers & Operations Research* 37(4):712–723.

Zhang R, Yun WY, Kopfer H (2010) Heuristic-based truck scheduling for inland container transportation. *OR spectrum* 32(3):787–808, ISSN 0171-6468.

Appendix. Detailed Results of Experiments

Table 14 Comparison on 25% scaled down real-life instances. Times represents the average evaluation time, while S.D. is the standard deviation of obtained HLDR values. Best heuristic solutions are in bold.

Instance		NP4-1	NP4-2	NP4-3	NP4-4	NP4-5
VNS-RLS	Best	**82.89%**	62.17%	75.78%	60.50%	79.45%
	Average	80.88%	61.19%	75.00%	59.28%	78.62%
	Times	387,006	285,553	298,205	292,664	412,737
	S.D.	1.05%	0.53%	0.64%	0.65%	0.41%
VNS-RL	Best	**82.89%**	62.42%	75.74%	60.52%	79.14%
	Average	80.29%	61.28%	74.88%	59.34%	78.56%
	Times	169,470	134,063	149,373	134,128	199,220
	S.D.	1.34%	0.60%	0.60%	0.56%	0.37%
VNS-RLS No Ec	Best	**82.89%**	62.32%	75.64%	59.76%	79.24%
	Average	81.51%	61.42%	74.92%	59.18%	78.48%
	Times	469,233	311,885	319,202	347,134	326,956
	S.D.	1.16%	0.60%	0.62%	0.35%	0.42%
VNS-RLS No Feasibility Indicator	Best	82.85%	**62.91%**	75.98%	**61.12%**	**79.61%**
	Average	**80.89%**	**61.59%**	74.93%	**59.42%**	**78.71%**
	Times	442,311	320,768	306,145	310,331	425,826
	S.D.	0.81%	0.53%	0.63%	0.75%	0.43%
VNS-RL No Feasibility Indicator	Best	81.50%	61.95%	**76.11%**	59.88%	79.18%
	Average	79.70%	61.16%	**75.19%**	59.27%	78.41%
	Times	153,843	145,666	171,405	154,865	153,056
	S.D.	1.22%	0.42%	0.69%	0.43%	0.48%
CPLEX		78.36%	65.14%	64.83%	54.39%	NF
Upper Bound		92.36%	97.04%	100%	97.72%	100%

Instance		NP6-1	NP6-2	NP6-3	NP6-4	NP6-5
VNS-RLS	Best	76.88%	73.23%	62.48%	**80.56%**	82.56%
	Average	74.85%	72.63%	61.99%	79.70%	80.60%
	Times	615,170	537,578	246,889	564,762	446,762
	S.D.	0.93%	0.39%	%0.29	0.51%	1.39%
VNS-RL	Best	76.31%	73.09%	**62.77%**	80.41%	**83.59%**
	Average	75.01%	72.50%	61.86%	79.35%	81.01%
	Times	305,856	232,937	128,263	267,918	216,844
	S.D.	0.86%	0.32%	0.37%	0.78%	1.44%
VNS-RLS No Ec	Best	76.24%	**73.39%**	62.32%	80.50%	82.44%
	Average	74.99%	**72.83%**	**62.06%**	**79.84%**	80.53%
	Times	698,514	624,078	253,037	541,548	365,435
	S.D.	0.96%	0.41%	0.20%	0.41%	1.72%
VNS-RLS No Feasibility Indicator	Best	**77.61%**	73.35%	**62.77%**	80.41%	82.79%
	Average	**75.52%**	72.70%	61.91%	79.64%	**81.78%**
	Times	653,283	485,454	274,900	497,273	445,694
	S.D.	1.21%	0.37%	0.30%	0.67%	0.92%
VNS-RL No Feasibility Indicator	Best	76.00%	73.13%	62.58%	**80.56%**	83.06%
	Average	74.87%	72.61%	61.99%	79.33%	81.74%
	Times	290,428	232,629	107,558	270,913	196,576
	S.D.	1.10%	0.47%	0.31%	0.69%	0.68%
CPLEX		NF	NF	54.30%	NF	66.11%
Upper Bound		NF	NF	95.20%	NF	98.39%

Instance		NP8-1	NP8-2	NP8-3	NP8-4	NP8-5
VNS-RLS	Best	**77.05%**	77.91%	75.66%	62.26%	72.27%
	Average	**75.35%**	77.00%	75.05%	62.08%	71.56%
	Times	735,909	481,493	388,220	415,835	540,940
	S.D.	0.83%	0.50%	0.34%	0.09%	0.47%
VNS-RL	Best	76.33%	**78.27%**	75.69%	62.25%	72.27%
	Average	74.54%	76.97%	75.03%	62.08%	71.25%
	Times	323,590	234,179	193,351	174,499	228,491
	S.D.	0.96%	0.47%	0.29%	0.11%	0.52%
VNS-RLS No Ec	Best	76.91%	77.76%	75.35%	60.90%	72.27%
	Average	74.72%	77.16%	74.93%	60.47%	**71.68%**
	Times	607,961	525,479	442,103	430,962	516,872
	S.D.	1.20%	0.37%	0.31%	0.32%	0.36%
VNS-RLS No Feasibility Indicator	Best	76.88%	78.08%	75.72%	**62.26%**	72.25%
	Average	74.84%	**77.28%**	74.86%	**62.11%**	71.54%
	Times	656,798	588,599	352,809	354,340	541,743
	S.D.	1.06%	0.37%	0.43%	0.13%	0.40%
VNS-RL No Feasibility Indicator	Best	76.77%	77.66%	**75.82%**	62.21%	**72.31%**
	Average	75.15%	76.88%	**75.19%**	62.06%	71.40%
	Times	323,912	211,260	213,402	186,282	197,478
	S.D.	0.74%	0.49%	0.34%	0.15%	0.46%
CPLEX		NF	NF	NF	NF	NF
Upper Bound		98.98%	100%	100%	NF	100%

Table 15 Comparison on 25% scaled down artificial instances. Times represents the average evaluation time, while S.D. is the standard deviation of obtained HLDR values. Best heuristic solutions are in bold.

Instance		LB4-1	LB4-2	TB4-3	TB4-4	LU4-5	LU4-6	TU4-7	TU4-8
VNS-RLS	Best	76.68%	82.67%	69.28%	66.04%	60.16%	60.92%	**48.75%**	**54.97%**
	Average	74.35	81.54	**68.05%**	64.70%	59.09%	60.64%	**48.56%**	54.52%
	Times	277,040	315,961	273,910	272,605	295,389	278,076	167,317	164,856
	S.D.	0.88%	0.83%	0.75%	0.65%	0.52%	0.22%	0.30%	0.31%
VNS-RL	Best	75.82%	83.08%	68.64%	66.33%	60.31%	60.91%	**48.75%**	**54.97%**
	Average	74.38%	81.33%	67.51%	64.83%	**59.35%**	60.43%	48.43%	54.61%
	Times	134,970	129,368	117,319	133,301	141,104	117,165	75,732	81,194
	S.D.	0.90%	0.91%	0.86%	0.72%	0.48%	0.33%	0.32%	0.29%
VNS-RLS No Ec	Best	**76.92%**	**83.42%**	69.08%	66.41%	**60.71%**	**61.08%**	**48.75%**	**54.97%**
	Average	74.80%	**81.61%**	67.78%	**64.95%**	59.29%	60.62%	48.54%	**54.68%**
	Times	313,707	280,849	286.059	298,651	321,835	290,082	166,248	193,536
	S.D.	0.95%	1.09%	0.65%	0.75%	0.64%	0.29%	0.30%	0.33%
VNS-RLS No Feasibility Indicator	Best	75.59%	**83.42%**	**69.37%**	**66.45%**	60.29%	60.96%	**48.75%**	**54.97%**
	Average	74.40%	81.58%	67.75%	64.79%	59.17%	**60.69%**	48.55%	54.54%
	Times	261,407	333,130	292,332	320,959	297,749	310,406	171,264	185,505
	S.D.	0.57%	1.05%	0.85%	0.77%	0.67%	0.20%	0.31%	0.34%
VNS-RL No Feasibility Indicator	Best	**76.92%**	83.08%	68.78%	66.00%	59.74%	60.87%	**48.75%**	**54.97%**
	Average	**74.85%**	81.58%	67.52%	64.78%	59.27%	60.56%	48.53%	54.55%
	Times	107,783	132,351	132,658	142,240	164,529	131,225	75,934	76,202
	S.D.	1.05%	1.12%	0.76%	0.84%	0.38%	0.24%	0.30%	0.17%
CPLEX		66.62%	76.41%	69.91%	69.30%	NF	58.65%	50.37%	55.36%
Upper Bound		100%	94.87%	86.31%	83.51%	79.94%	73.90%	52.17%	66.38%
Instance		**LB8-1**	**LB8-2**	**TB8-3**	**TB8-4**	**LU8-5**	**LU8-6**	**TU8-7**	**TU8-8**
VNS-RLS	Best	91.28%	93.60%	62.87%	65.28%	**65.76%**	66.37%	**56.72%**	**52.34%**
	Average	89.58%	91.90%	**61.81%**	62.98%	64.78%	65.39%	55.98%	**51.97%**
	Times	538,438	508,352	280,373	448,218	465,806	430,572	319,333	265,267
	S.D.	0.99%	1.07%	0.59%	1.17%	0.54%	0.51%	0.37%	0.21%
VNS-RL	Best	91.25%	93.11%	62.49%	64.26%	65.61%	66.05%	56.54%	52.12%
	Average	88.84%	91.20%	61.74%	62.54%	64.37%	65.29%	55.90%	51.93%
	Times	234,779	217,003	129,619	180,111	187,423	179,388	123,896	124,926
	S.D.	1.42%	1.13%	0.51%	0.90%	0.71%	0.37%	0.35%	0.13%
VNS-RLS No Ec	Best	91.25%	93.56%	**63.05%**	**66.31%**	**65.76%**	**66.58%**	56.46%	52.29%
	Average	**89.76%**	92.09%	61.78%	**63.25%**	**64.86%**	65.58%	55.79%	51.93%
	Times	492,628	547,853	296,837	517,855	438,295	439,782	269,164	281,479
	S.D.	0.95%	0.87%	0.54%	1.16%	0.44%	0.49%	0.29%	0.18%
VNS-RLS No Feasibility Indicator	Best	**91.94%**	**93.70%**	62.62%	65.82%	65.68%	66.03%	56.63%	52.16%
	Average	89.6%	91.88%	61.79%	62.96%	64.79%	65.46%	55.95%	51.94%
	Times	463,923	473,201	287,761	425,639	410,844	363,277	305,785	259,536
	S.D.	1.19%	1.08%	0.51%	0.99%	0.41%	0.41%	0.33%	0.15%
VNS-RL No Feasibility Indicator	Best	90.54%	93.62%	62.62%	63.41%	65.33%	66.37%	56.46%	52.11%
	Average	89.27%	**92.34%**	61.64%	62.49%	64.42%	**65.71%**	**56.01%**	51.94%
	Times	189,883	257,521	134,118	184,205	195,563	200,338	155,815	110,828
	S.D.	0.67%	0.75%	0.78%	0.75%	0.51%	0.35%	0.37%	0.13%
CPLEX		NF	NF	56.85%	52.40%	57.42%	NF	47.65%	50.74%
Upper Bound		100%	100%	82.33%	88.75%	78.33%	86.84%	71.59%	70.43%

Table 16 Comparison on 100% real-life instances. Times represents the average evaluation time, while S.D. is the standard deviation of obtained HLDR values. Best heuristic solutions are in bold.

Instance		NP4-1	NP4-2	NP4-3	NP4-4	NP4-5
VNS-RLS	Best	82.99%	69.78%	73.13%	66.76%	**80.82%**
	Average	**81.88%**	69.33%	72.11%	**66.06%**	80.37%
	Times	612,487	383,815	518,193	811,218	487,919
	S.D.	0.53%	0.19%	0.53%	0.42%	0.16%
VNS-RL	Best	81.87%	69.72%	72.85%	66.51%	80.74%
	Average	80.90%	69.38%	**72.44%**	65.76%	80.35%
	Times	358,785	260,903	371,269	497,124	354,045
	S.D.	0.80%	0.26%	0.28%	0.49%	0.23%
VNS-RLS No Ec	Best	**83.29%**	**69.85%**	72.90%	66.61%	80.65%
	Average	**81.88%**	**69.56%**	72.20%	65.91%	80.48%
	Times	779,504	575,894	661.384	923,891	718,219
	S.D.	0.55%	0.16%	0.38%	0.47%	0.17%
VNS-RLS No Feasibility Indicator	Best	82.89%	69.72%	**73.54%**	**66.78%**	80.78%
	Average	81.69%	69.51%	72.41%	65.99%	**80.55%**
	Times	818,229	436,507	590,170	955,539	737,512
	S.D.	0.64%	0.16%	0.49%	0.48%	0.20%
VNS-RL No Feasibility Indicator	Best	81.89%	69.67%	73.04%	66.27%	80.75%
	Average	81.08%	69.42%	72.10%	65.66%	80.49%
	Times	446,173	245,862	308,416	438,464	411,071
	S.D.	0.51%	0.19%	0.65%	0.50%	0.20%
Upper Bound		90.43%	70.23%	79.58%	73.72%	81.20%

Instance		NP6-1	NP6-2	NP6-3	NP6-4	NP6-5
VNS-RLS	Best	79.60%	74.10%	58.86%	**80.19%**	**80.15%**
	Average	78.96%	73.77%	58.39%	79.29%	78.44%
	Times	828,469	913,906	412,097	1,098,792	928,097
	S.D.	0.43%	0.23%	0.21%	0.49%	0.69%
VNS-RL	Best	79.54%	**74.23%**	58.95%	79.86%	80.09%
	Average	78.92%	73.59%	**58.81%**	79.32%	78.50%
	Times	546,222	666,544	301,455	654,549	442,473
	S.D.	0.42%	0.30%	0.10%	0.47%	1.02%
VNS-RLS No Ec	Best	79.64%	74.14%	58.94%	79.52%	79.99%
	Average	79.07%	73.72%	58.62%	79.10%	78.36%
	Times	1,030,825	1,163,006	513,974	1,053,682	984,987
	S.D.	0.47%	0.21%	0.23%	0.53%	0.99%
VNS-RLS No Feasibility Indicator	Best	**79.69%**	74.10%	59.00%	80.01%	80.11%
	Average	**79.31%**	73.74%	58.62%	**79.45%**	79.34%
	Times	1,163,262	1,015,544	621,863	1,095,930	1,162,651
	S.D.	0.28%	0.29%	0.20%	0.26%	0.49%
VNS-RL No Feasibility Indicator	Best	79.64%	74.20%	**59.11%**	79.84%	79.67%
	Average	79.20%	**73.80%**	58.79%	79.23%	**78.94%**
	Times	508,061	568,328	317,112	599,814	466,904
	S.D.	0.26%	0.24%	0.20%	0.44%	0.52%
Upper Bound		83.93%	76.67%	66.90%	80.97%	84.30%

Instance		NP8-1	NP8-2	NP8-3	NP8-4	NP8-5
VNS-RLS	Best	73.69%	75.09%	74.31%	61.94%	73.28%
	Average	73.10%	74.52%	73.77%	61.85%	72.84%
	Times	1,091,477	917,666	857,388	636,706	1,031,595
	S.D.	0.26%	0.32%	0.40%	0.05%	0.21%
VNS-RL	Best	73.28%	75.25%	74.43%	62.00%	73.60%
	Average	72.87%	74.69%	74.08%	61.91%	73.00%
	Times	567,895	460,155	514,308	398,169	487,284
	S.D.	0.29%	0.23%	0.23%	0.07%	0.32%
VNS-RLS No Ec	Best	**73.80%**	75.27%	74.20%	61.97%	73.62%
	Average	**73.48%**	74.86%	73.96%	61.91%	73.26%
	Times	1,498,392	978,695	867,663	693,779	1,189,550
	S.D.	0.15%	0.28%	0.22%	0.06%	0.35%
VNS-RLS No Feasibility Indicator	Best	73.70%	75.44%	74.60%	62.01%	**73.86%**
	Average	73.32%	**74.90%**	**74.13%**	61.90%	**73.42%**
	Times	1,324,345	1,133,670	974,108	578,979	1,033,072
	S.D.	0.26%	0.31%	0.24%	0.06%	0.37%
VNS-RL No Feasibility Indicator	Best	73.77%	**75.67%**	**74.65%**	**62.05%**	73.50%
	Average	73.12%	74.80%	73.94%	**61.95%**	73.14%
	Times	645,318	497,384	534,775	406,094	468,859
	S.D.	0.39%	0.48%	0.37%	0.06%	0.30%
Upper Bound		77.04%	77.55%	78.82%	62.53%	76.09%

Table 17 **Comparison on 100% artificial instances. Times represents the average evaluation time, while S.D. is the standard deviation of obtained HLDR values. Best heuristic solutions are in bold.**

Instance		LB4-1	LB4-2	TB4-3	TB4-4	LU4-5	LU4-6	TU4-7	TU4-8
VNS-RLS	Best	73.57%	78.02%	69.52%	**72.91%**	64.64%	67.89%	53.07%	53.78%
	Average	72.79%	77.52%	68.78%	**72.09%**	64.22%	67.50%	52.90%	53.58%
	Times	645,863	612,594	666,776	738,772	677,499	816,750	316,288	234,027
	S.D.	0.51%	0.37%	0.53%	0.51%	0.24%	0.26%	0.19%	0.09%
VNS-RL	Best	73.46%	78.60%	**69.90%**	72.19%	64.65%	67.92%	53.09%	53.77%
	Average	72.59%	77.25%	68.61%	71.27%	64.21%	67.54%	52.90%	53.56%
	Times	295,938	346,393	335,523	319,619	335,523	358,255	191,666	141,046
	S.D.	0.48%	0.68%	0.69%	0.48%	0.27%	0.44%	0.22%	0.10%
VNS-RLS No Ec	Best	73.52%	78.08%	69.32%	72.24%	64.67%	**68.12%**	**53.21%**	**53.80%**
	Average	72.93%	77.70%	68.54%	71.42%	**64.38%**	67.52%	**53.03%**	**53.61%**
	Times	642,796	617,656	616,237	635,130	724,154	782,608	399,970	290,599
	S.D.	0.32%	0.32%	0.42%	0.49%	0.20%	0.40%	0.16%	0.08%
VNS-RLS No Feasibility Indicator	Best	**73.76%**	**78.75%**	69.44%	72.52%	64.50%	68.08%	53.18%	53.68%
	Average	**72.99%**	**77.79%**	**68.84%**	71.95%	64.01%	**67.71%**	52.79%	53.53%
	Times	702,902	691,981	878,641	883,267	655,225	887,895	360,106	266,630
	S.D.	0.52%	0.47%	0.52%	0.40%	0.31%	0.37%	0.29%	0.08%
VNS-RL No Feasibility Indicator	Best	73.51%	78.31%	69.36%	72.39%	**64.75%**	68.11%	53.12%	53.77%
	Average	72.80%	77.44%	68.26%	71.34%	64.25%	67.60%	52.78%	53.57%
	Times	276,675	316,636	316,561	326.226	342,380	341,844	202,185	143,319
	S.D.	0.44%	0.41%	0.66%	0.87%	0.36%	0.26%	0.34%	0.14%
Upper Bound		79.47%	86.33%	84.05%	88.74%	74.11%	74.47%	64.05%	63.50%

Instance		LB8-1	LB8-2	TB8-3	TB8-4	LU8-5	LU8-6	TU8-7	TU8-8
VNS-RLS	Best	85.86%	**94.94%**	69.41%	66.08%	67.95%	68.40%	**59.72%**	54.36%
	Average	83.48%	**93.21%**	69.01%	65.24%	67.24%	67.87%	59.31%	54.23%
	Times	1,481,728	1,362,552	619,999	1,422,110	990,621	888,131	586,452	887,790
	S.D.	1.46%	0.82%	0.29%	0.81%	0.52%	0.29%	0.28%	0.12%
VNS-RL	Best	85.16%	93.19%	69.15%	**67.12%**	68.00%	68.32%	59.64%	54.24%
	Average	82.74%	91.96%	68.73%	65.21%	66.96%	67.84%	59.29%	53.99%
	Times	687,880	583,113	309,823	665,151	523,243	411,686	324,946	348,373
	S.D.	1.41%	0.85%	0.33%	1.29%	0.51%	0.40%	0.19%	0.17%
VNS-RLS No Ec	Best	85.49%	94.03%	69.59%	66.85%	67.81%	68.41%	59.60%	**54.50%**
	Average	84.11%	92.83%	**69.04%**	65.70%	67.20%	68.07%	59.21%	54.23%
	Times	1,443,635	1,134,773	669,136	1,478,978	1,114,876	1,026,285	572,065	859,770
	S.D.	0.95%	1.05%	0.38%	0.76%	0.34%	0.21%	0.21%	0.16%
VNS-RLS No Feasibility Indicator	Best	**87.49%**	94.12%	**69.90%**	66.66%	**68.07%**	**68.65%**	59.36%	54.42%
	Average	**85.17%**	92.71%	68.90%	65.66%	**67.29%**	**68.25%**	59.11%	**54.27%**
	Times	1,324,345	1,216,936	524,024	1,713,200	1,058,132	1,043,078	499,058	842,941
	S.D.	0.26%	0.79%	0.39%	0.82%	0.37%	0.29%	0.16%	0.12%
VNS-RL No Feasibility Indicator	Best	85.90%	93.85%	69.46%	66.58%	67.32%	68.64%	59.52%	54.38%
	Average	84.25%	92.35%	68.92%	**65.82%**	66.61%	**68.25%**	**59.33%**	54.05%
	Times	662,849	607,075	330,893	704,743	409,184	520,407	337,106	403,569
	S.D.	1.04%	0.82%	0.35%	0.51%	0.36%	0.30%	0.17%	0.14%
Upper Bound		98.26%	97.97%	87.06%	92.44%	74.27%	71.36%	70.29%	56.54%

Table 18 Comparison on 50% scaled down real-life instances. Times represents the average evaluation time, while S.D. is the standard deviation of obtained HLDR values. Best heuristic solutions are in bold.

Instance		NP4-1	NP4-2	NP4-3	NP4-4	NP4-5
VNS-RLS	Best	75.94%	**65.02%**	**73.18%**	62.49%	**79.91%**
	Average	74.56%	**63.33%**	72.15%	61.46%	79.36%
	Times	460,790	568,601	378,158	421,254	582,828
	S.D.	0.86%	0.88%	0.57%	0.66%	0.31%
VNS-RL	Best	75.69%	64.54%	72.88%	62.72%	79.64%
	Average	74.55%	62.84%	71.96%	61.76%	79.23%
	Times	241,986	224,495	184,163	259,344	272,266
	S.D.	0.72%	1.02%	0.66%	0.74%	0.31%
VNS-RLS No Ec	Best	76.06%	63.74%	72.77%	63.63%	79.68%
	Average	75.17%	63.00%	72.22%	61.84%	79.47%
	Times	534,540	513,641	376,939	518,689	555,183
	S.D.	0.62%	0.45%	0.44%	0.78%	0.14%
VNS-RLS No Feasibility Indicator	Best	75.80%	63.71%	72.91%	**62.80%**	79.77%
	Average	74.83%	63.04%	**72.30%**	**61.97%**	**79.51%**
	Times	477,073	497,737	457,465	562,605	577,746
	S.D.	0.59%	0.49%	0.44%	0.51%	0.11%
VNS-RL No Feasibility Indicator	Best	**76.58%**	64.53%	72.67%	62.49%	79.83%
	Average	**75.18%**	63.25%	72.14%	61.37%	79.35%
	Times	293,465	229,114	206,959	213,268	277,655
	S.D.	0.73%	0.66%	0.63%	0.54%	0.35%
Instance		**NP6-1**	**NP6-2**	**NP6-3**	**NP6-4**	**NP6-5**
VNS-RLS	Best	74.22%	71.86%	60.18%	78.22%	78.82%
	Average	73.33%	71.48%	**59.79%**	77.62%	77.13%
	Times	790,909	755,430	366,196	922,226	737,902
	S.D.	0.58%	0.24%	0.22%	0.53%	1.11%
VNS-RL	Best	74.09%	72.13%	60.04%	78.38%	77.84%
	Average	73.05%	71.56%	59.76%	77.62%	77.23%
	Times	368,065	419,235	178,172	386,823	318,487
	S.D.	0.58%	0.35%	0.30%	0.54%	0.80%
VNS-RLS No Ec	Best	**75.34%**	**72.03%**	**60.21%**	78.15%	78.05%
	Average	**73.86%**	**71.68%**	59.72%	77.73%	77.30%
	Times	936,874	826,244	398,098	833.024	808,691
	S.D.	0.72%	0.25%	0.37%	0.38%	0.81%
VNS-RLS No Feasibility Indicator	Best	75.23%	71.82%	**60.21%**	**78.66%**	78.98%
	Average	73.62%	71.45%	59.70%	**77.88%**	77.45%
	Times	833,619	604,648	340,517	913,326	769,152
	S.D.	0.73%	0.32%	0.26%	0.41%	0.85%
VNS-RL No Feasibility Indicator	Best	74.66%	71.98%	60.16%	78.02%	**80.02%**
	Average	73.68%	71.54%	59.75%	77.26%	**78.53%**
	Times	471,835	354531	174,494	360,102	365,608
	S.D.	0.50%	0.21%	0.41%	0.59%	1.18%
Instance		**NP8-1**	**NP8-2**	**NP8-3**	**NP8-4**	**NP8-5**
VNS-RLS	Best	**73.83%**	74.95%	74.54%	62.39%	72.71%
	Average	72.69%	73.97%	73.93%	**62.32%**	72.04%
	Times	949,066	916,414	685,003	277,510	937,414
	S.D.	0.62%	0.57%	0.37%	0.06%	0.40%
VNS-RL	Best	73.40%	74.29%	74.33%	62.35%	72.82%
	Average	72.53%	73.86%	**74.01%**	62.20%	72.04%
	Times	528,557	342,240	332,628	277,510	397,483
	S.D.	0.53%	0.36%	0.29%	0.09%	0.41%
VNS-RLS No Ec	Best	73.66%	74.66%	74.32%	62.36%	72.76%
	Average	**72.80%**	74.07%	73.99%	62.30%	72.00%
	Times	1,174,959	839.055	624,092	574,786	380,812
	S.D.	0.51%	0.30%	0.26%	0.06%	0.48%
VNS-RLS No Feasibility Indicator	Best	73.29%	**74.99%**	**74.87%**	**62.40%**	72.16%
	Average	72.68%	**74.32%**	74.14%	62.30%	71.76%
	Times	1,124,131	771,843	768,217	502,949	729,571
	S.D.	0.55%	0.45%	0.43%	0.07%	0.27%
VNS-RL No Feasibility Indicator	Best	73.54%	74.88%	74.65%	62.36%	**73.33%**
	Average	**72.80%**	74.30%	**74.01%**	62.25%	**72.13%**
	Times	480,227	338,567	341,863	244,973	380,812
	S.D.	0.68%	0.56%	0.34%	0.06%	0.53%

Table 19 Comparison on 50% scaled down artificial instances. Times represents the average evaluation time, while S.D. is the standard deviation of obtained HLDR values. Best heuristic solutions are in bold.

Instance		LB4-1	LB4-2	TB4-3	TB4-4	LU4-5	LU4-6	TU4-7	TU4-8
VNS-RLS	Best	74.57%	**76.86%**	68.49%	**69.41%**	61.03%	67.28%	55.63%	55.32%
	Average	73.68%	**76.06%**	67.39%	**67.95%**	60.29%	66.36%	55.48%	55.06%
	Times	411,992	352,122	312,086	427,614	505,017	469,358	240,935	241,367
	S.D.	0.66%	0.37%	0.69%	0.73%	0.42%	0.44%	0.17%	0.15%
VNS-RL	Best	74.37%	76.34%	68.39%	68.57%	61.05%	66.94%	**55.98%**	**55.45%**
	Average	73.46%	75.74%	67.44%	67.59%	60.42%	66.53%	**55.49%**	55.02%
	Times	214,815	178,610	182,916	179,805	232,713	242,746	140,668	104,870
	S.D.	0.98%	0.35%	0.57%	0.58%	0.44%	0.33%	0.26%	0.18%
VNS-RLS No Ec	Best	74.46%	76.40%	**68.60%**	68.92%	**61.71%**	**67.43%**	55.63%	55.33%
	Average	73.72%	75.94%	67.45%	67.89%	60.58%	**66.74%**	55.44%	54.98%
	Times	401,541	341,333	445,478	362,031	542,568	575,862	233,207	198,381
	S.D.	0.41%	0.40%	0.64%	0.73%	0.61%	0.37%	0.17%	0.16%
VNS-RLS No Feasibility Indicator	Best	74.35%	76.44%	68.46%	68.73%	61.22%	67.13%	55.66%	55.32%
	Average	**73.84%**	75.96%	**67.64%**	67.75%	60.67%	66.73%	55.47%	**55.09%**
	Times	412,689	375,605	442,350	427,133	494,563	505,842	260,674	269,741
	S.D.	0.34%	0.51%	0.50%	0.57%	0.33%	0.35%	0.13%	0.12%
VNS-RL No Feasibility Indicator	Best	**74.64%**	76.82%	68.18%	68.81%	61.45%	67.00%	55.76%	**55.45%**
	Average	73.50%	75.96%	67.49%	67.67%	**60.97%**	66.63%	55.46%	54.98%
	Times	194,032	174,671	227,019	189,330	231,117	241,260	129,217	106,969
	S.D.	0.58%	0.59%	0.57%	0.84%	0.40%	0.22%	0.25%	0.20%
Instance		**LB8-1**	**LB8-2**	**TB8-3**	**TB8-4**	**LU8-5**	**LU8-6**	**TU8-7**	**TU8-8**
VNS-RLS	Best	89.16%	93.65%	67.91%	65.89%	67.75%	67.81%	60.89%	53.90%
	Average	86.40%	92.25%	**67.23%**	64.56%	**67.28%**	67.01%	60.51%	53.73%
	Times	771,657	767,629	489,994	781,730	563,840	740,547	325,656	379,196
	S.D.	2.01%	1.13%	0.49%	0.76%	0.34%	0.51%	0.20%	0.08%
VNS-RL	Best	89.46%	92.72%	67.65%	65.39%	67.53%	67.34%	60.99%	53.76%
	Average	87.03%	91.44%	67.18%	64.69%	66.84%	66.71%	**60.67%**	53.56%
	Times	463,459	354,493	210,610	390,148	219,171	282,208	165,852	147,020
	S.D.	1.53%	0.97%	0.41%	0.53%	0.36%	0.38%	0.22%	0.14%
VNS-RLS No Ec	Best	89.08%	93.66%	67.61%	65.78%	67,59%	**67.87%**	60.73%	**53.99%**
	Average	87.50%	**92.36%**	67.06%	64.73%	67.09%	67.24%	60.49%	**53.79%**
	Times	913,824	764,714	386,613	715,519	551,729	649,507	312,937	405,527
	S.D.	0.69%	0.89%	0.29%	0.60%	0.47%	0.50%	0.18%	0.13%
VNS-RLS No Feasibility Indicator	Best	**90.51%**	**93.90%**	**67.99%**	**66.59%**	**67.93%**	67.69%	**61.00%**	53.87%
	Average	**88.21%**	92.31%	67.10%	**65.09%**	67.32%	**67.03%**	60.51%	53.67%
	Times	1,049,083	838,955	433,102	789,905	630,593	617,863	288,587	350,326
	S.D.	1.66%	1.29%	0.63%	0.88%	0.46%	0.60%	0.24%	0.17%
VNS-RL No Feasibility Indicator	Best	90.08%	93.06%	67.76%	65.96%	67.49%	67.81%	60.89%	53.85%
	Average	87.00%	91.88%	67.10%	64.80%	66.78%	66.94%	60.60%	53.70%
	Times	385,955	377,490	189,565	372,753	225,925	312,747	187,888	211,534
	S.D.	1.42%	0.65%	0.33%	0.81%	0.49%	0.54%	0.16%	0.08%