Information Sciences 394–395 (2017) 167–182

Contents lists available at ScienceDirect

Information Sciences

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

# A hybrid algorithm for a vehicle routing problem with realistic constraints

Defu Zhang[a], Sifan Cai[a], Furong Ye[a,*], Yain-Whar Si[b], Trung Thanh Nguyen[c]

[a] School of Information Science and Engineering, Xiamen University, Xiamen, China
[b] Department of Computer and Information Science, University of Macau, Macau, China
[c] Department of Maritime and Mechanical Engineering, Liverpool John Moores University, Liverpool, England United Kingdom

ARTICLE INFO

*Article history:*
Received 7 February 2016
Revised 10 February 2017
Accepted 14 February 2017
Available online 15 February 2017

*Keywords:*
Vehicle routing problem
Container loading
Tabu search
Artificial bee colony algorithm

ABSTRACT

Proliferation of multi-national corporations and extremely competitive business environments have led to an unprecedented demand for third-party logistics services. However, recent studies on the vehicle routing problem (VRP) have considered only simple constraints. They also do not scale well to real-world problems that are encountered in the logistics industry. In this paper, we introduce a novel vehicle routing problem with time window and pallet loading constraints; this problem accounts for the actual needs of businesses in the logistics industry such as the delivery of consumer goods and agricultural products. To solve this new VRP, we propose a hybrid approach by combining Tabu search and the artificial bee colony algorithm. A new benchmark data set is generated to verify the performance of the proposed algorithm because the proposed VRP has never been reported in the literature. Experiments are performed for a data set of Solomon's 56 vehicle routing problem with time windows. Our approach is superior to a number of other heuristic algorithms in a comparison on Solomon's VRPTW instances.



## 1. Introduction

The vehicle routing problem (VRP) is a classical problem from the logistics and transportation fields. It is concerned with route planning for vehicles that start from a central depot and go to a set of customers. Due to its wide range of applications in both commercial and public entities, the VRP is considered to be one of the most important problems in operational research [5]. The VRP was first proposed by Dantzig and Ramser [11]. Further improvements and variants of this problem have been extensively studied in recent years. Although various objectives were introduced to the classical VRP in recent years, the majority of these variants were mostly related to the addition of new constraints to the original problem. For example, *pallet* and *time window* are the two constraints that are most often used by a variety of researchers in the VRP. In the pallet constraint, goods of different sizes must be transported in boxes of standard size and limited capacity. In the time window constraint, goods must be delivered during a certain time window. A number of studies related to the above constraints have been reported in the literature. For pallet constraints, Leung et al. [25] solved heterogeneous fleet VRPs with two-dimensional loading constraints by using simulated annealing. Wei et al. [39] proposed an adaptive variable neighborhood search for a heterogeneous fleet VRP with three-dimensional loading constraints. For VRPs with time window constraints, Cherkesly et al. [6] developed branch-price-and-cut algorithms for the pickup and delivery problem with time

* Corresponding author.
*E-mail addresses:* dfzhang@xmu.edu.cn (D. Zhang), 277970699@qq.com (F. Ye).

http://dx.doi.org/10.1016/j.ins.2017.02.028
0020-0255/

windows and multiple stacks. Hifi and Wu [17] solved the VRP with time windows by considering the number of vehicles and the total distances in order. Gong et al. [15] developed a discrete particle swarm optimization approach for the vehicle routing problem with time windows.

However, these two constraints are normally considered separately in the existing research. In reality, both constraints co-exist in transportation problems. How to address the daily dispatching problem is generally recognized as the primary concern of most logistic companies. However, when designing simple VRPs, researchers often neglect some of the constraints that are encountered in real-world scenarios. In most situations, these constraints are considered separately. Therefore, VRPs that involve a single constraint usually suffer from several drawbacks. For example, the VRP with time windows focuses on constructing routes without accounting for the properties of the items to be shipped. Although there are VRPs with time windows that consider the capacity of vehicles, they still ignore the actual physical dimensions of the items. On the other hand, the VRPs with the pallet constraint focus only on constructing routes that have feasible loading conditions without considering the time factor. Due to the omission of the time factor, for some real-world scenarios, goods might not be delivered to the customers on time. Therefore, it is necessary to further investigate the realistic situations for the VRP by accounting for both the time window and pallet constraints (the three-dimensional loading problem in particular).

To the best of our knowledge, the VRP with both time window and three-dimensional loading constraints has been addressed only in the work of Zachariadis et al. [43]. The problem defined in [43] has specific constraints and includes a mix of different request types. This VRP (proposed in [43]) might not meet the dispatching requirements of some logistics companies, especially from the electronic commerce area. The reason is that the VRP proposed in [43] considers both pick-up and delivery activities, whereas logistics companies from electronic commerce usually concern the dispatching service only.

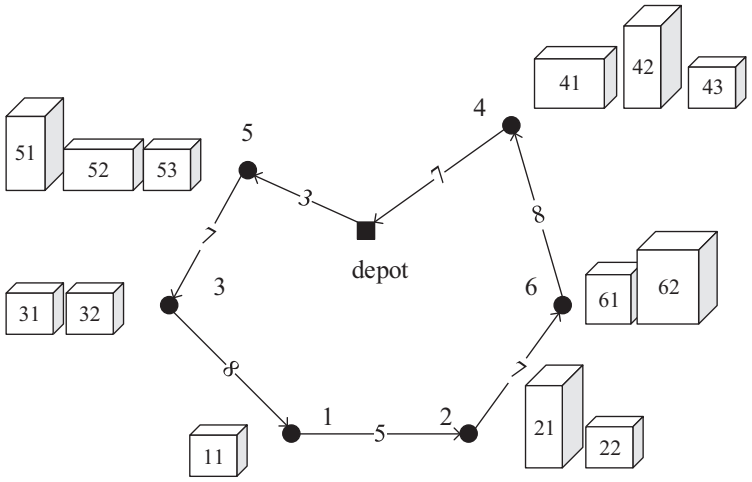
The problem introduced in this paper can be considered to be a VRP that includes two types of constraints: the time window and three-dimensional loading constraints. In contrast to [43], the problem addressed in this paper more closely mimics the real-world situations because it accounts for both the actual needs of the businesses (detailed below) and the unique types of depots and customers.

For the first type of constraint, the depot and customers are specified with a time window that includes exact start and end times. This means that vehicles must depart and return within the depot's work time window. A vehicle must also visit a customer between the specified start and end times. If a vehicle arrives before the start time of a customer, any unloading will be postponed until the customer is available.

In addition to the basic space constraint, other three-dimensional loading constraints, such as the fragility, supporting surface, and order of unloading, are also considered. Businesses involved in the shipping of electrical appliances and the distribution of fresh agricultural products can greatly benefit from the proposed solution because the proposed approach considers not only the time window but also the three-dimensional loading constraints from real-world scenarios. For the VRP with only the loading constraint, a container loading problem can be adopted to validate the feasibility of the loading. However, for the VRP with time window and pallet loading constraints (VRPTWP), the loading problem can be considered to be a three-dimensional bin loading problem in which a fixed number of rectangular items are loaded into larger rectangular boxes [16].

Earlier works on the VRP were based mainly on the exact approaches. At the same time, rapid developments in the logistics industry and the dynamic nature of today's business environment had a significant impact on the scale and complexity of the VRPs. Clearly, traditional exact approaches might not be able to match the scale of real-world situations. This circumstance has led to the development of heuristic/metaheuristic algorithms for the VRP. Tabu search is one of the most important methods in early research on heuristics. One of the earliest reports on solving the VRP using Tabu search was proposed by Gendreau et al. [13].Since then, many researchers have used Tabu search to solve VRPs. Tabu search and unified Tabu search were used by Taillard et al. [34] and Cordeau et al. [9] to solve VRPs with soft time windows as well as VRPs with time windows. Leung et al. [26] applied extended guided Tabu search and a new packing algorithm for the VRP with the two-dimensional loading constraint. The artificial bee colony algorithm introduced by Karaboga in [20] is a new and excellent algorithm for NP-hard problems. Over the past ten years, approaches based on the artificial bee colony algorithm have been increasingly adopted by researchers to solve VRPs. In [33], Szeto et al applied the artificial bee colony algorithm to solve the capacitated VRP. Later, Yao et al. [40] proposed an artificial bee colony algorithm with a scanning strategy for the periodic VRP. At the same time, other heuristics, such as simulated annealing [21] and genetic algorithms [18], have also been widely used to solve VRPs. Hybrid approaches that combine different metaheuristics were used to solve VRPs as well; for example, Bortfeldt et al. [4] described a hybrid algorithm for the VRP that involves clustered backhauls and loading constraints. Beheshti and Hejazi [3] proposed a hybrid algorithm of column generation and metaheuristics for variants of the VRP. Zhang et al. [45] developed a hybrid algorithm by combining an evolutionary local search with the recombination method for the VRP with three-dimensional loading constraints. Akpinar [1] proposed a hybrid large neighborhood search algorithm for the capacitated VRP.

The main contributions of our work are two-fold. First, we consider a new variant of the VRP that involves loading and time window constraints. This variant is designed to mimic real-world scenarios from the logistics industry, combining the two constraints for the first time. In contrast with previous research, we address the loading constraint with an efficient method. Second, we propose a hybrid algorithm by combining Tabu search and the artificial bee colony algorithm (Tabu-ABC). The performance of the algorithm was evaluated against other heuristics on a set of Solomon's 56 VRP with time windows (VRPTW). The novel hybrid algorithm takes advantage of two heuristics and performs more efficiently, which pro-




Time window and work time ( $[t_{si}, t_{ei}]$, $t_i$ ) of each vertex:
**depot:**[0,70],0
**1:**[15,23],2 **2:**[25,30],1 **3:**[10,14],1
**4:**[35,55],3 **5:**[2,8],2 **6:**[10,40],1

**Fig. 1.** An illustration of VRPTW routing.

vides a potential approach for problems in other domains. In addition, the set of benchmark data generated for the VRPTWP from this work can be adopted by other interested parties for further research in this area.

The remainder of this paper is organized as follows. In Section 2, the VRPTWP is introduced in detail. In Section 3, we describe the new hybrid algorithm (Tabu-ABC) in detail. In Section 4, the computational results of the proposed algorithm are reported, and finally, we conclude our work in Section 5.

## 2. The problem

Let $G(V, A)$ be an undirected graph, where $V = \{0, 1, \ldots, n\}$ is the set of vertices and $A$ is the set of edges. Let $C_{ij}$ be the transportation cost between vertices $i$ and $j$. Vertex $0$ in $V$ is called the depot. The main objective of solving the VRP is to search for a solution that covers the shortest distance but traverses every vertex. The traversal must start and end at the depot.

Recall that the VRP to be addressed in this paper accounts for both the time window and loading constraints to reflect real-world situations. In this context, we assume that there is a customer at each vertex and that each customer needs a set of items from the depot. Each set of items is specified with a pre-defined total weight. Each item can be considered to be a three-dimensional cuboid of length $l_i$, width $w_i$ and height $h_i$. At the depot vertex, there are some vehicles available for carrying goods, and each of them has a fixed loading space (a container) of dimension $L \times W \times H$, where $L$, $W$ and $H$ are the length, width and height of the loading space, respectively. In addition, each vehicle is specified with a weight capacity $D$. Each vertex can be either the depot or a customer, and each has a specific work time window $[t_{si}, t_{ei}]$ and work time $t_i$. In addition, all of the work at a specific vertex must be performed within each vertex's work time window. An example of time window in a route is given in Fig. 1.

In the VRPTWP, the following conditions must be met to satisfy the loading constraints:

(i) Each customer (vertex) is visited only once. In other words, one customer belongs to only one route.
(ii) All of the routes must start and end at the depot.
(iii) All of the routes must depart and return within the time window of the depot.
(iv) All of the customers (vertices) are available during their work time window.
(v) All of the items that are needed by the customers in a route are loaded onto the vehicle that serves that route.

Before a vehicle can depart from the depot, it is necessary to ensure that all of the needed items are loaded on the vehicle. This can be considered as a process for verifying the feasibility of the three-dimensional loading problem while accounting for the following conditions:

(i) All of the items must be completely loaded and fit into the container of a vehicle, i.e., the edges of the items and the container do not intersect one another.
(ii) Items are not allowed to overlap.

(iii) The bottom of each item must be sufficiently supported by the bottom of the other items or by the bottom of the container.
(iv) The surface of each item must be in parallel with the surface of the container.

In addition to the above basic conditions, we add the following conditions to better reflect real-world situations:

(v) *Orientation:* The items have fixed vertical orientation, which means that they have fixed bottom surfaces.
(vi) *Capacity:* The sum of the items' weights is less than or equal to the vehicle's loading capacity.
(vii) *Fragility:* Nonfragile items cannot be loaded on top of fragile items.
(viii) *LIFO:* If customer $i$ is visited earlier than customer $j$, then the items of customer $j$ should be packed earlier than those of customer $i$.

Mathematically, our objective is to find a minimum travel distance, which can be expressed as follows:

$$\min \sum_{i=1}^{r} \left( \sum_{j=1}^{n_i-1} C_{j(j+1)} \right) \tag{1}$$

where $r$ is the number of routes (the number of vehicles is equal to the number of routes because each vehicle serves a unique route), $n_i$ is the number of vertexes in route $i$, and $C_{j(j+1)}$ denotes the cost of traveling from vertex $j$ to vertex $j+1$ (in this paper, the traveling cost is equal to the distance).

The time windows constraints can be described as follows:

$$t_{ai} < t_{ei} \tag{2}$$

where

$$t_{ai} = t_{l(i-1)} + tC_{i(i-1)}, \tag{3}$$

$$t_{li} = \begin{cases} t_{ai} + t_i, & if\ t_{ai} > t_{si} \\ t_{si} + t_i, & otherwise \end{cases} \tag{4}$$

where $t_{si}$ and $t_{ei}$ are the start and end times of vertex $i$ in the route, $t_{ai}$ is the time that the vehicle arrives at vertex $i$, $t_{li}$ is the time that the vehicle leaves from vertex $i$, $t_i$ is the time cost of working for vertex $i$ in the route, and $tC_{i(i-1)}$ is the time cost of traveling from vertex $i$ to vertex $i$–1.

Because the loading constraint is met with many conditions and handled with a heuristic algorithm, there is no need to provide the mathematical formulation for this constraint. Note that the VRPTWP can be considered as a combination of two NP-hard sub-problems. Therefore, the VRPTWP can be classified as a more difficult NP-hard problem. In the following section, we describe a heuristic algorithm to address this problem.

## 3. The proposed approach

According to the description given in Section 2, the VRPTWP can be treated as a combination of a VRP with three-dimensional loading and time window constraints. In this section, a two-stage approach is proposed. First, we consider multiple strategies to solve the three-dimensional loading problem. Next, a hybrid algorithm is used to solve the VRPTW by combining Tabu search with an artificial bee colony approach.

### 3.1. Three-dimensional loading problem

One of the key aspects of the proposed approach is the use of an innovative method to judge whether boxes (items) that are needed by customers along a route can be loaded onto the vehicle. Such a feasibility test has a significant effect on the outcome because it is repeatedly invoked by the main algorithm described in Section 3.2.

#### 3.1.1. Loading positions

The container (of a vehicle) is placed in a three-dimensional system of coordinates (see Fig. 2.), and the origin of the coordinates is located at the bottom-left corner of the container. $L$, $W$ and $H$ represent the length, width and height of the container. To fully utilize the available space, the boxes to be loaded are required to be close to the container or to the boxes that are already loaded. Because the boxes can be rotated in the horizontal direction, when a box $i$ of length $l_i$, width $w_i$ and height $h_i$ is being loaded into the container, its values along the $x$, $y$ and $z$ axes $\langle\ l_i', w_i', h_i'\ \rangle$ can be considered to be $\langle\ l_i, w_i, h_i\ \rangle$ or $\langle\ w_i, l_i, h_i\ \rangle$.

We denote $B = \{b_1, b_2, \ldots, b_n\}$ as a set of boxes. First, $b_1$ will be loaded if there is an available loading position at (0, 0, 0). Next, there are three available loading positions for $b_2$, namely, ($l_1'$, 0, 0), (0, $w_1'$, 0) and (0, 0, $h_1'$ ). Suppose that $b_2$ is loaded at position ($l_1'$, 0, 0); then, ($l_1'$, 0, 0) is deleted, and another three available loading positions ($l_1' + l_2'$, 0, 0), ($l_1'$, $w_2'$, 0) and ($l_1'$, 0, $h_2'$ ) will be generated. Therefore, five loading positions are now available for $b_3$. In the general case, if the $i^{th}$ box is loaded at location ($x$, $y$, $z$), then ($x$, $y$, $z$) will be deleted from the available loading positions list. Subsequently, ($x + l_i'$, $y$, $z$), ($x$, $y + w_i'$, $z$) and ($x$, $y$, $z + h_i'$ ) will be added to the available loading positions list (see Fig. 1.). When a box is loaded,

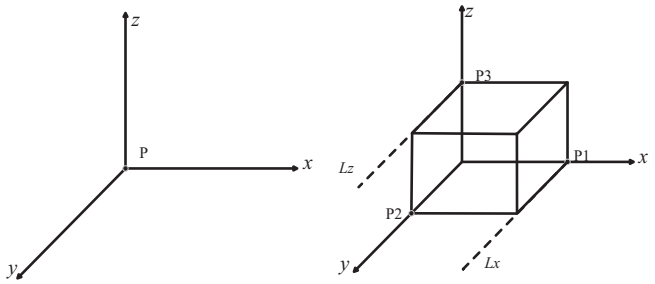



**Fig. 2.** Updating process of loading positions.

**Algorithm 1** Verifying the feasibility of three-dimensional loading.

**PalletisationFeasibility(*B*, Vehicle)**
1. $I = \{(0, 0, 0)\}$, $Lz = Lx = 0$;
2. **for** $i = 0$ to $n$
3. $flag = false$;
4. **for** $(x, y, z) \in I$
5. **if** $x + l_i' \le Lx$, $z + h_i' \le Lz$ and $b_i$ can be loaded on $(x, y, z)$
6. $flag = true$, **go to** line 19;
7. **if** $Lx = 0$ or $Lx = L$
8. **if** $b_i$ can be loaded on $(0, 0, Lz)$
9. $x = 0$, $y = 0$, $flag = true$, $z = Lz$, $Lz = Lz + h_i'$, $Lx = l_i'$;
10. **else**
11. $Lz = H$, $Lx = L$, $i = i - 1$;
12. **else**
13. **for** $(x, y, z) \in I$
14. **if** $x = Lx$ and $y = 0$
15. **if** $z + h_i' \le Lz$ and $b_i$ can be loaded on $(x, y, z)$
16. $Lx = Lx + l_i'$, $flag = true$, **go to** line 19;
17. **else**
18. $Lx = L$, $i = i - 1$;
19. **if** $flag = true$
20. load $b_i$ on $(x, y, z)$, $I = I / (x, y, z)$, move $b_i$ with the translational operator and mark $b_i$'s new position $(x', y', z')$, $I = I \cup (x' + l_i', y', z')$, $(x', y' + w_i', z')$, $(x', y', z' + h_i')$;
21. **else**
22. **return** *false*;
23. **return** *true*;

an available loading position will be deleted, and three new available loading positions will be added. Therefore, there will be $2(i\text{-}1) + 1$ available loading positions for box $b_i$. If $b_i$ cannot be loaded to any of the available loading positions, then it is assumed that boxes $\{b_1, b_2, \ldots, b_n\}$ cannot be packed in the container.

### 3.1.2. Reference line

Two reference lines are considered in this study to control the used space and to pack efficiently. *Lz* and *Lx* denote the two reference lines for the *z* axis and *x* axis, respectively. When checking whether box $b_i$ can be loaded at position $(x, y, z)$, we must ensure that the conditions stated in Section 2 are not violated. In addition, loading must also satisfy the conditions $z + h_i < Lz$ and $x + l_i < Lx$. Once an available loading position is deemed to be feasible, $b_i$ will be loaded, and the available loading positions list will be updated. The following two situations are considered when none of the positions are available: (1) If $Lx < L$, *Lx* will be increased to *L*; (2) if $Lz < H$, *Lz* will be increased to *H*. If a feasible position for $b_i$ is not found after (1) and (2), then we can conclude that the boxes cannot be packed into the container.

### 3.1.3. Translational operator

Once an available loading position is chosen and the box is loaded, the position of the box will be adjusted by decreasing the values of *x*, *y*, and *z* until it is blocked by other boxes or the container.

### 3.1.4. Loading algorithm

In this paper, we adopt the loading algorithm (Algorithm 1) from [44]. *I* is a list of available loading positions, which are sorted in increasing order based on *x*. The parameters *y* and *z* are used to break ties when the values of *x* for the loading positions are the same. We maintain the order of the list *I* while it is updated. Additionally, a *flag* variable is used to express whether a box can be loaded into the container or not. The loading algorithm accepts the ordered boxes *B* and the container of the vehicle as the inputs. The algorithm will return *true* if all of the boxes can be loaded into the container.

**Algorithm 2** Initialization of *VRPTWP*.

```
Initialization (C)
1.  Route_Num = 0, c = C, S = φ;
2.  while c ≠ φ
3.      Select customer c_i from C randomly.
4.      if Route_Num = 0
5.          Route_num = Route_num + 1;
6.          Generate a new route s, S = S + s;
7.          Insert c_i to s, c = c/ c_i;
8.      else if c_i can be inserted into a route in S
9.          Select s that causes least increase of cost after insertion;
10.         Insert c_i into s in the position that causes least increase of cost after insertion;
11.         c = c/ c_i;
12.     else
13.         Route_num = Route_num + 1;
14.         if Route_num > number of vehicles
15.             Go to line 1;
16.         else
17.             Generate a new route s, S = S + s;
18.             Insert c_i to s, c = c/ c_i;
```

*C* is the set of customer, *S* is the set of routes and Route_Num is the number of routes.

First, the original available loading position is set to (0, 0, 0), and *Lz* and *Lx* are initialized to 0. The boxes are readied in such a way that they can be loaded into the container in order. In this algorithm, each box is first tested to see whether it can be loaded into the container without exceeding the reference lines. If it cannot be loaded, then the algorithm changes the value of *Lx*. When *Lx* is equal to 0 or *L*, the algorithm increases *Lz*. When *Lz* is equal to *H* and if the box still cannot be loaded, the algorithm increases *Lx*. When the algorithm attempts to increase the value of *Lx*, all of the positions that satisfy $x = Lx$ and $y = 0$ are tested for the ability to load the box. Once *Lx* is equal to *L* and the box cannot be loaded, the algorithm returns *false*. When a box is loaded successfully, the selected loading position is deleted from the list *I*. After the box is moved with adjusting of the position, three new available loading positions are generated. The algorithm will return *true* if all of the boxes are loaded successfully; otherwise, it will return *false*.

### 3.2. Overall structure of the VRPTWP solution

Recall that the VRPTWP consists of two problems, namely, the VRP with three-dimensional loading constraints (3L-CVRP) and the VRPTW. Because the former problem was addressed in Section 3.1, we are now ready to solve the VRPTWP. In our approach, we use a local search to improve the solution. We define six neighborhood structures. One of them is randomly selected during each iteration, and the algorithm attempts to find a better solution, which has a lower cost, during each iteration. However, this strategy could lead the algorithm to be trapped in a local optimum. To avoid local optima, we adopt the Tabu search and artificial bee colony algorithms.

#### 3.2.1. Construction of the initial VRPTWP solution

Our algorithm begins by generating an initial feasible solution (explained below). Once this initialization step is completed, further improvement can be performed to achieve better results. Note that all of the constraints described in Section 2 must be satisfied by this initial solution.

In this algorithm, customers are randomly inserted into the routes one by one. During the insertion process, the customer is inserted at the position where the increase in cost is minimal provided that all of the constraints mentioned in Section 2 have been satisfied. After a new customer is inserted and if all existing routes cannot be assigned for that customer, a new route will be set up. If the total number of routes exceeds the number of available vehicles, the algorithm will restart. Otherwise, the algorithm continues until all of the customers are processed. The process of construction is described in Algorithm 2.

#### 3.2.2. Neighborhood solutions

When we attempt to find a better solution, we can reassign the positions of the vertices along different routes to construct a new solution. At each step, we choose a neighborhood structure randomly. Six neighborhood structures are applied in this paper and are defined as follows:

(i) *Swapping*: In this strategy, the locations of two customers are exchanged. Two customers can be either on the same route (see Fig. 3(a).) or on different routes (see Fig. 3(b).).
(ii) *Relocation*: With this strategy, a customer is moved to another position. The new position can be on its original route (see Fig. 3(c).) or on different route (see Fig. 3(d).).
(iii) *Routes swapping*: Each route is divided into two sub-routes by a vertex (customer). The two sub-routes (the part that is visited after the specific vertex) of the two routes are exchanged entirely (see Fig. 3(f).).
(iv) *Route reversal*: A sub-route reverses the order of customers (see Fig. 3(e).).

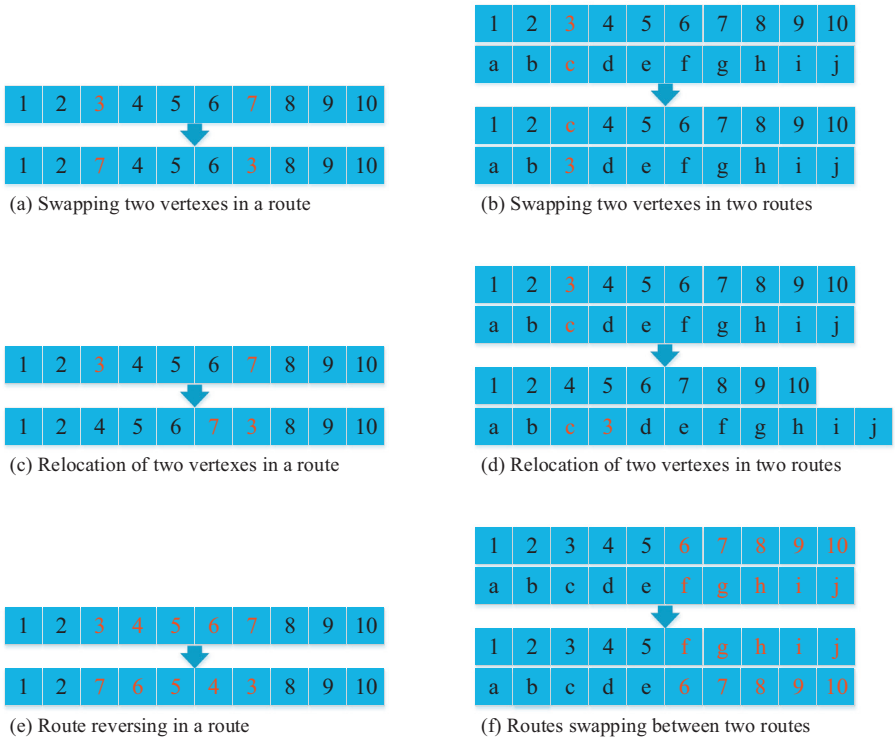



**Fig. 3.** Neighborhood structures.

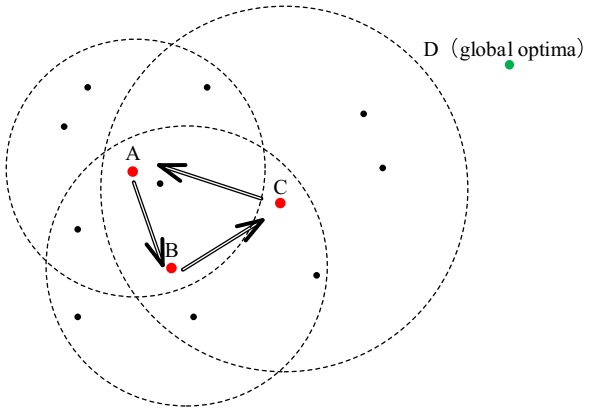



**Fig. 4.** Trap of local optima.

*3.2.3. Tabu-ABC*

Tabu-ABC is a combination of the Tabu search and artificial bee colony algorithm. It uses Tabu search to rapidly generate a high quality solution that is used by the artificial bee colony algorithm. At the same time, the artificial bee colony algorithm takes advantage of Tabu search to increase the variety of food sources.

Tabu search is designed to search for the best solution in its neighborhood, even if there is no better solution. This approach could pose the risk of getting stuck in a local optimum (Fig. 4). A, B, C and D are four different solutions. In A's neighborhood, B is the best solution. C is the best solution in B's neighborhood, and A is the best in C's neighborhood. According to the proposed algorithm, the best solution that is found can only be B, C and A. In this situation, the local search becomes trapped in a loop, and the solution cannot be improved further. Although D is the true best solution, it is missed by the algorithm. This case highlights the problem of the search being trapped in a local optimum.

To avoid this trap, a Tabu list is introduced to prohibit previously visited customers from being revisited by the algorithm. Once we have found a better solution with the neighborhood structure, all of the recently exchanged customers are inserted into a Tabu list. In the subsequent iterations of searching, these customers will not be selected unless the algorithm can

**Algorithm 3** The framework of Tabu-ABC.

**Tabu-ABC**

1. Find original food sources $x_i,\ i = 1, 2, \ldots,\ n$, with initialization;
2. Evaluate the fitness of each food source $f(x_i),\ i = 1, 2, \ldots,\ n$;
3. Improve the fitness of food sources with Tabu search (Searching for new food sources in neighborhood, if a new food source is better than old one and not in the Tabu tenure, replace the old one with it);
4. No update times: $l_1 = l_2 = \cdots = l_n = 0$;
5. **While** not satisfy termination condition **do**
6. Employed bees stage: search $x_i'$ in neighborhood of $x_i$, **if** $f(x_i')$ is greater than $f(x_i)$, replace $x_i$ with $x_i'$, $l_i = 0$; **else** $l_i = l_i + 1$;
7. Onlookers stage: Select a $x_i'$ according to the traditional roulette wheel selection; find the best $x_i'$ in neighborhood of $x_i$, **if** $f(x_i')$ is greater than $f(x_i)$, replace $x_i$ with $x_i'$, $l_i = 0$; **else** $l_i = l_i + 1$;
8. Scout bees stage: for each food source $x_i$, if $l_i = limit$, find a new food sources that similarity with all existed food sources is lower than the predetermined limit, and replace $x_i$ with it;
9. **End while**.

obtain a solution that is better than the current best solution. The maximum length of the Tabu list is called the Tabu tenure.

The artificial bee colony algorithm is a class of swarm intelligence techniques. Honey bees from the algorithm are classified into three types: employed bees, onlookers and scouts. The job of employed bees is to exploit the food sources. They gather and share information with the onlookers. According to the information shared by the employed bees, the onlookers select a food source that has a higher quality. Hence, food sources that have good quality will be chosen by the onlookers. The scout bees randomly explore new food sources. When the onlookers and scout bees find a new food source, they become employed bees.

Because Tabu search starts with a random solution, its performance can be easily influenced by the initial solution. Therefore, we introduce the artificial bee colony algorithm to alleviate this shortcoming. The artificial bee colony algorithm can generate a set of initial solutions for Tabu search by eliminating the influence of the initial solution. Because Tabu search can assist the artificial bee colony in generating better food sources, a hybrid algorithm (Tabu-ABC) is proposed in this paper.

Tabu-ABC (Algorithm 3) starts by generating random solutions as food sources and associating each source with employed bees. Before this associating event, Tabu search is applied to improve the initial solutions (food sources). Then, each employed bee determines new food sources in the neighborhood of its associated food source. If it finds a new and better food source, it will switch from the old source to the new source. After all of the employed bees finish their work, which occurs at a fixed iteration, they share their information with the onlookers. The onlookers then select food sources according to the traditional roulette wheel selection method. After the onlookers have selected their food sources, they explore, as well as evaluate, food sources that are nearby to the chosen food source. For each old food source, if a new and better food source is found, the old source is replaced by the new source. Additionally, a food source is abandoned if it has not shown improvement in a predefined number of iterations. At that time, the employed bee is transformed into a scout and will be associated with a new food source. Here, we apply the Tabu approach again. The similarity of new food sources and old food sources cannot exceed a predefined threshold. New food sources are randomly generated until one of them satisfies the similarity condition. In the context of VRPTWP as stated in this paper, route a's similarity to route b is defined as the length of their longest common subsequence divided by the length of route a. Solution A's similarity to solution B is the average of the highest similarities of each of A's routes to B's routes. The algorithm is terminated when it reaches a predefined number of iterations.

## 4. Computational results

To obtain effective solutions for the VRPTWP and to provide a basis of comparison for further study, we construct a set of benchmarks for the VRPTWP and report their computational results. To generate the benchmarks, we combine two famous instances. The first is the set of instances proposed in [14] for the 3L-CVRP, and the other is the well-known instances of the VRPTW introduced by Solomon [32]. The position information in the 3L-CVRP instances is replaced one by one with the 27 instances (C1, C2, and R101 - R110) of Solomon. At the same time, the time window information is also imported. To maintain the feasibility of the instances, the maximum number of vehicles allowed is double. We generated a set of benchmarks[1] and tested the proposed approach. We believe that the computational results obtained can be used as baseline approaches for any future developments in this field.

In this section, the computational results on the VRPTWP and VRPTW are presented. The proposed algorithm is coded in C++ and runs on a machine with an Intel Core i5 CPU, 2.6 GHz/8 G of RAM.

[1] https://github.com/maomiT/VRPTWP

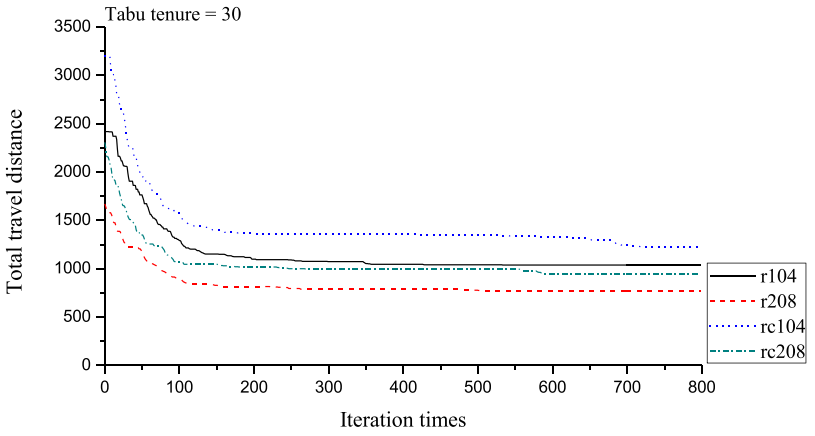



**Fig. 5.** Process of Tabu search on some instances.

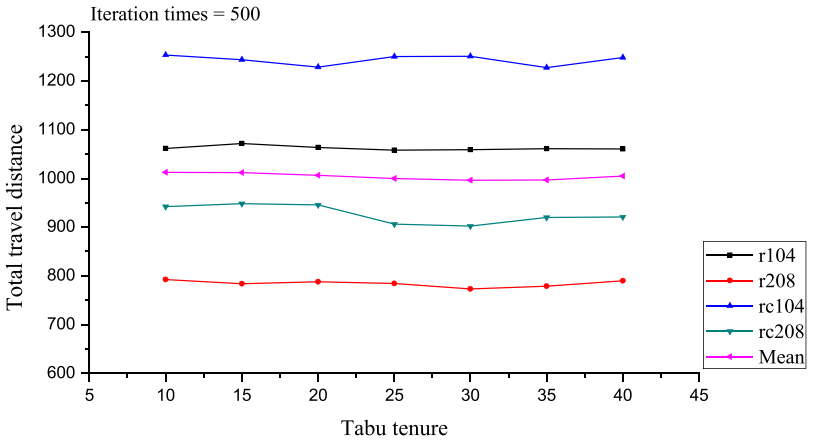



**Fig. 6.** Results on different *Tabu tenure*.

### *4.1. Sensitivity analysis of the parameters*

To analyze the influence of the parameters, we performed many experiments with different settings. To find the relationship between the time consumption and the quality of the solutions, we applied Tabu search on four instances with 800 iterations in 10 runs. Fig. 5 presents the process of Tabu search on instances r104, r208, rc104 and rc208. The horizontal and vertical axes represent the number of iterations and the total travel distance (the cost of the solution). Fig. 5 shows that the improvements achieved are very small after 300 iterations. The lowest total travel distances found for each instance are 1037.95, 758.38, 1224.74 and 946.74, respectively. In Fig. 6, the average results from different Tabu tenures are shown. Obviously, the travel distance with the Tabu tenure of 30 is shorter than the travel distance with the other Tabu tenures in most cases. To test more parameters of Tabu-ABC, Fig. 7 presents the influence of the update restriction (the maximum allowable number of iterations without an update), and Fig. 8 presents the influence of the similarity restriction (the least similarity that allows scout bees to accept a new source). According to Fig. 7, the update restriction and similarity restriction show little influence on the final results. Although different values of the update restriction could cause a large influence on specific instances (such as r104 and rc208), they still show similar results on average. When the similarity restriction is 0.7, the average result of the four instances is the shortest, and the results are similar to the other sets. In fact, according to the figures, we can find that Tabu-ABC is robust, and the parameters show a minor effect on the total travel distance. For the time cost, for each test on the VRPTWP instance, the length of the Tabu tenure is set to 30, and Tabu search is conducted for 300 iterations. The group size of the food sources is also set to 10, and the limit of the number of no updating iterations is set to $10n$ ($n$ is the number of customers). The limit on the sources' similarity is set to 0.7, and the ABC is conducted for $50n$ iterations. For each test on the VRPTW, Tabu search is conducted for 500 iterations, while the population size of the food sources is set to 20. The ABC is conducted for $200n$ iterations, and the other parameters' settings are the same as the settings on the VRPTWP instances. The parameter settings are shown in Table 1.

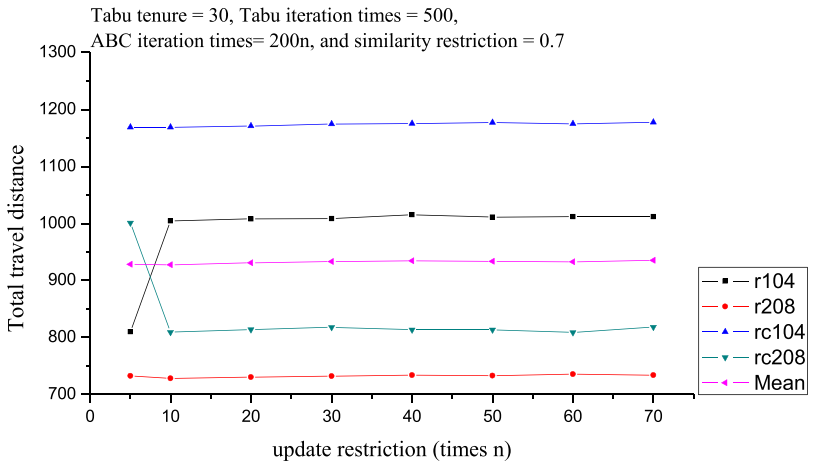



**Fig. 7.** Results on different update restriction.

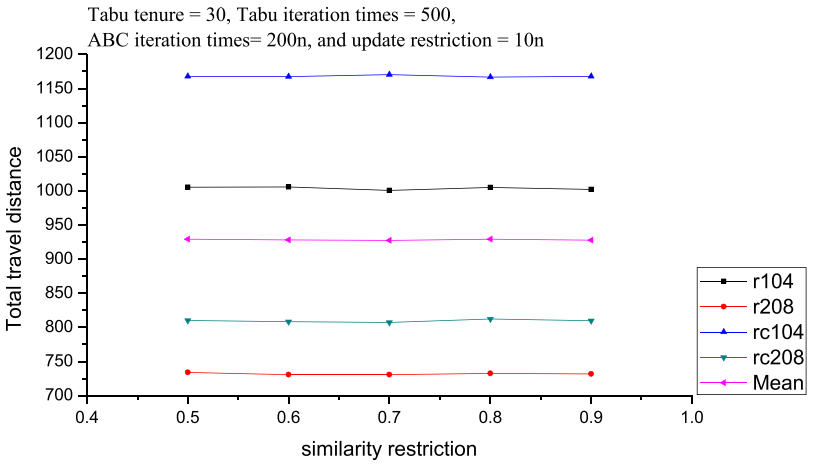



**Fig. 8.** Results on different similarity restriction.

**Table 1**
Parameter settings.

| | |
|---|---|
| Tabu tenure | 30 |
| Tabu search iteration times | 300 in the VRPTWP and 500 in the VRPTW |
| Group size of food sources | 10 in the VRPTWP and 20 in the VRPTW |
| Limit of no update times | $10n$, $n=$ the number of customers |
| The limit of sources' similarity | 0.7 |
| ABC iteration times | $50n$ in the VRPTWP and $200n$ in the VRPTW<br>$n=$ the number of customers |

### *4.2. Computational results on the VRPTWP instances*

The results on the VRPTWP are presented in Table 2, in which "NV" represents the number of vehicles, "TD" means the total travel distance (solution cost), and "CPU" denotes the computational time (in second). The algorithm has allocated 10 independent runs for each instance. The proposed Tabu-ABC algorithm is computationally expensive because it addresses three-dimensional loading problems. Some computational results on the three-dimensional loading problem are presented in [44]. According to Table 2, a greater fleet size does not necessarily result in a greater cost. As seen in VRPTWP15 and VRPTWP22, the lowest cost actually results in a greater fleet size compared with the average fleet size over all of the runs. Similar situations are shown in Tables 5 and 6. In fact, the cost of buying more vehicles is far cheaper than the cost of traveling more distance in the long term.

### *4.3. Computational results on the VRPTW instances*

The main contributions of our work consist of the new VRPTWP problem and a novel strategy called Tabu-ABC. To evaluate the performance of the algorithm, we performed extensive experiments on the set of Solomon's 100 customers

**Table 2**
Results of the VRPTWP.

| Data set | Best | | Mean | | Gap | Sd | CPU |
|---|---|---|---|---|---|---|---|
| | NV | TD | NV | TD | | | |
| VRPTWP01 | 5 | 322.33 | 5 | 322.33 | 0 | 0 | 325.10 |
| VRPTWP02 | 5 | 295.29 | 5 | 299.89 | 0.02 | 3.61 | 232.28 |
| VRPTWP03 | 5 | 303.60 | 5 | 303.60 | 0 | 0 | 430.27 |
| VRPTWP04 | 6 | 380.19 | 6 | 380.80 | 0 | 0.57 | 371.88 |
| VRPTWP05 | 7 | 416.35 | 7 | 417.66 | 0 | 2.95 | 545.72 |
| VRPTWP06 | 7 | 408.99 | 7 | 409.09 | 0 | 0.07 | 306.38 |
| VRPTWP07 | 7 | 407.91 | 7 | 409.89 | 0 | 1.81 | 604.70 |
| VRPTWP08 | 7 | 425.90 | 8 | 425.90 | 0 | 0 | 637.32 |
| VRPTWP09 | 8 | 530.50 | 10 | 532.63 | 0 | 1.80 | 549.60 |
| VRPTWP10 | 10 | 668.74 | 11 | 669.11 | 0 | 0.83 | 956.40 |
| VRPTWP11 | 11 | 619.34 | 9 | 626.27 | 0.01 | 4.21 | 1032.98 |
| VRPTWP12 | 9 | 688.60 | 10.40 | 690.10 | 0 | 1.56 | 671.37 |
| VRPTWP13 | 11 | 626.72 | 8.40 | 630.43 | 0.01 | 4.40 | 1347.24 |
| VRPTWP14 | 8 | 765.37 | 12 | 766.78 | 0 | 1.93 | 1221.68 |
| VRPTWP15 | 12 | 713.23 | 10.60 | 715.72 | 0 | 3.41 | 1091.86 |
| VRPTWP16 | 11 | 746.67 | 11.80 | 748.15 | 0 | 0.83 | 429.32 |
| VRPTWP17 | 15 | 994.28 | 15.20 | 997.27 | 0 | 4.55 | 484.99 |
| VRPTWP18 | 18 | 1206.51 | 18 | 1207.60 | 0 | 1.49 | 636.18 |
| VRPTWP19 | 16 | 1211.99 | 15.60 | 1217.63 | 0 | 4.87 | 783.59 |
| VRPTWP20 | 24 | 1644.56 | 23.20 | 1652.96 | 0.01 | 6.52 | 1512.11 |
| VRPTWP21 | 23 | 1603.88 | 22 | 1612.42 | 0.01 | 9.29 | 2174.74 |
| VRPTWP22 | 26 | 1811.19 | 26.20 | 1817.75 | 0 | 5.83 | 2170.98 |
| VRPTWP23 | 24 | 1654.13 | 24 | 1675.25 | 0.01 | 15.22 | 1950.79 |
| VRPTWP24 | 21 | 1644.55 | 21.40 | 1649.61 | 0 | 7.22 | 1745.54 |
| VRPTWP25 | 27 | 1836.02 | 26.80 | 1856.52 | 0.01 | 16.43 | 2877.48 |
| VRPTWP26 | 32 | 2144.47 | 32 | 2160.37 | 0.01 | 10.42 | 3250.49 |
| VRPTWP27 | 28 | 2002.63 | 29.60 | 2030.71 | 0.01 | 21.35 | 3037.26 |

Gap = (Mean TD − Best TD)/Best TD, and Sd is the standard deviation of 15 times TD.

VRPTW, and the results were compared with other heuristic approaches. Solomon's VRPTW was divided into six sets, C1, C2, R1, R2, RC1 and RC2. The customers in sets C1 and C2 are clustered in groups. In sets R1 and R2, they are uniformly distributed, and in sets RC1 and RC2, they are semi-clustered. The proposed algorithm was executed for 10 independent runs on each instance.

To prove the effectiveness, Tabu-ABC was compared with nine other heuristic approaches. The comparison is presented in Table 3, which shows the average best total distance and the number of vehicles for each set. Obviously, the proposed algorithm achieves the best results for C1, R2 and RC2, and the average result is also better than some other heuristics. In Table 4, we compare the average mean total distance and the number of vehicles for each set. Our algorithm still achieves the best average solution for four sets (R1, R2, RC1 and RC2).

In Table 5, we compare Tabu-ABC with heuristics from some recently published papers on the VRPTW. From the experiment results, we can observe that in most instances, the proposed algorithm achieves the best solutions. We also compare our work with best-known solutions, which are listed in Table 6. Most of the best-known results are summarized in [42]. Some of these results are updated based on our recent findings. From Table 6, we can observe that our algorithm achieves better solutions in 15 instances. In 4 instances, it achieves the best-known solution. The performance of the proposed algorithm is also similar to that of the best-known solutions in other instances. In addition, Tabu-ABC achieves better results on four specific instances in Fig. 5, which reflects that Tabu-ABC improves the pure Tabu effectively. Because the CPU time in Table 6 is large, we test the algorithm with different parameters. From Table 7, the algorithm achieves good results with less CPU time. The results also indicate that the parametric values slightly influence the performance of the Tabu-ABC approach.

## 5. Conclusions

This paper introduces a new vehicle routing problem with time window and pallet loading constraints (VRPTWP), which was taken from the logistics industry. The VRPTWP consists of two sub NP-hard problems, namely, the three-dimensional loading problem and the VRPTW. In addition, the VRPTWP addressed in this paper considers time window constraints and closely reflects real-world situations. To the best of our knowledge, the VRPTWP is a new problem that has never been addressed before. To find the best solutions for the VRPTWP, a new algorithm called Tabu-ABC is proposed in this paper. Tabu-ABC is a hybrid algorithm based on Tabu search and the ABC. We also created a set of new benchmarks for the VRPTWP. The experimental results show that the proposed approach is highly effective in comparison to other heuristics on Solomon's VRPTW instances. For future work, we would like to develop more efficient approaches to solve the VRPTWP, especially in terms of improving the CPU time cost of Tabu-ABC. Because Tabu-ABC can be easily modified and extended for different requirements, we are planning to use this approach in other problem areas.

## Acknowledgments

The authors would like to thank the anonymous referees for their valuable comments that help improve this paper. This work is partially supported by the National Nature Science Foundation of China (Grant no. 61672439 and 61272003) and by the University of Macau under grant MYRG2015-00054-FST, and is also partially supported by a Seed-Corn grant awarded by the Chartered Institute of Logistics and Transport and two grants from the British Council, a UK-ASEAN Knowledge Partnership grant and a Newton Institutional Links grant.

## Appendix

(Tables 3–7).

**Table 3**
Comparison among different heuristics on the VRPTW (Average of best solutions).

| Date set | | Tan et al. (2006) [35] | Yu et al. (2011) [42] | Cordeau and Maischberger (2012) [10] | Gong et al. (2012) [15] |
|---|---|---|---|---|---|
| C1 | NV | 10 | 10 | 10 | 10 |
| | TD | 828.71 | 829.01 | 828.38 | 835.91 |
| C2 | NV | 3 | 3.3 | 3 | 3 |
| | TD | 590.07 | 590.78 | 589.86 | 593.41 |
| R1 | NV | 12.92 | 13.1 | 12 | 12.58 |
| | TD | 1187.35 | 1196.96 | 1209.19 | 1232.28 |
| R2 | NV | 3.55 | 4.6 | 2.73 | 3 |
| | TD | 951.74 | 951.36 | 951.17 | 1016.66 |
| RC1 | NV | 12.38 | 12.7 | 11.5 | 12.13 |
| | TD | 1355.37 | 1380.55 | 1385.9 | 1385.47 |
| RC2 | NV | 4.25 | 5.6 | 3.25 | 3.38 |
| | TD | 1068.26 | 1095.84 | 1120.53 | 1169.07 |
| Avg. | NV | 6.59 | 7.04 | 6.07 | 6.30 |
| | TD | 996.92 | 1007.42 | 1014.17 | 1038.80 |
| | | Barbucha. (2014) [2] | Luo et al. (2015) [28] | Yassen et al. (2015) [41] | Tabu-ABC |
| C1 | NV | 10 | 10 | – | 10 |
| | TD | 828.38 | 828.38 | 838.47 | 828.38 |
| C2 | NV | 3 | 3 | – | 3 |
| | TD | 589.86 | 589.86 | 605.41 | 590.39 |
| R1 | NV | 11.92 | 11.92 | – | 13.75 |
| | TD | 1232.13 | 1210.34 | 1207.76 | 1187.90 |
| R2 | NV | 3.09 | 2.73 | – | 4.64 |
| | TD | 922.48 | 951.03 | 977.19 | 891.24 |
| RC1 | NV | 12 | 11.5 | – | 13.13 |
| | TD | 1355.36 | 1384.16 | 1381.96 | 1361.08 |
| RC2 | NV | 3.38 | 3.25 | – | 5.5 |
| | TD | 1106 | 1119.24 | 1099.12 | 1017.47 |
| Avg. | NV | 7.23 | 7.07 | – | 8.34 |
| | TD | 1005.70 | 1013.84 | 1018.32 | 979.41 |

**Table 4**
Comparison among different heuristics on the VRPTW (average of mean solutions).

| Date set | | Chiang and Russell (1997) [7] | Lau et al. (2003) [24] | Tan et al. (2006) [35] | Yu et al. (2011) [42] |
|---|---|---|---|---|---|
| C1 | NV | 10 | 10 | – | 10 |
| | TD | 828.38 | 828.38 | 837.21 | 841.92 |
| C2 | NV | 3 | 3 | – | 3.3 |
| | TD | 591.42 | 589.86 | 632.42 | 612.75 |
| R1 | NV | 12.17 | 12 | – | 13.1 |
| | TD | 1204.19 | 1217.73 | 1240.31 | 1213.16 |
| R2 | NV | 2.73 | 2.73 | – | 4.6 |
| | TD | 986.32 | 967.75 | 1068.57 | 952.3 |
| RC1 | NV | 11.88 | 11.63 | – | 12.7 |
| | TD | 1397.44 | 1382.42 | 1381.23 | 1415.62 |
| RC2 | NV | 3.25 | 3.25 | – | 5.6 |
| | TD | 1229.54 | 1129.19 | 1154.88 | 1120.37 |
| Avg. | NV | 7.17 | 7.10 | – | 8.22 |
| | TD | 1039.55 | 1019.22 | 1052.44 | 1026.02 |
| | | Cordeau and Maischberger (2012) [10] | Gong et al. (2012) [15] | Luo et al. (2015) [28] | Tabu-ABC |
| C1 | NV | 10 | 10 | 10 | 10 |
| | TD | 828.94 | 856.44 | 828.38 | 828.73 |

(*continued on next page*)

**Table 4** (*continued*)

| Date set | | Chiang and Russell (1997) [7] | Lau et al. (2003) [24] | Tan et al. (2006) [35] | Yu et al. (2011) [42] |
|---|---|---|---|---|---|
| C2 | NV | 3 | 3.03 | 3 | 3 |
| | TD | 590.85 | 612.93 | 589.86 | 591.45 |
| R1 | NV | 12.02 | 13.01 | 11.92 | 13.82 |
| | TD | 1213.57 | 1263.25 | 1210.75 | 1195.49 |
| R2 | NV | 2.73 | 3.1 | 2.7 | 4.5 |
| | TD | 959.62 | 1073.72 | 951.51 | 902.88 |
| RC1 | NV | 11.55 | 12.66 | 11.5 | 13.56 |
| | TD | 1386.39 | 1400.97 | 1384.62 | 1373.25 |
| RC2 | NV | 3.25 | 3.59 | 3.25 | 5.47 |
| | TD | 1130.27 | 1228.95 | 1119.63 | 1028.92 |
| Avg. | NV | 7.09 | 7.57 | 7.06 | 8.39 |
| | TD | 1018.27 | 1072.71 | 1014.13 | 986.79 |

**Table 5**
Comparison among four heuristics.

| Algorithm | HSFLA (2015) [28] | | CPLA (2014) [2] | | PITSH (2012) [10] | | Tabu-ABC | |
|---|---|---|---|---|---|---|---|---|
| Data set | TD | NV | TD | NV | TD | NV | TD | NV |
| C101 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 |
| C102 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 |
| C103 | 828.06 | 10 | 828.06 | 10 | 828.07 | 10 | 828.07 | 10 |
| C104 | 824.78 | 10 | 824.78 | 10 | 824.78 | 10 | 824.78 | 10 |
| C105 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 |
| C106 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 |
| C107 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 |
| C108 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 |
| C109 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 | 828.94 | 10 |
| C201 | 591.56 | 3 | 591.56 | 3 | 591.56 | 3 | 591.56 | 3 |
| C202 | 591.56 | 3 | 591.56 | 3 | 591.56 | 3 | 591.56 | 3 |
| C203 | 591.17 | 3 | 591.17 | 3 | 591.17 | 3 | 591.17 | 3 |
| C204 | 590.6 | 3 | 590.6 | 3 | 590.6 | 3 | 594.89 | 3 |
| C205 | 588.88 | 3 | 588.88 | 3 | 588.88 | 3 | 588.88 | 3 |
| C206 | 588.49 | 3 | 588.49 | 3 | 588.49 | 3 | 588.49 | 3 |
| C207 | 588.29 | 3 | 588.29 | 3 | 588.29 | 3 | 588.29 | 3 |
| C208 | 588.32 | 3 | 588.32 | 3 | 588.32 | 3 | 588.32 | 3 |
| R101 | 1650.8 | 10 | 1656.21 | 19 | 1650.8 | 19 | 1643.18 | 20 |
| R102 | 1486.12 | 17 | 1501.97 | 17 | 1486.12 | 17 | 1460.26 | 18 |
| R103 | 1292.67 | 13 | 1295.6 | 13 | 1294.23 | 13 | 1217.39 | 15 |
| R104 | 1007.31 | 9 | 1017.38 | 9 | 981.2 | 10 | 987.61 | 11 |
| R105 | 1377.11 | 14 | 1381.89 | 14 | 1377.11 | 14 | 1363.91 | 15 |
| R106 | 1252.03 | 12 | 1258.76 | 12 | 1252.62 | 12 | 1247.90 | 13 |
| R107 | 1104.66 | 10 | 1117.85 | 10 | 1104.66 | 10 | 1087.50 | 12 |
| R108 | 960.88 | 9 | 976.06 | 9 | 963.99 | 9 | 961.85 | 11 |
| R109 | 1194.73 | 11 | 1229.71 | 11 | 1194.73 | 11 | 1152.99 | 13 |
| R110 | 1118.84 | 10 | 1196.49 | 10 | 1118.84 | 10 | 1091.50 | 12 |
| R111 | 1096.73 | 10 | 1123.64 | 10 | 1096.73 | 10 | 1067.46 | 12 |
| R112 | 982.14 | 9 | 1030.02 | 9 | 989.27 | 9 | 973.25 | 10 |
| R201 | 1252.37 | 4 | 1253.02 | 4 | 1252.37 | 4 | 1174.69 | 6 |
| R202 | 1191.7 | 3 | 1086.08 | 4 | 1191.7 | 3 | 1046.10 | 5 |
| R203 | 939.5 | 3 | 945.8 | 3 | 941.08 | 3 | 884.02 | 5 |
| R204 | 825.52 | 2 | 752.13 | 3 | 825.52 | 2 | 750.40 | 4 |
| R205 | 994.43 | 3 | 1017.93 | 3 | 994.43 | 3 | 960.75 | 5 |
| R206 | 906.14 | 3 | 920.37 | 3 | 906.14 | 3 | 900.97 | 4 |
| R207 | 890.61 | 2 | 815.26 | 3 | 890.61 | 2 | 809.72 | 4 |
| R208 | 726.82 | 2 | 729.42 | 2 | 726.82 | 2 | 723.14 | 5 |
| R209 | 909.16 | 3 | 916.33 | 3 | 909.16 | 3 | 863.12 | 5 |
| R210 | 939.37 | 3 | 943.1 | 3 | 939.37 | 3 | 927.54 | 5 |
| R211 | 885.71 | 2 | 767.82 | 3 | 885.71 | 2 | 763.22 | 4 |
| RC101 | 1696.95 | 14 | 1626.09 | 15 | 1696.95 | 14 | 1646.17 | 16 |
| RC102 | 1554.75 | 12 | 1486.17 | 13 | 1554.75 | 12 | 1481.61 | 14 |
| RC103 | 1261.67 | 11 | 1268.79 | 11 | 1261.67 | 11 | 1280.76 | 12 |
| RC104 | 1135.48 | 10 | 1136.27 | 10 | 1135.48 | 10 | 1162.03 | 11 |
| RC105 | 1629.44 | 13 | 1542.29 | 14 | 1633.72 | 13 | 1545.30 | 16 |
| RC106 | 1424.73 | 11 | 1394.1 | 12 | 1424.73 | 11 | 1401.17 | 14 |
| RC107 | 1230.48 | 11 | 1234.06 | 11 | 1232.2 | 11 | 1235.28 | 12 |
| RC108 | 1139.82 | 10 | 1155.1 | 10 | 1147.69 | 10 | 1136.35 | 11 |
| RC201 | 1406.94 | 4 | 1435.27 | 4 | 1406.94 | 4 | 1271.78 | 7 |
| RC202 | 1365.64 | 3 | 1162.8 | 4 | 1367.09 | 3 | 1116.21 | 6 |
| RC203 | 1049.62 | 3 | 1062.32 | 3 | 1050.64 | 3 | 941.81 | 5 |
| RC204 | 798.46 | 3 | 799.08 | 3 | 798.46 | 3 | 801.87 | 4 |
| RC205 | 1297.65 | 4 | 1303.68 | 4 | 1297.65 | 4 | 1165.82 | 7 |
| RC206 | 1146.32 | 3 | 1155.33 | 3 | 1153.61 | 3 | 1072.85 | 5 |
| RC207 | 1061.14 | 3 | 1095.37 | 3 | 1061.14 | 3 | 977.11 | 5 |
| RC208 | 828.14 | 3 | 834.16 | 3 | 828.71 | 3 | 792.33 | 5 |

**Table 6**
Detail results of our algorithm and comparison with best-known solutions.

| | Best-known | | | This work | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Data set | NV | TD | Authors | Best TD | NV | Mean TD | NV | Gap | Sd | CPU |
| C101 | 10 | 827.3 | Desrochers, Desrosiers, and Solomon (1992) [12] | 828.94 | 10 | 828.94 | 10 | 0 | 0 | 3592.54 |
| C102 | 10 | 827.3 | Desrochers et al. (1992) [12] | 828.94 | 10 | 828.94 | 10 | 0 | 0 | 668.07 |
| C103 | 10 | 826.3 | Tavares, Machado, Pereira and Costa (2003) [37] | 828.07 | 10 | 828.07 | 10 | 0.00 | 0.00 | 252.88 |
| C104 | 10 | 822.9 | Tavares et al. (2003) [37] | 824.78 | 10 | 827.93 | 10 | 0.54 | 5.57 | 140.18 |
| C105 | 10 | 827.3 | Tavares et al. (2003) [37] | 828.94 | 10 | 828.94 | 10 | 0 | 0 | 1204.79 |
| C106 | 10 | 827.3 | Desrochers et al. (1992) [12] | 828.94 | 10 | 828.94 | 10 | 0 | 0 | 1184.74 |
| C107 | 10 | 827.3 | Tavares et al. (2003) [37] | 828.94 | 10 | 828.94 | 10 | 0 | 0 | 782.00 |
| C108 | 10 | 827.3 | Tavares et al. (2003) [37] | 828.94 | 10 | 828.94 | 10 | 0 | 0 | 498.37 |
| C109 | 10 | 827.3 | Tavares et al. (2003) [37] | 828.94 | 10 | 828.94 | 10 | 0 | 0 | 252.69 |
| C201 | 3 | 589.1 | Cook and Rich (1999) [8] | 591.56 | 3 | 591.56 | 3 | 0 | 0 | 4155.89 |
| C202 | 3 | 589.1 | Cook and Rich (1999) [8] | 591.56 | 3 | 591.56 | 3 | 0 | 0 | 630.48 |
| C203 | 3 | 591.17 | Li and Lim (2003) [27] | 591.17 | 3 | 591.31 | 3 | 0.35 | 3.66 | 278.99 |
| C204 | 3 | 590.6 | Potvin and Bengio (1996) [29] | 594.89 | 3 | 603.16 | 3 | 2.95 | 10.12 | 152.45 |
| C205 | 3 | 588.88 | Potvin and Bengio (1996) [29] | 588.88 | 3 | 588.88 | 3 | 0.00 | 0.00 | 1731.04 |
| C206 | 3 | 588.49 | Lau et al. (2003) [24] | 588.49 | 3 | 588.49 | 3 | 0.00 | 0.00 | 1181.93 |
| C207 | 3 | 588.29 | Rochat and Taillard (1995) [30] | 588.29 | 3 | 588.29 | 3 | 0 | 0 | 853.12 |
| C208 | 3 | 588.03 | Tan, Chew and Lee. (2006) [35] | 588.32 | 3 | 588.32 | 3 | 0 | 0 | 476.75 |
| R101 | 18 | 1607.7 | Desrochers et al. (1992) [12] | 1643.18 | 20 | 1645.82 | 20 | 0.39 | 5.48 | 1137.93 |
| R102 | 17 | 1434 | Desrochers et al. (1992) [12] | 1460.26 | 18 | 1463.91 | 18.08 | 0.43 | 4.00 | 434.26 |
| R103 | 13 | 1175.67 | Lau, Lim, and Liu (2001) [23] [21] | 1217.39 | 15 | 1223.27 | 14.92 | 0.30 | 2.41 | 398.70 |
| R104 | 10 | 974.2 | Tan et al. (2006) [35] | 987.61 | 11 | 1002.54 | 11.33 | 0.92 | 6.04 | 227.74 |
| R105 | 15 | 1346.12 | Kallehauge, Larsen, and Madsen (2006) [19] | 1363.91 | 15 | 1372.01 | 15.75 | 0.68 | 5.45 | 641.26 |
| R106 | 13 | 1234.6 | Cook and Rich (1999) [8] | 1247.90 | 13 | 1256.45 | 13.42 | 0.93 | 4.76 | 457.90 |
| R107 | 11 | 1051.84 | Kallehauge et al. (2006) [19] | 1087.50 | 12 | 1097.41 | 12.00 | 1.03 | 4.46 | 336.94 |
| R108 | 10 | 954.03 | Tan et al. (2006) [35] | 961.85 | 11 | 965.82 | 10.83 | 1.25 | 5.91 | 220.71 |
| R109 | 12 | 1013.2 | Chiang and Russell (1997) [7] | 1152.99 | 13 | 1163.07 | 13.00 | 1.23 | 7.20 | 365.31 |
| R110 | 12 | 1068 | Cook and Rich (1999) [8] | 1091.50 | 12 | 1100.83 | 12.17 | 1.49 | 8.51 | 340.76 |
| R111 | 12 | 1048.7 | Cook and Rich (1999) [8] | 1067.46 | 12 | 1076.61 | 12 | 0.98 | 7.16 | 302.90 |
| R112 | 10 | 953.63 | Rochat and Taillard (1995) [30] | 973.25 | 10 | 978.16 | 10.67 | 0.71 | 6.47 | 240.36 |
| R201 | 8 | 1198.15 | Tan et al. (2001) [36] | 1174.69 | 6 | 1178.90 | 6.08 | 0.46 | 4.85 | 547.67 |
| R202 | 6 | 1077.66 | Tan et al. (2001) [36] | 1046.10 | 5 | 1053.44 | 5.08 | 1.41 | 7.88 | 331.63 |
| R203 | 5 | 933.286 | Tan et al. (2001) [36] | 884.02 | 5 | 896.05 | 4.92 | 1.43 | 7.77 | 273.62 |
| R204 | 3 | 752.13 | Barbucha (2014) [2] | 750.40 | 4 | 758.13 | 4 | 1.52 | 6.50 | 230.77 |
| R205 | 3 | 994.42 | Rousseau, Gendreau, and Pesant (2002) [31] | 960.75 | 5 | 975.83 | 4.92 | 1.36 | 7.61 | 290.46 |
| R206 | 3 | 833 | Thangiah, Osman, and Sun (1994) [38] | 900.97 | 4 | 908.18 | 4.25 | 1.61 | 6.76 | 226.36 |
| R207 | 3 | 814.78 | Rochat and Taillard (1995) [30] | 809.72 | 4 | 826.74 | 4 | 2.15 | 11.31 | 234.11 |
| R208 | 2 | 729.42 | Barbucha (2014) [2] | 723.14 | 5 | 732.31 | 3.42 | 1.37 | 6.99 | 198.33 |
| R209 | 3 | 855 | Thangiah et al. (1994) [38] | 863.12 | 5 | 882.22 | 4.92 | 1.35 | 6.71 | 226.84 |
| R210 | 3 | 943.10 | Barbucha (2014) [2] | 927.54 | 5 | 938.63 | 4.92 | 2.09 | 9.31 | 232.11 |
| R211 | 2 | 767.82 | Barbucha (2014) [2] | 763.22 | 4 | 781.23 | 4 | 2.18 | 7.61 | 189.63 |
| RC101 | 15 | 1619.8 | Kohl, Desrosiers, Madsen, Solomon, and Soumis (1999) [22] | 1646.17 | 16 | 1656.01 | 16.33 | 0.66 | 6.93 | 663.29 |
| RC102 | 13 | 1470.26 | Tan et al. (2006) [35] | 1481.61 | 14 | 1488.79 | 14.83 | 0.81 | 7.29 | 441.74 |
| RC103 | 12 | 1196.12 | Tan et al. (2006) [35] | 1280.76 | 12 | 1298.32 | 12.08 | 1.94 | 18.50 | 308.63 |
| RC104 | 10 | 1135.48 | Cordeau, Laporte, and Mercier (2001) [9] | 1162.03 | 11 | 1168.13 | 11.00 | 0.60 | 5.43 | 217.15 |
| RC105 | 14 | 1542.29 | Barbucha (2014) [2] | 1545.30 | 16 | 1554.79 | 16.17 | 1.57 | 8.50 | 545.30 |
| RC106 | 13 | 1371.69 | Tan et al. (2006) [35] | 1401.17 | 14 | 1413.38 | 14.00 | 0.82 | 7.41 | 301.56 |
| RC107 | 11 | 1222.16 | Tan et al. (2006) [35] | 1235.28 | 12 | 1256.98 | 12.50 | 1.96 | 11.91 | 277.52 |
| RC108 | 11 | 1133.82 | Luo, Li, Chen and Liu (2015) [28] | 1136.35 | 11 | 1149.65 | 11.17 | 1.67 | 9.76 | 235.20 |
| RC201 | 6 | 1134.91 | Tan et al. (2006) [35] | 1271.78 | 7 | 1284.59 | 6.92 | 0.78 | 5.71 | 476.51 |
| RC202 | 5 | 1130.53 | Tan et al. (2006) [35] | 1116.21 | 6 | 1122.97 | 5.75 | 0.71 | 5.25 | 350.15 |
| RC203 | 4 | 1026.61 | Tan et al. (2006) [35] | 941.81 | 5 | 951.30 | 5.08 | 0.79 | 6.33 | 226.52 |
| RC204 | 3 | 799.08 | Barbucha (2014) [2] | 801.87 | 4 | 809.09 | 4 | 2.22 | 8.13 | 162.89 |
| RC205 | 5 | 1295.46 | Tan et al. (2006) [35] | 1165.82 | 7 | 1172.80 | 7 | 2.23 | 16.34 | 335.38 |
| RC206 | 4 | 1112.2 | Yu, Yang and Yao. (2011) [42] | 1072.85 | 5 | 1082.93 | 5.50 | 1.15 | 8.40 | 272.82 |
| RC207 | 4 | 1040.67 | Tan et al. (2006) [35] | 977.11 | 5 | 998.46 | 5.42 | 2.34 | 9.82 | 211.82 |
| RC208 | 3 | 829.69 | Rousseau et al. (2002) [31] | 792.33 | 5 | 809.23 | 4.67 | 2.11 | 8.65 | 191.15 |

Gap = (Mean TD − Best TD)/Best TD, and Sd is the standard deviation of 15 times TD.

**Table 7**
Detail comparison between two parameters setting.

| Data set | Tabu tenure = 30, Tabu iteration times = 500, ABC iteration times = 200$n$, similarity restriction = 0.7 and update restriction = 10$n$ | | | | | Tabu tenure = 30, Tabu iteration times = 500, ABC iteration times = 200$n$, similarity restriction = 0.6, and update restriction = 50$n$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Best TD | NV | Mean TD | NV | CPU | Best TD | NV | Mean TD | NV | CPU |
| C101 | 828.94 | 10 | 828.94 | 10 | 3592.54 | 828.94 | 10 | 828.94 | 10 | 1070.42 |
| C102 | 828.94 | 10 | 828.94 | 10 | 668.07 | 828.94 | 10 | 828.94 | 10 | 280.49 |
| C103 | 828.07 | 10 | 828.07 | 10 | 252.88 | 828.07 | 10 | 828.07 | 10 | 116.66 |
| C104 | 824.78 | 10 | 827.93 | 10 | 140.18 | 824.78 | 10 | 829.26 | 10 | 60.94 |
| C105 | 828.94 | 10 | 828.94 | 10 | 1204.79 | 828.94 | 10 | 828.94 | 10 | 478.01 |
| C106 | 828.94 | 10 | 828.94 | 10 | 1184.74 | 828.94 | 10 | 828.94 | 10 | 389.41 |
| C107 | 828.94 | 10 | 828.94 | 10 | 782.00 | 828.94 | 10 | 828.94 | 10 | 322.39 |
| C108 | 828.94 | 10 | 828.94 | 10 | 498.37 | 828.94 | 10 | 828.94 | 10 | 185.52 |
| C109 | 828.94 | 10 | 828.94 | 10 | 252.69 | 828.94 | 10 | 828.94 | 10 | 83.06 |
| C201 | 591.56 | 3 | 591.56 | 3 | 4155.89 | 591.56 | 3 | 591.56 | 3 | 1337.71 |
| C202 | 591.56 | 3 | 591.56 | 3 | 630.48 | 591.56 | 3 | 591.56 | 3 | 348.48 |
| C203 | 591.17 | 3 | 591.31 | 3 | 278.99 | 591.17 | 3 | 593.21 | 3 | 132.09 |
| C204 | 594.89 | 3 | 603.16 | 3 | 152.45 | 590.6 | 3 | 608.03 | 3 | 53.45 |
| C205 | 588.88 | 3 | 588.88 | 3 | 1731.04 | 588.88 | 3 | 588.88 | 3 | 619.37 |
| C206 | 588.49 | 3 | 588.49 | 3 | 1181.93 | 588.49 | 3 | 588.49 | 3 | 473.96 |
| C207 | 588.29 | 3 | 588.29 | 3 | 853.12 | 588.29 | 3 | 588.29 | 3 | 322.58 |
| C208 | 588.32 | 3 | 588.32 | 3 | 476.75 | 588.32 | 3 | 588.32 | 3 | 234.39 |
| R101 | 1643.18 | 20 | 1645.82 | 20 | 1137.93 | 1644.5 | 20 | 1650.89 | 20.27 | 521.88 |
| R102 | 1460.26 | 18 | 1463.91 | 18.08 | 434.26 | 1463.52 | 18 | 1469.81 | 18.07 | 187.46 |
| R103 | 1217.39 | 15 | 1223.27 | 14.92 | 398.70 | 1224.44 | 15 | 1228.17 | 14.8 | 141.13 |
| R104 | 987.61 | 11 | 1002.54 | 11.33 | 227.74 | 1001.89 | 12 | 1011.12 | 11.8 | 85.29 |
| R105 | 1363.91 | 15 | 1372.01 | 15.75 | 641.26 | 1369.07 | 16 | 1378.44 | 15.93 | 235.14 |
| R106 | 1247.90 | 13 | 1256.45 | 13.42 | 457.90 | 1251.71 | 13 | 1263.34 | 13.73 | 174.85 |
| R107 | 1087.50 | 12 | 1097.41 | 12.00 | 336.94 | 1093.99 | 12 | 1105.3 | 11.93 | 130.82 |
| R108 | 961.85 | 11 | 965.82 | 10.83 | 220.71 | 958.44 | 11 | 970.46 | 10.87 | 87.48 |
| R109 | 1152.99 | 13 | 1163.07 | 13.00 | 365.31 | 1160.06 | 13 | 1174.37 | 13.27 | 162.67 |
| R110 | 1091.50 | 12 | 1100.83 | 12.17 | 340.76 | 1095.34 | 12 | 1111.61 | 12.13 | 154.76 |
| R111 | 1067.46 | 12 | 1076.61 | 12 | 302.90 | 1073.29 | 12 | 1083.78 | 12.2 | 131.07 |
| R112 | 973.25 | 10 | 978.16 | 10.67 | 240.36 | 981.67 | 11 | 988.63 | 10.87 | 118.97 |
| R201 | 1174.69 | 6 | 1178.90 | 6.08 | 547.67 | 1178.91 | 6 | 1184.33 | 6.07 | 196.05 |
| R202 | 1046.10 | 5 | 1053.44 | 5.08 | 331.63 | 1041.48 | 5 | 1056.15 | 5.2 | 119.36 |
| R203 | 884.02 | 5 | 896.05 | 4.92 | 273.62 | 889.4 | 5 | 902.09 | 4.53 | 85.53 |
| R204 | 750.40 | 4 | 758.13 | 4 | 230.77 | 748.15 | 4 | 759.52 | 4 | 69.77 |
| R205 | 960.75 | 5 | 975.83 | 4.92 | 290.46 | 968.89 | 5 | 982.09 | 4.8 | 116.61 |
| R206 | 900.97 | 4 | 908.18 | 4.25 | 226.36 | 897.2 | 4 | 911.63 | 4 | 78.55 |
| R207 | 809.72 | 4 | 826.74 | 4 | 234.11 | 812.61 | 4 | 830.07 | 4 | 73.13 |
| R208 | 723.14 | 5 | 732.31 | 3.42 | 198.33 | 725.58 | 4 | 735.53 | 3.13 | 65.18 |
| R209 | 863.12 | 5 | 882.22 | 4.92 | 226.84 | 872.65 | 5 | 884.44 | 4.93 | 90.78 |
| R210 | 927.54 | 5 | 938.63 | 4.92 | 232.11 | 919.41 | 5 | 938.66 | 4.8 | 95.88 |
| R211 | 763.22 | 4 | 781.23 | 4 | 189.63 | 769.28 | 4 | 786.04 | 4 | 77.49 |
| RC101 | 1646.17 | 16 | 1656.01 | 16.33 | 663.29 | 1650.2 | 16 | 1661.04 | 16.4 | 271.72 |
| RC102 | 1481.61 | 14 | 1488.79 | 14.83 | 441.74 | 1481.48 | 14 | 1493.5 | 14.8 | 193.72 |
| RC103 | 1280.76 | 12 | 1298.32 | 12.08 | 308.63 | 1288.36 | 12 | 1313.42 | 12.13 | 143.69 |
| RC104 | 1162.03 | 11 | 1168.13 | 11.00 | 217.15 | 1174.46 | 11 | 1181.55 | 11.27 | 92.34 |
| RC105 | 1545.30 | 16 | 1554.79 | 16.17 | 545.30 | 1535.94 | 15 | 1560.12 | 16.53 | 195.66 |
| RC106 | 1401.17 | 14 | 1413.38 | 14.00 | 301.56 | 1411.51 | 14 | 1423.01 | 13.87 | 163.77 |
| RC107 | 1235.28 | 12 | 1256.98 | 12.50 | 277.52 | 1236.27 | 12 | 1260.54 | 12.07 | 137.43 |
| RC108 | 1136.35 | 11 | 1149.65 | 11.17 | 235.20 | 1131.4 | 11 | 1150.3 | 11.4 | 132.3 |
| RC201 | 1271.78 | 7 | 1284.59 | 6.92 | 476.51 | 1285.14 | 7 | 1295.15 | 6.73 | 174.15 |
| RC202 | 1116.21 | 6 | 1122.97 | 5.75 | 350.15 | 1116.58 | 6 | 1124.5 | 5.87 | 122.95 |
| RC203 | 941.81 | 5 | 951.30 | 5.08 | 226.52 | 944.87 | 5 | 952.32 | 5 | 82.79 |
| RC204 | 801.87 | 4 | 809.09 | 4 | 162.89 | 791.76 | 4 | 809.33 | 4.2 | 56.04 |
| RC205 | 1165.82 | 7 | 1172.80 | 7 | 335.38 | 1163.29 | 7 | 1189.25 | 6.6 | 115.08 |
| RC206 | 1072.85 | 5 | 1082.93 | 5.50 | 272.82 | 1070.82 | 6 | 1083.09 | 5.47 | 98.9 |
| RC207 | 977.11 | 5 | 998.46 | 5.42 | 211.82 | 980.87 | 5 | 1003.8 | 5.13 | 85.6 |
| RC208 | 792.33 | 5 | 809.23 | 4.67 | 191.15 | 799.11 | 4 | 816 | 4.73 | 81.55 |

## References


[1] S. Akpinar, Hybrid large neighborhood search algorithm for capacitated vehicle routing problem, Expert Syst. Appl. 61 (2016) 28–38.
[2] D. Barbucha, A cooperative population learning algorithm for vehicle routing problem with time windows, Neurocomputing 146 (2014) 210–229.
[3] A.K. Beheshti, S.R. Hejazi, A novel hybrid column generation-metaheuristic approach for the vehicle routing problem with general soft time window, Inf. Sci. 316 (20) (2015) 598–615.
[4] A. Bortfeldt, T. Hahn, D. Männel, L.M. Hybrid, algorithms for the vehicle routing problem with clustered backhauls and 3D loading constraints, Eur. J. Oper. Res. 243 (1) (2015) 82–96.
[5] J. Caceres-Cruz, P. Arias, D. Guimarans, D. Riera, A.A. Juan, Rich vehicle routing problem: survey, ACM Comput. Surv. 47 (2) (2015) 32:1-28.
[6] M. Cherkesly, G. Desaulniers, S. Irnich, G. Laporte, Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and multiple stacks, Eur. J. Oper. Res. 250 (3) (2016) 782–793.
[7] W.C. Chiang, R.A. Russell, A reactive tabu search metaheuristic for the vehicle routing problem with time windows, INFORMS J. Comput. 9 (4) (1997) 417–430.
[8] W. Cook, J.L. Rich, A parallel cutting-plane algorithm for the vehicle routing problem with time windows, Computational and Applied Mathematics Department, Rice University, Houston, TX, 1999 Technical Report.
[9] J.-F. Cordeau, G. Laporte, A. Mercier, A unified tabu search heuristic for vehicle routing problems with time windows, J. Oper. Res. Soc. 52 (8) (2001) 928–936.
[10] J.-F. Cordeau, M. Maischberger, A parallel iterated tabu search heuristic for vehicle routing problems, Comput. Oper. Res. 39 (2012) 2033–2050.
[11] G.B. Dantzig, J.H. Ramser, The truck dispatching problem, Manage. Sci. 6 (1) (1959) 80–91.
[12] M. Desrochers, J. Desrosiers, M. Solomon, A new optimization algorithm for the vehicle routing problem with time windows, Oper. Res. 40 (2) (1992) 342–354.
[13] M. Gendreau, A. Hertz, G. Laporte, A tabu search heuristic for the vehicle routing problem, Manage. Sci. 40 (10) (1994) 1276–1290.
[14] M. Gendreau, M. Iori, G. Laporte, S. Martello, A tabu search algorithm for a routing and container loading problem, Transport. Sci. 40 (3) (2006) 342–350.
[15] Y.J. Gong, Jun Zhang, O. Liu, R.-Z. Huang, H.S.-H. Chung, Y.-H. Shi, Optimizing the vehicle routing problem with time windows: a discrete particle swarm optimization approach, IEEE Trans. Syst. Man Cybern. Part C 42 (2) (2012) 254–267.
[16] M. Hifi, I. Kacem, S. Nègre, L. Wu, A linear programming approach for the three-dimensional bin-loading problem, Electr. Notes Discr. Math. 36 (2010) 993–1000.
[17] M. Hifi, L. Wu, A hybrid metaheuristic for the vehicle routing problem with time windows, in: 2014 International Conference on Control, Decision and Information Technologies (CoDIT), Metz, 2014, pp. 188–194.
[18] J.H. Holland, Adaptation in Natural and Artificial systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence, MIT Press, Cambridge, MA, 1992.
[19] B. Kallehauge, J. Larsen, O.B.G. Madsen, Lagrangian duality applied on vehicle routing with time windows, Comput. Oper. Res. 33 (5) (2006) 1464–1487.
[20] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report TR06, Erciyes University, 2005.
[21] S. Kirkpatrick, Optimization by simulated annealing: quantitative studies, J. Stat. Phys. 34 (1984) 975–986.
[22] N. Kohl, J. Desrosiers, O.B.G. Madsen, M. Solomon, F. Soumis, Two-path cuts for the vehicle routing problem with time windows, Transport. Sci. 33 (1) (1999) 101–116.
[23] H.C. Lau, Y.F. Lim, Q.Z. Liu, Diversification of search neighborhood via constraint-based local search and its applications to VRPTW, 3rd International Workshop on Integration Of AL and OR Techniques, 2001.
[24] H.C. Lau, M. Sim, K.M Teo, Vehicle routing problem with time windows and a limited number of vehicles, Eur. J. Oper. Res. 148 (3) (2003) 559–569.
[25] S.C.H. Leung, Z. Zhang, D. Zhang, X. Hua, M.K. Lim, A meta-heuristic algorithm for heterogeneous fleet vehicle routing problems with two-dimensional loading constraints, Eur. J. Oper. Res. 225 (2) (2013) 199–210.
[26] S.C.H. Leung, X. Zhou, D. Zhang, J. Zheng, Extended guided tabu search and a new packing algorithm for the two-dimensional loading vehicle routing problem, Comput. Oper. Res. 38 (1) (2011) 205–215.
[27] H. Li, A. Lim, Local search with annealing-like restarts to solve the VRPTW, Eur. J. Oper. Res. 150 (1) (2003) 115–127.
[28] J.P. Luo, X. Li, M.R. Chen, H.W. Liu, A novel hybrid shuffled frog leaping algorithm for vehicle routing problem with time windows, Inf. Sci. 316 (2015) 266–292.
[29] J.Y. Potvin, S. Bengio, The vehicle routing problem with time windows part II: genetic search, INFORMS J. Comput. 8 (2) (1996) 165–172.
[30] Y. Rochat, É.D. Taillard, Probabilistic diversification and intensification in local search for vehicle routing, J. Heurist. 1 (1) (1995) 147–167.
[31] L.M. Rousseau, M. Gendreau, G. Pesant, Using constraint-based operators to solve the vehicle routing problem with time windows, J. Heurist. 8 (1) (2002) 43–58.
[32] M.M. Solomon, Algorithms for the vehicle routing and scheduling problems with time window constraints, Oper. Res. 35 (2) (1987) 254–265.
[33] W.Y. Szeto, Yongzhong Wu, SinC. Ho, An artificial bee colony algorithm for the capacitated vehicle routing problem, Eur. J. Oper. Res. 215 (1) (2011) 126–135.
[34] É. Taillard, P. Badeau, M. Gendreau, F. Guertin, J.Y. Potvin, A tabu search heuristic for the vehicle routing problem with soft time windows, Transport. Sci. 31 (2) (1997) 170–186.
[35] K.C. Tan, Y.H. Chew, L.H. Lee, A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows, Comput. Optim. Appl. 34 (1) (2006) 115–151.
[36] K.C. Tan, L.H. Lee, K. Ou, Artificial intelligence heuristics in solving vehicle routing problems with time window constraints, Eng. Appl. Artif. Intell. 14 (6) (2001) 825–837.
[37] J. Tavares, P. Machado, F.B. Pereira, E Costa, On the influence of GVR in vehicle routing, in: Proceedings of the 2003 ACM Symposium on Applied Computing, New York, 2003, pp. 753–758.
[38] S.R. Thangiah, I.H. Osman and T. Sun, Hybrid genetic algorithm, simulated annealing and tabu search methods for vehicle routing problems with time windows, Computer Science Department, Slippery Rock University, Technical Report SRU CpSc-TR-94-27, 1994.
[39] L. Wei, Z. Zhang, L. Andrew, An adaptive variable neighborhood search for a heterogeneous fleet vehicle routing problem with three-dimensional loading constraints, Comput. Intell. Mag. 9 (4) (2014) 18–30.
[40] B.Z. Yao, P. Hu, M.H. Zhang, S. Wang, Artificial bee colony algorithm with scanning strategy for the periodic vehicle routing problem, Simulation 89 (6) (2013) 762–770.
[41] E.T. Yassen, M. Ayob, M.Z.A. Nazri, N.R. Sabar, Meta-harmony search algorithm for the vehicle routing problem with time windows, Inf. Sci. 325 (2015) 140–158.
[42] B. Yu, Z.Z. Yang, B.Z. Yao, A hybrid algorithm for vehicle routing problem with time windows, Expert Syst. Appl. 38 (1) (2011) 435–441.
[43] E.E. Zachariadis, C.D. Tarantilis, C.T. Kiranoudis, Designing vehicle routes for a mix of different request types, under time windows and loading constraints, Eur. J. Oper. Res. 229 (2) (2013) 303–317.
[44] D. Zhang, L. Wei, Q. Chen, H. Chen, A combinational heuristic algorithm for the three-dimensional packing problem, J. Softw. 18 (9) (2007) 2083–2089.
[45] Z. Zhang, L. Wei, A. Lim, An evolutionary local search for the capacitated vehicle routing problem minimizing fuel consumption under three-dimensional loading constraints, Transport. Res. Part B 82 (2015) 20–35.