Flex Serv Manuf J (2010) 22:61–82
DOI 10.1007/s10696-010-9061-4

Simulated annealing for the vehicle routing problem with two-dimensional loading constraints

Stephen C. H. Leung · Jiemin Zheng · Defu Zhang · Xiyue Zhou

Published online: 25 June 2010

Abstract This paper addresses the capacitated vehicle routing problem with two-dimensional loading constraints (2L-CVRP). The 2L-CVRP is a combination of the two most important problems in distribution logistics, which are loading of freight into vehicles, and the successive routing of the vehicles to satisfy customer demand. The objective is to minimize the transportation cost. All vehicles must start and terminate at a central depot, and the transported items carried by the vehicles must be feasibly packed into the loading surfaces of the vehicles. A simulated annealing algorithm to solve the problem is presented, in which the loading component of the problem is solved through a collection of packing heuristics. A novel approach to plan packing is employed. An efficient data structure (Trie) is used to accelerate the algorithm. The extensive computational results prove the effectiveness of the algorithm.

Keywords Vehicle routing · Loading constraints · Simulated annealing

1 Introduction

This paper addresses the capacitated vehicle routing problem with two-dimensional loading constraints, which is denoted as 2L-CVRP (Iori 2004; Iori 2005). The 2L-CVRP is a variant of one of the most frequently studied combinatorial optimization problems, the capacitated vehicle routing problem (CVRP) (Prins 2004; Tarantilis and Kiranoudis 2002; Tarantilis 2005; Mester and Bräysy 2007).

S. C. H. Leung
Department of Management Sciences, City University of Hong Kong, Kowloon, Hong Kong

J. Zheng · D. Zhang (✉) · X. Zhou
Department of Computer Science, Xiamen University, Xiamen 361005, China
e-mail: dfzhang@xmu.edu.cn

The CVRP is the well-known variant of the vehicle routing problem (VRP) (Dantzig and Ramser 1959; Toth and Vigo 2002a).

The VRP is to find the minimum cost routes to be traveled by a fleet of vehicles. All of the vehicles must start and terminate at a central depot. Each customer must be visited once, by one vehicle only. The VRP is a well known integer programming problem which falls into the category of NP-hard problems. A large number of heuristics have been proposed to solve the problem. These algorithms are separated into two main categories—classical heuristics and metaheuristics. Tabu search strategy is the most widely used and efficient algorithm. Several variants of the standard Tabu search have been presented. In the last decade, many nature inspired metaheuristic algorithms have been proposed for the VRP, for instance, genetic algorithms and ant colony optimization. For more detailed descriptions of these algorithms, survey papers by Bodin et al. (1983); Fisher (1995); Gendreau et al. (1997); Laporte et al. (2000); Laporte and Semet (2002); Tarantilis (2005); Li et al. (2009); Laporte (2009) may be referred.

The simulated annealing algorithm is widely used for the VRP and its variants. Osman (1993) developed a hybrid simulated annealing and Tabu search algorithm for VRP. Chiang and Russell (1996) proposed simulated annealing metaheuristics for the vehicle routing problem with time windows. Breedam (1995) reported simulated annealing-based improvement methods to solve VRP. Moghaddam et al. (2006) presented a hybrid simulated annealing algorithm for capacitated vehicle routing problem with the independent route length. Lin et al. (2009) considered the application of a simulated annealing (SA) heuristic to the truck and trailer routing problem (TTRP) which is a variant of the VRP.

The CVRP is the well-known variant of the VRP where all vehicles are identical and have the same maximum loading capacity. A large number of heuristic and metaheuristic solution methods have been proposed for the problem. For more details, we refer to some extensive survey papers (Laporte and Semet 2002; Gendreau et al. 2002; Cordeau et al. 2002, 2005). For exact solution methods, the reader is referred to Toth and Vigo (2002b), Fukasawa et al. (2006) and Baldacci et al. (2008).

The 2L-CVRP combines the CVRP with a loading problem that is closely related to the well-known two-dimensional bin packing problem (2BPP). The 2BPP belongs to the category of combinatorial optimization problems, and is also a NP Hard problem. It finds the minimum number of identical rectangular bins to load a given set of rectangular items. For more algorithms or reviews on the bin packing problems, interested readers are referred to (Martello and Vigo 1998; Boschetti and Mingozzi 2003a, b; Pisinger and Sigurd 2007; Wei et al. 2009).

The 2L-CVRP was first solved by an exact algorithm which used the branch-and-cut technique (Iori et al. 2007). In the test dataset, their approach can deal with the instances with no more than 30 customers and 91 items in 1 day of CPU time. As for the larger scale problems, a metaheuristic approach was proposed by Gendreau et al. (2008). Precisely, the Tabu search was employed for routing aspects of the problem. Usually, the means of lower bounds, heuristics, local search and a truncated branch-and-bound were used to check the loading feasibility. 180 problem instances were tested in their work. The number of customers went up to 255 and

the items up to 786. Recently, a new method, the guided Tabu search (Zachariadis et al. 2009), which combines the Tabu search with guided local search, was presented. For checking the feasibility of loading, a collection of packing heuristics was used. To accelerate the algorithm, two strategies that reduced the neighborhoods explored, and record of the loading feasibility information, were employed. A nature inspired metaheuristic algorithm, an effective heuristic based on ant colony optimization, has been proposed by Fuellerer et al. (2009). The costs of four different loading configurations were compared in this work.

The importance of the 2L-CVRP is mainly reflected in two aspects. Theoretically, composed of two NP-hard optimization problems (CVRP and 2BPP), it is also a high complexity NP-hard problem. For practical applications, this problem may exist at many companies. An efficient method to solve this problem can significantly reduce costs for the companies.

The purpose of the paper is to solve the 2L-CVRP efficiently. A simulated annealing metaheuristics is employed to solve the problem. As for the loading feasibility aspect, a new heuristic packing algorithm is used. When constructing the initial solution, a new, quicker method is introduced. In order to accelerate the algorithm, a Trie structure (Ferenc 2004; Zachariadis et al. 2009) is proposed to store the information of loading feasibility. The proposed methodology is tested on a 180 2L-CVRP benchmark data set introduced by Gendreau et al. (2008), and the results show that the proposed methodology is more efficient.

The rest of this paper is organized as follows. After the introduction, a clear detailed description of the problem is provided. In Sect. 3, the proposed algorithm is presented. Computational results are described in Sect. 4. Finally, conclusions are summarized.

2 Description of the problem

The 2L-CVRP is defined as follows. A completely undirected graph G is given. Let $G = \{V_0, E\}$, in which $V_0 = V \cup \{0\}$ is a set of $n + 1$ vertices corresponding to the depot (vertices 0) and the customers (vertices $1, 2, \ldots, n$) and E is the set of edges (i,j), $(i,j \in V_0)$. For each edge $(i,j) \in E$, the associated traveling cost C_{ij} is defined, which corresponds to the cost of transition from i to j. In the central depot, a set of K identical vehicles is available. Each vehicle has a weight capacity D and a rectangular loading surface of length L and width W. The loading surface is denoted as $A = WL$. Each customer i $(i = 1, 2, \ldots, n)$ demands a set of m_i rectangular items, denoted as IT_i. Each item $I_{ir} \in IT_i$ $(r = 1, 2, \ldots, m_i)$ has a specific length l_{ir} and width w_{ir}. In addition, we denote $a_i = \sum_{r=1}^{m_i} l_{ir} w_{ir}$ as the total area of the items of customer i. The total weight of the items in the set IT_i is equal to d_i. The items have a fixed orientation. They must be loaded with their l-edges and w-edges parallel to the corresponding edges of the vehicle. Each set IT_i must be loaded into a single vehicle. All customers must be served by using no more than K vehicles. In other words, each customer must be visited by one, and only one vehicle, once. Every vehicle starts from, and ends at, the central depot. The weight of the items loaded in a vehicle must not exceed the capacity of the vehicle D. All the items in a vehicle

must be loaded in area A. Overlapping loading is not permitted. The 2L-CVRP is to find the minimum traveling cost of K vehicles.

The problem described in the previous paragraph is called the *Unrestricted* 2L-CVRP. There is another variant of the problem, called the *Sequential* 2L-CVRP. In the *Sequential* 2L-CVRP, an additional constraint is imposed. The additional constraint is as follows. When a vehicle is visiting customer i, all of the items in the set of IT_i can be unloaded from the vehicle by means of forklift trucks parallel to the length dimension of the vehicle surface, without moving other items required by other customers. In this paper, we study both variants of the problem, the *Unrestricted* 2L-CVRP, and the *Sequential* 2L-CVRP.

3 The proposed algorithm

The proposed algorithm for the solution of the 2L-CVRP employs simulated annealing metaheuristics to explore the solution space, and to optimize the objective. Regarding loading constraints of the problem, the bundle of packing heuristics designed in Zachariadis et al. (2009) is used. We also add a new packing algorithm to the bundle. The efficiency of the added heuristic algorithm will be shown by experimental results. In Sect. 3.1, the packing heuristics bundle is introduced. In Sect. 3.2, the methodology for constructing the initial solution is proposed. In Sect. 3.3, the definition of three neighborhood structures is described. In Sect. 3.4, simulated annealing metaheuristics for the 2L-CVRP is developed. In Sect. 3.5, we discuss how to accelerate the algorithm.

3.1 The bundle of packing heuristics

Before altering the route-sequence of the customers, the loading feasibility must be checked. Six packing heuristics are employed. Five of them were used in Zachariadis et al. (2009). A brief review of them is given here; for detailed introduction, the reader is referred to Zachariadis et al. (2009). Then we focus on the newly added heuristic algorithm.

Given a route-sequence of customers $C = (c_1, c_2, \ldots, c_t), c_1, c_2, \ldots, c_t \in V$, customer c_1 will be visited first, then c_2, and c_t the last. All corresponding sets of items $IT_{c_i}, (i = 1, 2, \ldots, t)$ must be loaded on the surface area of the vehicle A. In order to increase the probability of obtaining a feasible loading, two orderings (Ord_{Seq}, Ord_{Un}) of all items in sets $IT_{c_i}, (i = 1, 2, \ldots, t)$ will be generated and tested. The Ord_{Seq} ordering is produced by sorting the sets IT_{c_i} in the order of $i = t, t-1, \ldots, 2, 1$, and in each set IT_{c_i}, sorting the items by decreasing surface area. The Ord_{Un} ordering is generated by simply sorting all items in sets $IT_{c_i}, (i = 1, 2, \ldots, t)$ by decreasing surface area.

For the first five heuristics, the items are selected for loading on the surface one by one, from the ordering. Available loading positions for the items are listed, in the list denoted as *posList*. In the beginning, *posList* = {(0, 0)}, only the bottom-left corner is the available loading position. After an item is loaded, its loading position is deleted from the *posList*, and a maximum of four new loading positions are

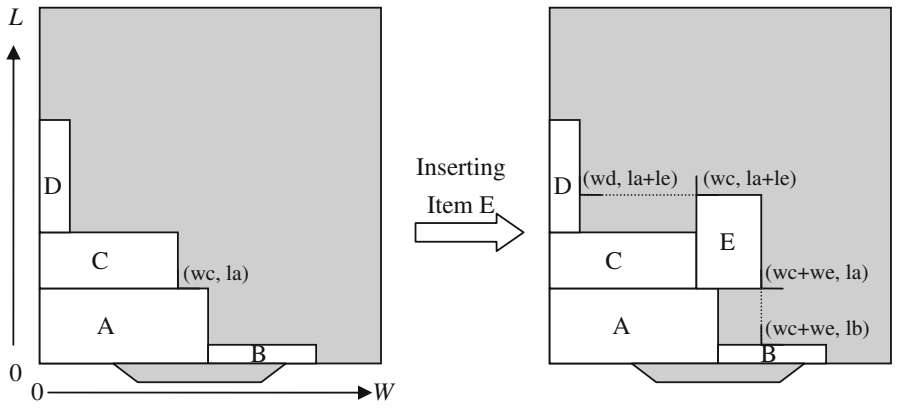


Fig. 1 Inserting an item into the loading surface

generated and inserted into the *posList*. For instance, in Fig. 1, when item E is loaded, loading position (wc, la) is erased from the *posList*, while new loading positions (wc, la + le) (wc + we, la) (wd, la + le) and (wc + we, lb) are added to the *posList*.

The position selected for an item must be feasible, which means when the item is loaded at the position, it should not lead to any overlaps (for both *Unrestricted* and *Sequential* problem versions), or sequence constraint violations (for the *Sequential* problem version). Each of the five heuristics has a different criterion to select a position, from available feasible positions, to load an item.

Heur1: *Bottom-Left Fill* (*W*-axis) (Chazelle 1983) selects the one with the minimum *W*-axis coordinate, breaking ties by minimum *L*-axis coordinate.

Heur2: *Bottom-Left Fill* (*L*-axis) (Chazelle 1983) gives priority to the one with the minimum *L*-axis coordinate, breaking ties by minimum *W*-axis coordinate.

Heur3: *Max Touching Perimeter* heuristic (Lodi et al. 1999) selects the position where, if the item is loaded, it has the max total touching perimeter with the edges of the already loaded items, and the loading surface of the vehicle.

Heur4: *Max Touching Perimeter No Walls* heuristic (Lodi et al. 1999) evaluates the total perimeter as the sum of the common edges of the item to be loaded, with edges of already loaded items, but not for edges common with the loading surface of the vehicle.

Heur5: *Min Area* heuristic (Zachariadis et al. 2009) selects the one whose corresponding rectangular surface is the minimum.

Next, a novel heuristic Heur6 is introduced. In this heuristic, we check available positions from the lowest position to the highest position to place the remaining items. As shown in Fig. 2, the shaded area depicts areas where either items are already loaded, or area considered as wasted area. Let the horizontal line marked by

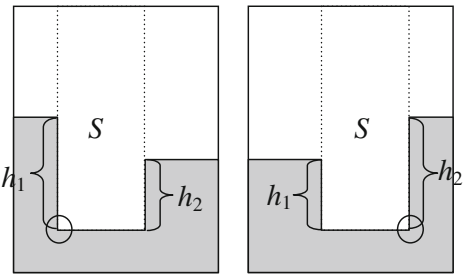


Fig. 2 The corner to place item

the circle be the lowest available position. We try to place an item with the maximum fitness value in the area of S. The calculation of the fitness value is given in the following paragraphs. Let h_1 denote the height of the wall to the left of S, and h_2 denote the height of the wall to the right of S. In case $h_1 \geq h_2$, the item will be placed at the left-bottom corner of S. For another case where $h_1 < h_2$, the item will be placed at the right-bottom corner of S. If all of the remaining items cannot be placed on the lowest position, the second lowest will be considered (see Fig. 3.). If there exists a remaining item whose height is larger than the height of S, the packing is considered unfeasible, and the heuristic algorithm is terminated.

The question, then, is which kind of placement is better. It is obvious that a placement is good if it can decrease the number of corner positions. So the concept of fitness value is presented to evaluate whether a placement is good or not. If a placement can fit more corner positions, the corresponding rectangle for this placement is given a larger fitness value. We design the fitness value as follows.

(1) In Fig. 4, item R is placed at the position marked by the circle. If the width of the item is equal to w, and its length is h_1, for case one, or h_2 for case two, the fitness value of R is 4.

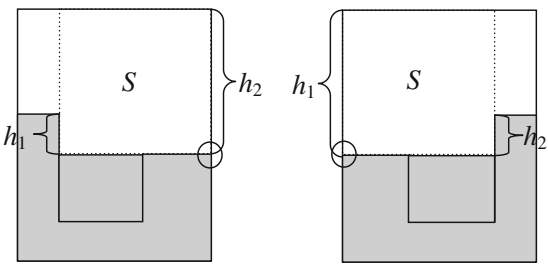


Fig. 3 The second lowest position to

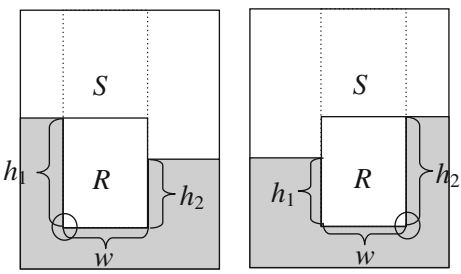


Fig. 4 Fitness value = 4

(2) In Fig. 5, if the width of item R is w, and its length is equal to h_2 for case one, or h_1 for case two, the fitness value of R is 3. Although Fig. 4 and Fig. 5 fit the same number of corners, we give the priority to Fig. 4. Because the items are always placed on the higher wall side, if a placement fit the higher wall, it is considered as a better placement.

(3) In Fig. 6, if the width of item R is equal to w, and its length is equal to neither h_1 nor h_2, the fitness value of R is 2.

(4) In Fig. 7, if the width of item R is not equal to w, and its length is equal to h_1 for case one, or h_2 for case two, the fitness value of R is 1.

(5) In Fig. 8, if the width of item R is not equal to w, and the length of the item is not equal to h_1 for case one, or h_2 for case two, the fitness value of R is 0.

If there are more than two items having the same maximum fitness value, we give priority to the one having the maximum length.

As mentioned before, there are two different orderings, Ord_{Un} and Ord_{Seq}. When ordering Ord_{Un} is put into Heur6, Heur6 selects the best fitting item from all of the item sets $IT_{c_i}, (i = 1, 2, \ldots, t)$. While for ordering Ord_{Seq}, the items are loaded by

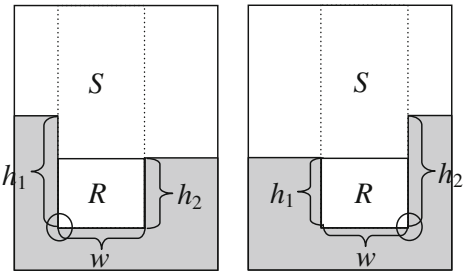


Fig. 5 Fitness value = 3

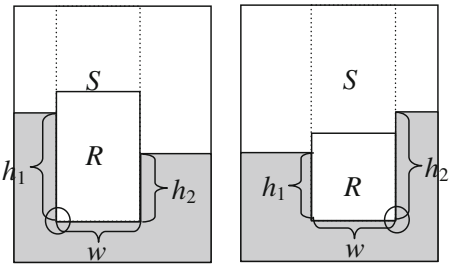


Fig. 6 Fitness value = 2

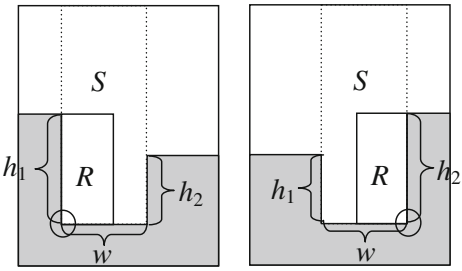
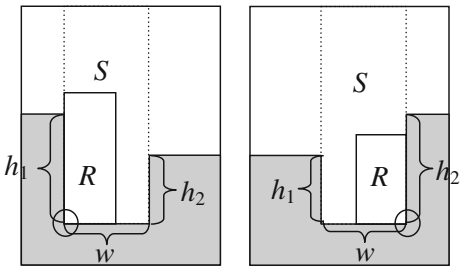


Fig. 7 Fitness value = 1

Fig. 8 Fitness value = 0



the order of the item sets. Heur6 selects the best fitting item from item set IT_{c_t} first, and load it. When all items in set IT_{c_t} are loaded, it turns to item set $IT_{c_{t-1}}$, and so on, until all items are loaded.

All of the first five heuristics load the items one by one, according to the orderings. The sixth heuristic packing algorithm selects the best fitting item from the remaining items, so it may increase the probability of obtaining a feasible loading.

The given route sequence is checked by the six heuristics in the order presented (Heur1 comes first, while Heur6 will be the last). The two orderings, Ord_{Un} and Ord_{Seq}, are put into the packing heuristics for each version of the problem, i.e., the *Unrestricted* 2L-CVRP and the *Sequential* 2L-CVRP. If all packing heuristics fail to produce a feasible loading with these two orderings, the route examined is considered to be violative of loading constraints of the problem.

3.2 Constructing the initial solution

At first, all of the n customers are sorted in decreasing value of $a_i (i = 1, 2, \ldots, n)$, and the sequence is recorded in $OrdCustomer_i (i = 1, 2, \ldots, n)$. The unoccupied loading surface of each vehicle is calculated and denoted as $freeArea_i (i = 1, 2, \ldots, K)$. Customers are selected successively, to be inserted into the routes one by one, according to the ordering *OrdCustomer*. The feasibility of inserting a customer c into every position of every route is checked. If a customer c can be inserted into a set of feasible routes, its items are assigned to be loaded by the vehicle with the minimum *freeArea*, and inserted into the corresponding route at the position that leads to the minimum cost increase. If it comes up against a customer $OrdCustomer_t$ who cannot be inserted into any of the K routes, it is exchanged with $OrdCustomer_j$ (j is selected randomly from 1 to $t-1$). Then the construction is tried again with the new ordering *OrdCustomer*.

Feasible initial solutions for all *Unrestricted* and *Sequential* 2L-CVRP problem instances can be constructed by this method very quickly.

3.3 Three neighborhood structures

To move from the current to the subsequent solution, we use three types of moves. Each of them defines a neighbourhood structure $NS_i (i = 1,2,3)$. These three structures are also used by Zachariadis et al. (2009). A neighbourhood structure is selected randomly out of the three in the same probability, and the subsequent

solution is a stochastically feasible solution among the neighbourhood solutions of this type.

NS_1 customer relocation (Waters 1987): in this type, a customer is relocated to another position of the current solution.

NS_2 route exchange (Waters 1987): in this type, a pair of customers exchange their positions in the current solution.

NS_3 route interchanging (Croes 1958; Lin 1965): in this type, a pair of customers is selected. If they are in the same route, positions of customers between them including themselves are reversed. In the other case, when they are in different routes, the part of the first route, before the selected customer, is connected to the part of the second route, after the other selected customer, including itself, and vice versa.

3.4 The simulated annealing metaheuristics for the 2L-CVRP

Simulated annealing (SA) is an algorithmic approach for solving combinatorial optimization problems (Kirkpatrick et al. 1983; Cerny 1985). Simulated annealing randomizes the local search procedure, and, in some instances, allows for solution changes that worsen the solution. This constitutes an attempt to reduce the probability of becoming trapped in a locally suboptimal solution. The probability of accepting a worsened solution is determined by a control parameter (T), called temperature. At the beginning of the algorithm, T is assigned an initial value T_0. The deterministic cooling schedule is designed according to the following formula:

$$T_k = 0.9T_{k-1}$$

The simulated annealing algorithm attempts to transform the current solution S_{cur} into one of its neighbors S_{new}. This process can be modeled as a Markov chain. For each temperature value, we get one Markov chain. The length of each Markov chain, denoted as *Len*, is a parameter of the algorithm that needs tuning. In *Len* iterations, if the cost of the new solution is less than the current solution, the new solution is accepted unconditionally. If the cost of the new solution is higher than the current solution, the probability to accept the new solution is P.

$$P = \exp\left(\frac{\cos t(S_{cur}) - \cos t(S_{new})}{T}\right)$$

If a solution remains unchanged for 2*Len iterations, the algorithm is terminated.

3.5 Accelerating the algorithm

In order to accelerate the algorithm, a Trie is employed to record the loading feasibility of routes already examined. This strategy helps avoid unnecessary calls to the packing heuristic bundle. To ascertain if a given route violates the loading constraints of the problem, first, it is checked whether it is in the Trie tree. If it is in the tree, the feasibility of the route is given by the tree. If not, the heuristic bundle is executed and the result is inserted into the Trie tree. For a large scale instance, the

physical memory required for storing the loading information may exceed the available memory. So before inserting a route into the Trie tree, the percent of memory in use is queried. If it exceeds a threshold, the tree is destructed, and rebuilt.

4 Computational results

We have implemented the algorithm in C++ and run it on a 2.4 GHz Core Duo notebook with 2,048 MB RAM under Windows XP. In order to verify the performance of the proposed algorithm, the proposed algorithm was tested by 360 2LCVRP (180 for the *Unrestricted* and 180 for the *Sequential* version of the problem) benchmark problem instances from the literature (Iori 2004; Iori et al. 2007; Gendreau et al. 2008; Zachariadis et al. 2009). The datasets are available at http://www.or.deis.unibo.it/research.html.

These instances were derived from 36 classical CVRP instances, whose description can be found in Toth and Vigo (2002a). The customer demands, which are a set of two-dimensional, weighted and rectangular items, are expressed into the instances. Each of the 36 instances has five classes of item sets. In Class 1, each customer is assigned one item with unit width and length. In Classes 2–5, the number of items, demanded by customer i, m_i is generated by a uniform distribution on a certain interval (see Table 1, column 2). Each item is classified into one of the three shape categories, with equal probability. The three categories are *vertical* (the relative lengths are greater than the relative widths), *homogeneous* (the relative lengths and widths are generated in the same intervals), and *horizontal* (the relative lengths are smaller than the relative widths). The dimensions (width and length) of an item are uniformly distributed into the ranges determined by this item's shape category (see Table 1, columns 3–8). The values $L = 40$ and $W = 20$ were chosen for the dimensions of the loading area. The numbers of customers and items, in the instances for Classes 2–5, are shown in Table 2. For details of the datasets, the reader is referred to Gendreau et al. (2008) and Zachariadis et al. (2009).

There are only two parameters, the initial value T_0, and the length of each Markov chain *Len*, in the proposed algorithmic framework. The values of them were determined through an extensive calibration procedure. In order to cover cases with diverse customer set sizes and item characteristics, 30 problems (instances 6, 12, 24, 30, 36, Classes 1–5) were tested by the proposed algorithm with different

Table 1 The characteristics of items of classes 2–5 instances

Class	m_i	Vertical		Homogeneous		Horizontal	
		Length	Width	Length	Width	Length	Width
2	[1,2]	[0.4L, 0.9L]	[0.1W, 0.2W]	[0.2L, 0.5L]	[0.2W, 0.5W]	[0.1L, 0.2L]	[0.4 W, 0.9 W]
3	[1,3]	[0.3L, 0.8L]	[0.1W, 0.2W]	[0.2L, 0.4L]	[0.2W, 0.4W]	[0.1L, 0.2L]	[0.3 W, 0.8 W]
4	[1,4]	[0.2L, 0.7L]	[0.1W, 0.2W]	[0.1L, 0.4L]	[0.1W, 0.4W]	[0.1L, 0.2L]	[0.2W, 0.7 W]
5	[1,5]	[0.1L, 0.6L]	[0.1W, 0.2W]	[0.1L, 0.3L]	[0.1W, 0.3W]	[0.1L, 0.2L]	[0.1W, 0.6W]

Table 2 The characteristics of Classes 2–5 instances

Inst	n	Number of items of classes 2–5				Inst	n	Number of items of classes 2–5			
		2	3	4	5			2	3	4	5
1	15	24	31	37	45	19	50	82	103	134	157
2	15	25	31	40	48	20	71	104	151	178	226
3	20	29	46	44	49	21	75	114	164	168	202
4	20	32	43	50	62	22	75	112	154	198	236
5	21	31	37	41	57	23	75	112	155	179	225
6	21	33	40	57	56	24	75	124	152	195	215
7	22	32	41	51	55	25	100	157	212	254	311
8	22	29	42	48	52	26	100	147	198	247	310
9	25	40	61	63	91	27	100	152	211	245	320
10	29	43	49	72	86	28	120	183	242	299	384
11	29	43	62	74	91	29	134	197	262	342	422
12	30	50	56	82	101	30	150	225	298	366	433
13	32	44	56	78	102	31	199	307	402	513	602
14	32	47	57	65	87	32	199	299	404	497	589
15	32	48	59	84	114	33	199	301	407	499	577
16	35	56	74	93	114	34	240	370	490	604	720
17	40	60	73	96	127	35	252	367	507	634	762
18	44	66	87	112	122	36	255	387	511	606	786

n number of customers

parameters values. The experiment results are summarized in Table 3. In the table, the results are the average of 30 problems. There is no obvious correlation between the two parameters. So we valued each of them separately. After extensively calibration tests, known from Table 3, the appropriate value for T_0 is set as 10, and 20,000 for *Len*.

We compare our proposed algorithm with Tabu search (TS) algorithm (Gendreau et al. 2008), guided Tabu search (GTS) algorithm (Zachariadis et al. 2009), and ant colony optimization (ACO) algorithm (Fuellerer et al. 2009). The TS was coded in C and tested on a Pentium IV 1.7 GHz. The GTS was coded in Visual C#, executed on a Pentium IV 2.4 GHz with 1 GB of RAM under Windows XP. The ACO was coded using ANSI C++ and performed on a Pentium IV 3.2 GHz.

The results are shown in Tables 4, 5, 6, 7. In these tables, the SA is the proposed algorithm, *s* is the average value of the solution obtained, *%gap* is the percent improvement over the corresponding algorithm solution value, and *tot* is the total time required for processing the algorithm (in seconds).

As can be seen in Tables 4, 5, 6, 7. Table 4 reports the results obtained on the pure CVRP instances. The solution values achieved by the SA and GTS are almost the same, on average. They are better than that achieved by the TS. But for the more difficult problems in Classes 2–5, the proposed algorithm beats the TS and GTS. As shown in Tables 5 and 6, for the *Unrestricted* 2L-CVRP, the SA achieved on average 3.19% improvement over the TS, 1.38% over the GTS, and in all of the

Table 3 Calibration experiment results for the parameters

T_0	*s* 5	*tot*	*s* 10	*tot*	*s* 15	*tot*	*s* 20	*tot*
Unrestr seq	1078.73	355.3	**1072.28**	397.5	1080.42	494.9	1078.34	473.5
	1111.82	530.3	**1106.00**	546.5	1120.53	727.2	1133.88	692.7
Len	10,000		15,000		20,000		25,000	
Unrestr seq	1082.31	217.0	1080.13	339.5	**1072.28**	397.5	1077.29	496.9
	1124.55	342.0	1120.17	446.4	**1106.00**	546.5	1118.19	737.0

s is the average value of the solution obtained

tot is the total time required for processing the algorithm (in seconds)

instances, except Instance 6, SA achieved a better solution than the GTS. While for the *Sequential* 2L-CVRP, the improvement was 3.57% over TS, and 2% over GTS on average, and also only in one instance (Instance 8), the solution of the proposed algorithm was defeated by the GTS. As demonstrated in Tables 4, 5, 6, 7 all of the instances can be solved by SA within 1.5 h. So the executing time is acceptable.

The GTS produced solutions with higher average quality compared to those obtained by TS. So we focused on the comparison between the proposed algorithm SA and GTS. Figure 9 shows the percent improvement of SA over GTS. The x-axes displays instances 1–36, and the y-axes denotes the improvement. As known from the figure, for the larger scale instance, the performance of the SA is the better. For the more complex, the more constraints problem, from the CVRP, to the *Unrestricted* 2L-CVRP, and then the *Sequential* 2L-CVRP, the SA is more efficient.

The performance of the proposed algorithm SA with respect to ACO is shown in Table 7, where the runtime limit of ACO is 3,600 s. ACO performs better than SA for the *Unrestricted* 2L-CVRP on average, while SA performs better than ACO for the *Sequential* 2L-CVRP on average. In particular, for the Unrestricted 2L-CVRP and the Sequential 2L-CVRP, the average running time of SA (Core Duo notebook, 2.4 GHz) on Classes 1–5 is 298.5 and 463.1 s, respectively, while that of ACO (Pentium IV, 3.2 GHz) is 902.0 and 1067.7, so SA computes faster than ACO. In addition, for the sequential instances with the number of customers more than 100, SA consistently outperforms ACO.

In order to demonstrate the efficiency of the proposed packing algorithm Heur6, we design tests for the problems, using the heuristics bundle that includes Heur6, for comparing it with the bundle that does not include Heur6. The results are shown in Table 8, where *s*, *tot* and *%gap* are as before. The results are improved by 1.17% for the *Unrestricted* 2L-CVRP and 1.35% for the *Sequential* 2L-CVRP on average. It can be noted that adding packing algorithm Heur6 in SA is not necessary to increase the computational time. It can be explained as follows: In the SA metaheuristics, there are a number of iterations. In each iteration, packing heuristics Heur1, Heur2, …, and Heur6 (as Heur1 comes first, while Heur6 will be the last) will be involved. Moreover, Heur6 is able to increase the number of feasible solutions in the neighbourhoods of current solution in each iteration of SA. Hence, in each iteration, with the feasible solutions obtained by Heur6 in previous iteration, a better solution

Table 4 Computational results on the pure CVRP instances of Class 1

Inst	Class 1							
	s_{SA}	tot_{SA}	s_{TS}	tot_{TS}	$\%gap_{TS}$	s_{GTS}	tot_{GTS}	$\%gap_{GTS}$
1	**278.73**	12.0	**278.73**	2.0	0.00	**278.73**	5.2	0.00
2	**334.96**	9.3	**334.96**	0.0	0.00	**334.96**	2.4	0.00
3	**358.40**	11.6	359.77	3.5	0.38	**358.40**	6.8	0.00
4	430.89	8.9	**430.88**	0.1	0.00	**430.88**	1.7	0.00
5	**375.28**	10.7	**375.28**	1.4	0.00	**375.28**	2.1	0.00
6	**495.85**	15.6	**495.85**	0.3	0.00	**495.85**	3.4	0.00
7	**568.56**	9.3	**568.56**	0.5	0.00	**568.56**	1.3	0.00
8	**568.56**	7.3	**568.56**	0.5	0.00	**568.56**	0.6	0.00
9	**607.65**	25.2	**607.65**	0.4	0.00	**607.65**	1.8	0.00
10	**535.80**	26.5	538.79	6.1	0.55	**535.80**	9.1	0.00
11	**505.01**	20.3	**505.01**	2.5	0.00	**505.01**	6.3	0.00
12	**610.00**	43.9	610.57	28.5	0.09	**610.00**	9.6	0.00
13	**2006.34**	16.8	**2006.34**	29.9	0.00	**2006.34**	10.0	0.00
14	**837.67**	26.7	**837.67**	22.2	0.00	**837.67**	26.4	0.00
15	**837.67**	24.7	**837.67**	1.7	0.00	**837.67**	12.7	0.00
16	**698.61**	50.1	**698.61**	2.7	0.00	**698.61**	22.3	0.00
17	**861.79**	64.2	862.62	59.0	0.10	863.27	58.6	0.17
18	**723.54**	37.8	**723.54**	81.9	0.00	730.85	77.0	1.00
19	**524.61**	56.7	**524.61**	128.8	0.00	**524.61**	169.1	0.00
20	**241.97**	96.3	**241.97**	253.6	0.00	244.54	339.3	1.05
21	693.56	62.2	688.18	325.0	−0.78	**687.60**	359.1	−0.87
22	744.92	81.4	**740.66**	2070.7	−0.57	**740.66**	1160.9	−0.57
23	844.88	110.1	860.47	2210.1	1.81	**839.07**	2544.1	−0.69
24	**1029.25**	126.0	1048.91	866.9	1.87	1035.33	1680.7	0.59
25	830.83	91.6	830.26	2371.0	−0.07	**829.45**	1715.8	−0.17
26	**819.56**	49.8	**819.56**	3597.6	0.00	**819.56**	1743.1	0.00
27	1098.91	89.8	1099.95	355.9	0.09	**1097.63**	1556.9	−0.12
28	1054.66	206.0	1078.27	985.2	2.19	**1042.12**	1383.4	−1.20
29	1190.19	196.0	**1179.01**	3080.0	−0.95	1188.15	2790.4	−0.17
30	1050.64	174.4	1061.55	1834.4	1.03	**1037.05**	1393.1	−1.31
31	**1361.82**	451.7	1464.04	288.8	6.98	1421.20	1089.8	4.18
32	1339.50	219.2	1352.61	1780.8	0.97	**1328.68**	1528.4	−0.81
33	1334.20	239.8	1361.51	2531.7	2.01	**1328.19**	2628.1	−0.45
34	**719.89**	339.0	858.94	1941.9	16.19	719.91	1495.3	0.00
35	888.00	204.5	992.86	766.7	10.56	**877.04**	1549.0	–
36	603.69	426.4	678.87	1530.9	11.07	**594.10**	2299.3	−1.61
Avg					1.49			−0.06

Table 5 Computational results for the *Unrestricted* 2L-CVRP (Classes 2–5)

Inst	Class 1							
	s_{SA}	tot_{SA}	s_{TS}	tot_{TS}	$\%gap_{TS}$	s_{GTS}	tot_{GTS}	$\%gap_{GTS}$
1	**290.17**	15.3	291.60	4.2	0.49	295.74	3.5	1.88
2	341.35	13.2	**341.02**	0.1	−0.10	341.89	2.0	0.16
3	377.37	21.7	**377.35**	1.6	−0.01	384.49	1.2	1.85
4	**435.01**	18.3	437.45	0.5	0.56	441.45	3.8	1.46
5	**379.49**	22.6	380.20	5.0	0.19	382.22	7.5	0.71
6	501.02	21.0	501.02	7.2	0.00	**499.47**	7.0	−0.31
7	**698.65**	26.1	700.34	6.3	0.24	703.49	6.8	0.69
8	703.12	29.3	**694.99**	11.2	−1.17	705.59	10.2	0.35
9	**612.01**	27.5	619.69	3.6	1.24	615.65	7.7	0.59
10	**694.18**	70.1	700.39	36.0	0.89	713.00	16.7	2.64
11	**732.87**	70.7	739.04	55.7	0.83	740.40	31.4	1.02
12	**613.94**	46.0	620.62	49.0	1.08	616.83	93.7	0.47
13	**2555.37**	29.0	2598.20	57.5	1.65	2599.40	68.5	1.69
14	**1007.08**	100.9	1047.72	375.8	3.88	1036.77	299.1	2.86
15	**1192.77**	108.0	1201.38	156.7	0.72	1197.83	138.3	0.42
16	**701.27**	49.7	702.03	20.5	0.11	702.29	132.4	0.15
17	**864.05**	57.8	866.37	64.9	0.27	864.26	45.4	0.02
18	**1063.48**	125.3	1085.84	589.3	2.06	1076.81	397.7	1.24
19	**763.78**	155.4	772.25	633.7	1.10	776.91	570.7	1.69
20	**541.35**	375.2	564.67	954.5	4.13	551.71	839.7	1.88
21	**1039.65**	247.2	1066.21	460.1	2.49	1050.43	203.8	1.03
22	**1061.26**	256.9	1087.46	1191.2	2.41	1076.11	1584.6	1.38
23	**1073.10**	280.2	1104.72	2032.4	2.86	1091.18	1750.3	1.66
24	**1140.63**	206.4	1187.62	1454.1	3.96	1149.12	842.4	0.74
25	**1394.44**	417.4	1436.09	1205.8	2.90	1418.87	965.4	1.72
26	**1369.02**	417.7	1404.49	1173.9	2.53	1395.63	1403.9	1.91
27	**1375.52**	353.2	1450.18	521.3	5.15	1396.60	875.6	1.51
28	**2671.74**	540.2	2738.31	2051.2	2.43	2737.01	1836.4	2.38
29	**2270.18**	671.1	2474.33	1406.5	8.25	2315.20	1653.4	1.94
30	**1839.20**	718.7	1948.72	1185.4	5.62	1889.84	2852.4	2.68
31	**2372.69**	810.0	2506.99	2375.8	5.36	2413.45	4100.7	1.69
32	**2315.32**	860.4	2486.43	1664.8	6.88	2351.69	3597.8	1.55
33	**2381.78**	891.2	2504.00	1843.2	4.88	2455.79	3431.2	3.01
34	**1220.20**	1229.9	1466.06	1359.1	16.77	1233.46	3843.3	1.08
35	**1479.41**	1581.2	1765.30	2061.7	16.19	1498.78	4183.9	1.29
36	**1755.62**	1658.1	1909.88	2265.8	8.08	1801.41	4004.3	2.54
Avg					3.19			1.38

Table 6 Computational results for the *Sequential* 2L-CVRP (Classes 2–5)

Inst	Class 1							
	s_{SA}	tot_{SA}	s_{TS}	tot_{TS}	$\%gap_{TS}$	s_{GTS}	tot_{GTS}	$\%gap_{GTS}$
1	**297.57**	23.2	299.09	2.6	0.51	304.22	4.4	2.19
2	**345.23**	15.2	**345.23**	0.4	0.00	346.71	2.6	0.43
3	389.69	25.7	**385.30**	3.8	−1.14	393.35	4.0	0.93
4	444.21	20.7	**443.42**	1.4	−0.18	444.62	4.4	0.09
5	387.59	34.6	**384.06**	4.1	−0.92	396.36	11.4	2.21
6	503.66	25.0	**502.78**	5.1	−0.18	505.04	5.0	0.27
7	**719.95**	29.4	721.90	15.5	0.27	723.83	31.9	0.54
8	716.62	31.6	722.73	32.8	0.85	**715.72**	33.0	−0.13
9	**621.23**	33.9	624.06	10.6	0.45	622.20	17.3	0.16
10	722.72	81.9	**714.90**	43.5	−1.09	727.86	49.4	0.71
11	**760.39**	92.0	773.45	99.0	1.69	768.15	94.8	1.01
12	**618.52**	45.1	631.85	58.8	2.11	628.62	131.1	1.61
13	**2641.58**	40.6	2687.03	49.0	1.69	2679.34	87.8	1.41
14	**1061.10**	134.2	1101.49	146.0	3.67	1092.78	173.2	2.90
15	**1214.81**	156.0	1240.89	165.4	2.10	1234.21	74.2	1.57
16	**703.27**	66.5	704.85	28.0	0.22	707.56	58.1	0.61
17	**864.26**	67.0	866.50	88.9	0.26	865.20	91.5	0.11
18	**1091.02**	161.3	1116.17	566.5	2.25	1102.25	390.7	1.02
19	**790.14**	235.7	802.48	365.2	1.54	800.94	541.2	1.35
20	**553.90**	656.8	581.81	808.9	4.80	576.58	848.9	3.93
21	**1072.87**	371.2	1110.19	1702.2	3.36	1106.33	1539.9	3.02
22	**1105.15**	440.6	1130.33	1573.8	2.23	1128.61	2208.3	2.08
23	**1104.31**	483.2	1186.36	675.8	6.92	1151.24	1190.6	4.08
24	**1174.85**	299.5	1248.43	2642.5	5.89	1206.62	2200.3	2.63
25	**1449.54**	650.2	1480.63	2336.5	2.10	1479.03	3807.6	1.99
26	**1406.55**	730.0	1471.74	1554.6	4.43	1475.96	1984.8	4.70
27	**1414.90**	511.8	1524.22	1308.2	7.17	1463.34	1226.1	3.31
28	**2751.72**	831.1	2858.53	2576.9	3.74	2835.30	2932.7	2.95
29	**2357.68**	1181.6	2575.28	1162.5	8.45	2460.74	2026.9	4.19
30	**1935.82**	1118.8	2076.20	2021.4	6.76	2031.94	2127.5	4.73
31	**2464.52**	1429.5	2592.17	2102.2	4.92	2554.37	3163.4	3.52
32	**2410.73**	1561.3	2605.10	2305.2	7.46	2462.45	5009.8	2.10
33	**2486.88**	1559.3	2610.55	2221.2	4.74	2565.41	4474.1	3.06
34	**1288.92**	2151.4	1546.06	2184.4	16.63	1303.49	5371.4	1.12
35	**1601.00**	2390.3	1985.44	2223.1	19.36	1675.35	5004.2	4.44
36	**1842.87**	2242.7	1946.66	2626.3	5.33	1864.73	5340.4	1.17
Avg					3.57			2.00

Table 7 Computational results of SA and ACO (Classes 1–5)

Inst	n	The *Unrestricted* 2L-CVRP			The *Sequential* 2L-CVRP		
		s_{SA}	s_{ACO}	$\%gap_{ACO}$	s_{SA}	s_{ACO}	$\%gap_{ACO}$
1	15	287.88	**284.82**	−1.08	293.80	**291.83**	−0.68
2	15	340.07	**339.81**	−0.08	**343.18**	343.22	0.01
3	20	373.57	**372.73**	−0.23	383.43	**378.17**	−1.39
4	20	**434.19**	437.11	0.67	441.55	**439.07**	−0.56
5	21	378.65	**378.28**	−0.10	385.13	**381.63**	−0.92
6	21	499.98	**497.02**	−0.60	502.10	**499.77**	−0.47
7	22	672.63	**671.36**	−0.19	689.67	**678.22**	−1.69
8	22	676.21	**666.62**	−1.44	687.01	**684.37**	−0.39
9	25	**611.14**	611.15	0.00	618.51	**614.88**	−0.59
10	29	662.51	**655.55**	−1.06	685.34	**668.48**	−2.52
11	29	687.30	**678.51**	−1.30	709.31	**690.22**	−2.77
12	30	613.15	**612.74**	−0.07	616.81	**615.90**	−0.15
13	32	2445.56	**2424.14**	−0.88	2514.53	**2480.04**	−1.39
14	32	973.20	**966.55**	−0.69	1016.41	**1007.92**	−0.84
15	32	1121.75	**1105.76**	−1.45	**1139.38**	1145.96	0.57
16	35	700.74	**699.92**	−0.12	702.34	**701.09**	−0.18
17	40	**863.60**	864.62	0.12	**863.77**	864.92	0.13
18	44	995.49	**984.70**	−1.10	1017.52	**1003.84**	−1.36
19	50	715.95	**707.89**	−1.14	737.03	**728.89**	−1.12
20	71	481.48	**472.45**	−1.91	491.51	**484.23**	−1.50
21	75	970.43	**952.79**	−1.85	997.01	**987.54**	−0.96
22	75	997.99	**988.21**	−0.99	1033.10	**1018.76**	−1.41
23	75	1027.45	**1012.19**	−1.51	**1052.43**	1054.69	0.21
24	75	1118.35	**1109.38**	−0.81	1145.73	**1134.90**	−0.95
25	100	1281.72	**1266.07**	−1.24	1325.79	**1320.94**	−0.37
26	100	1259.13	**1245.32**	−1.11	**1289.15**	1309.67	1.57
27	100	1320.20	**1303.41**	−1.29	1351.70	**1341.25**	−0.78
28	120	**2348.33**	2363.21	0.63	**2412.31**	2468.98	2.30
29	134	**2054.18**	2061.22	0.34	**2124.18**	2187.90	2.91
30	150	1681.49	**1665.76**	−0.94	**1758.78**	1796.54	2.10
31	199	2170.51	**2141.48**	−1.36	**2243.98**	2282.91	1.71
32	199	2120.16	**2101.41**	−0.89	**2196.48**	2260.37	2.83
33	199	2172.26	**2158.10**	−0.66	**2256.34**	2315.51	2.56
34	240	1120.14	**1112.06**	−0.73	**1175.11**	1198.04	1.91
35	252	1361.12	**1354.65**	−0.48	**1458.40**	1516.70	3.84
36	255	**1525.24**	1543.62	1.19	**1595.03**	1632.44	2.29
Avg		1085.10	1078.07	−0.67	1118.16	1125.83	0.05

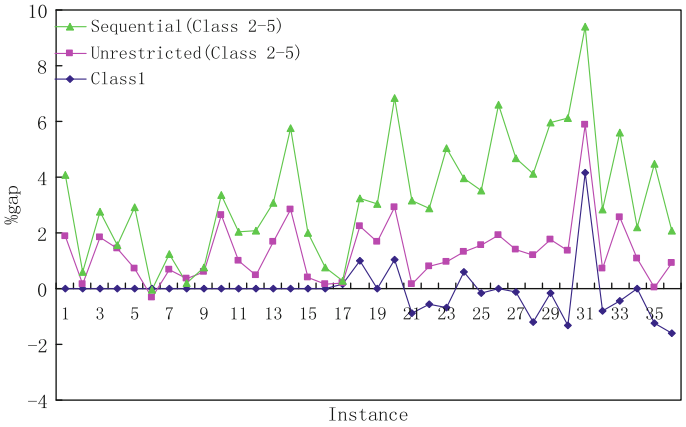


Fig. 9 The percent improvement of SA over GTS

Table 8 Computational results of two packing heuristics (Class 2–5)

Inst	The *Unrestricted* 2L-CVRP					The *Sequential* 2L-CVRP				
	From *Heur1* to *Heur5*		From *Heur1* to *Heur6*		$\%gap_{H6}$	From *Heur1* to *Heur5*		From *Heur1* to *Heur6*		$\%gap_{H6}$
	s	*tot*	*s*	*tot*		*s*	*tot*	*s*	*tot*	
1	294.73	17.4	**290.17**	15.3	1.55	299.79	20.2	**297.57**	23.2	0.74
2	341.89	12.8	**341.35**	13.2	0.16	345.23	14.8	345.23	15.2	0.00
3	382.94	21.8	**377.37**	21.7	1.46	392.31	25.8	**389.69**	25.7	0.67
4	441.45	19.3	**435.01**	18.3	1.46	444.21	28.3	444.21	20.7	0.00
5	381.13	23.5	**379.49**	22.6	0.43	391.97	33.8	**387.59**	34.6	1.12
6	501.02	21.5	501.02	21.0	0.00	503.8	28.1	**503.66**	25.0	0.03
7	699.29	25.2	**698.65**	26.1	0.09	721.62	28.0	**719.95**	29.4	0.23
8	705.4	27.5	**703.12**	29.3	0.32	719.14	27.7	**716.62**	31.6	0.35
9	615.64	28.3	**612.01**	27.5	0.59	621.23	43.7	621.23	33.9	0.00
10	707.96	76.7	**694.18**	70.1	1.95	730.6	74.4	**722.72**	81.9	1.08
11	740.58	80.9	**732.87**	70.7	1.04	764.6	92.5	**760.39**	92.0	0.55
12	616.83	39.8	**613.94**	46.0	0.47	623.87	53.8	**618.52**	45.1	0.86
13	2589.64	29.2	**2555.37**	29.0	1.32	2659.27	39.5	**2641.58**	40.6	0.67
14	1038.3	106.6	**1007.08**	100.9	3.01	1075.89	128.5	**1061.1**	134.2	1.38
15	1197.24	116.5	**1192.77**	108.0	0.37	1223.04	154.1	**1214.81**	156.0	0.67
16	702.29	47.4	**701.27**	49.7	0.15	705.51	57.2	**703.27**	66.5	0.32
17	868.09	60.3	**864.05**	57.8	0.46	864.75	62.2	**864.26**	67.0	0.06
18	1076.4	138.2	**1063.48**	125.3	1.20	1094.98	151.8	**1091.02**	161.3	0.36
19	778.15	169.8	**763.78**	155.4	1.85	800.19	255.7	**790.14**	235.7	1.26
20	544.27	441.5	**541.35**	375.2	0.54	**553.11**	655.7	553.9	656.8	−0.14
21	1053.38	284.2	**1039.65**	247.2	1.30	1087.58	399.9	**1072.87**	371.2	1.35
22	1078.66	296.3	**1061.26**	256.9	1.61	1123.48	487.3	**1105.15**	440.6	1.63

Table 8 continued

Inst	The *Unrestricted* 2L-CVRP					The *Sequential* 2L-CVRP				
	From *Heur1* to *Heur5*		From *Heur1* to *Heur6*		$\%gap_{H6}$	From *Heur1* to *Heur5*		From *Heur1* to *Heur6*		$\%gap_{H6}$
	s	*tot*	*s*	*tot*		*s*	*tot*	*s*	*tot*	
23	1082.12	328.2	**1073.1**	280.2	0.83	1124.19	486.6	**1104.31**	483.2	1.77
24	1153.63	242.8	**1140.63**	206.4	1.13	1182.77	340.4	**1174.85**	299.5	0.67
25	1413.59	467.2	**1394.44**	417.4	1.35	1474.43	688.6	**1449.54**	650.2	1.69
26	1386.83	527.5	**1369.02**	417.7	1.28	1421.98	748.7	**1406.55**	730.0	1.08
27	1400.34	407.4	**1375.52**	353.2	1.77	1449.34	560.9	**1414.9**	511.8	2.38
28	2737.09	634.9	**2671.74**	540.2	2.39	2786.51	962.8	**2751.72**	831.1	1.25
29	2298.56	763.9	**2270.18**	671.1	1.23	2391.95	1149.0	**2357.68**	1181.6	1.43
30	1877.62	840.6	**1839.2**	718.7	2.05	2004.97	1169.1	**1935.82**	1118.8	3.45
31	2404.07	998.9	**2372.69**	810.0	1.31	2540.08	1411.7	**2464.52**	1429.5	2.97
32	2340.55	1002.1	**2315.32**	860.4	1.08	2498.01	1509.9	**2410.73**	1561.3	3.49
33	2420.18	1089.2	**2381.78**	891.2	1.59	2571.96	1487.5	**2486.88**	1559.3	3.31
34	1240.37	1497.0	**1220.2**	1229.9	1.63	1349.34	2033.6	**1288.92**	2151.4	4.48
35	1504.23	1951.0	**1479.41**	1581.2	1.65	1676.89	1912.5	**1601**	2390.3	4.53
36	1780.23	1923.1	**1755.62**	1658.1	1.38	1896.79	2226.5	**1842.87**	2242.7	2.84
Avg	1177.63	410.0	1161.89	347.9	1.17	1225.43	543.1	1203.22	553.6	1.35

may be obtained. Finally, the total number of iterations of SA may be reduced. That is why the total executing time may be decreased by using packing heuristics Heur1 to Heur6. After adding the packing algorithm Heur6, for each *Unrestricted* 2L-CVRP instance, the average time required by SA decreases from 410.0 s to 347.9 s, while for the *Sequential* 2L-CVRP instance, it increases from 543.1 s to 553.6 s. So adding the proposed packing algorithm Heur6 improved the performance of SA algorithm without taking much more total computational time for processing the whole algorithm.

The results obtained by the proposed algorithm for all tests are reported in Tables 9 and 10 of the Appendix.

5 Conclusions

In this paper we have examined two versions of the 2L-CVRP, the *Unrestricted* 2L-CVRP and the *Sequential* 2L-CVRP. To the best of our knowledge, it is the first time for the simulated annealing metaheuristics to be used in this problem. From the computational results, it is shown that SA achieves good results. A new packing heuristic algorithm is introduced, whose efficiency is showed by the experiment results. For constructing the initial solution, a stochastic algorithm is employed. The future work is to further improve the performance of this algorithm, especially for Class 1, and extend it to solve the capacitated vehicle routing problem with three-dimensional loading constraints.

Appendix

See Tables 9 and 10.

Table 9 Results obtained for the *Unrestricted* 2L-CVRP

Instance	Class1	Class2	Class3	Class4	Class5
1	278.73	285.50	299.70	296.75	278.73
2	334.96	334.96	353.48	342.00	334.96
3	358.40	387.70	398.90	364.45	358.40
4	430.89	430.89	430.89	447.37	430.89
5	375.28	375.28	383.52	383.88	375.28
6	495.85	498.16	504.68	505.38	495.85
7	568.56	725.46	706.99	703.49	658.64
8	568.56	718.18	748.83	723.65	621.85
9	607.65	607.65	607.65	625.10	607.65
10	535.80	694.70	628.22	754.76	699.05
11	505.01	717.32	733.64	834.77	645.75
12	610.00	621.28	610.00	614.24	610.23
13	2006.34	2703.84	2491.78	2609.81	2416.04
14	837.67	1078.10	1037.83	986.84	925.56
15	837.67	1094.91	1179.96	1261.71	1234.50
16	698.61	698.61	698.61	709.27	698.61
17	861.79	870.86	861.79	861.79	861.79
18	723.54	1046.89	1115.74	1146.30	944.97
19	524.61	784.06	797.31	809.68	664.07
20	241.97	552.98	548.36	574.34	489.75
21	693.56	1050.05	1174.10	1012.80	921.66
22	744.92	1068.13	1104.84	1115.07	957.00
23	844.88	1071.26	1141.35	1110.84	968.94
24	1029.25	1216.03	1120.87	1157.34	1068.26
25	830.83	1463.10	1438.03	1467.33	1209.31
26	819.56	1330.00	1411.41	1468.02	1266.66
27	1098.91	1379.59	1446.60	1373.65	1302.25
28	1054.66	2705.21	2808.38	2740.05	2433.33
29	1190.19	2283.11	2211.08	2379.96	2206.58
30	1050.64	1882.77	1935.96	1939.11	1598.97
31	1361.82	2388.70	2443.76	2540.45	2117.84
32	1339.50	2377.97	2381.73	2419.24	2082.34
33	1334.20	2393.48	2512.06	2519.44	2102.13
34	719.89	1227.83	1289.35	1267.62	1095.99
35	888.00	1460.40	1548.33	1586.15	1322.74
36	603.69	1776.71	1906.05	1764.13	1575.60

Table 10 Results obtained for the *Sequential* 2L-CVRP

Instance	Class1	Class2	Class3	Class4	Class5
1	278.73	300.08	307.53	296.75	285.93
2	334.96	347.73	356.24	342.00	334.96
3	358.40	403.93	413.63	382.81	358.40
4	430.89	451.98	446.61	447.37	430.89
5	375.28	388.72	397.08	389.28	375.28
6	495.85	499.08	504.68	515.05	495.85
7	568.56	762.26	709.72	713.28	694.54
8	568.56	730.87	752.42	736.73	646.46
9	607.65	611.49	640.63	625.13	607.65
10	535.80	721.77	658.49	798.20	712.43
11	505.01	742.88	782.10	852.87	663.70
12	610.00	629.82	610.00	624.01	610.23
13	2006.34	2763.95	2567.96	2743.53	2490.87
14	837.67	1140.40	1118.46	1009.00	976.53
15	837.67	1116.85	1187.59	1289.67	1265.14
16	698.61	698.61	706.61	709.27	698.61
17	861.79	870.86	861.79	862.62	861.79
18	723.54	1077.68	1122.90	1182.84	980.64
19	524.61	798.37	827.14	847.24	687.80
20	241.97	556.93	556.56	595.56	506.55
21	693.56	1094.31	1201.06	1048.64	947.48
22	744.92	1097.86	1162.00	1148.50	1012.23
23	844.88	1093.76	1166.17	1151.76	1005.56
24	1029.25	1243.14	1169.68	1191.85	1094.71
25	830.83	1500.57	1503.24	1527.20	1267.13
26	819.56	1344.73	1438.66	1519.48	1323.33
27	1098.91	1424.15	1481.29	1407.20	1346.94
28	1054.66	2735.56	2870.37	2852.44	2548.51
29	1190.19	2341.01	2300.88	2420.50	2368.32
30	1050.64	1943.73	2088.80	2029.50	1681.24
31	1361.82	2430.78	2530.60	2671.37	2225.34
32	1339.50	2451.20	2464.30	2530.60	2196.81
33	1334.20	2465.03	2618.09	2655.45	2208.93
34	719.89	1307.73	1347.77	1352.35	1147.81
35	888.00	1516.69	1647.97	1853.62	1385.72
36	603.69	1887.97	1991.55	1832.86	1659.08

References

Baldacci R, Christofides N, Mingozzi A (2008) An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. Math Program 115:351–385

Bodin L, Golden B, Assad A, Ball M (1983) The state of the art in the routing and scheduling of vehicles and crews. Comput Oper Res 10:63–212

Boschetti MA, Mingozzi A (2003a) The two-dimensional finite bin packing problem. Part I: new lower bounds for the oriented case. 4OR Q J Oper Res 1:27–42

Boschetti MA, Mingozzi A (2003b) The two-dimensional finite bin packing problem. Part II: New lower and upper bounds. 4OR Q J Oper Res 1:135–147

Breedam AV (1995) Improvement heuristics for the vehicle routing problem based on simulated annealing. Eur J Oper Res 86:480–490

Cerny V (1985) Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. J Optim Theory Appl 45:41–51

Chazelle B (1983) The bottom-left bin packing heuristic: an efficient implementation. IEEE Trans Comput C-32:697–707

Chiang WC, Russell RA (1996) Simulated annealing metaheuristics for the vehicle routing problem with time windows. Ann Oper Res 63:3–27

Cordeau JF, Gendreau M, Laporte G, Potvin JY, Semet F (2002) A guide to vehicle routing heuristics. J Oper Res Soc 53:512–522

Cordeau JF, Gendreau M, Hertz A, Laporte G, Sormany JS (2005) New heuristics for the vehicle routing problem. In: Langevin A, Riopel D (eds) Logistics systems: design and optimization. New York, Springer, pp 279–297

Croes G (1958) A method for solving traveling salesman problems. Oper Res 6:791–812

Dantzig GB, Ramser JH (1959) The truck dispatching problem. Manage Sci 6:80–91

Ferenc B (2004) Surprising results of trie-based FIM algorithms. In: Bart G, Mohammed JZ, Roberto B (ed) Proceedings of the IEEE ICDM workshop on frequent itemset mining implementations (FIMI'04), vol. 126 of CEUR workshop proceedings, Brighton, UK

Fisher ML (1995) Handbooks in operations research and management science. In: Ball MO, Magnanti TL, Momma CL, Nemhauser GL (eds) Network routing, vol 8. Amsterdam, North-Holland, pp 1–33

Fuellerer G, Doerner KF, Hartl RF, Iori M (2009) Ant colony optimization for the two-dimensional loading vehicle routing problem. Comput Oper Res 36:655–673

Fukasawa R, Longo H, Lysgaard J, Marcus PA, Reis M, Uchoa E, Werneck RF (2006) Robust branch-and-cut-and-price for the capacitated vehicle routing problem. Math Program 106:491–511

Gendreau M, Laporte G, Potvin JY (1997) Vehicle routing: modern heuristics. In: Aarts EHL, Lenstra JK (eds) Local search in combinatorial optimization. Wiley, Chichester, pp 311–336

Gendreau M, Laporte G, Potvin JY (2002) Metaheuristics for the capacitated VRP. In: Toth P, Vigo D (eds) The vehicle routing problem. Monographs on discrete mathematics and applications. SIAM, Philadelphia, pp 129–154

Gendreau M, Iori M, Laporte G, Martello S (2008) A tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. Networks 51:4–18

Iori M (2004) Metaheuristic algorithms for combinatorial optimization problems. PhD thesis, DEIS, University of Bologna

Iori M (2005) Metaheuristic algorithms for combinatorial optimization problems. 4OR Q J Oper Res 3:163–166

Iori M, Salazar Gonzalez JJ, Vigo D (2007) An exact approach for the vehicle routing problem with two dimensional loading constraints. Transp Sci 41:253–264

Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680

Laporte G (2009) Fifty years of vehicle routing. Transp Sci 43:408–416

Laporte G, Semet F (2002) Classical heuristics for the capacitated VRP. In: Toth P, Vigo D (eds) The vehicle routing problem. Monographs on discrete mathematics and applications. SIAM, Philadelphia, pp 109–128

Laporte G, Gendreau M, Potvin JY, Semet F (2000) Classical and modern heuristics for the vehicle routing problem. Int Tran Oper Res 7:285–300

Li XY, Tian P, Leung SCH (2009) An ant colony optimization metaheuristic hybridized with tabu search for open vehicle routing problems. J Oper Res Soc 60(7):1012–1025

Lin S (1965) Computer solutions of the traveling salesman problem. Bell Syst Tech J 44:2245–2269

Lin SW, Yu VF, Chou SY (2009) Solving the truck and trailer routing problem based on a simulated annealing heuristic. Comput Oper Res 36:1683–1692
Lodi A, Martello S, Vigo D (1999) Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems. INFORMS J Comput 11:345–357
Martello S, Vigo D (1998) Exact solution of the two-dimensional finite bin packing problem. Manage Sci 44:388–399
Mester D, Bräysy O (2007) Active-guided evolution strategies for large scale capacitated vehicle routing problems. Comput Oper Res 34(10):2964–2975
Moghaddam RT, Safaei N, Gholipour Y (2006) A hybrid simulated annealing for capacitated vehicle routing problems with the independent route length. Appl Math Comput 176:445–454
Osman IH (1993) Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. Ann Oper Res 41:421–451
Pisinger D, Sigurd M (2007) Using decomposition techniques and constraint programming for solving the two-dimensional bin packing problem. INFORMS J Comput 19:36–51
Prins C (2004) A simple and effective evolutionary algorithm for the vehicle routing problem. Comput Oper Res 31(12):1985–2002
Tarantilis CD (2005) Solving the vehicle routing problem with adaptive memory programming methodology. Comput Oper Res 32(9):2309–2327
Tarantilis CD, Kiranoudis CT (2002) Bone route: an effective memory-based method for effective fleet management. Ann Oper Res 115:227–241
Toth P, Vigo D (2002a) The vehicle routing problem. Monographs on discrete mathematics and applications. SIAM, Philadelphia
Toth P, Vigo D (2002b) Branch-and-bound algorithms for the capacitated VRP. In: Toth P, Vigo D (eds) The vehicle routing problem. SIAM, Philadelphia, pp 29–52
Waters CDJ (1987) A solution procedure for the vehicle scheduling problem based on iterative route improvement. J Oper Res Soc 38:833–839
Wei L, Zhang D, Chen Q (2009) A least wasted first heuristic algorithm for rectangular packing problem. Comput Oper Res 36(5):1608–1614
Zachariadis EE, Tarantilis CD, Kiranoudis CT (2009) A guided tabu search for the vehicle routing problem with two-dimensional loading constraints. Eur J Oper Res 195:729–743

Author Biographies

Stephen C. H. Leung is an Assistant Professor in the Department of Management Sciences at the City University of Hong Kong. He received his PhD in Management Science and Operational Research from City University of Hong Kong. Before joining the teaching profession, Dr. Leung was a consultant in Wilbur Smith Associate Ltd. He had been involved in a variety of projects about transport modeling and traffic impact assessment. His current research interests are in logistics and supply chain management, operations management, transportation and RFID applications.

Jiemin Zheng is a postgraduate student in the Computer Science department of Xiamen University. He received the B.S. in computer science department from Xiamen University. His research interests include computational intelligence and finance data mining.

Defu Zhang is an associate professor in the Computer Science department of Xiamen University. He received the B.S. and M.S. in computational mathematics from Xiangtan University, and Ph.D. in computer software and its theory from Huazhong University of Science & Technology. He worked at the Longtop as a PostDoc in the data mining group from 2006 to 2008. His research interests include computational intelligence and finance data mining.

Xiyue Zhou is a postgraduate student in the Computer Science department of Xiamen University. He received the B.S. in computer science department from Xiamen University. His research interests are finance data mining and computation intelligence.