2008 International Conference on Management Science & Engineering (15th)
September 10-12, 2008 Long Beach, USA

A New Tabu Search Heuristic Algorithm for the Vehicle Routing Problem with Time Windows

QI Ming-yao[1], MIAO Li-xin[1], ZHANG Le[2], XU Hua-yu[2]
1 Graduate School at Shenzhen, Tsinghua University, P.R.China, 518055
2 Department of Mathematical Sciences, Tsinghua University, P.R.China, 100084

Abstract: This paper describes a new design of Tabu Search (TS) algorithm for solving the Vehicle Routing Problem with Time Windows (VRPTW). Since VRPTW is a well known NP-hard problem, heuristic algorithms such as Tabu Search are always used to get a good approach. The former published designs of TS usually focus on the neighbor structure, the relaxation to the objective function or the multi-period algorithms. This paper has two contributions. First, it designs an objective value-based Tabu List structure to help decreasing the tabu list size and escape from local optima. it still adopts some tactics like adaptive tabu size, randomly selected neighbor structure and so on. Second, from a problem oriented point of view, it shows that, with this self-adaptive Tabu List, even five kinds of simple neighbor structures and a single period algorithm with original objective function can get very good solutions. We tested this algorithm with Solomon's VRPTW benchmark problems, and 7 of the best known solutions are updated. Another advantage of this algorithm is its efficiency. It runs on an average of 80 seconds on a normal personal computer for a solution of a problem with 100 customers.

Keywords: Tabu search, vehicle routing problem, time window, heuristic algorithm, genetic algorithm

1 Introduction

The Vehicle Routing Problem (VRP) focuses on the optimal arrangement or schedule of a fleet of vehicles while serving scattered customers. When customer has his own time period and only during this period the service can be accepted, the problem is called Vehicle Routing Problem with Time Windows (VRPTW).

The VRP was first introduced in 1959. It is still a very important problem that is broadly studied now because of its wide use and hardness. Even the simplest form of VRP is Strongly NP-Complete. Thus, heuristic algorithms are always introduced to get a good solution. Tabu Search (TS) is one of the most important heuristics.

Tabu Search was first introduced by F. Glover (1989, 1990)[1][2]. TS is a promotion of neighbor search algorithm, using some memory method to prevent local optimal solutions. The most important fact in TS is the use of Tabu List, which lists the information of the solutions that is not allowed to reach while doing a neighbor search. This is the essential difference that tells Tabu Search from simple neighbor search. After it has been developed, TS becomes popular used in solving many difficult problems, including the VRPTW.

Supported by the National Natural Science Foundation of China (70702003) and Natural Science Foundation of China Guangdong Province (7301721)

2 Literature review

The VRP (Vehicle Routing Problem, or VSP, Vehicle Scheduling Problem) lies at the heart of distribution systems, because of its important role and wide use. It is also a very difficult problem, since the simplest form of VRP is Strongly NP-Complete, i.e. there is no even pseudo-polynomial time algorithm. Thus, a lot of heuristic algorithms are introduced to solve the VRP. Tabu Search (TS) is one of such algorithms.

Tabu Search was first proposed by Glover in 1986, and became a mature algorithm in 1989 and 1990 by Glover's two articles [1] [2]. In [1], Glover constructed the simple TS model through the well-known hill climbing heuristic algorithm. He gave also a general way of constructing and updating the Tabu List, as well as the aspiration criteria. He then gave an example of a partition problem, and discussed some possible improvement of the algorithm, including consideration of the dual role between Tabu List restrictions and aspiration criteria, choosing between the dominant and deficient moves, strategic oscillation of the objective function, use of intermediate and long-term memory functions and a probabilistic way of searching the current best solution. Last, he applied the algorithm in several TSP instances.

Glover in [2] continued his discussion of Tabu Search, focusing on the refinement of the algorithm. He first discussed dynamic strategies for managing a tabu list rather than the simple circular list. These strategies include the C-sequence method and the reverse elimination method for both single and paired attribute moves, to prevent any essentially same solutions from entering the tabu list. He also mentioned the method of introducing more than one tabu list, and indicated an

1-4244-2388-0/08/$20.00 ©2008 IEEE

example in TSP. Next he discussed the structure of the move sets, and gave a way of dividing the Tabu Search into sequential stages. Last, he applied the Tabu Search into mixed integer programming.

These two articles are the very classical works of the TS algorithm. Especially in [1], Glover introduced several general rules for TS, which are still the basic principles of the design of this algorithm. The two articles focus mainly on the theory and the basic routines of the operations of the methods, giving few actual calculating examples. Later articles, by the given rules, improved the algorithm and applied it in varies problems. Next we consider only the application of Tabu Search in VRP.

Laporte et al. (2000) published a survey of heuristics for the VRP[3]. In this survey, he mentioned 6 successful implementation of Tabu Search from 1993 to 1998, including the well-known Taburoute method of Gendreau et al. (1994)[4], a new neighborhood structure and a way of local improvements of Xu and Kelly (1996)[5], etc.

Gendreau et al. (1994) introduced the Taburoute method, a very involved implementation of TS to solve the VRP with capacity and route length constraints[4]. This method requires running the TS for three times with different parameters, the first for initialization, the second for solution improvement and the last for intensification. Each time running the TS, they used a special insertion procedure called GENI, which was developed by Gendreau et al. (1992) [6]. This procedure helps to find the neighbor of the current solution. In this article, they introduced a random length of the tabu list, i.e. the number of steps one solution is tabu is not a fixed number, but a random number in some predetermined interval. Concurrently, an additional objective function with a penalty component as well as the traditional objective function is used, thus the algorithm can start from a feasible or infeasible initial solution. Last they compared the result of the Taburoute method with the existing 12 heuristic algorithms of the same problem using different instances, and in most cases, Taburoute acts better than most of the former algorithms.

Xu and Kelly (1996) developed a more complicated structure of neighborhood[5]. They constructed a net flow model to express the adding/removing of nodes between routes, and using 3-opt exchanges during solution improvements. This algorithm uses a lot of control parameters, and these parameters are adjusted dynamically during the circulation. In this algorithm, they also used a frequency-based function as a penalty to the cost of each node, and a repository of best solutions is memorized to periodically restart the search and reinitialize the control parameters. These are the two ways to raise diversification in there article.

In 1998, Barbarosoglu and Ozgur developed a new Tabu search heuristic to solve the single-depot vehicle routing problem of a distribution company carrying goods from a depot to a set of dedicated dealers. It extends the Tabu search with a new neighborhood generation procedure. In case there exists a clustering pattern in the locations of the dealers, the classical procedure of neighborhood generation would perform rather poorly. Barbarosoglu and Ozgur (1999) proposed a new heuristic called TANEC to improve the neighborhood construction, where the scattering pattern had been taken into account[7]. Some work are also reported to apply constraint programming to VRPTW and harvest good efficient neighbor search [8][9].

Recent articles always focus on the extension forms of the traditional VRP and on the model rather than on the algorithm. Ho and Haugland (2004) introduced a VRP model with time window and split deliveries[10]. In this model, demand of one customer can be divided into parts and fulfilled by more than one vehicle in the same time window. They used the common way of constructing the neighborhood, and designed the TS algorithm. A similar model of multi-compartment VRP was set up by Fallahi et al. (2008)[11]. In this model, each vehicle has different compartments dedicated to each product, respectively.

A recent improvement of the algorithm was introduced in 2007 by Cordeau et al[12]. In [12], they combined a classical Tabu search with perturbation operators that helped escape from local optima. The reformative algorithm, which mainly focuses on the perturbation mechanisms, is simple yet general because of the randomization and diversity. Therefore, it does not rely on the availability of a feasible initial solution and can start from any status, and thus it can be used to provide feasible initial solutions to more sophisticated algorithms. Simultaneously, it is robust and can be easily implemented.

We can conclude that the TS is a direct evolution of the local search algorithm, thanks to the use of Tabu List. The two most important points to get better solution are regional intensification and global diversification. In the design of all the algorithms mentioned above, they were trying to raise the two, through different ways. Generally, intermediate and long term memory functions are introduced. If we consider the tabu list as a kind of short term memory function, then the design of the TS algorithm framework can be then considered as the design of three memory functions with different terms. These three functions set up a framework of the TS algorithm. In this framework, a lot of other algorithms can be embedded, and this is what most of the implementations did. From this point of view, the main contribution of the TS algorithm is right this framework. Under this framework, varies scholars have developed different kinds of TS. New neighborhood structure, new searching algorithms or new data structures, these are all improvements of the embedded algorithms.

The former works mentioned above has designed good implementations of TS algorithm and generated very good solutions on VRPTW, but the complexity of neighbor structure has lowered the efficiency. Although parallel algorithms have been used for acceleration, for example, Badeau (1997)[13] and Garcia (1994)[14], they

have the disadvantage of relying on specific computers, and also, they are much more complicated. As we show above, nearly all of the former researches on TS algorithm focus on the neighbor structure, different intermediate or long-term memory functions, relaxation of objective function or multi-period algorithm, but the Tabu Lists are mainly the traditional move-tabu or solution-tabu. In the following chapters, we will give a new TS algorithm that uses another tabu method and a set of very simple neighbor structure. This algorithm is characterized as to be easy to implement, and still efficient enough to reach or even exceed the former results very quickly.

3 Problem formulation

We consider the standard VRPTW. The model satisfies: 1. Single depot, every vehicle starts and returns to this depot; 2. Consider the vehicle capacity constraint and the maximum vehicle number constraint. 3. Demand of every customer does not exceed the capacity of one vehicle, and one customer is served only by one vehicle; 4. If a vehicle arrives before the time window, it will wait until the service is possible, and arrival after the time window is not allowed. From 4, we can assume that all vehicle starts from the depot at time 0. According to Solomon's paper (1997)[15] After the complete vehicle schedules have been created, we can adjust the depot departure time separately for each vehicle to eliminate any unnecessary waiting time. The problem has two objectives. The first objective is to minimize the number of vehicles, and the second objective is to minimize the total cost of all routes.

Suppose that:

V: number of vehicles that can be used.

C: number of customers.

W: weight capacity of each vehicle.

C_i: demand of customer i.

S_{ij}: expense from customer i to customer j. Here we suppose the expense is equivalent to the distance.

T_{ij}: time consuming from customer i to customer j.

S_i: the service time in customer i.

E_i: the lower bound of the time window of customer i.

L_i: the upper bound of the time window of customer i.

P_i: the penalty for late arrival after L_i.

Then the decision variables will include:

K: number of vehicles that actually used.

n_k: number of customers that vehicle k would serve.

a_j^k: the j^{th} customer among those customers served by vehicle k.

T_i: the time when the vehicle arrives at customer i. For all i, if $T_i < E_i$, set $T_i = E_i$, for we assume that if the vehicle come earlier than the permitted service time, it should wait until the service is available. We can calculate T_i as:

$$T_{a_j^k} = \begin{cases} t_{0a_1^k}, j = 1 \\ T_{a_{j-1}^k} + t_{a_{j-1}^k a_j^k}, j = 2,...,n_k \end{cases} \quad (1)$$

$Cost_i$: the cost spent on customer i, including the penalty if the vehicle arrives after L_i.

The formulation of the VRPTW discussed in this paper is:

Minimize COST;

$$COST = \sum_{k=1}^{K} (S_{0a_1^k} + \sum_{j=1}^{n_k - 1} S_{a_j^k a_{j+1}^k} + S_{a_{n_k}^k 0}) + \sum_{i=0}^{C} P_i \quad (2)$$

Subject to

$1 \le K \le C$.

$\sum_{k=1}^{K} n_k = K$, $1 \le n_k \le C$.

$\{a_j^k: 1 \le k \le K, 1 \le j \le n_k\} = \{1,2,3..., C\}$

$\sum_{i=1}^{n_k} c_{a_i^k} \le W$

Now we discuss the design of the TS algorithm in the next section.

4 Design of algorithm

4.1 Representation of a solution

Let 0 represent the depot and 1, 2, …, C represent the C customers. Suppose there are V vehicles in total that be used. We represent a solution with a number sequence which contains from 1 to n and C+1 zeros. The first and the last number must be 0, and the sub-sequence between two adjacent zeros represents the service order, or the route, of one vehicle. For example, the sequence

0 5 9 0 4 6 0 2 1 3 0 10 7 8 0 0

represents a route shown in Fig.1 and indicates that there are maximum 5 vehicles.

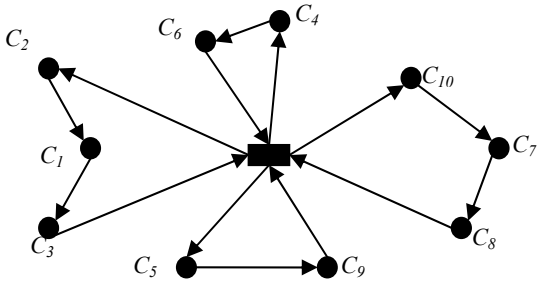

Fig.1 Representation of a solution

4.2 Initial solution

Some algorithms may choose to use infeasible initial solutions (2002)[16], here we use a greedy method to get the initial feasible solution for the reason that we don't adopt relaxation . The steps are:

Step 1: We start from a vehicle, while not violate the capacity constraint, as well as the time window constraint, search one un-served customer with minimum cost, and add to the route.

Step 2: When no such customers found, let the vehicle return the depot, and start another vehicle.

Step 3: When all customers are arranged, finish the process, otherwise go to step 1.

4.3 Tabu List

Tabu list is one of the key factors that determine the quality of a TS algorithm. The most popular tabu list is constructed by those recently visited solutions, or the moves to a solution. If the size of the tabu list is too large, it will spend more time to compare with the current solution one by one, but if the size of the tabu list is too small, it will be very hard to escape from local optima. As for the combinational optimization problem like VRPTW problem, especially for as large instance as having over 100 customers, the solution space is very huge. Even the forbidding of a few hundred or thousand solutions works very poorly to escape from local optima and reach the global satisfying solution.

In this paper, we give a new Tabu List design. The Tabu List memorizes intervals that contain each object value of the solutions reached by the search algorithm. To make the maximum object value be our target, we take *A-COST* as our object function, with A referring to a big constant that guarantee object value is nonnegative. If a solution's objective value is in any of the intervals, it is considered as tabu. As showed in Fig.2, when a solution with objective value z is reached, we tabu the interval $[z-h, z+h]$ for a specific T steps. By using tabu of object value intervals, instead of solution collection, the algorithm can quickly escape from local optimal solutions, otherwise, as we know, the quantity of solutions value in $[z-h, z+h]$ is enormous and even we have a very big sized traditional tabu list, it still take a long time to get out of this area.

Meanwhile, we should be very cautious to set the objective value interval as tabu, since if the interval is too large, the global optima may be forbidden, or if it is too small, fewer bad solutions can be excluded. Moreover, when the object value is far away from the global optima, we should set a wider interval, and once the object value is approaching the global optima, the interval should be very narrow. Therefore, we designed a self-adaptive interval plan that we think is similar to Gendreau's idea of random tabu size[5], although we have different tabu contents. In our plan, for two predetermined parameters α and β, when we find a solution, let the change of objective value between the current solution and the previous one be H, if $H>0$, we choose $h=\alpha H$; else we choose $h=-\beta H$. In real experiments, we also constraint h into an interval $[h_{min}, h_{max}]$, if h is too large or too small, let it be the nearest bound. Later in next chapter, we will show that this Tabu List can greatly improve the algorithm.

4.4 Neighbor structure

Instead of adopting a fixed neighbor structure, we used five simple neighbor structures and each time select one randomly from them. This method, to our consideration, is very similar to the Variable Neighbor Search[17][18][19], and also our approach is very easy to implemented.

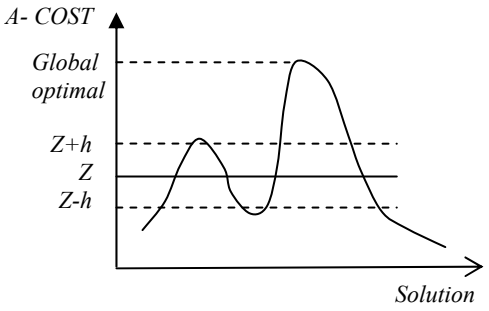

Fig.2 The object value based tabu list

The five neighbor structures are as follows:

(1) Swap

We simply swap two points of the solution. These two points can either be in the same route or different routes, even could be 0. And they are randomly selected. For example, for a two routes solution like below, we exchange the customer 4 from route one with the customer 7 from the other route.

(0 1 2 3 4 5 0 6 7 8 0) -> (0 1 2 3 7 5 0 6 4 8 0)

(2) 2-Opt

The 2-Opt movement is to change two arcs of the same route into two other arcs that don't formerly belong to the route. Its effect on our presentation is to reverse the direction of a sub-sequence in one route. The two points of the sub-sequence are randomly selected.

(0 1 2 3 4 5 6 7 8 9 0) -> (0 1 2 3 7 6 5 4 8 9 0)

(3) Reinsert

We remove one point and reinsert it into another position. The point could be either a customer or a zero. The point and the position to insert are randomly selected.

(0 1 2 3 4 5 0 6 7 8 0) -> (0 1 2 3 5 0 6 7 4 8 0)

(4) Exchange

We change the 'tail' of two routes, as Fig.3 shows. The positions of the tails are randomly selected.

(0 1 2 3 4 5 0 6 7 8 9 0) -> (0 1 2 8 9 0 6 7 3 4 5 0)

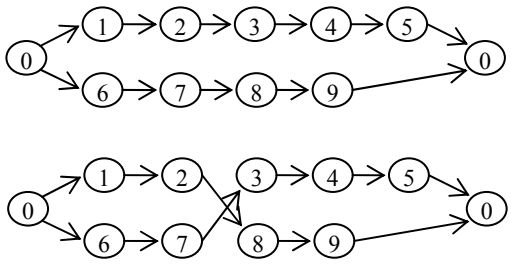

Fig.3 Exchange between two routes

(5) Reverse-exchange

We change the 'tail' of one route with the 'head' of another route, and reverse the directions, as Fig.4 shows.

The position of head and tail are randomly selected.

(0 1 2 3 4 5 0 6 7 8 9 0) -> (0 1 2 7 6 0 5 4 3 8 9 0)

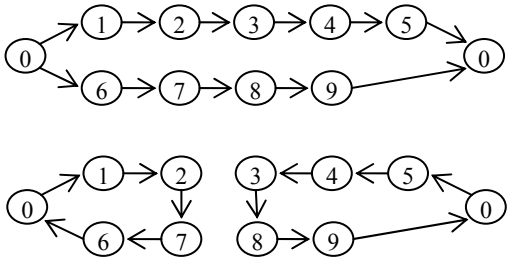

Fig.4 Reverse-exchange between two routes

While doing a neighbor search, not all these neighbors are searched. In practice, we search only a specific number n_1 of neighbors, and they are randomly selected in these 5 different kinds. Only the best n_2 neighbors are considered.

4.5 Calculation procedure

When all n_2 moves are tabu, choose the best among them to be the next solution; when the so far best solution has not updated for M iterations, stop.

So far we have developed all detail of a Tabu Search algorithm. The complete procedure is as Fig.5 shows:

Step 1: Get the initial solution, set cursol (current solution) = sofarbest (so far best solution) = initial solution.

Step 2: Choose the best n_2 solutions among n_1 neighbors of cursol; choose the best non-tabu solution among the n_2 solutions and assign to cursol. If all n_2 solutions are tabu, choose the best one to be cursol. Update the Tabu List.

Step 3: If cursol is better than sofarbest, sofarbest = cursol. If sofarbest has not updated for M iterations, stop. Else go to Step 2.

5 Calculation results

We test our algorithm using Solomon's VRPTW instances (Solomon 1987). We choose the parameters as: α=0.01, β=0.02, T=7, h_{min}=0.05, h_{max}=1, n_1=5000, n_2=10, M=1000. These parameters are chosen after several tests, and the results show that these are most efficient values. With these values, our algorithm, coded with Visual C++ 6.0, runs on average 85 seconds (total running time) on Intel Pentium 4 Mobile CPU 1.6GHz and 512MB of RAM. Bräysy and Gendreau (2002) [20] mentioned the CPU time of several former good algorithms, but most of them are time-consuming. For example, on Pentium CPU 400M, the running time is 30 minutes. Considering the factor of CPU promotion, our algorithm is still very efficient.

According to the results of the best known solutions of Solomon's VRPTW reported at http://www.sintef.no/static/ am/ opti/ projects/ top/ vrp/ bknown.html, our algorithm has updated 7 of them, as showed in Tab.1.

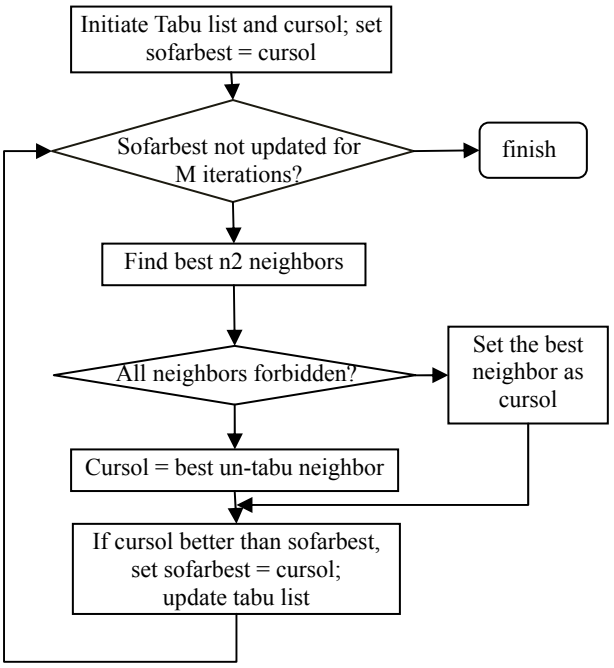

Fig.5 The calculation procedure

And for all other C problems, i.e. C101~C109, C201, C202, C205~C208, our algorithm has generated the same solution as reported above.

Tab.1 Calculation results

Problem	Former best		Our result	
	Vehicle number	Total distance	Vehicle number	Total distance
C203	3	591.17	3	589.55
C204	3	590.6	3	588.22
R101	19	1645.79	19	1645
R102	17	1486.12	17	1445.2
R103	13	1292.68	13	1263.5
RC104	10	1135.48	10	1132.1
RC107	11	1230.48s	11	1216.1

From the results we can see that, with even the five simple neighbor structures, the specific Tabu List design can get very good solutions. We have also tested our algorithm with other frequently used Tabu List, for example, move-tabu, insertion-tabu and solution-tabu, but the results are far behind this. For the most frequently used neighbor structures, the quantity of a solution's neighbors is very large, and most changes of objective value due to moves to next solutions always scatter around a small region. If the Tabu List only tabus one move or one solution, even one objective value, it cannot effectively prevent the cycling back to the local optima. Our Tabu List tabus a number of intervals of objective values; these will include a lot of solutions, and can thus reduce the length of Tabu List. That's why the TS algorithm with this Tabu List design is fast and effective.

6 Conclusions and further research

In this paper, we design a tabu search algorithm that uses simple neighbor structures, original objective functions and just one calculation period. Even no intermediate or long-memory functions are introduced. But our new design of the Tabu List makes it a very powerful and efficient TS algorithm. This algorithm is very easy to implement, and thus has very wide use in both research and practice.

Further research will probably be around the combination of this Tabu Search Algorithm with other heuristics algorithms like Genetic Algorithm (GA). Actually we have made experiments in two direction on such hybrid approach: one is taking TS as the mutation algorithm so as to improve the GA performance, and the other is that, once the solution has not been updated for specific iterations in TS loops, we take GA to find a better solution with the so far best as the seed, since GA can always find a at least not worse solution. Our efforts have seen some progress till now. Other research area we think is the dynamic vehicle routing problems considering time-dependent traffic status or real time customer demands, which is one of the key targets set in our funded projects.

References

[1]F. Glover. Tabu search – part I[J]. ORSA Journal on Computing, 1989, 1(3):190-206.

[2]F. Glover, Tabu search – part II[J]. ORSA Journal on Computing, 1990, 2(1):4-32.

[3]G. Laporte, M. Gendreau, J. Y. Potvin, F. Semet. Classical and modern heuristics for the vehicle routing problem[J]. International Transactions in Operational Research, 2000, 7(4):285-300.

[4]M. Gendreau, A. Hertz, G. Laporte. A tabu search heuristic for the vehicle routing problem[J]. Management Science, 1994, 40(10):1276-1290.

[5]J. Xu, J. Kelly. A Network flow-based tabu search heuristic for the vehicle routing problem[J]. Transportation Science, 1996, 30(4):379-393.

[6]M. Gendreau, A. Hertz, G. Laporte. New insertion and postoptimization procedures for the traveling salesman problem[J]. Operations Research, 1992, 40(6):1086-1094.

[7]G. Barbarosoglu, D. Ozgur. A tabu search algorithm for the vehicle routing problem[J]. Computers & Operations Research, 1999, 26(3):255-270.

[8]G. Pesant, M. Gendreau. A constraint programming framework for local search methods[J]. Journal of Heuristics, 1999, 5(3):255-279.

[9]L. M. Rousseau, M. Gendreau, G. Pesant. Using constraint-based operators to solve the vehicle routing problem with time windows[J]. Journal of Heuristics, 2002, 8(1):43-58.

[10]S. C. Ho, D. Haugland. A tabu search heuristic for the vehicle routing problem with time windows and split deliveries[J]. Computers & Operations Research, 2004, 31(12):1947-1964.

[11]A. E. Fallahi, C. Prins, R. W. Calvo. A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem[J]. Computers & Operations Research, 2008, 35(5) :1725-1741.

[12]J. F. Cordeau, G. Laporte, F. Pasin. Iterated tabu search for the car sequencing problem[J]. European Journal of Operational Research, In Press.

[13]P. Badeau, F. Guertin, M. Gendreau, J. Y. Potvin. A parallel tabu search heuristic for the vehicle routing problem with time windows[J]. Transportation Research Part C, 1997, 5(2):109-122.

[14]B. L. Garcia, J. Y. Potvin, J. M. Rousseau. A parallel implementation of the tabu search heuristic for vehicle routing problems with time window constraints[J]. Computers & Operations Research, 1994, 21(9): 1025-1033.

[15]M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints[J]. Operation Research, 1987, 35(2):254-265.

[16]J. F. Cordeau, G. Laporte, A. Mercier. A unified tabu search heuristic for vehicle routing problems with time windows[J]. The Journal of the Operational Research Society, 2001, 52(8):928-936.

[17]N. Mladenovi´c, P. Hansen. Variable neighborhood search. Computers and Operations Research[J]. 1997, 24(11):1097-1100.

[18]P. Hansen, N. Mladenovi´c. Variable neighborhood search: Principles and applications[J]. European Journal of Operational Research, 2001, 130(3): 449-467.

[19]M. Polacek，R. F. Hartl，K. Doerner，M. Reimann. A variable neighborhood search for the multi depot vehicle routing problem with time windows[J]. Journal of Heuristics, 2004, 10(6):613-627.

[20]O. Bräysy, M. Gendreau. Tabu search heuristics for the vehicle routing problem with time windows[J]. Sociedad de Estadistica e Investigacion Operativa, 2002, 10(2): 211-237.