Computer Science and Information Technology 2(2): 108-119, 2014
DOI: 10.13189/csit.2014.020208 http://www.hrpub.org

Three Strategies Tabu Search for Vehicle Routing Problem with Time Windows

Abdel-Rahman Hedar[1], **Mohammed Abdallah Bakr**[2,*]

[1]Faculty of Computers and Information, Assiut University, Assiut, 71526, Egypt
[2]Faculty of Science, Assiut University, Assiut, 71526, Egypt
*Corresponding Author: ma_bakr85@yahoo.com

Abstract In the vehicle routing problem with time window (VRPTW), the objective is to minimize the number of vehicles and then minimize the total time travelled. Each route starts at the depot and ends at a customer, visiting a number of customers, each once, en route, without returning to the depot. The demand of each customer must be completely fulfilled by a single vehicle. The total demand serviced by each vehicle must not exceed vehicle capacity. An effective tabu search for vehicle routing with time window (TSTS-VRPTW) heuristic for this problem is proposed. The TSTS-VRPTW is based on three function MOVE, EXCHANGE, SWAP. Computational results on Solomon$'s$ benchmarks that consist of six different datasets show that the proposed TSTS-VRPTW is comparable in terms of solution quality to the best performing published heuristics.

Keywords Vehicle routing, Time windows, Service costs

1 Introduction

The vehicle routing problem with time windows (VRPTW) is a crucial issue in transportation and logistics systems [46], [43]. The problem can be defined as designing routes for a fleet of vehicles to serve a set of customers. Each route starts from and terminates at the depot. Each customer is visited once and only once by exactly one vehicle. The routes should satisfy the vehicle capacity and the time window constraints. Besides, the VRPTW has two objectives. The primary objective is to minimize the number of vehicle (NV) routes. The secondary objective is to minimize the total travel distance (TD) with the same number of routes. This paper develops effective tabu search (TSTS-VRPTW) for solving VRPT. It is worth noting that tabu search has been applied to various other vehicle routing problems. The remainder of the paper is organized as follows: (Section 2) provides a comprehensive literature review on methods proposed for the VRPT. (Section 3) present a formal definition of the VRPTW. (Section 4) elaborates the proposed hybrid (TSTS-VRPTW) and provides a detailed description of all algorithmic components. Computational experiments assessing the proof of concept and the quality of the proposed method, along with a comparative performance analysis, are presented in (Section 5), finally conclusions are drawn and pointers for further research are suggested as (section 6).

2 Literature Review

There is a wide range of literature on the VRP as it is a highly relevant, yet complicated problem. We refer to [30], [31] for exact, heuristic and metaheuristic algorithms, and to [2] for recent exact algorithms applied to the VRP. A related problem also frequently seen in practice and studied is the VRPTW. In his seminal paper, [43] extended a number of VRP heuristic methods for the VRPTW. The interested reader is referred to [5], [6] and [12] for an overview of various solution methods applied to the VRPTW. Most of the models developed for the VRP and the VRPTW in the literature considered deterministic parameters such as deterministic travel times, demands and service times.
A comprehensive survey on the stochastic VRP can be found in [17]. The authors argued that uncertainty can be seen in various components of the VRP: stochastic travel times, stochastic demands and/or stochastic customers. In [30], the VRP with stochastic travel and service times was considered. The authors introduced three distinct formulations based on stochastic programming and developed a branch-and-cut approach. [29] developed a solution procedure by inserting a branch-and-cut algorithm into a Monte Carlo solution approach to solve large-sized problems effectively. [47] studied the VRP with the travel times resulting from a stochastic process due to the traffic congestion. The developed queueing models were solved by means of a Tabu Search metaheuristic. Reference [44] studied the stochastic VRP where uncertain customer demands were considered.
Stochastic versions of the VRPTW are introduced more

recently. Reference [1] considered the VRPTW with uncertain travel times. The objective was to minimize the total cost which included the penalty costs due to the early and late arrivals, the operation costs and the fixed cost of vehicles used. A genetic algorithm was proposed to solve the described problem. Reference [38] also studied the VRPTW where the travel times were random variables with a known probability distribution. The number of vehicles used and the total distance traveled were minimized along with the penalties due to arrivals outside the time windows. The authors developed a Tabu Search method. The VRPTW with stochastic travel and service times was studied by [34]. Two formulations based on stochastic programming were proposed. A heuristic algorithm based on Tabu Search was developed to obtain the results effectively. The models in these studies placed emphasis on the customers and considered all cost components together regardless of their relations and differences. Since efficiency plays an important role in operations, we separate out the cost components into transportation costs and service costs. We develop a one-stage model which enables different combinations of these two cost components with respect to the company preferences. Additionally, in our model the time window violations and the overtime of the drivers are handled by the objective function. A technique similar to that applied in [44] is used in our study to calculate the penalties incurred for early and late servicing. Our model takes into account penalties proportional to the expected duration of the earliness and lateness derived from the arrival time distributions.

The classical routing problems and their stochastic versions have been widely solved by applying the Tabu Search metaheuristic to obtain good solutions within a reasonable time. The interested reader is referred to [19] ,[20] for the details about this metaheuristic. [16], [17] implemented the Tabu Search method for the VRP. Reference[10] proposed the adaptive memory, which turned out to be very effective for Tabu Search applications in the VRP. For the VRPTW, some implementations of the Tabu Search method come from [9], [14], and [45]. Our Tabu Search implementation is based on the algorithm given in [9]. We however apply a different function in the local search algorithm to evaluate the solutions since the violations of the time windows are handled by the objective function. In the process of evaluating the solutions, the stochasticity of the problem is taken into consideration. Furthermore, the Tabu Search algorithm is improved by adding a medium-term memory which focuses the search on the promising solutions in the neighborhood. This intensification mechanism operates by restarting from the best feasible solution, and searching its neighborhood effectively by means of a list which includes the moves previously applied from that solution. This structure makes our medium-term memory application different from the intensification approaches in which solutions are generated by extracting good routes from high-quality solutions on-hand (see [36]).

The interested reader is referred to [9], [36], [38], and [45] for a post-optimization heuristic applied in the VRPTW. In [9] ,[45], the authors used a heuristic method developed by [15] for the traveling salesman problem with time windows by modifying the GENIUS procedure [15]). In the post-optimization phase of their heuristic, each node of a route was successively removed and re-inserted to improve the solution on-hand. As a post-optimization method, [36] solved a set partitioning model at the end of the diversification and intensification techniques to improve the solution by using the routes already generated. In [38], a post-optimization method was applied to optimize the waiting times at each customer by using a generalized reduced gradient method.

3 Vehicle Routing Problem

Capacitated vehicle routing problem (CVRP), which is the most elementary version of the VRP, can be defined as follows [13]: n customers are waiting to be served, each of which requires a quantity q_i of goods $(i = 1, 2, ..., n)$. A depot has a fleet to deliver the goods. Each vehicle has a capacity of Q units, *i.e.*, the total demand of customers served on each route could not exceed Q. Therefore, the vehicle has to periodically return to the depot for reloading, which is equivalent to dispatching a new vehicle for delivery. Besides, each customer must be visited once and only once by exactly one vehicle. A solution of the CVRP is a collection of routes that a vehicle starts from the depot, serves some specific customers, and terminates at the depot.

Define $G = (V, A)$ as a directed complete graph, where $V = (c_0, c_1, ..., c_n)$ is the node set, and $A = \{\langle c_i, c_j \rangle \mid c_i, c_j \in V, c_i \neq c_j \}$ is the arc set. c_0 represents the depot, and c_i represents the customer i $(i = 1, 2, ..., n)$. Each node is associated with a demand quantity q_i, among which node c_0 is associated with $q_0 = 0$. Each arc $\langle c_i, c_j \rangle$ is associated with a travel time t_{ij}, which is represented by the Euclidean distance d_{ij} between nodes c_i and c_j $(d_{ij} = d_{ji})$. The objective of the CVRP is to minimize the total travel time of all the vehicles.

The VRPTW is an extension of the CVRP. In the VRPTW, nodes are associated with more properties, and the solution has to satisfy more constraints. A service time s_i is considered; therefore, the vehicle has to stay at the location of customer c_i for a time interval at least s_i ($s_0 = 0$ is associated with the depot c_0) for service. A time window $[e_i , l_i]$ during which the service has to start is considered. Therefore, when a vehicle arrives at customer c_i earlier than e_i , it has to wait until the beginning of the time window to serve the customer. On the other hand, if a vehicle cannot arrive at c_i before l_i , the vehicle cannot serve c_i . At this time, customer c_i should be served by another vehicle. For depot c_0 , the time window is defined as that e_0 is the earliest start time, and l_0 is the latest return time of all the vehicles. The VRPTW has two objectives. The primary objective is to minimize the number of the vehicle routes v . The secondary objective is to minimize the total TD with the same number of routes. The VRPTW can be stated mathematically as follows. Define variable

$$x_{ij}^k = \begin{cases} 1 & \text{if vehicle } k \text{ travels directly from } i \text{ to } j \\ 0 & \text{otherwise} \end{cases}$$

$$y_i^k = \begin{cases} 1 & \text{if customer } i \text{ is served by vehicle } k \\ 0 & \text{otherwise} \end{cases}$$

The goal of the VRPTW is to minimize

$$minZ_1 = \upsilon \tag{1}$$

and

$$minZ_2 = \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=1}^{\upsilon} t_{ij} \times x_{ij}^k \tag{2}$$

s.t.

$$\sum_{i=0}^{n} x_{ij}^k = y_j^k \quad \forall k = 1, ..., \upsilon, \quad \forall j = 1, ..., n \tag{3}$$

$$\sum_{j=0}^{n} x_{ij}^k = y_i^k \quad \forall k = 1, ..., \upsilon, \quad \forall i = 1, ..., n \tag{4}$$

$$\sum_{i=0}^{n} y_i^k \times q_i \leq Q \quad \forall k = 1, ..., \upsilon \tag{5}$$

$$\sum_{k=1}^{\upsilon} y_i^k = 1 \quad \forall i = 1, ..., n \tag{6}$$

$$\sum_{k=1}^{\upsilon} y_0^k = \upsilon \tag{7}$$

$$t_i + w_i + s_i + t_{ij} = t_j \quad \forall i, j = 0, 1, .., n \quad i \neq j \tag{8}$$

$$e_j \leq t_j \leq l_j \quad \forall j = 0, 1, ..., n \tag{9}$$

$$w_i = max\{e_i - t_i, 0\} \quad \forall j = 0, 1, ..., n \tag{10}$$

Constraint (5) denotes that the quantity of goods that each vehicle carries could not exceed the capacity Q. Constraint (6) stands for that each customer can only be served by one vehicle. Constraint (7) represents that all the routes start from the depot. Formulas (8)–(10) define the time window constraint, where t_i is the time when the vehicle arrives at node i ; w_i is the waiting time of a vehicle at the customer location until the time e_i ; s_i is the service time; and t_{ij} is the travel time between nodes i and j . The CVRP is NP-hard [46], and in addition, the VRPTW has more constraints and is, therefore, harder to solve [43]. In [39], Savelsberg proved that to find a feasible solution for the VRPTW is NP-hard. Researches on the VRPTW have great theoretical and practical values.

3.1 Tabu Search Optimization

Tabu search algorithm was proposed by Glover [19]. In 1986, he pointed out the controlled randomization in SA (Simulated annealing) to escape from local optima and proposed a deterministic algorithm [18]. In a parallel work, a similar approach named "steepest ascent/mildest descent" has been proposed by [22]. In the 1990s, the tabu search algorithm became very popular in solving optimization problems in an approximate manner. Nowadays, it is one of the most widespread S-metaheuristics. The use of memory, which stores information related to the search process, represents the particular feature of tabu search.

TS behaves like a steepest LS (Local search) algorithm, but it accepts non improving solutions to escape from local optima when all neighbors are non improving solutions. Usually, the whole neighborhood is explored in a deterministic manner, whereas in SA a random neighbor is selected. As in local search, when a better neighbor is found, it replaces the current solution. When a local optima is reached, the search carries on by selecting a candidate worse than the current solution. The best solution in the neighborhood is selected as the new current solution even if it is not improving the current solution. Tabu search may be viewed as a dynamic transformation of the neighborhood. This policy may generate cycles; that is, previous visited solutions could be selected again.

To avoid cycles, TS discards the neighbors that have been previously visited. It memorizes the recent search trajectory. Tabu search manages a memory of the solutions or moves recently applied, which is called the tabu list. This tabu list constitutes the short-term memory. At each iteration of TS, the short-term memory is updated. Storing all visited solutions is time and space consuming. Indeed, we have to check at each iteration if a generated solution does not belong to the list of all visited solutions. The tabu list usually contains a constant number of tabu moves. Usually, the attributes of the moves are stored in the tabu list.

By introducing the concept of solution features or move features in the tabu list, one may lose some information about the search memory. We can reject solutions that have not yet been generated. If a move is "good," but it is tabu, do we still reject it? The tabu list may be too restrictive; a non generated solution may be forbidden. Yet for some conditions, called aspiration criteria, tabu solutions may be accepted. The admissible neighbor solutions are those that are non tabu or hold the aspiration criteria.

The steps of TS [23]are

- Step 1: Choose an initial solution i in S. Set $i^* = i$ and k=0.
- Step 2: Set $k = k + 1$ and generate a subset V^* of solution in $N(i, k)$ such that either one of the Tabu conditions is violated or at least one of the aspiration conditions holds.
- Step 3: Choose a best j in i^* and set $i = j$.

- Step 4: If $f(i) < f(i^*)$ then set $i^* = i$.
- Step 5: Update Tabu and aspiration conditions.
- Step 6: If a stopping condition is met then stop. Else go to Step 2.

4 Three Strategy Tabu Search for Vehicle Routing Problem with Time Window

4.1 Initial Solution

In order to obtain an initial solution, we have developed a procedure, that constructs a feasible solution to the VRPTW by finding a feasible assignment of customers to vehicles. The customers are assigned randomly to selected route. The proposed procedure can be summarized as follows :
Step 1: make a number of routes equal to number of vehicles (all vehicle have same capacity, Nv and n represent number of vehicles and number of nodes sequentially).
Step 2: for $i = 1$ to $n - 1$
select random node, assign it to current route. If capacity of current route match vehicle capacity assigned to new route.
Step 3: for $i = 1$ to Nv
calculate length of each route.
Step 4: for $i = 1$ to Nv
select first node in route and swap it by all next node, in each swap find length of route. Choose shortest length of route.

4.2 Formulation of the Fitness Function

A fitness function is a particular type of objective function that quantifies the optimality of solution. The shorter the route, the higher is the fitness value. Hence, we formulated the fitness function as follows :

$$F = TD * \prod_{i=1}^{NV} \max[1, (RQ/Q)] \qquad (11)$$

Where TD is total length of route, RQ is all demand of nodes in route .

4.3 TSTS-VRPTW Algorithm

4.3.1 Move Type :

- choose two routs R_1 & R_2 randomly.
- find the smallest distance between two nodes $n_1 \in R_1$ and $n_2 \in R_2$.
- relocates the node n_1 before position of node n_2, that is R_1 = (0 1 $\underline{5}$ 9 12)$\rightarrow$ R_2 = (0 $\underline{2}$ 10 6 14) , R_1 = (0 1 9 12)$\rightarrow$ R_2 = (0 $\underline{5}$ $\underline{2}$ 10 6 14),the algorithm shown in Fig. 1.

4.3.2 Exchange Type :

- choose two routs R_1 & R_2 randomly.
- find the smallest distance between two nodes $n_1 \in R_1$ and $n_2 \in R_2$.
- swap nodes n_1 and n_2, that is R_1 = (0 1 $\underline{5}$ 9 12)$\rightarrow$ R_2 = (0 $\underline{2}$ 10 6 14), R_1 = (0 1 $\underline{2}$ 9 12)$\rightarrow$ R_2 = (0 $\underline{5}$ 10 6 14), the algorithm shown in Fig. 2.

4.3.3 Swap Type :

- choose three nodes n_1, n_2 & $n_3 \in R_1$ randomly.
- find all probabilities of swap these three nodes, that is R_1 = (0 1 $\underline{5}$ $\underline{14}$ $\underline{9}$ 12), R_1 = (0 1 $\underline{5}$ $\underline{9}$ $\underline{14}$ 12).....,R_1 = (0 1 $\underline{9}$ $\underline{14}$ $\underline{5}$ 12).
- calculate the length of each probability of route R_1.
- choose the smallest distance of route R_1, the algorithm shown in Fig. 3.

4.4 Stopping Criteria

Since the algorithm is open-ended, so the stopping criteria are always needed. It may run forever as the optimum is unknown. For this problem, algorithm will stop searching right after it complete diversify on the repetition and we also confining the maximum iteration up to 1000 to prevent from wasting time. This is because, in case if there are too many repetitions, the algorithm may run forever until it completed diversify but in the same time the solution are not improve after such number of iteration. Since the optimum is unknown, so the maximum number of iteration is needed.

5 Numerical Results

This section presents the numerical experimental results for the TSTSTW. The experiments are done on Solomon$'s$ benchmarks that consist of six different datasets [43]. Datasets C have clustered customers, and the time windows are generated by the known vehicle routes for these clustered customers. Datasets R have randomly generated customers that are distributed uniformly over a square. Datasets RC have mixed customers of datasets C and R. Datasets $C1$, $R1$, and $RC1$ have narrow time windows and small delivery capacity, which allow few customers to be served by one vehicle. In contrast, datasets $C2$, $R2$, and $RC2$ have large time windows and capacity, which allow more customers to be served by one vehicle.

5.1 Experimental Settings

The proposed solution method introduce three parameters, namely Pm, Pe and Ps. This parameters define the ratio of each function to take place in scope of solution.The three parameters values changes from 0 to 1, where Pm, Pe and Ps represent move, exchange and swap function. In solution we set three parameter

equal to .3.
Our algorithm is coded in Matlab and runs on a machine with Intel Pentium 4 CPU, 2.45 GHz/6 GB of RAM. In the remaining of this section, the experimental results are presented. The results obtained by the TSTS-VRPTW are compared with the best-so-far results that are reported at Solomon$'s$ web page: `http://web.cba.neu.edu/~msolomon/problems.htm`. These best known results are derived from 24 different algorithms that include exact, heuristic, and metaheuristic algorithms. The algorithms consist of KDMSS [28], RT [36], TBGGP [45], CLM [9], S97 [41], LLH [33], IKMUY [26], BVH [3], DLP [11], CC [10], GTA [13], BB [4], S98 [42], CR [8], L [32], HG [25], SSSD [40], H [24], KLM [28], RGP [37], M [35], IV [27], C [7] and S-PSO-VRPTW [21].

5.2 Results and Comparisons

Tables 1,2 and 3 show the result comparisons, in which "NV" represents the NV routes and "TD" represents the total TD. "Author" denotes who have proposed the best results. The best result are presented. The best result are marked in bold indicate that, for the instance, the TSTS-VRPTW provides new best result. In other cases, if the NV routes that we obtained is equal to that of the best known, which means that the TSTS-VRPTW produces the best result, the best result is presented in italic.
1) 25-Customer Instances: As shown in Table I, for 25-customer instances, the TSTS-VRPTW produces 29 out of 56 new best results, among which 12 come from the R1-type datasets, nine come from the C1-type datasets, and eight come from the RC1-type datasets. Compared with the previous algorithms that provided the best known, the TSTS-VRPTW has better performance in exploring irregular search space (datasets R and RC). Moreover, the proposed algorithm is very good at reducing the required NV. For the instances with large time windows and capacities, the required NV is very small. Besides the 29 new best known, the TSTS-VRPTW obtains another 14 solutions that are less than current best known solutions. Therefore, the proposed algorithm performs well in optimizing 43 out of the 56 25-customer instances.
In Table 1, it can be observed that the TSTS-VRPTW comprehensively outperforms S-PSO-VRPTW and The comparisons in Table I and the aforementioned investigation show that the proposed TSTS-VRPTW presents a much better performance than S-PSO-VRPTW.
2) 50-Customer Instances: As shown in Table 2, for 50-customer instances, the TSTS-VRPTW produces 31 out of 56 new best results, among which eight come from the RC1-type datasets, 12 come from the R1-type datasets, nine come from the C1-type datasets, and two come from the RC2-type datasets. Besides, for 17 of the remaining 25 instances, the TSTS-VRPTW produces best results compared with the algorithms in the literature. Therefore, the TSTS-VRPTW is a very powerful technique to solve almost all the Solomon$'s$ 50-customer instances.
3) 100-Customer Instances: In Table 3, results of the TSTS-VRPTW that optimizes Solomon$'s$ 100-customer in-stances are presented. Compared with the best known results that are summarized on Solomon$'s$ website, which is gained by 24 different algorithms, the TSTS-VRPTW obtains 38 best solutions, among which two come from R2-type datasets, nine come from C1-type datasets, eight come from RC1-type datasets, seven come from RC2-type datasets, 12 come from R1-type datasets.

6 Conclusion

In this paper, we have proposed a set-based TS to solve the VRPTW. In the TSTS-VRPTW, the search space is a universal set of arcs in the complete graph of the VRPTW. Each particle's position that represents a set of delivery routes for a fleet of vehicles has been defined as a subset of arcs and built constructively. In this representation, the characteristics of the VRPTW have been fully embodied. During the construction of particle position, the constraints of the VRPTW have been taken into account, and a time-oriented NNH has bee applied. Furthermore, a novel decision making method has been proposed to handle the primary and secondary objectives of the VRPTW. This method is not only appropriate for the TSTS-VRPTW, but also potentially useful in other approaches to solve the VRPTW.
The proposed TSTS-VRPTW is one of the few TS algorithm that was tested on all Solomon$'s$ benchmarks. Experimental results illustrate the effectiveness and efficiency of the proposed algorithm, for we provide better results than the existing best known results for Solomon's instances. Moreover, the TSTS-VRPTW obtains promising results on almost all the tested instances, which shows the robustness of the algorithm.

Table 1. RESULT COMPARISONS FOR SOLOMON'*S* 25-CUSTOMER INSTANCES

	TSTS-VRPTW		**Best known**			**S-PSO-VRPTW**			
problem						**Best**		**Mean**	
	NV	**TD**	**Authors**	**NV**	**TD**	**NV**	**TD**	**NV**	**TD**
R101	**3**	**302**	KDMSS	8	617.1	8	618.33	8	618.33
R102	**3**	**298.92**	KDMSS	7	547.1	7	548.11	7	555.14
R103	**3**	**300.22**	KDMSS	5	454.6	4	473.39	4	475.21
R104	**3**	**304.33**	KDMSS	4	416.9	4	418.3	4	422.47
R105	**3**	**317.1**	KDMSS	6	530.5	5	556.72	5	556.3
R106	**3**	**309.69**	KDMSS	3	465.4	5	466.48	5	473.22
R107	**3**	**300.71**	KDMSS	4	424.3	4	425.27	4	435.44
R108	**3**	**310.318**	KDMSS	4	397.3	4	405.39	4	416.14
R109	**3**	**308.26**	KDMSS	5	441.3	4	460.52	4	467.25
R110	**3**	**310.56**	KDMSS	4	444.1	4	445.8	4	446.45
R111	**3**	**303.85**	KDMSS	5	428.8	4	429.7	4	438.66
R112	**3**	**298.20**	KDMSS	4	393	4	394.1	4	400.93
R201	3	*299.64*	CR+KLM	4	463.3	2	523.66	2	532.23
R202	3	*307.00*	CR+KLM	4	410.5	2	455.53	2	466.97
R203	3	*304.92*	CR+KLM	3	391.4	2	408.89	2	421.86
R204	3	314.04	IV+C	2	355	1	389.91	1	398.32
R205	3	*320.86*	CR+KLM	3	393	1	501.83	1	507.75
R206	3	*316.70*	CR+KLM	3	374.4	1	413.21	1	431.91
R207	3	*301.55*	KLM	3	361.6	1	402.28	1	409.43
R208	3	320.23	IV+C	1	328.2	1	329.33	1	335.77
R209	3	309.43	KIM	2	370.7	1	438.24	1	444.62
R210	3	*307.43*	CR+KLM	3	404.6	1	513.98	1	526.85
R211	3	310.75	KLM	2	350.9	1	361.69	1	369.86
C101	**3**	**127.28**	KDMSS	3	191.3	3	191.81	3	191.81
C102	**3**	**121.57**	KDMSS	3	190.3	3	190.74	3	192.29
C103	**3**	**120.28**	KDMSS	3	190.3	3	190.74	3	194.12
C104	**3**	**136.69**	KDMSS	3	186.9	3	187.45	3	192.45
C105	**3**	**127.28**	KDMSS	3	191.3	3	191.81	3	191.81
C106	**3**	**120.34**	KDMSS	3	191.3	3	191.81	3	191.81
C107	**3**	**120.28**	KDMSS	3	191.3	3	191.81	3	196.09
C108	**3**	**120.28**	KDMSS	3	191.3	3	191.81	3	192.22
C109	**3**	**120.08**	KDMSS	3	191.3	3	191.81	3	191.84
C201	3	116.7568	CR+L	2	214.7	2	215.54	2	216.98
C202	3	173.95	CR+L	2	214.7	1	223.31	1	230.4
C203	3	161.53	CR+L	2	214.7	1	223.31	1	229.89
C204	3	177.91	CR+KLM	2	213.1	1	221.28	1	230.17
C205	3	163.38	CR+L	2	214.7	1	297.45	1	297.55
C206	3	184.27	CR+L	2	214.7	1	285.39	1	290.29
C207	3	176.92	CR+L	2	214.5	1	274.78	1	276.26
C208	3	160.78	CR+L	2	214.5	1	229.84	1	230.12
RC101	**3**	**181.64**	KDMSS	4	461.1	4	462.16	4	462.39
RC102	**3**	**186.61**	KDMSS	3	351.8	3	352.74	3	353.12
RC103	**3**	**180.71**	KDMSS	3	332.8	3	333.92	3	334.3
RC104	**3**	**181.47**	KDMSS	3	306.6	3	307.14	3	307.31
RC105	**3**	**184.25**	KDMSS	4	411.3	4	412.38	4	413.73
RC106	**3**	**181.47**	KDMSS	3	345.5	3	346.51	3	348.64
RC107	**3**	**191.25**	KDMSS	3	298.3	3	298.95	3	303.42
RC108	**3**	**185.70**	KDMSS	3	294.5	3	294.99	3	298.2
RC201	3	*182.62*	CR+L	3	360.2	2	432.3	2	438.7
RC202	3	*216.27*	CR+KLM	3	338	2	376.12	2	382.94
RC203	3	*183.39*	IV+C	3	326.9	1	432.55	1	435.29
RC204	3	*182.11*	C	3	299.7	1	327.33	1	331.4
RC205	3	*188.48*	L+KLM	3	338	2	386.15	2	407.6
RC206	3	*185.1*	KIM	3	324	1	482.02	1	484.79
RC207	3	*183.86*	KLM	3	298.3	1	478.97	1	482.22
RC208	3	*180.70*	C	2	269.1	1	309.85	1	313.17

Table 2. RESULT COMPARISONS FOR SOLOMON'*S* 50-CUSTOMER INSTANCES

	TSTS-VRPTW		**Best known**			**S-PSO-VRPTW**			
problem						**Best**		**Mean**	
	NV	**TD**	**Author**	**NV**	**TD**	**NV**	**TD**	**NV**	**TD**
R101	**4**	**547.79**	KDMSS	12	1044	11	1100.72	11.8	1060.11
R102	**4**	**536.32**	KDMSS	11	909	10	923.71	10	927.13
R103	**4**	**543.89**	KDMSS	9	772.9	8	790.17	8	808.99
R104	**4**	**571.32**	KDMSS	6	625.4	6	631.58	6	638.83
R105	**4**	**527.25**	KDMSS	9	899.3	8	983.49	9	933.67
R106	**4**	**541.59**	KDMSS	5	793	7	865.93	7.5	851.17
R107	**4**	**503.25**	KDMSS	7	711.1	6	737.1	6.1	751.77
R108	**4**	**549.60**	CR+KLM	6	617.7	6	624.29	6	632.73
R109	**4**	**507.32**	KDMSS	8	786.8	7	801.97	7	815.44
R110	**4**	**510.29**	KDMSS	7	697	7	710.4	7	731.79
R111	**4**	**509.47**	CR+KLM	7	707.2	6	756.35	6.9	722.55
R112	**4**	**521.45**	CR+KLM	6	630.2	6	638.49	6	645.75
R201	3	*503.69*	CR+KLM	6	791.9	2	953.29	2.6	911.86
R202	3	*503.23*	CR+KLM	5	698.5	2	815.21	2	833.9
R203	3	*484.98*	IV+C	5	605.3	2	668.36	2	688.84
R204	3	*526.31*	IV	2	506.6	2	518.37	2	525.78
R205	3	*500.24*	IV+C	4	690.1	2	756.38	2	773.24
R206	3	*494.75*	IV+C	4	632.4	2	661.55	2	676.25
R207	3	554.35				2	593.95	2	611.04
R208	3	524.39				2	508.41	2	518.83
R209	3	*529.48*	IV+C	4	600.6	2	658.28	2	671.54
R210	3	*544.17*	IV+C	4	6454	2	670.99	2	682.7
R211	3	*497.98*	IV+DLP	3	535.3	2	62.74	2	576.03
C101	**5**	**270.58**	KDMSS	5	362.4	5	363.25	5	363.25
C102	**5**	**258.17**	KDMSS	5	361.4	5	362.17	5	373.68
C103	**5**	**248.88**	KDMSS	5	361.4	3	362.17	5	368.85
C104	**5**	**254.37**	KDMSS	5	358	5	358.88	5	362.88
C105	**5**	**261.34**	KDMSS	5	362.4	5	363.25	5	363.25
C106	**5**	**252.04**	KDMSS	5	362.4	5	363.25	5	363.25
C107	**5**	**252.04**	KDMSS	5	362.4	5	363.25	5	374.87
C108	**5**	**257.83**	KDMSS	5	362.4	5	363.25	5	366.72
C109	**5**	**251.72**	KDMSS	5	362.4	5	363.25	5	365.17
C201	3	*250.81*	CR+L	3	360.2	2	441.96	2	444.96
C202	3	380	CR+KLM	3	360.2	2	403.81	2	407.25
C203	3	367	CR+KLM	3	359.8	2	402.52	2	406.11
C204	3	335.72	KLM	2	350.1	2	356.77	2	364.02
C205	3	*341.89*	CR+KLM	3	359.8	2	429.12	2	445.45
C206	3	*333.28*	CR+KLM	3	359.8	2	412.5	2	437.66
C207	3	*320.84*	CR+KLM	3	359.4	2	426.13	2	444.62
C208	3	326.37	CR+KLM	2	350.5	2	352.29	2	353.56
RC101	**5**	**518.93**	KDMSS	8	944	8	945.58	8	946.66
RC102	**5**	**498.22**	KDMSS	7	822.5	7	823.97	7	827.89
RC103	**5**	**442.52**	KDMSS	6	710.9	6	712.91	6	714.67
RC104	**5**	**411.16**	KDMSS	5	545.8	5	546.51	5	547.95
RC105	**5**	**562.57**	KDMSS	8	855.3	8	856.97	8	860.5
RC106	**5**	**434.36**	KDMSS	6	723.2	6	724.65	6	729.62
RC107	**5**	**402.28**	KDMSS	6	642.7	6	645.7	6	653.54
RC108	**5**	**335.87**	KDMSS	6	598.1	6	599.7	6	613.41
RC201	**3**	**380.54**	L+KLM	5	684.8	3	838.76	3	855.61
RC202	3	*359.45*	IV+C	5	613.6	2	867.26	2	889.5
RC203	3	*364.52*	IV+C	4	555.3	2	674.44	2	680.46
RC204	3	*401.57*	DLP	3	444.2	2	479.22	2	485.41
RC205	**3**	**359.02**	IV+C	5	630.2	3	765.02	3	775.99
RC206	3	*381.7*	IV+C	5	610	2	755.13	2	764.65
RC207	3	*424.97*	C	4	558.6	2	655.81	2	677.21
RC208	3	389.49				2	498.79	2	518.8

Table 3. RESULT COMPARISONS FOR SOLOMON'*S* 100-CUSTOMER INSTANCES

	TSTS-VRPTW		**Best known**			**S-PSO-VRPTW**			
Problem	**NV**	**TD**	**authors**	**NV**	**TD**	**Best**		**Mean**	
						NV	**TD**	**NV**	**TD**
R101	**8**	**877.05**	H	19	1645.79	19	1652.00	19	1657.89
R102	**8**	**875.09**	RT	17	1486.12	17	1500.81	17.8	1506.63
R103	**8**	**821.38**	LLH	13	1292.68	14	1242.65	14	1268.59
R104	**8**	**860.88**	M	9	1007.24	10	1042.22	10.5	1097.77
R105	**8**	**851.52**	RT	14	1377.11	14	1385.08	14.3	1405.28
R106	**8**	**903.73**	M	12	1251.98	12	1294.87	12.9	1287.03
R107	**8**	**845.51**	S97	10	1104.66	11	1123.98	11	1160.02
R108	**8**	**776.09**	BB	9	960.88	10	1011.68	10.1	1066.29
R109	**8**	**890.95**	HG	11	1194.73	12	1211.63	12.4	1252.79
R110	**8**	**862.74**	M	10	1118.59	11	1190.36	12	1190.38
R111	**8**	**866.58**	RGP	10	1096.72	11	1102.99	11.3	1163.27
R112	**8**	**818.96**	GTA	9	982.14	10	1029.12	10.8	1103.07
R201	**3**	**964.81**	HG	4	1252.37	4	1274.97	4	1298.28
R202	**3**	**967.19**	RGP	3	1191.70	3	1247.03	3.5	1259.84
R203	3	1126.20	M	3	939.54	3	1052.71	3	1100.80
R204	3	1021.90	BVH	2	825.52	3	844.16	3	928.04
R205	3	1070.00	RGP	3	994.42	3	1061.46	3	1135.70
R206	3	1077.00	SSSD	3	906.14	3	1016.35	3	1065.61
R207	3	1052.30	BVH	2	893.33	3	946.78	3	1036.86
R208	3	1036.70	M	2	726.75	2	834.72	2.6	880.31
R209	3	970.64	H	3	909.16	3	1003.19	3	1076.95
R210	3	1072.60	M	3	939.34	3	1040.54	3	1090.36
R211	3	909.07	BVH	2	892.71	3	861.32	3	938.15
C101	**10**	**728.70**	RT	10	828.94	10	828.94	10	828.94
C102	**10**	**750.30**	RT	10	828.94	10	829.71	10	850.84
C103	**10**	**696.64**	RT	10	828.06	10	851.37	10	886.34
C104	**10**	**651.48**	RT	10	824.78	10	868.52	10	958.49
C105	**10**	**689.20**	RT	10	828.94	10	828.94	10	830.44
C106	**10**	**716.50**	RT	10	828.94	10	828.94	10	828.94
C107	**10**	**774.11**	RT	10	828.94	10	828.94	10	865.30
C108	**10**	**721.04**	RT	10	828.94	10	828.94	10	829.08
C109	**10**	**761.74**	RI	10	828.94	10	828.94	10	829.58
C201	3	819.66	RI	3	591.56	3	591.56	3	621.78
C202	3	1017.60	RI	3	591.56	3	591.56	3	616.38
C203	3	757.36	RI	3	591.17	3	591.17	3	605.84
C204	3	852.70	RI	3	590.60	3	615.43	3.2	678.08
C205	3	906.19	RI	3	588.88	3	588.88	3	600.31
C206	3	811.08	RI	3	588.49	3	588.88	3	592.95
C207	3	844.33	RI	3	588.29	3	591.35	3	596.54
C208	3	913.05	RI	3	588.32	3	588.49	3	591.59
RC101	**9**	**1111.00**	TBGGP	14	1696.94	15	1641.20	15	1668.88
RC102	**9**	**1097.10**	TBGGP	12	1554.75	13	1510.95	13.8	1506.81
RC103	**9**	**895.30**	S98	11	1261.67	11	1294.74	11.7	1305.92
RC104	**9**	**1059.80**	CLM	10	1135.48	10	1190.55	10.8	1219.32
RC105	**9**	**988.04**	BB	13	1629.44	14	1603.71	14.9	1581.80
RC106	**9**	**1029.00**	BB	11	1424.73	12	1410.93	12.8	1416.16
RC107	**9**	**1094.50**	S97	11	1230.48	11	1249.80	11.3	1277.97
RC108	**9**	**1043.50**	TBGGP	10	1139.82	11	1181.87	11	1230.92
RC201	**3**	**1017.80**	M	4	1406.91	4	1423.52	4	1472.71
RC202	**3**	**889.79**	CC	3	1367.09	4	1193.59	4	1286.52
RC203	**3**	**998.54**	CC	3	1049.62	3	1123.42	3	1222.81
RC204	3	1055.00	M	3	798.41	3	894.12	3	964.50
RC205	**3**	**1018.30**	M	4	1297.19	4	1321.43	4	1382.20
RC206	**3**	**913.97**	H	3	1146.32	3	1307.90	4	1225.40
RC207	**3**	**926.98**	BVH	3	1061.14	3	1130.37	3.7	1192.55
RC208	**3**	**1019.30**	IKMUY	3	828.14	3	958.24	3	1084.93

Appendix
Code for Three Function Move, Exchange and Swap

```
function SolM = nodeMove(Sol,Nv,tij,li)
-- Sol Initial solution.
SolM = Sol;
rCheck =0;
while rCheck == 0
    RouteRand = randperm(Nv);
    R1 = RouteRand(1);
    R2 = RouteRand(2);

    if (length(Sol{R1}) ~= 1)&&(tij(R1))<(li(R1))
        rCheck = 1;
    end
end
R1size = length(Sol{R1});
R2size = length(Sol{R2});
global DM;
-- DM Distance matrix between each node and other.
node1=Sol{R1}(1,1);
node2=Sol{R2}(1,1);
mind=DM(node1,node2);
minn1=1;
minn2=1;
for i=1:R1size
    node1=Sol{R1}(1,i);
    for j=1:R2size
        node2=Sol{R2}(1,j);
        if(DM(node1,node2)< mind) &&(tij(R1))<(li(R1))
            mind=DM(node1,node2);
            minn1=i;
            minn2=j;
        end
    end
end
-- Move the node.
node_cut=minn1;
node_move=minn2;
SolM{R1} = [];
SolM{R2} = [];
index1 = 1;
while index1 <= R1size
    if index1 ~= node_cut
        SolM{R1} = [SolM{R1} Sol{R1}(1,index1)];
    end
    index1 = index1 + 1;
end
index2 = 1;
for k = 1: R2size+1
    if k == node_move
        SolM{R2} = [SolM{R2} Sol{R1}(1,node_cut)];
    else
        SolM{R2} = [SolM{R2} Sol{R2}(1,index2)];
        index2 = index2 + 1;
    end
end
```

Figure 1. Algorithm 1: Move type

```
function SolE = nodeEx(Sol,Nv,tij,li)
SolE = Sol;
rcheck = 0;
RouteRand = randperm(Nv);
R1 = RouteRand(1);
R2 = RouteRand(2);
R1size = length(Sol{R1});
R2size = length(Sol{R2});
global DM;
node1 = Sol{R1}(1,1);
node2 = Sol{R2}(1,1);
mind = DM(node1,node2);
minn2 = 1;
for i = 1:R1size
   node1 = Sol{R1}(1,i);
   for j = 1:R2size
        node2 = Sol{R2}(1,j);
   if(DM(node1,node2)< mind)&&((tij(R1))<(li
            mind = DM(node1,node2);
            minn2 = j;
        end
    end
end
node_E2 = minn2;
node1 = Sol{R1}(1,1);
node2 = Sol{R2}(1,1);
mind = DM(node1,node2);
minn1 = 1;
for i = 1:R1size
   node1 = Sol{R1}(1,i);
   for j = 1:R2size
        node2 = Sol{R2}(1,j);
   if(DM(node2,node1)< mind)&&((tij(R1))<(li
            mind = DM(node1,node2);
            minn1 = i;
        end
    end
end
node_E1 = minn1;
var = SolE{R1}(1,node_E1);
SolE{R1}(1,node_E1) = SolE{R2}(1,node_E2);
SolE{R2}(1,node_E2) = var;
```

Figure 2. Algorithm 2: Exchange type

```
function SolW = nodeSwap(Sol,Nv,tij,li)
RouteRand = randperm(Nv);
R1size=0;
i=1;
while(R1size<3)
R1 = RouteRand(i);
i=i+1;
R1size = length(Sol{R1});
end
NodeRand = randperm(R1size);
Node = zeros(1,3);
index1 = NodeRand(1);
index2 = NodeRand(2);
index3 = NodeRand(3);
Node(1,1) = Sol{R1}(1,index1);
Node(1,2) = Sol{R1}(1,index2);
Node(1,3) = Sol{R1}(1,index3);

Svalue = GetpathLenght(Sol,i);
-- Find total distance of route
SolW = Sol;
for i = 1:3
    SolM = Sol;
    index1 = NodeRand(i);
    SolM{R1}(1,index1) = Node(1,1);
    for j = 1:3
        if(i==j)
            continue;
        end
        index2 = NodeRand(j);
        SolM{R1}(1,index2)=Node(1,2);
        for K=1:3
             if(K==j || K==i)
                continue;
             end
             index3 = NodeRand(K);
             SolM{R1}(1,index3)=Node(1,3);
             L=GetpathLenght(SolM,i);
             if(L<Svalue)&&(tij(index3))<(li(index3))
                 SolW = SolM;
             end
        end
    end
end
function Length = GetpathLenght(Sol,R)
global DM;
Rsize = length(Sol{R});
Length = 0;
for i = 1:Rsize-1
    node = Sol{R}(i);
    next_node = Sol{R}(i+1);
    Length = Length + DM(node,next_node);
end
```

Figure 3. Algorithm 3: Swap type

REFERENCES

[1] N. Ando and E. Taniguchi. Travel time reliability in vehicle routing and scheduling with time windows. *Networks and Spatial Economics*, 6:293–311, 2006.

[2] R. Baldacci, P. Toth, and D. Vigo. Exact algorithms for routing problems under vehicle capacity constraints. *Annals of Operation Research*, 175(1):213–245, 2010.

[3] R. Bent and P.V. Hentenryck. A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Science*, 38(4):515–530, 2004.

[4] J. Berger and M. Barkaoui. A parallel hybrid genetic algorithm for the vehicle routing problem with time windows. *Computers and Operations Research*, 31(12):2037–2053, 2004.

[5] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part‖: Metaheuristics. *Transportation Science*, 39(1):119–139, 2005.

[6] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part|: Route construction and local search algorthims. *Transportation Science*, 39(1):104–118, 2005.

[7] A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computer Operation Research*, 33(10):2972–2990, 2006.

[8] W. Cook and J.L. Rich. A parallel cutting plane algorithm for the vehicle routing problem with time windows. Working paper, Department of Computer and Applied Mathematics, Rice University, Houston, Taxas, United States of America, 1999.

[9] J. Cordeau, G. Laporte, and A. Mercier. A unified tabu search heuristic for vehicle routing with time windows. *journal of the Operation Research Society*, 52:928–936, 2001.

[10] Z.J. Czech and P. Czarnas. A parallel simulated annealing for the vehicle routing problem with time windows. In *10th Euromicro Workshop Parallel, Distributed Network-Based Process*, pages 376–383, 2002.

[11] E. Danna and C.L. Pape. *Column Generation*, chapter Accelerating branch-and-price with local search: A case study on the vehicle routing problem with time windows, pages 99–130. Handbook of Constraint Programming. Kluwer Academic Publishers, 2005.

[12] J. Desrosiers, Y. Dumas, MM. Solomon, and F. Soumis. *Network Routing*, volume 8, chapter Time Constrained Routing and Scheduling, pages 35–139. Handbooks in Operations Research and Management Science, Amsterdam: North-Holland, 1995.

[13] L.M. Gambardella, *É*. Taillard, and G. Agazzi. *New Ideas in Optimization*, chapter MACS-VRPTW: A Multiple ant Colony System for Vehicle Routing Problems with Time Windows, pages 63–76. McGraw-Hill, New York, 1999.

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

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

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

[17] M. Gendreau, G. Laporte, and R. Sèguin. Stochastic vehicle routing. *European Journal of Operation Research*, 88:3–12, 1996.

[18] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operation Research*, 13(5):533—-549, 1986.

[19] F. Glover. Tabu search, part|. *ORSA Journal on Computing*, 1(3):190–206, 1989.

[20] F. Glover. Tabu search, part‖. *ORSA Journal on Computing*, 2(1):4–32, 1990.

[21] Y.J. Gong, J. Zhang, O. Liu, R.Z. Huang, H.S.H Chung, and Y.H. Shi. Optimizing the vehicle routing problem with time windows: A discrete particle swarm optimization approach. *Ieee Transactions on Systems, Man, and Cybernetcs—part C: Applications and Reviews*, 42(2):254–267, March 2012.

[22] P. Hansen. The steepest ascent mildest descent heuristic for combinatorial programming. Capri, Italy, 1986. Numerical Methods in Combinatorial Optimization.

[23] A. Hertz, E. Taillard, and D. Werra. A tutorial on tabu search. Technical report, Colorado State University, http://www.cs.colostate.edu/whitley/ČS640/hertz92tutorial.pdf, April 2005.

[24] J. Homberger. *Verteilt-parallele Metaheuristiken zur Tourenplanung*. Wiesbaden: Gaber, Germany, 2000.

[25] J. Homberger and H. Gehring. Two evolutionary metaheuristics for the vehicle routing problem with time windows. *Information System of Operation Research*, 37:297–318, 1999.

[26] T. Ibaraki, M. Kubo, T. Masuda, T. Uno, and M. Yagiura. Effective local search algorithms for the vehicle routing problem with general time windows. Working paper, Department of Applied Mathematics and Physics, Kyoto University, Japan, 2001.

[27] S. Irnich and D. Villeneuve. he shortest path problem with k-cycle elim-ination ($k \geq 3$): Improving a branch-and-price algorithm for the vrptw. *Information Journal of Computer*, 18(3):391–496, 2006.

[28] B. Kallehauge, J. Larsen, and O.B.G. Madsen. Lagrangean duality applied on vehicle routing with time windows - experimental results. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2001.

[29] A. Kenyon and D. Morton. Stochastic vehicle routing with random travel times. *Transportation Science*, 37(1):69–82, 2003.

[30] G. Laporte. The vehicle routing problem: an overview of exact and approximate algorithms. *European journal of the Operation Research*, 59:345–358, 1992.

[31] G. Laporte. What you should know about the vehicle routing problem. *Naval Research Logistics*, 54:811–819, 2007.

[32] J. Larsen. *Parallelization of the vehicle routing problem with time windows*. Ph.d., Department of Mathematics Modelling, Technical University of Denmark, Lyngby, Denmark, 1999.

[33] H. Li, A. Lim, and J. Hung. Local search with annealing-like restarts to solve the vrptw. Working paper, Department of Computer Science, National University, Singapore, 2001.

[34] X. Li, P. Tian, and S. Leung. Vehicle routing problems with time windows and stochastic travel and service times: Models and algorithm. *International Journal of Production Economics*, 125:137–145, 2010.

[35] D. Mester. An evolutionary strategies algorithm for large scale vehicle routing problem with capacitate and time windows restrictions. Working paper, The Institute of Evolution, Haifa, Israel, 2002.

[36] Y. Rochat and E.D. Taillard. Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1(1):147–167, 1995.

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

[38] RA. Russell and TL. Urban. Vehicle routing with soft time windows and erlang travel times. *Journal of the Operational Research Society*, 58:1220–1228, 2008.

[39] M. Savelsbergh. *private communication.* 1984.

[40] G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt, and G. Dueck. Record breaking optimization results using the ruin and recreate principle. *journal of Computer Physic*, 159(2):139–171, 2000.

[41] P. Shaw. A new local search algorithm providing high quality solutions to vehicle routing problems. Working paper, Department of Computer Science, Strathclyde University, Glasgow, Scotland, 1997.

[42] P. Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In Maher M. and Puget J.F., editors, *Principle and Practice of Constraint Programming*, pages 417–431, London,UK, 1998. Springer-Verlag.

[43] MM. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987.

[44] W. Stewart and B. Golden. Stochastic vehicle routing: A comprehensive approach. *European Journal of Operational Research*, 14:371–385, 1983.

[45] E. Taillard, P. Badeau, M. Gendreau, F. Guertin, and J. Potvin. A tabu search heuristic for the vehicle routing problems with soft time windows. *Computers and Operations Research*, 31(2):170–186, 1997.

[46] P. Toth and D. Vigo. *The Vehicle Routing Problem.* SIAM, Philadelphia, PA, 2001.

[47] T. Van Woensel, L. Kerbache, H. Peremans, and N. Vandaele. Routing with dynamic travel times: A queueing approach. *European Journal of Operational Research*, 186(3):990—-1007, 2008.