Journal of the Eastern Asia Society for Transportation Studies, Vol. 6, pp. 2822 - 2836, 2005

A HYBRID ANT COLONY SYSTEM FOR VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

Chia-Ho CHEN
Ph.D. Student
Department of Industrial Engineering and Management
Yuan Ze University
135 Yuan-Tung Road, Chungli,
Taiwan 320, R.O.C.
Fax: +886-3-4638907
E-mail: s919506@mail.yzu.edu.tw

Ching-Jung TING
Assistant Professor
Department of Industrial Engineering and Management
Yuan Ze University
135 Yuan-Tung Road, Chungli,
Taiwan 320, R.O.C.
Fax: +886-3-4638907
E-mail: ietingcj@saturn.yzu.edu.tw

Abstract: The Vehicle Routing Problem with Time Windows (VRPTW) is an important problem occurring in many logistics systems. The objective of VRPTW is to serve a set of customers within their predefined time windows at minimum cost. Ant Colony System algorithm (ACS) that is capable of searching multiple search areas simultaneously in the solution space is good in diversification. On the other hand, Simulated Annealing algorithm (SA) is a local search technique that has been successfully applied to many NP-hard problems. A hybrid algorithm (IACS-SA) that combines an improved ACS with SA is proposed in this paper. The algorithm has been tested on 56 Solomon benchmark problems. The results show that our IACS-SA is competitive with other meta-heuristic approaches in the literature. The results also indicate that such a hybrid algorithm outperforms the individual heuristic alone.

Key Words: logistics, ant colony system, simulated annealing, hybrid algorithm

1. INTRODUCTION

The routing and scheduling of vehicles represent an important component of distribution and transportation system. Vehicle Routing Problem (VRP) is a well-know combinatorial problem with considerable economic significance in logistics system. Many different types of VRP can be generated in terms of the demands and restrictions of practical applications. The Vehicle Routing Problem with Time Windows (VRPTW) is a generalization of the VRP where the service of a customer must be started at a predefined time windows. The VRPTW has been applied extensively in practice, such as bank deliveries, postal deliveries, school bus routing, security patrol service and industrial refuse collection. Moreover, the VRPTW has been proved to be NP-hard and exact algorithms cannot find the optimal solution for large VRPTW within reasonable computational times. Thus many heuristic approaches have been proposed in the literature. Many meta-heuristic approaches like Simulated Annealing (Czech and Czarnas, 2002; Li and Lim, 2003), Genetic Algorithms (Potvin and Bengio, 1996; Berger et al., 2001; Chen et al., 2001; Tan et al., 2001; Ting and Huang, 2004), Tabu Search (Potvin et al., 1996; Chiang and Russell, 1997; Taillard et al., 1997), and Ant Colony System (Gambardella et al., 1999) have been presented to find the near-optimal solutions within reasonable time. These meta-heuristic approaches are developed by imitating artificial intelligence, biological evolution and/or physics phenomenon.

Among these meta-heuristic approaches, Ant System (AS) is a newer distributed meta-heuristic first introduced by Colorni et al. (1991). The AS approach is based on the behavior of real ants searching for food. Real ants communicate with each other using an aromatic essence called pheromone that they laid down on the path they traverse. The selection of the pheromone trail reflects the length of the paths as well as the quality of the food source found. Ant algorithms have been applied to many combinatorial problems successfully, including traveling salesman problem (Dorigo et al., 1996; Dorigo and Gambardella, 1997a, 1997b), quadratic assignment problem (Maniezzo ea al., 1994; Dorigo et al., 1996; Gambardella et al., 1999; Stützle and Dorigo, 1999; Talbi et al., 2001), job-shop scheduling (Colorni et al., 1994; Dorigo et al., 1996), vehicle routing problem (Bullnheimer et al., 1998; Bullnheimer et al., 1999, Gambardella et al., 1999), sequential ordering problem (Gambardella and Dorigo, 1997) and graph coloring problem (Costa and Hertz, 1997).

Bullnheimer et al. (1998) were the first researchers that used AS to solve the VRP. They presented a hybrid Ant System algorithm (HAS) that added the 2-opt heuristic and then based on Saving Algorithm to construct routes of ants. However, the results of HAS were not as good as other meta-heuristic approaches. Bullnheimer et al. (1999) developed an improved AS (IAS) for the VRP. They applied the idea of candidate lists (Dorigo and Gambardella, 1997a, 1997b) to construct vehicle routes. Candidate lists can concentrate the search on promising candidates that can be better used for further iterations. Comparisons on a set of standard problems showed that the performance of IAS is significantly better than AS for VRP, and it outperformed SA and Neural Network. Gambardella et al. (1999) defined a hybrid Ant System algorithm for VRP (HAS-VRP), which was inspired by ACS. Results obtained by HAS-VRP were competitive with those of the best-known algorithms and new upper bounds have been found for well-known problem instances. Furthermore, Gambardella et al. (1999) proposed a multiple Ant Colony System to vehicle routing problem with time windows (MACS-VRPTW) and improved some of the best-known solutions in the literature. In MACS-VRPTW, one colony minimizes the number of vehicles while the other colony minimizes the traveled distances.

This paper focuses on the study of a hybrid of two search heuristics, ACS and SA, on VRPTW. ACS that is capable of searching multiple search areas simultaneously in the search space is good in diversification. On the other hand, SA is a local search technique that has been successfully applied to many NP-hard problems. We propose an improved ACS algorithm (IACS) for VRPTW possessed a new route construction rule, a new pheromone update rule and diverse local search approaches (2-opt and Insertion Move). Then, we create a hybrid algorithm (IACS-SA) that combines the strengths of both search heuristics. Finally our IACS-SA is tested by Solomon’s 56 VRPTW benchmark problems and compared the performance with other meta-heuristics.

The rest of the paper is organized as follows. Section 2 introduces the vehicle routing problem with time windows formulation. Section 3 describes the improved ant colony system that incorporates with a new state transition rule, a new pheromone updating rule and diverse local search approaches: 2-opt (Lin, 1965) and insertion move. Section 4 introduces the procedure of SA. Section 5 proposes the hybrid algorithm of IACS-SA. Computational results on Solomon’s 56 VRPTW benchmark problems with IACS-SA and comparison against other meta-heuristics are reported in section 6. Finally, concluding remarks are made.

2. MATHEMATICAL FORMULATION

In the vehicle routing problem with time windows (VRPTW), a fleet of m identical vehicles delivers goods to n customers whose demands are known. All vehicle routes must start and end at the central depot with the same capacity and the maximum service time limitations. Every customer must be visited exactly once by one vehicle. At each customer, the start of the service must be within a given time interval. Moreover, a vehicle is permitted to arrive before the beginning of the time window, and wait until service is possible. However, it is not permitted to arrive after the end of the time window. The objective is to minimize the total traveled distance. The typical mathematical model of the VRPTW can be formulated as follows:

Inputs:

b_j^k : the time vehicle k starts service at node j, $j \in N$, $b_j^k = \max\{e_j, b_i^k + t_{ij}\}$

c_{ij} : travel distance of edge (i, j)

d_i : demand of node i, $i \in N$

e_i : the earliest start time for node i

l_i : the latest start time for node i

N : the set of nodes, 0 denotes the depot

Q : capacity of vehicles

t_{ij} : travel time of edge (i, j), and t_{ij} includes the service time at node i

V : set of vehicles

Decision variable:

$$x_{ij}^k = \begin{cases} 1 & \text{: if vehicle k visits node j immediately after node i, i} \neq \text{j} \\ 0 & \text{: otherwise} \end{cases}$$

Model:

$$Min \quad \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}^k \tag{1}$$

$$s.t. \quad \sum_{k \in V} \sum_{i \in N} x_{ij}^k = 1 \qquad \forall j \in N \setminus \{0\} \tag{2}$$

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

$$\sum_{i \in N} d_i \sum_{j \in N} x_{ij}^k \leq Q \qquad \forall k \in V \tag{4}$$

$$\sum_{i \in N} x_{ih}^k - \sum_{j \in N} x_{hj}^k = 0 \qquad \forall h \in N \setminus \{0\}, \forall k \in V \tag{5}$$

$$\sum_{j \in N \setminus \{0\}} x_{0j}^k = 1 \qquad \forall k \in V \tag{6}$$

$$\sum_{i \in N \setminus \{0\}} x_{i0}^k = 1 \qquad \forall k \in V \tag{7}$$

$$x_{ij}^k (b_i^k + t_{ij}) \leq b_j^k \qquad \forall i \in N, \forall j \in N, \forall k \in V \tag{8}$$

$$e_i \leq b_i^k \leq l_i \qquad \forall i \in N, \forall k \in V \tag{9}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall i \in N, \forall j \in N, \forall k \in V \tag{10}$$

The objective function (1) is to minimize the total traveled distance. Constraints (2) and (3) insure that each node is served exactly once by only one vehicle. Constraint (4) states that the total demand of any vehicle route cannot exceed the vehicle capacity. Constraint (5) represents that a vehicle must leave the node that it has just entered. Constraints (6) and (7) denote that each vehicle must leave and return to the depot. Constraint (8) expresses that the vehicle k cannot arrive at j before $b_i^k + t_{ij}$ if it travels from i to j. Constraint (9) ensures that the start of the service must be within the time windows at each node. Constraint (10) is the integrality constraint.

3. IMPROVED ANT COLONY SYSTEM

The Improved Ant Colony System (IACS) was first proposed by Ting and Chen (2004). The IACS, which is based on the ACS proposed by Dorigo and Gambardella (1997), includes four steps as follows:

Step 1: Set parameters and initialize the pheromone trails.
Step 2: Each ant builds the solution by the state transition rule and carries out local pheromone update.
Step 3: Apply the local search to improve the ants' solution.
Step 4: Update the global pheromone information.

3.1 Pheromone trails Initialization

In our research, the initial pheromone level of each edge is evaluated by eq. (11).

$$\tau_0 = (n * L_{nn})^{-1} \tag{11}$$

where n is the number of nodes and L_{nn} is the tour length produced by the Nearest Neighbor (NN) heuristic. The Nearest Neighbor heuristic procedures that we use for generating the initial solution are as follows:

Step 1: Randomly start with one node that has not been visited as the beginning of a new route.
Step 2: If there are not any nodes that its time window is conformed, go to Step 1. Otherwise, find the node that is closest to the last node and has not been visited, then add this node to the route.
Step 3: If all nodes are visited, stop. Otherwise, repeat Step 2 until the vehicle capacity or maximum service time constraint is violated; then, drop the last node from the current route, and go to Step 1.

3.2 Solution Construction

In the original ACS, each ant moves from present node i to the next node v according to the rule given by (12).

$$v = \begin{cases} \arg \max\limits_{j \in U} \left[(\tau_{ij})^{\alpha} (\eta_{ij})^{\beta} \right] & q \le q_0 \\ V & q > q_0 \end{cases}$$
$$V: P_{ij} = \frac{(\tau_{ij})^{\alpha} (\eta_{ij})^{\beta}}{\sum\limits_{j \in U} (\tau_{ij})^{\alpha} (\eta_{ij})^{\beta}} \tag{12}$$

where U is the set of nodes which are not visited yet, τ_{ij} is the pheromone of edge (i, j), η_{ij} denotes the savings of combining nodes i and j on one tour as opposed to serving them on two different tours. Thus, the η_{ij} is calculated as follows:

$$\eta_{ij} = d_{i0} + d_{0j} - d_{ij} \tag{13}$$

where d_{ij} denotes the distance between nodes i and j, and node 0 is the depot; and α, β are the parameters that determine the relative influence of pheromone versus distance (β, α>0). Moreover, q is a random number following uniform distribution in [0, 1], and q_0 is a pre-defined parameter ($0 \le q_0 \le 1$). If $q \le q_0$ then the best next node is chosen according to *arg*.

On the contrary, the next node is chosen according to *V.* Thus, the parameter q_0 determines the relative importance of exploitation (*arg*) versus exploration (*V*).

In our IACS, the best solution so far will be preserved and becomes the 1st solution in the next generation. Thus, we only reconstruct *b*-1 solutions (*b* is the number of ants) in each generation. The tour construction in IACS algorithm is as follows:

Step 1: Generate the ant's starting node randomly.
Step 2: Chose next node according to the tour construction rule by eq. (12).
Step 3: Repeat step 2 until the ant visits all nodes.
Step 4: Divide the tour according to the vehicle capacity and time windows restrictions.

3.3 Local Search

In original ACS, after constructing all ants' tours, local search is applied to improve each solution. However, local search is a time-consuming procedure of ACS. In our IACS, we apply local search to the best solution in the current iteration with two different types of local search: 2-opt and insertion move. Figure 1 illustrates a 2-opt exchange. Edges (3, 6) and (0, 4) are dropped from the current route. Then, we must introduce edges (0, 6) and (3, 4), since introducing edge (6, 4) leads to a subtour and introducing edge (3, 6) leads to the original route. Note that link symmetry is required here since the direction on edges (4, 5) and (5, 6) is reversed. The Insertion move is illustrated as figure 2. One node that is selected will be inserted in the same route (figure 2a) or in the other route (figure 2b) to reduce the route length.

In our research, we apply 2-opt to improve the solution followed by the Insertion move to the 2nd solution. The procedures of local search in IACS are as follows:

Step 1: Sort the solutions 2~*b* in ascending orders according to the objective function value.
Step 2: Carry out local search on the 2nd solution.

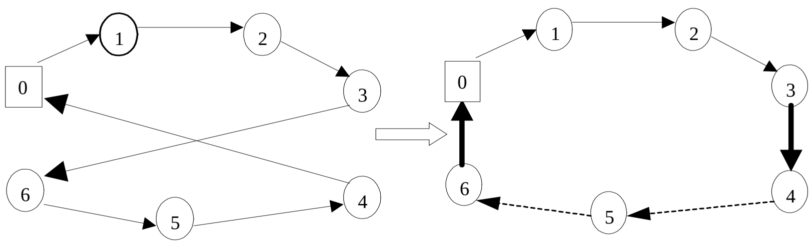
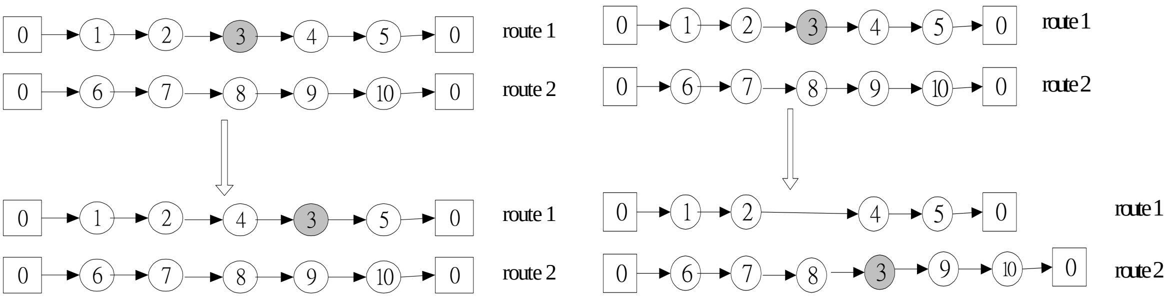


Figure 1. 2-opt



(a) Insert in the same tour (b) Insert in the other tour

Figure 2. Insertion Move

3.4 Pheromone Update

The pheromone updating of ACS includes global and local updating rules. Following Dorigo and Gambardella (1997), local updating rule in eq. (14) is applied to change pheromone level of edges while building a solution.

$$\tau_{ij}^{new} = (1-\rho)\tau_{ij}^{old} + \rho\tau_0 \tag{14}$$

where $0 \le \rho \le 1$ is a user-defined parameter called evaporation coefficient, and τ_0 is initial pheromone as defined in eq. (11). After improving the 2nd solution by local search, our global updating rule is applied to the first two solutions. The rule is described as follow:

$$\tau_{ij}^{new} = (1-\rho)\tau_{ij}^{old} + \rho\sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{15}$$

where

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{L_{u+1} - L_k}{L_{u+1}} & \text{if (i, j)} \in \text{tour obtained by ant k} \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

where L_k is the length of tour obtained by ant k, u is the number of solutions whose global pheromone will be updated and equals to 2 in this study.

3.5 Overall Procedure of IACS

Figure 3 shows the flowchart of IACS for VRPTW, and the procedures of IACS are described as follows:

- Step1: Set parameters.
- Step2: Generate an initial solution using Nearest Neighbor heuristic.
- Step3: Apply the local search (2-opt and Insertion Move) to the initial solution and let it to be the solution 1 of population. g = 1, h = 2.
- Step4: Construct solutions based on the route construction rule and progress local pheromone update. h = h + 1.
- Step5: If h > *b*, h = 2 and go to Step 6. Otherwise, go to Step 4.
- Step6: Sort the solutions 2~*b* in ascending order and apply local search (2-opt and Insertion Move) to 2nd solution.
- Step7: Apply the global pheromone update for solutions 1~u.
- Step8: Record the best solution so far and let it to be the solution 1 in the next generation. g = g + 1.
- Step9: If the stopping criterion (maximum number of generations, G, in this paper) is met, stop, then output the best solution. Otherwise, go to Step 4.

4. SIMULATED ANNEALING ALGORITHM

Simulated Annealing (SA) was first used to search the feasible solutions of an optimization problem by Kirkpatrick et al. (1983). SA is inspired by an analogy between the physical annealing of solid and combinatorial problems. In the physical process a solid is first melted at high temperature and then cooled very slowly to obtain a perfect lattice structure corresponding to a minimum energy state. SA transfers this process to a local search algorithm for combinatorial optimization problems. It does so by associating the set of solutions of the problem attacked with the states of the physical system, the objective function with the physical energy of the solid and the optimal solutions with the minimum energy states.

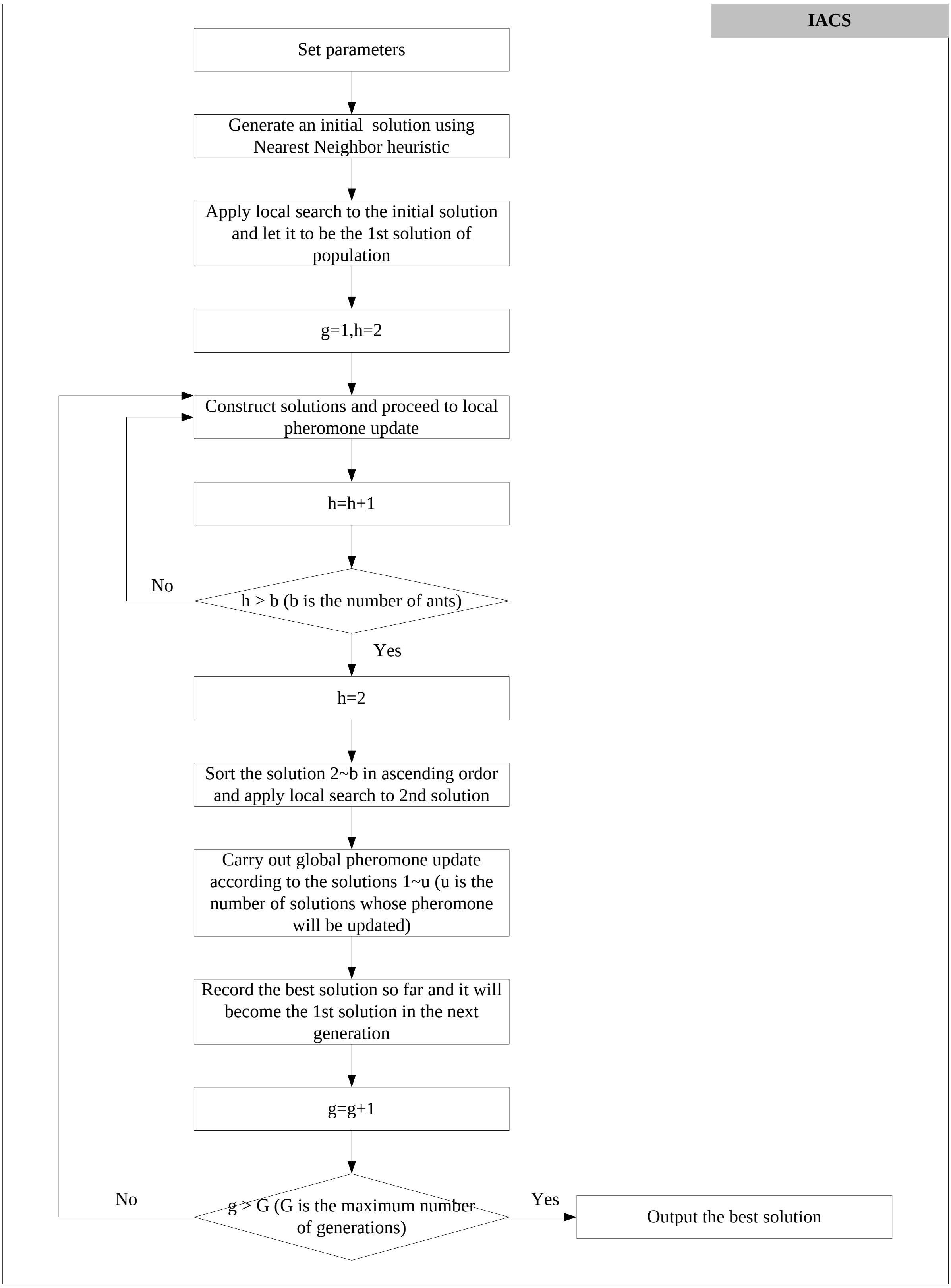


Figure 3. The flowchart of IACS

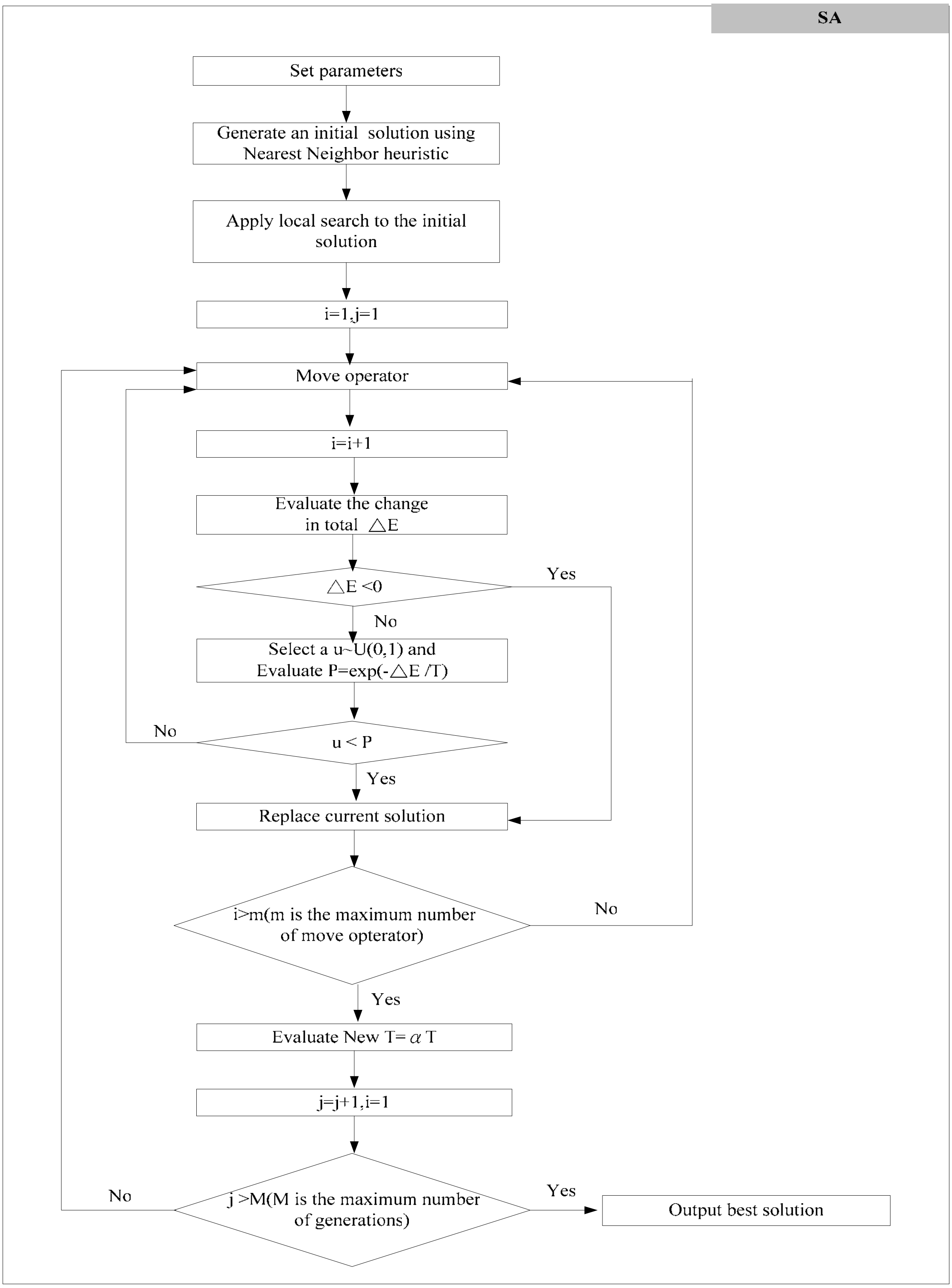


Figure 4. The flowchart of SA

SA is a local search strategy which tries to avoid local minima by accepting worse solutions with some probability. In our research, the procedure of SA is illustrated as figure 4 and described below:

Step 1: Set parameters: T (initial temperature), γ (cooling parameter), m (maximum number of move operator), M (maximum number of iterations).
Step 2: Generate an initial solution x^0 using Nearest Neighbor heuristic. Set $x = x^0$.
Step 3: Apply the local search (2-opt and Insertion) to the initial solution. $i = 1$, $j = 1$.
Step 4: Compute the objective function value of current solution $f(x)$.
Step 5: a. If $i \le m$, apply the move operator (2-opt exchange and Insertion move) to current solution to generate new solution x', and $i = i+1$, then go to step 5b. Otherwise, go to step 6.
b. Evaluate $\triangle E = f(x') - f(x)$. If $\triangle E \le 0$, go to step 5d; otherwise, go to step 5c.
c. Select a random variable $u \sim U(0,1)$. If

$$u < P(\Delta E) = \exp(-\Delta E / T) \tag{17}$$

then go to step 5d; otherwise, go to step 5a.
d. Accept the exchange, set $x = x'$ and $f(x) = f(x')$, then go to step 5a.
Step 6: If $j \le M$, evaluate $T = \gamma T$ and $j = j+1$, then go to step 3. Otherwise, stop.

5. HYBRID ALGORITHM IACS-SA

To improve the solutions constructed by ants, the IACS applies the local search to solutions. Nevertheless, if the solution have been trapped into local optima, it is ineffective to use local search to improve it. SA is a local search based algorithm, but it can help the solution out local optima. However, SA is a single search approach, so the initial solution of SA will influence its solution quality. Therefore, the concept of IACS-SA is to combine advantages of IACS and SA to search high quality solution for VRPTW. In IACS-SA, IACS can provide a good initial solution for SA, and SA can assist IACS to escape from the local optima. The framework of IACS-SA is shown in figure 5. We first use Nearest Neighbor heuristic to generate the initial solution for both IACS and SA, and then the initial solution is improved by local search. In each iteration, both IACS and SA solve the VRPTW respectively. IACS generates new solutions according to tour construction rule and the solutions improved by local search, and on the other hand SA applies the move operator (2-opt exchange and Insertion move) to improve the solution m times and then the temperature cools down. Afterward IACS and SA will communicate their best solution so far to each other. If the solutions obtained by them are unequal, the inferior one will be substituted by the other one. This procedure will be performed for a pre-specified number of iterations. IACS-SA outputs the best solution till now. According to our experiment the IACS-SA can find a good solution within small iterations. Finally, the best solution will be further improved by SA.

6. NUMERICAL ANALYSIS

6.1 Benchmark Problems

We conduct computational experiments on Solomon's 56 benchmark problems. These problems were generated in six classes: R1, R2, C1, C2, RC1 and RC2. All problems have 100 customers, a central depot, capacity constraints and time window constraints. The customers are randomly distributed in R1 and R2 problems, while in C1 and C2 problems they are clustered. In RC1 and RC2 problems, customers are mixed with both clustered and randomly distributed. The C1, R1 and RC1 problems have short scheduling horizon, while C2, R2 and RC2 have longer scheduling horizon. We summarize information of the Solomon benchmark problems in table 1. Columns 2-7 show the number of problems in each type, the vehicle capacity, schedule horizon, service times of each node, the node distribution and the elasticity of time window.

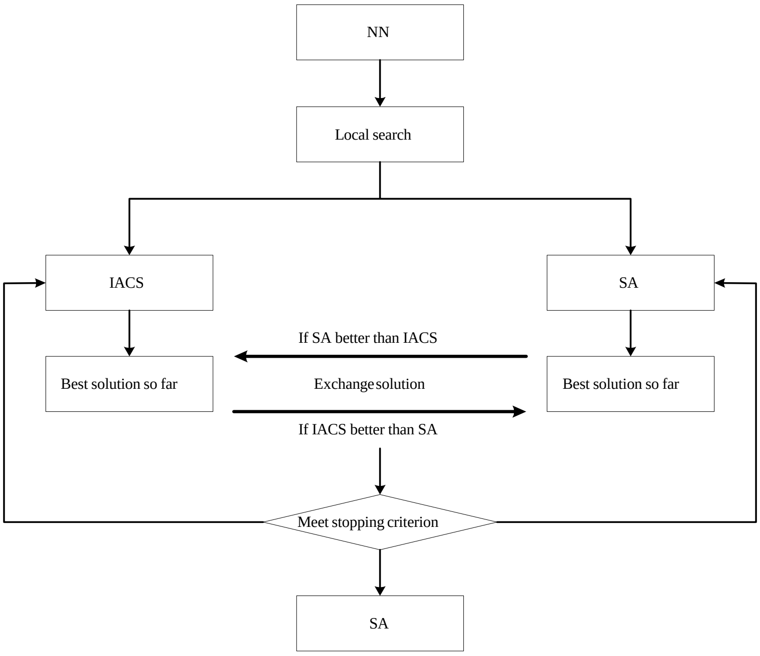


Figure 5. The framework of IACS-SA

Table 1. The information of Solomon's VRPTW benchmark problems

Type	Number of problems	Vehicle capacity	Schedule horizon	Service time	Node distribution	Time windows
R1	12	200	230	10	randomly	narrow
R2	11	1000	1000	10	randomly	large
C1	9	200	1236	90	clustered	narrow
C2	8	700	3390	90	clustered	large
RC1	8	200	240	10	Mixed	narrow
RC2	8	1000	960	10	Mixed	large

6.2 Computational Results

In our research, we treat the hard window case, where a vehicle must wait till the earliest start time if it arrives early. Furthermore, arrival time of the vehicle cannot exceed the latest service time of the customer. The overall aim in our research is to reduce the total distance traveled by the vehicles. The IACS-SA is coded in Borland C++ Builder 5.0 and executed on a Pentium Ⅲ 1000 MHz PC equipped with 128 MB of RAM. The IACS-SA parameters used for VRP instances are b=n/10, α=1, β=1, ρ=0.1, q_0=0.1, u=2, G=n/2, T=10, γ=0.99, m=n*n and M=3n. We summarize the computational results that include the best known solutions, literatures, solutions obtained by IACS-SA and the deviations of traveled distance from the best know solutions (RPD-relative percentage deviation) in tables 2 and 3. In addition, table 4 shows the RPD of average total distances of IACS-SA against the best-known results, as well as the number of better solutions we have obtained. As shown in tables 2, 3 and 4, the IACS-SA has yielded better or close routes as compared to the best-known solutions in 33 out of 56 problem instances (59%). There are 14 (25%) new best solutions produced. Moreover, the average traveled distance obtained by IACS-SA for R2 and RC2 problems are better than best known results.

Another observation from the results is the vehicle fleet size. Our objective function of VRPTW only considers the total distance traveled by vehicles so that it is possible for IACS-SA to reduce the total traveled distance by increasing the number of vehicles. Nevertheless, the average number of vehicles obtained by IACS-SA for all six problem types against the best-known results is increased by less than one vehicle.

Table 2. The computational results of type 1 problems

No.	Best known	Literature	IACS-SA	RPD
R101	1607.7[a]/18[b]	Desrochers et al., 1992	1670.66/19	3.92
R102	1434/17	Desrochers et al., 1992	1535.52/17	7.08
R103	1207/13	Thangiah et al., 1994	1227.97/14	1.74
R104	982.01/10	Rochat and Taillard, 1995	996.88/10	1.51
R105	1377.11/14	Rochat and Taillard, 1995	1365.23[#]/15	**-0.86**
R106	1252.03/12	Rochat and Taillard, 1995	1247.72[#]/13	**-0.34**
R107	1120.85/10	Homberger and Gehring, 1999	1088.73[#]/11	**-2.87**
R108	968.59/9	Taillard et al., 1997	958.31[#]/10	**-1.06**
R109	1013.16/12	Chiang and Russell, 1997	1154.55/12	13.96
R110	1080.36/11	Rochat and Taillard, 1995	1089.73/12	0.87
R111	1099.46/10	Homberger and Gehring, 1999	1111.95/11	1.14
R112	953.63/10	Rochat and Taillard, 1995	995.48/10	4.39
Average	1174.66/12.17		1203.56/12.83	2.46
C101	827.3/10	Desrochers et al., 1992	828.94[*]/10	0.20
C102	827.3/10	Desrochers et al., 1992	828.94[*]/10	0.20
C103	828.06/10	Rochat and Taillard, 1995	828.06[*]/10	**0.00**
C104	824.78/10	Rochat and Taillard, 1995	828.20[*]/10	0.41
C105	828.94/10	Potvin and Bengio, 1996	828.94[*]/10	**0.00**
C106	827.3/10	Desrochers et al., 1992	828.94[*]/10	0.20
C107	827.3/10	Desrochers et al., 1992	828.94[*]/10	0.20
C108	827.3/10	Desrochers et al., 1992	828.94[*]/10	0.20
C109	828.94/10	Potvin and Bengio, 1996	828.94[*]/10	**0.00**
Average	827.47/10		828.76/10.00	0.16
RC101	1642.82/15	Chiang and Russell, 1997	1653.25/15	0.63
RC102	1540.97/13	Chiang and Russell, 1997	1483.77[#]/14	**-3.71**
RC103	1110/11	Thangiah et al., 1994	1284.73/11	15.74
RC104	1135.83/10	Rochat and Taillard, 1995	1159.37/11	2.07
RC105	1637.15/13	Homberger and Gehring, 1999	1539.06[#]/15	**-5.99**
RC106	1395.37/12	Chiang and Russell, 1997	1389.66[#]/12	**-0.41**
RC107	1230.54/11	Taillard et al., 1997	1247.48/11	1.38
RC108	1139.82/10	Taillard et al., 1997	1153.40/11	1.19
Average	1354.06/11.88		1363.84/12.50	1.36

[a]：Traveled distance
[b]：Number of vehicles
[*]：Equal or close to the best know solutions
[#]：Better than the best know solution

Table 5 compares the average vehicle numbers, traveled distance and computational time in seconds obtained by IACS-SA with other seven meta-heuristic algorithms. The algorithms considered are: the Tabu Search (TS-P) of Potvin et al. (1996), the Tabu Search (TS-T) of Taillard et al. (1997), the Multiple Ant Colony System (MACS) of Gambardella et al. (1999), the Genetic Algorithm (GA) of Berger et al. (2001), the Hybrid Genetic Algorithm (HGA) of Chen et al. (2001), the hybrid SA and TS (SATS) of Tan et al. (2001) and the tabu-embedded

SA (TESA) of Li and Lim (2003). From table 5, our IACS-SA yields best results for R1, R2, C2, RC1 and RC2 and produces the lowest cumulated total traveled distance among eight algorithms. Taillard's TS and Gambardella's MACS obtained the best results for C1 and C2. It is obviously that the IACS-SA outperforms these competing heuristics in terms of total traveled distance. Effects of computer performance are influenced by many factors such as CPU speed, memory capacity, operation system and programming language. Therefore, a fair transformation of computational time is difficult to establish. However, our hybrid algorithm takes about 378 seconds to run for each instances, so its computational time is reasonable.

Table 3. The computational results of type 2 problems

No.	Best known	Literature	IACS-SA	RPD
R201	1252.37[a]/4[b]	Homberger and Gehring, 1999	1276.61/4	1.94
R202	1198.45/3	Homberger and Gehring, 1999	1101.91[#]/4	**-8.06**
R203	942.64/3	Homberger and Gehring, 1999	954.27/3	1.23
R204	854.88/2	Homberger and Gehring, 1999	762.92[#]/3	**-10.76**
R205	1013.47/3	Homberger and Gehring, 1999	1033.31/3	1.96
R206	833/3	Thangiah et al., 1994	915.13/3	9.86
R207	814.78/3	Rochat and Taillard, 1995	819.57/3	0.59
R208	726.82/2	Gambardella et al. 1999	735.51/2	1.20
R209	855/3	Thangiah et al., 1994	916.49/3	7.19
R210	955.39/3	Homberger and Gehring, 1999	944.39[#]/3	**-1.15**
R211	910.09/2	Homberger and Gehring, 1999	794.46[#]/3	**-12.71**
Average	941.54/2.82		932.23/3.09	**-0.79**
C201	591.56/3	Potvin and Bengio, 1996	591.56[*]/3	**0.00**
C202	591.56/3	Potvin and Bengio, 1996	591.56[*]/3	**0.00**
C203	591.17/3	Rochat and Taillard, 1995	591.17[*]/3	**0.00**
C204	590.6/3	Potvin and Bengio, 1996	590.6[*]/3	**0.00**
C205	588.88/3	Potvin and Bengio, 1996	588.88[*]/3	**0.00**
C206	588.49/3	Potvin and Bengio, 1996	588.49[*]/3	**0.00**
C207	588.29/3	Rochat and Taillard, 1995	588.29[*]/3	**0.00**
C208	588.32/3	Rochat and Taillard, 1995	588.32[*]/3	**0.00**
Average	589.86/3		589.86/3.00	0.00
RC201	1249/4	Thangiah et al., 1994	1444.15/4	15.62
RC202	1164.25/4	Taillard et al., 1997	1165.22[*]/4	0.08
RC203	1060.45/3	Homberger and Gehring, 1999	949.47[#]/4	**-10.47**
RC204	798.46/3	Gambardella et al. 1999	798.97[*]/3	0.06
RC205	1302.42/4	Homberger and Gehring, 1999	1343.70/4	3.17
RC206	1158.81/3	Thangiah et al., 1996	1085.00[#]/4	**-6.37**
RC207	1068.86/3	Gambardella et al. 1999	1003.60[#]/4	**-6.11**
RC208	833.4/3	Gambardella et al. 1999	848.36/3	1.80
Average	1079.46/3.38		1079.81/3.75	**-0.28**

[a]：Traveled distance
[b]：Number of vehicles
[*]：Equal or close to the best know solutions
[#]：Better than the best know solution

Table 4 Percentage deviation and number of new best solutions

Pro. type	Best known	Distance (RPD)	No. of new best
R1	1174.66	1203.56 (2.46%)	4/12
R2	941.54	932.23 (-0.79%)	4/11
C1	827.47	828.76 (0.16%)	0/9
C2	589.86	590.49 (0.00%)	0/8
RC1	1354.06	1365.35 (1.36%)	3/8
RC2	1079.46	1079.81 (-0.28%)	3/8
All	56087	56429 (0.55%)	14/56

Table 5 Comparison of IACS-SA with other meta-heuristics

Problem Type	**R1**	**R2**	**C1**	**C2**	**RC1**	**RC2**	**Total**
	12.60[a]	3.10	10.00	3.00	12.60	3.40	427
TS-P[1]	1294.70[b]	1185.90	861.00	602.50	1465.00	1476.10	64679
	639[c]	722	435	431	586	662	32957
	12.17	2.82	10.00	3.00	11.50	3.38	410
TS-T[2]	1209.35	980.27	**828.38**	**589.86**	1389.22	1117.44	57953
	13774	20232	14630	16375	11264	11596	833390
	12.00	2.73	10.00	3.00	11.63	3.25	407
MACS[3]	1217.73	967.75	**828.38**	**589.86**	1382.42	1129.19	57525
	1800	1800	1800	1800	1800	1800	100800
	11.92	2.73	10.00	3.00	11.50	3.25	405
GA[4]	1221.10	975.43	828.48	589.93	1389.89	1159.37	57523
	-	-	-	-	-	-	-
	13.20	5.00	10.10	3.25	13.50	5.00	478
HGA[5]	1227	980	861	619	1427	1223	59405
	-	-	-	-	-	-	84000
	13.10	4.60	10.00	3.30	12.70	5.60	470
SATS[6]	1213.16	952.30	841.92	612.75	1415.62	1120.37	57799
	-	-	-	-	-	-	15400
	12.08	2.91	10.00	3.00	11.75	3.25	411
TESA[7]	1215.14	953.43	828.38	589.86	1385.47	1142.48	57467
	1474	3882	201	1220	916	2669	100639
	12.83	3.09	10.00	3.00	12.50	3.75	432
Ours	**1203.56**	**932.23**	828.76	**589.86**	**1363.84**	**1079.81**	**56429**
	425	437	239	363	403	370	21146

[a]: Number of vehicles
[b]: Traveled distance
[c]: Computational time in seconds
[1]: TS-P, seconds on a SPARC10 workstation
[2]: TS-T, seconds on a Sun Sparc 10
[3]: MACS, second on a Sun UltraSparc 1 167 MHz
5: HGA, seconds on a Pentium Ⅱ 330 MHz PC
6: SATS, seconds on a Pentium Ⅱ 330 MHz PC
7: TESA, seconds on a Pentium Ⅲ 545 MHz PC

7. CONCLUSION

In this research, we have presented a hybrid algorithm (IACS-SA) that combines the improved ant colony system (IACS) and simulated annealing algorithm (SA). Based on the computational results of Solomon's 56 benchmark instances, our hybrid algorithm produces 14 out of 56 new best solutions against the best-known results. Moreover, the average RPD of all instances is merely 0.55%. Compared with other meta-heuristic algorithms, IACS-SA performs best on problem types R1, R2, C2, RC1 and RC2, and yields the lowest total traveled distance. Thus, it is believed that the performance of IACS-SA is not influenced by different types of instances and it is among the best heuristics in solving VRPTW problems. In addition, the solving speed of IACS-SA is reasonable in terms of computational time. However, our objective function of VRPTW does not involve the vehicle fleet size. Hence, improving IACS-SA to reduce traveled distance and number of vehicles simultaneously is in progress. Another work is to apply the IACS-SA to real-word problem to determine its applicability and practicability.

REFERENCES

Berger, J., Barkaoui, M. and Bräysy, O. (2001) A parallel hybrid genetic algorithm for the vehicle routing problem with time windows, Working paper, Defense Research Establishment Valcartier, Canada.

Bullnheimer, B., Hartl, R. F. and Strauss, C. (1999) An improved ant system for the vehicle routing problem, **Annals of Operations Research, Vol. 89**, 319-328.
Bullnheimer, B., Hartl, R.F. and Strauss, C. (1998) Applying the ant system to the vehicle routing problem, In S. Voss, S. Martello, I.H. Osman, and C. Roucairol, editors, Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, Kluwer, Boston, MA, 109-120.
Chen, C.H and. Ting, C.J. (2004) An improved ant colony system algorithm for the vehicle routing problem, Working Paper 2004-001, Department of Industrial Engineering and Management, Yuan Ze University, Taiwan, R.O.C.
Chen, T.K., Lee, L.H. and Ke, O. (2001) Hybrid genetic algorithm in solving vehicle routing problem with window constraints, **Asia-Pacific Journal of Operational Research, Vol. 18**, 121-130.
Chiang, W.C. and Russell, R.A. (1997) A reactive tabu search metaheuristics for the vehicle routing problem with time windows, **INFORMS Journal on Computing, Vol. 9**, 417-430.
Clarke, G. and Wright, J.W. (1964) Scheduling of vehicles from a central depot to a number of delivery points, **Operations Research, Vol. 12**, 568-581.
Colorni, A., Dorigo, M. and Maniezzo, V. (1991) Distributed optimization by ant colonies, In F. Varela and P. Bourgine, editors, **Proceedings of the European Conference on Artificial Life**, Elsevier, Amsterdam.
Colorni, A., Dorigo, M., Maniezzo, V. and Trubian, M. (1994) Ant system for job-shop scheduling, **Belgian Journal of Operations Research, Statistics and Computer Science, Vol. 34, No. 1**, 39-53.
Costa, D. and Hertz, A. (1997) Ants can colour graphs, **Journal of the Operational Research Society, Vol. 48**, 295-305.
Czech, Z.J. and Czarnas, P. (2002) A parallel simulated annealing for the vehicle routing problem with time windows, **Proceedings 10th Euromicro Workshop on Parallel**, Distributed and Network-based Processing, Canary Islands, Spain, 376-383.
Desrochers, M., Desrosiers, J. and Solomon, M.M. (1992) A new optimization algorithm for the vehicle routing problem with time windows, **Operations Research, Vol. 40**, 342-354.
Dorigo M. and Gambardella, L.M. (1997) Ant colonies for the traveling salesman problem, **BioSystems, Vol. 43**, 73-81.
Dorigo, M. and Gambardella, L.M. (1997) Ant colony system: A cooperative learning approach for the traveling salesman problem, **IEEE Transactions on Evolutionary Computation, Vol. 1**, 53-66.
Dorigo, M., Maniezzo, V. and Colorni, A. (1996) Ant system: Optimization by a colony of cooperating agents, **IEEE Transactions on Systems, Man and Cybernetics Part B, Vol. 26**, 29-41.
Gambardella, L.M. and Dorigo, M. (1997) HAS-SOP: A hybrid ant system for the sequential ordering problem, Tech. Rep. No. IDSIA 97-11, IDSIA, Lugano, Switzerland.
Gambardella, L.M., Tai*l*lard, E. and Agazzi, G. (1999) MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows, in New Ideas in Optimization, D. Corne, M. Dorigo and F. Glover (eds), McGraw-Hill, London, 63-76.
Gambardella, L.M., Taillard, E. and Dorigo, M. (1999) Ant colonies for the quadratic assignment problem, **Journal of the Operational Research Society, Vol. 50**, 167-176.
Homberger, J. and Gehring, H. (1999) Two evolutionary metaheuristics for the vehicle routing problem with time windows, **INFOR, Vol. 37**, 297-318.
Li, H. and Lim, A. (2003) Local search with annealing-like restarts to solve the VRPTW, **European Journal of Operational research, Vol. 150**, 115-127.
Lin, S. (1965) Computer solution of the traveling salesman problem, **Bell System Computer Journal, Vol. 44**, 2245-2269.

Maniezzo, V., Colorni, A. and Dorigo, M. (1994) The ant system applied to the quadratic assignment problem, Tech. Rep. IRIDIA/94-28, Université Libre de Bruxelles, Belgium.
Potvin, J.Y. and Bengio, S. (1996) The vehicle routing problem with time windows - Part II: Genetic search, **INFORMS Journal on Computing, Vol. 8**, 165-172.
Potvin, J.Y., Kervahut, T., Garcia, B.L. and Rousseau, J.M. (1996) The vehicle routing problem with time windows, Part I: tabu search, **INFORMS Journal on Computing, Vol. 8, No. 2**, 158-164.
Rochat, Y. and Taillard, E.D. (1995) Probabilistic diversification and intensification in local search for vehicle routing, **Journal of Heuristics, Vol. 1**, 147-167.
Rosenkrantz, D., Sterans, R. and Lewis, P. (1974) Approximate algorithms for the traveling salesperson problem, **In 15th Annual IEEE Symposium of Switching and Automata Theory**, 33-42.
Solomon, M.M. (1987) Algorithms for the vehicle routing and scheduling problems with time window constrains, **Operational Research, Vol. 35, No. 2**, 254-265.
Stützle, T. and Dorigo, M. (1999) ACO algorithms for the quadratic assignment problem**,** In D. Corne, M. Dorigo and F. Glover, editors, **New Ideas in Optimization**, McGraw-Hill.
Taillard, E., Badeau, P., Gendreau, M., Geurtin, F. and Potvin, J.Y. (1997) A tabu search heuristic for the vehicle routing problem with time windows, **Transportation Science, Vol. 31**, 170-186.
Talbi, E.G., Roux, O., Fonlupt, C. and Robillard, D. (2001) Parallel ant colonies for the quadratic assignment problem, **Future Generation Computer Systems, Vol. 17, No. 4**, 441-449.
Tan, K.C., Lee, L.H. and Ou, K. (2001) Artificial intelligence heuristics in solving vehicle routing problems with time window constraints, **Engineering Applications of Artificial, Vol. 14**, 825-837.
Thangiah, S.R., Osman, I.H., Vinayagaoorthy, R. and Sun, T. (1994) Algorithms for vehicle routing problems with time deadlines, **American Journal of Mathematical and Management Science, Vol. 13**, 323-355.
Thangiah, S.R., Potvin, J.Y. and Sun, T. (1996) Heuristics approaches to vehicle routing with backhauls and time windows, **Computers and Operations Research, Vol. 23**, 1043-1057.
Ting, C.J. and Huang, C.H. (2004) An improved genetic algorithm for vehicle routing problem with time windows, **International Journal of Industrial Engineering**, Forthcoming.