Ant colony system for a VRP with multiple time windows and multiple visits

Daniela Favaretto *

Elena Moretti †

Paola Pellegrini ‡

Department of Applied Mathematics

University Ca' Foscari of Venice

Dorsoduro 3825/E

I-30123 Venice

Italy

Abstract

The Vehicle routing problem with time windows is frequently found in literature, while multiple time windows are not often considered. In this paper a mathematical formulation of the vehicle routing problem with multiple time windows is presented, taking into account periodic constraints. An algorithm based on Ant Colony System is proposed and implemented. Computational results related to a purpose-built benchmark are finally reported.

Keywords and phrases : *Logistics, vehicle routing problem, multiple time windows, ant colony system.*

1. Introduction

The *Vehicle Routing Problem with Time Windows* (*VRPTW*) is defined as the problem of minimizing costs when a fleet of homogeneous vehicles has to distribute goods from a depot to a set of customers satisfying time windows and capacity constraints [4]. The objective of the problem is usually to minimize the total length of the subtours.

* *E-mail*: `favaret@unive.it`

† *E-mail*: `emoretti@unive.it`

‡ *E-mail*: `paolap@pellegrini.it`

Journal of Interdisciplinary Mathematics
Vol. 10 (2007), No. 2, pp. 263–284

A variant of the problem is proposed in this paper: the difference between the two problems consists in the objective functions and in the presence of some constraints. In the formulation here presented, the total weighted time, traveling and waiting time, has to be minimized, satisfying constraints related to multiple time windows and periodicity.

The *VRPTW* has been solved in literature both with exact [11], heuristic [4], [6], [15] and meta-heuristic [1], [2] algorithms.

In this paper, an algorithm based on Ant Colony System is proposed and implemented for the problem considered. Some experiments are presented, tackling instances of the *Vehicle Routing Problem with Multiple Time Windows* (*VRPMTW*) of a benchmark derived from literature ([11]) and modified for taking into account the presence of multiple time windows for which, as far as we know, no results are currently available.

In Section 2 the Vehicle Routing Problem with Multiple Time Windows and periodic constraints is formulated. Section 3 deals with the approach of Ant Colony System while Section 4 gives the description of the procedure proposed. In Section 5 computational experience related to a purpose-built benchmark is reported and discussed. Finally, conclusions and hints for future research are presented in Section 6.

2. Mathematical formulation

Consider a *Vehicle Routing Problem with Time Windows* (*VRPTW*) having the following features:

- each customer has multiple time windows, the number of which varies from 1 to TW;
- each customer may require to be served many times during the period considered, $[0, T]$;
- each subtour cannot be longer than a fixed value P.

The problem can be formulated as a *VRPTW* [6] with the addition of some variables related to the multiplicity of time windows and some specific constraints taking into account the request of periodic visits during the period. The formulation proposed is inspired by the one presented in [5].

Let $G = (V, E)$ be a graph where V is the set of nodes and E is the set of not oriented edges. More precisely, $V = \{0\} \cup M$, where 0 is the depot and $M = \{1, 2, ..., m\}$ is the set of customers, and $E = (\{0\} \times M) \cup (M \times M)$, where $M \times M$ is the set of edges connecting the customers and $\{0\} \times M$ is the set of edges connecting the customers and the depot.

A positive demand q_i and a set of w_i time windows $[e_i^v, l_i^v] \subset [0, T]$, $1 \leq v \leq w_i$, $w_i \leq TW$, such that $e_i^v < l_i^v$ and $l_i^v < e_i^{v+1}$, $v \in \{1, 2, \ldots, w_i - 1\}$, are associated to each customer $i \in M$.

Let $M' \subseteq M$ be the set of customers requiring to be served more than once in the period $[0, T]$. To every customer belonging to M' is associated one dummy customer for each requested visit, except the first one. To each dummy customer, time windows equal to the ones of the real customer which they are referred to, are associated.

Let Z be the set of dummy customers, $N = Z \cup M$ and $A = (\{0\} \times N) \cup (N \times N)$. In this way all the customers in N must be served only once in the considered period choosing one of the requested time windows in a suitable way as will be discussed in the following.

For each $i \in M'$ and $j \in M' \cup Z$ define $o_{i,j}$ in the following way:

$$o_{i,j} = \begin{cases} 1, & \text{if } j \text{ is a dummy customer associated to } i \text{ or } i = j, \\ 0, & \text{otherwise.} \end{cases}$$

Moreover, for each $i \in M'$ define π_i such that:

$$\pi_i = \begin{cases} 1, & \text{if customer } i \text{ requires visits having suitable time distance,} \\ 0, & \text{otherwise.} \end{cases}$$

Two visits are defined to have a suitable time distance if the width of the interval between them is at least equal to $\left\lfloor \frac{T}{\sum_{j \in N} o_{i,j}} \right\rfloor$; this value is the bigger integer not greater than the ratio between the time horizon and the total number of visits that customer i requires. A customer is allowed to impose this restriction only if he indicates appropriate time windows.

To each edge $(i, j) \in A$ is associated a weight $t_{i,j}$, representing the time required to travel from node i to node j increased by the time necessary for the service at i, if i is different from 0.

In the time interval $[0, T]$, H_0 disjoint subperiods are considered, which may represent for instance different days in a planning horizon corresponding to a week. Let $H = \{1, 2, \ldots, H_0\}$ be the set of such subperiods. Moreover, it is required that each subtour begins and finishes in the same subinterval.

Let C be the cost of a waiting time unit and K the set of available vehicles each having capacity Q.

Let $x_{i,j}^{h,k}$ and u_i^v be binary variables having the following meaning:

$$x_{i,j}^{h,k} = \begin{cases} 1, & \text{if the vehicle } k \text{ visits customer } j \text{ immediately} \\ & \text{after customer } i \text{ in the subperiod } h, \\ 0, & \text{otherwise} \end{cases}$$

and

$$u_i^v = \begin{cases} 1, & \text{if customer } i \text{ is served in his } v\text{-th time window,} \\ 0, & \text{otherwise,} \end{cases}$$

$(i, j) \in A$, $k \in K$, $h \in H$ and $v \in \{1, \ldots, w_i\}$.

Let S_i be the instant of time in which the service and W_i the waiting time concerning customer i, $i \in N$.

The model can be formulated as follows

$$\min \sum_{h \in H} \sum_{k \in K} \sum_{(i,j) \in A} t_{i,j} \cdot x_{i,j}^{h,k} + C \cdot \sum_{i \in N} W_i + F \sum_{k \in K} \sum_{h \in H} \sum_{j \in N} x_{0,j}^{h,k}, \tag{1}$$

$$\text{subject to} \sum_{h \in H} \sum_{k \in K} \sum_{j \in N} x_{i,j}^{h,k} = 1, \quad i \in N, \tag{2}$$

$$\sum_{h \in H} \sum_{k \in K} \sum_{i \in N} x_{i,j}^{h,k} = 1, \quad j \in N, \tag{3}$$

$$\sum_{k \in K} \sum_{j \in N} x_{0,j}^{h,k} \leq |K|, \quad h \in H, \tag{4}$$

$$\sum_{k \in K} \sum_{i \in N} x_{i,0}^{h,k} = \sum_{k \in K} \sum_{j \in N} x_{0,j}^{h,k}, \quad h \in H, \tag{5}$$

$$\sum_{v=1}^{w_i} u_i^v = 1, \quad i \in N, \tag{6}$$

$$\sum_{v=1}^{w_i} u_i^v \cdot e_i^v \leq S_i, \quad i \in N, \tag{7}$$

$$\sum_{v=1}^{w_i} u_i^v \cdot l_i^v \geq S_i, \quad i \in N, \tag{8}$$

$$S_i + t_{i,j}^k + W_j - S_j \leq B(1 - x_{i,j}^{h,k}), \quad (i, j) \in N \times N,\ k \in K,\ h \in H, \tag{9}$$

$$S_i + t_{i,j}^k + W_j - S_j \geq B(x_{i,j}^{h,k} - 1), \quad (i, j) \in N \times N, k \in K, h \in H, \tag{10}$$

$$\sum_{i \in N \cup \{0\}} \sum_{j \in N} q_j \cdot x_{i,j}^{h,k} \leq Q, \quad k \in K,\ h \in H\,, \tag{11}$$

$$u_z^v - u_j^v \leq 1,\ 1 \leq v \leq w_i, i \in M',\ z, j \in M' \cup Z : \\ o_{i,z} = 1, o_{i,j} = 1, z \neq j\,, \tag{12}$$

$$\left| \sum_{v=1}^{w_z} u_z^v \cdot e_z^v - \sum_{v=1}^{w_j} u_j^v \cdot e_j^v \right| \geq \left\lfloor \frac{T}{\sum_{j \in N} o_{i,j}} \right\rfloor \pi_i\,, \\ i \in M', z, j \in M' \cup Z : o_{i,z} = 1,\ o_{i,j} = 1, z \neq j\,, \tag{13}$$

$$S_j - S_i - P \leq B(2 - x_{0,i}^{h,k} - x_{j,0}^{h,k})\,, \\ (i,j) \in N \times N, i \neq j,\ k \in K,\ h \in H\,, \tag{14}$$

$$x_{i,j}^{h,k} \in \{0,1\}, \quad (i,j) \in A,\ k \in K\,, \tag{15}$$

$$u_i^v \in \{0,1\}, \quad i \in N,\ 1 \leq v \leq w_i\,, \tag{16}$$

$$W_i \geq 0, \quad i \in N\,, \tag{17}$$

$$S_i \geq 0, \quad i \in N\,. \tag{18}$$

Constraints (2) and (3) restrict the assignment of each customer to exactly one vehicle route. Constraint (4) means that at most a fixed number $|K|$ of vehicles in each time interval h can leave the depot. Constraint (5) implies that the number of vehicles which have left the depot is equal to the number of vehicles coming back to the depot in each time interval h. Constraints (6), (7), (8), (9), (10) and (11) ensure the schedule feasibility with respect to time considerations and capacity constraints. B is an arbitrary large value. Constraint (12) implies that dummy customers of the same real customer are not visited in the same time window. Constraint (13) means that different visits of the same real customer can't be chosen arbitrarily if he required them to be suitably separated (i.e., if $\pi_i = 1$): the time distance between two visits must be greater than or equal to a fixed value depending on the total number of required services and on the length of the time period considered, as previously explained. It has to be remarked that if $\pi_i = 0$ for some $i \in M'$ the constraint is trivially satisfied. Constraint (14) guaranties that each circuit belongs to only one time interval h and that its total time length is less than or equal to a fixed value P. Binary conditions and nonnegative constraints on the variables are expressed by the last constraints (from (15) to (18)).

The objective function (1) is the sum of the weighted routing and waiting time and the fixed cost of the vehicles used. Since the cost of using an additional vehicle (F) is quite high compared with the other transportation costs, setting the value of F big enough the incidence of the third term is much stronger than the one of the other ones. The need of consideration of the waiting time follows immediately the choice of minimization of the temporal duration of the tours. In many real situations a main objective is the minimization of the total time spent by the drivers for completing the services: a tour which is shorter in terms of distance traveled and implying some hours more than another one is often not preferable.

3. Ant colony system

Ant Colony System is an *Ant Colony Optimization* (ACO) algorithms, where ACO is a meta-heuristic which studies a set of artificial ants cooperating to the solution of an optimization problem, by the exchange of information via pheromone deposited on graph edges [9].

Among the problems strictly related to the one considered in this paper, the first one to which this method has been applied is the *Traveling Salesman Problem* (*TSP*) [7], [8], [16], then algorithms have been proposed for *VRP* [1], [2] and *VRPTW* [12].

The aim of this paper is to propose an algorithm to solve the *VRPMTW*[1] formulated in the previous section, using the Ant Colony System method.

Following the literature ([12]), the Vehicle Routing Problem is transformed into a *TSP* considering a number of depots equal to the number of vehicles which must be used to serve the customers. The initial graph G presented in the previous section is modified adding as many nodes as is the number of vehicles used minus one, moreover the arcs between each fictitious depot and each customer are duplicated and the distances between copies of the depot are set equal to infinity. To each edge two weights are associated: $\tau(i, j)$, called pheromone, which is modified at each iteration by artificial ants, and $\eta(i, j)$, the value of which depends on the length of the edge.

At the beginning f ants are located in the same depot. Each ant generates a complete tour by choosing the nodes according to a probabilistic transition rule. Ants prefer to move to nodes connected by shortest edges with high amount of pheromone. When the ants move the level

[1]For ease of exposure in the following the problem formulated in Section 2 will be referred to as *Vehicle Routing Problem with Multiple Time Windows* (or *VRPMTW*), not including in this denomination periodic constraints that are nonetheless a relevant part of it.

of pheromone on the edges used is modified using a local updating rule. Once all the ants have completed their tours a global pheromone updating rule is applied, increasing the pheromone level on the edges which belong to the current best tour. Then the whole process is repeated.

ACS ends when one of the following conditions becomes true: a fixed number of solutions has been generated, a fixed computational time has elapsed or a fixed number of iterations with no improvement of the objective function has been performed.

The state transition rule shows how ant k in node i chooses node j to move to. Let q_0 be a fixed parameter $(0 \leq q_0 \leq 1)$ and q be a random number uniformly distributed in $[0, 1]$.

If $q \leq q_0$ ant k in node i chooses node j such that

$$j = \arg \max_{u \in J_k(i)} \{[\tau(i, u)] \cdot [\eta(i, u)]^{\beta}\}, \tag{19}$$

where

$\tau(i, u)$ is the level of pheromone associated to the edge (i, u);

$\eta(i, u)$ is a function of the length of the edge (i, u);

β is a parameter which determines the relative importance of η versus pheromone;

$J_k(i)$ is the set of nodes which can be visited by ant k leaving from i.

Remark that the index k associated to ants is completely different from the one associated to vehicles and this will not generate confusion in the following.

If $q > q_0$, the choice of ant k in node i is random. Each node j has a probability to be chosen equal to

$$p_k(i, j) = \begin{cases} \dfrac{[\tau(i, j)] \cdot [\eta(i, j)]^{\beta}}{\sum\limits_{u \in J_k(i)} [\tau(i, j)] \cdot [\eta(i, u)]^{\beta}}, & \text{if } j \in J_k(i), \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

This means that, if q is greater than q_0, each node has a probability of being chosen proportional to its desirability.

If q is greater than q_0, this process is called *exploration*, otherwise it is called *exploitation*.

When f hamiltonian circuits are determined, the pheromone level is modified on all edges by a global modification. In this context, only the ant that has found the shortest route deposits pheromone on the edges it went through, so that the choices of the following agents will be positively affected by those ants which have obtained the best solutions.

4. Description of the approach

The approach used to tackle the Vehicle Routing Problem with Multiple Time Windows has been called *MACS-VRPMTW*, where *MACS* means Multiple Ant Colony System.

The algorithm is quite similar to the one proposed in [12], where two types of colonies of ants minimize simultaneously two different objective functions: the number of vehicles and the total cost. More precisely, the first colony must determine a feasible solution, if it exists, with a fixed number of vehicles; such colony will be called *ACS-VEI*. The second one tries to improve the solution found with the minimum number of vehicles; such colony will be called *ACS-TIME*. This procedure is slightly different from the one proposed in [9].

These two kinds of colony work in a very similar way: analyzing each node with respect to the constraints imposed by the model (capacity of each vehicle, time windows, etc.) each ant builds a list of feasible movements and chooses the one indicated by a probabilistic rule similar to the one described in Section 3.

The first step of the algorithm finds a feasible solution by an heuristic based on nearest neighbor [15]. A feasible solution is represented by a list of nodes starting from a depot and alternating customer-nodes and depot-ones; the last element of the list is the last node touched before the last vehicle goes back to the depot. In this way the number of depots is exactly equal to the number of vehicles required and each subroute starts with a depot and ends with a customer. On the other hand the value of the objective function includes also the time needed for each vehicle to go back to the initial depot.

Let ψ^{gb} be the current optimal solution (globally best) and s be the number of vehicles used.

The second step requires that an *ACS-VEI* colony of ants is activated to find a feasible solution with $s - 1$ vehicles. The search will be repeated decreasing of one unit the number of vehicles used at each iteration, until no feasible solution can be found.

In this way the algorithm determines the minimum number of vehicles with which ants are able to find a feasible solution given the established stop-criteria. With this number of vehicles the *ACS-TIME* colony is activated to find the shortest route.

It may happen that in this phase an ant finds a route that has as the last node a depot, which means that it has found a solution with one vehicle less. In this case this last depot-node is dropped and the ants will work with the new graph obtained. In this situation, *ACS-TIME* may find a solution which uses a number of vehicles smaller than the one used by *ACS-VEI*.

4.1 *Set of feasible customers and time windows*

Let $J_k(i)$ be the set of feasible customers if the last node visited by ant k is i. If i is a depot, $J_k(i)$ will be the set of all customers not yet visited. If i is a customer, $J_k(i)$ will be the set of all not visited customers j such that:

- the capacity constraint is satisfied;
- there exists v, $1 \leq v \leq w_j$, such that $[e_j^v, l_j^v]$ belongs to the subperiod in which the subtour has begun and l_j^v is greater than or equal to the instant in which j can be reached;
- the subtour obtained inserting customer j has a duration smaller than or equal to P.

If this set is empty and at least $\dfrac{3}{4}$ of the capacity of a vehicle is used, $J_k(i)$ contains also all non visited depots.

4.2 *State transition rule*

The state transition rule used in the algorithm is a variant of the one described in Section 3. In this paper, in which multiple time windows are considered, a new parameter ϑ is introduced to express a measure of desirability taking into account the number of time windows which are successive to the current instant. This parameter will be defined in the next subsection.

If $q \leq q_0$ ant k in node i chooses node j such that

$$j = \arg\max_{u \in J_k(i)} \{[\tau(i,u)] \cdot [\eta(i,u)]^\beta \cdot [\vartheta(i,u)]^\gamma\}, \tag{21}$$

where

$\vartheta(i,u)$ is a function of the width and of the number of the available time windows of node u being in node i;

ν is a parameter which determines the relative importance of ϑ versus pheromone τ and the heuristic measure η.

If $q > q_0$, the choice of ant k in node i is random. Each node j has a probability to be chosen equal to

$$p_k(i,j) = \begin{cases} \dfrac{[\tau(i,j)] \cdot [\eta(i,j)]^\beta \cdot [\vartheta(i,u)]^\nu}{\sum\limits_{u \in J_k(i)} [\tau(i,u)] \cdot [\eta(i,u)]^\beta \cdot [\vartheta(i,u)]^\nu}, & \text{if } j \in J_k(i), \\ 0, & \text{otherwise.} \end{cases} \tag{22}$$

4.3 *Evaluation of* η

The value of $\eta(i,j)$ takes into account not only the time t_{ij} necessary to go from node i to node j, but also the urgency of serving customer j, given by the time interval between the present moment and the one in which the chosen time window closes; moreover, the number of times in which node j has not been touched in the previous (unfeasible) routes is considered.

More precisely,

$$\eta(i,j) = \frac{1}{\max\{1, \gamma\delta_{ij}(\hat{l}_j - \text{now}) - IN_j^3\}}, \tag{23}$$

where

now is the current time,

$\hat{e}_j$ and $\hat{l}_j$ are the beginning and the end of the time window chosen for visiting customer j,

$\delta_{ij} = \max\{\text{now} + t_{ij}, \hat{e}_j\}$-now is the width of the time interval elapsing before the beginning of the service to customer j,

IN_j is the number of times which node j has not been inserted in a tour,

γ is a scale factor used to have homogeneous quantities.

Formula (23) for $\eta(i,j)$ is the same as the one proposed by Gambardella *et al.* in [12], a part from the presence of the coefficient γ and from the raising to the third power of IN_j, which is introduced to privilege the choice of customers difficult to visit.

4.4 *Evaluation of* ϑ

The value of $\vartheta(i,j)$ depends on the time still available to visit node j being in node i at the instant now and on the number of time windows of customer j. More precisely,

$$\vartheta(i,j) = \begin{cases} 1, & \text{if } SW_j = \emptyset, \\ \left(\max\left\{1, \xi\left[\sum_{v \in SW_j} (l_j^v - e_j^v)\right] \cdot (l_j^{\max} - \text{now} - t_{ij}) - w_j^3\right\}\right)^{-1}, & \\ \quad \text{otherwise}, & \end{cases} \tag{24}$$

where

$SW_j = \{v \in \{1, \ldots, w_j\} \mid l_j^v > \hat{l}_j\}$ is the set of the time windows subsequent the one chosen,

$l_j^{\max}$ is the end of the last time window of customer j,

ξ is a scale factor introduced to have homogeneous quantities.

Remark that if there is no time windows successive to the current instant, the value of ϑ must be set equal to 1 so that the criterion on which the next node is chosen relies completely only on η and τ. This heuristic measure allows to consider the multiplicity of time windows: the urgency of node j depends both on the number and on the width of the time windows not yet closed.

4.5 *Pheromone updating rules*

Pheromone updating rules are the following:

- *local updating rule*

$$\tau(i,j) = (1-\rho) \cdot \tau(i,j) + \rho\tau_0,$$

with

ρ parameter such that $0 < \rho < 1$,

τ_0 initial level of pheromone;

- *global updating rule*

$$\tau(i,j) = (1-\alpha) \cdot \tau(i,j) + \alpha\Delta\tau(i,j),$$

with

$$\Delta\tau(i,j) = \begin{cases} (L_{\psi_{gb}})^{-1}, & \text{if } (i,j) \in \psi^{gb}, \\ 0, & \text{otherwise,} \end{cases}$$

α parameter such that $0 < \alpha < 1$;

$L_{\psi^{gb}}$ length of the globally best tour ψ^{gb}.

Each time a new *ACS-VEI* colony is activated the pheromone level is set equal to τ_0 on each edge, where $\tau_0 = \frac{1}{|N| \cdot L_{NN}}$ with L_{NN} length of the solution found with the nearest neighbor algorithm; on the other hand, each time a colony completes its task the pheromone level present on each edge is recorded.

Once the minimum number of vehicles necessary to the algorithm to find a feasible solution is determined, before activating the *ACS-TIME* colony the trail of pheromone present on each edge is updated. The aim of this operation is to go back to the situation that led the ants to the construction of a feasible solution with the minimum number of vehicles found. Computational experiments have shown that in this way the ants of the *ACS-TIME* colony are able to find a feasible solution much faster, and so they have more time and more iterations to try to improve it. Remark that this procedure is different from the one used in [12], where the level of pheromone is initialized before activating *ACS-TIME*.

4.6 *The algorithm*

In Figures from 1 to 4 the algorithm is depicted.

In *MACS-VRPTW* ([12]), the *ACS-VEI* and *ACS-TIME* colonies are activated in a different way. For every feasible solution found by *ACS-VEI* with s vehicles, *ACS-TIME* searches a better solution with s vehicles. The procedure is repeated decreasing the number of vehicles used. In the algorithm proposed, *ACS-TIME* is started only once, when the final number of vehicles has been identified by *ACS-VEI*.

5. Computational results

The algorithm above described was coded in Visual Basic and run on an Athlon XP 1600, 1.39 Ghz.

Referring to the results obtained in [10] the values chosen for α, ρ and q_0 are the following:

$$\alpha = 0.1\,, \quad \rho = 0.9\,, \quad q_0 = 0.9\,.$$

Figure 1

The algorithm

1. /* Initialization */
    /* ψ^{gb} *is the best feasible solution found,*
    ψ^k *is current solution,*
    $\psi^{ACS\text{-}VEI}$ *is the best solution found with the current number of vehicles,*
    N *is the set of nodes of the graph,*
    s is the number of vehicles found by nearest neighbor,
    L_{NN} *is the total cost of the tour obtained by nearest neighbor,*
    L_{gb} *is the total cost of the best tour found,*
    L_{ψ_k} *is the total cost of the tour found by ant k* */
Initialize variables
Call Nearest Neighbor
Create s depot-nodes

2. /* Main Loop */
Do
    Delete a depot-node
    Call *ACS-VEI*
While *ACS-VEI* finds a feasible solution
Create a depot-node
Call *ACS-TIME*

Figure 2

***ACS-VEI* procedure**

1. /* *Initialization of pheromone trail* */

$$\tau_{ij} = \frac{1}{|N| \cdot L_{NN}}, \ \forall \ (i, j) \in \psi^{gb}$$

Do
  2. For each ant k
      Call Tour Building Procedure
      If # visited_customers $(\psi^k) = |N|$ Then
        $\psi^{gb} = \psi^k$
        $\tau_{\text{record}}(i, j) = \tau(i, j)$
        Exit Procedure
      Else
        If # visited_customers $(\psi^k) >$ # visited_customers $(\psi^{ACS\text{-}VEI})$ Then
          $\psi^{ACS\text{-}VEI} = \psi^k$
        End If
      End If
      For each node $j \notin \psi^k$
        $IN_j = IN_j + 1$
    End for each
  End for each

  3. /* *Global pheromone updating with* ψ^{gb} */

$$\tau(i, j) = \begin{cases} (1-\alpha) \cdot \tau(i, j) + \alpha (L_{\psi^{gb}})^{-1}, & \text{if } (i, j) \in \psi^{gb}, \\ (1-\alpha) \cdot \tau(i, j), & \text{otherwise} \end{cases}$$

    /* *Global pheromone updating with* $\psi^{ACS\text{-}VEI}$ */

$$\tau(i, j) = \begin{cases} (1-\alpha) \cdot \tau(i, j) + \alpha (L_{\psi^{ACS\text{-}VEI}})^{-1}, & \text{if } (i, j) \in \psi^{ACS\text{-}VEI}, \\ (1-\alpha) \cdot \tau(i, j), & \text{otherwise} \end{cases}$$

While a stop criterion is met

Figure 3

***ACS-TIME* procedure**

```
1. /* Initialization of pheromone trail */
τ(i, j) = τ_record(i, j)

Do
  2. For each ant k
     Call Tour Building Procedure
     If # visited_customers (ψ^k) = |N| Then
       If the last visited node is a depot-node Then
         Delete that depot-node
         ψ^gb = ψ^k
       Else
       If L_{ψ^k} < L_{ψ^gb} Then
         ψ^gb = ψ^k
       End If
     End If
   End If
  End for each

  3. /* Global pheromone updating */
```

$$\tau(i,j) := \begin{cases} (1-\alpha)\cdot\tau(i,j) + \alpha(L_{\psi^{gb}})^{-1}, & \text{if } (i,j)\in\psi^{gb}, \\ (1-\alpha)\cdot\tau(i,j), & \text{otherwise} \end{cases}$$

While a stop criterion is met

Moreover the values chosen for the other parameters are:

$f = 80$,
maximum number of iterations $= 10000$,
maximum computational time allowed $= 250$,
maximum number of iterations without improvement $= 155$.

Preliminary tests have been performed for different values of β and ν, the exponents of η and ϑ in the state transition rule. The values tested were $\beta \in \{0, 0.5, 1, 5, 10, 20, 30\}$ and $\nu \in \{0, 1, 2, 4, 5, 6, 7\}$ and all the combinations were checked. The selected quantities are those that reached the best results in terms of average relative error on a set of instances constructed as explained in Section 5.1:

$$\beta = 20\,, \quad \nu = 2\,.$$

Observing the state transition rule reported in formulas 21 and 22, it is observable that these two measures imply that the accent is posed first on the pheromone level on the arcs, which has implicitly exponent 1, then to the heuristic ϑ and finally on η (since these values are all smaller than one, the higher is the exponent, the smaller is the relevance attributed to them).

The values of the coefficients γ and ξ have been fixed equal to 1.

Figure 4

The tour building procedure

```
/* TourBuildingProcedure for the ant k */

Initialize ψ_k
Repeat
  1. /* Definition of compatible customers */
    For each not visited customer-node j
      If S_i + t_{i,j} ≤ l_j &
        total time from last depot ≤ P &
        the visit is sufficiently distant from the others to the same customer, if required, &
        capacity constraint is satisfied Then
          j is compatible
          compute η(i, j) and ϑ(i, j)
      End if
    End for each
    For each not visited depot-node j
      If no compatible nodes has been found or
        cumulated demand ≥ 3/4 · capacity Then
          j is compatible
          compute η(i, j) and ϑ(i, j)
      End if
    End for each
  If there are compatible customers Then
    2. /* Choice, of the node to visit */
        Draw q
        If q ≤ q_0 Then
          exploitation
      Else
          exploration
      End if
      Insert j in ψ^k
      Update remaining capacity
    3. /* Local pheromone updating */
      τ(i, j) := (1 − ρ) · τ(i, j) + ρΔτ(i, j)
  End if
Until no compatible customers are found
```

5.1 *Construction of the benchmark*

Since as far as we know it is not possible to find in literature results related to a benchmark for *Vehicle Routing Problem with Multiple Time Windows* (*VRPMTW*), we have created a set of instances for which the optimal solution is known. To this aim we have considered the two instances reported in literature by Fisher [11] for the *VRP* without time windows and with a fixed number of identical vehicles, not considering customers requiring periodic visits. In both cases the fixed number of vehicles is the minimum possible, considering the capacity of the vehicles and the total demand.

The *VRP* without time windows has been transformed into a *VRP* with multiple time windows in the following way: some time windows have been created such that the instant in which the customers are served in the optimal solution of *VRP* is contained in at least one of the time windows generated. In this way, using the same number of vehicles, the optimal solution of *VRP* without time windows is feasible and optimal for *VRP* with multiple time windows. Moreover it has to be remarked that in the optimal solution the waiting time in each node is zero which makes it optimal also for the objective function (1).

For each instance reported by Fisher [11] 28 instances have been created using the algorithm depicted in Figure 5. In this way 8 instances for each of the following kinds are obtained:

- one time window per customer with width from 100 to 300 minutes (instances 1-4, 29-32),
- one time window per customer with width from 50 to 100 minutes (instances 5-8, 33-36),
- at most two time windows per customer with width from 70 to 100 minutes, separated by a distance which varies from 50 to 100 minutes (instances 9-12, 37-40),
- at most four time windows per customer with width from 30 to 200 minutes, separated by a distance which varies from 50 to 100 minutes (instances 13-16, 41-44),
- at most five time windows per customer having width from 50 to 100 minutes, separated by a distance which varies from 15 to 50 minutes (instances 17-20, 45-48),
- at most six time windows per customer having width from 50 to 200 minutes, separated by a distance which varies from 30 to 100 minutes (instances 21-24, 49-52),
- at most ten time windows per customer having width from 40 to 80 minutes, distant from 15 to 50 minutes (instances 25-28, 53-56).

The number of vehicles used in both sets of instances is 4, and the service time is set equal to 0. The maximum duration of each tour is chosen in order to have the feasibility of tours implying a total time equal to the triple of the optimal one.

Figure 5
Creation of instances

```
/* W is the maximum number of time windows per customer, W_j is the number of time windows
assigned to customer j, arrival_j is the arrival instant according to the optimal tour,
dist_max and dist_min are the maximum and the minimum distance, between two consecutive
time windows of the same customer, width_max and width_min are the maximum and minimum width
allowed for a time window, q, d, w, n and a are random number related to the number of time windows,
the distance between time windows, the width of time windows, the time window and the instant in which
arrival_j falls, respectively */

For each node j ∈ N
  Draw q
  W_j = Round(q · W)
  Calculate arrival_j
  Draw a and n
  e_j^{n·W_j} = Round(arrival_j − {a · [width_min + Round(w · [width_max − width_min])]})
  l_j^{n·W_j} = Round(arrival_j + {[1 − a] · [width_min + Round(w · [width_max − width_min])]})
  For v = 1 to n · W_j − 1
    Draw w and d
    l_j^v = e_j^{v+1} − dist_min + Round(d · [dist_max − dist_min])
    e_j^v = l_j^v − width_min + Round(w · [width_max − width_min])
  Next v
  For v = n · W_j to W_j
    Draw w and d
    e_j^v = l_j^{v−1} + dist_min + Round(d · [dist_max − dist_min])
    l_j^v = e_j^v + width_min + Round(w · [width_max − width_min])
  Next v
  For v = 1 to W_j
    If e_j^v < 0 Then
      e_j^v = 0
    End If
    If l_j^v < 0 Then
      l_j^v = 0
    End If
  Next v
End for each
```

5.2 *Results of the benchmark*

In Table 1 the results of the computational experiences are reported. The optimal value function and the number of nodes for the instances numbered from 1 to 28 are 614 and 71 respectively and for the instances numbered from 29 to 56 are 991 and 44, respectively. The first column contains the number of instance. In the second one the value of the objective function obtained as the best result between two runs using *MACS-VRPMTW* is reported. Columns four, six and eight contain the values of the objective function obtained respectively by nearest neighbor and by setting $\beta = 0$ and $\nu = 2$ and $\beta = 20$ and $\nu = 0$ in *MACS-VRPMTW*. Remark that when $\beta = 0$ the transition rule takes into account only τ and ϑ, while if $\nu = 0$ the transition rule takes into account only τ and η. In columns three, five, seven and nine the relative error referred to the previous ones is computed. It has to be remarked that when the time windows are multiple, the use of η and ϑ together gives better results, except for only one instance, compared with the ones obtained using separately η or ϑ (setting $\beta = 0$ or $\nu = 0$, respectively).

In Table 2 the results about the relative error are reported.

From the computational experience it can be argued that the time elapsed increases with the maximum number of time windows and of course with the number of customers; moreover the ratio between the two values referred to the two kinds of instance is almost equal to 2, which can be interpreted as a kind of proportionality between computational time and number of customers.

6. Conclusions

The construction of the instances taken as benchmark is certainly unusual, but in absence of more classical source of data, the device used has made it possible to measure the performance of the algorithm.

It has to be pointed out that in all the considered instances the number of ants used is 80, which is high with respect to the number of nodes. In fact, as it was underlined in [10], the larger is the number of ants, the greater is the effect of the local updating rule of the pheromone and, then, the greater is the number of explored solutions. In this way, it may be that it is easier to jump out of a local minimum, even if of course this is not assured. On the other hand, the possibility of falling in a local minimum is a typical problem of many meta-heuristics.

Even if it is not possible to judge the results obtained in an absolute way, it can be observed that the introduction of the heuristic measure ϑ and its combination with the modification of the formulation of η have a strong impact on the performances of the algorithm, allowing a significant reduction of the average relative error in the instances considered. These elements, then, represent a satisfactory first step that may open the way to further analysis of the Vehicle Routing Problem with Multiple Time Windows using the ACO meta-heuristics. Possible developments can be the hybridization of the algorithm with a local search procedure and the comparison of its performances with those of other approaches.

Table 1

Results of the benchmark

Instance	MACS VRPMTW $\beta=20, \nu=2$	Relative error	Nearest Neighbor	Relative error	MACS VRPMTW $\beta=0, \nu=2$	Relative error	MACS VRPMTW $\beta=20, \nu=0$	Relative error
1	761	23.99	1303	112.30	1292	110.51	761	23.99
2	727	18.45	1296	111.16	760	23.83	727	18.45
3	727	18.45	1381	125.01	1366	122.57	727	18.45
4	692	12.75	1385	125.66	1376	124.19	692	12.75
5	807	31.49	1099	79.06	1043	69.94	807	31.49
6	770	25.46	1076	75.31	1101	79.39	770	25.46
7	798	30.02	1120	82.48	1123	82.97	1123	82.97
8	736	19.92	1080	75.97	1093	78.08	1093	78.08
9	878	43.05	1177	91.77	878	43.05	1158	88.68
10	905	47.45	1326	116.05	905	47.45	1263	105.78
11	808	31.65	1301	111.97	1263	105.78	1263	105.78
12	883	43.87	1308	113.12	1299	111.65	1299	111.65
13	889	44.85	1351	120.12	1336	117.68	1336	117.68
14	886	44.36	1286	109.53	1277	108.06	1277	108.06
15	842	37.19	1285	109.37	1272	107.25	1272	107.25
16	912	48.59	1258	104.97	1229	100.24	1229	100.24
17	806	31.32	1357	121.10	1164	89.65	1164	89.65
18	839	36.70	1192	94.21	1183	92.75	944	53.81
19	804	31.00	1173	91.12	1164	89.65	1164	89.65
20	859	39.96	1206	96.50	1188	93.56	1188	93.56
21	866	41.10	1307	112.95	1301	111.97	1301	111.97
22	848	38.17	1291	110.35	1282	108.88	1282	108.88
23	852	38.82	1264	105.95	1240	102.04	1240	102.04
24	881	43.54	1342	118.65	1329	116.54	1329	116.54
25	841	37.03	1108	80.53	1094	78.25	1094	78.25
26	825	34.42	1103	79.71	1086	76.94	1086	76.94
27	838	36.54	1210	97.15	1201	95.68	1201	95.68
28	751	22.36	1152	87.70	1130	84.11	838	36.54
29	1215	22.60	2048	106.66	1982	100.00	1210	22.10
30	1055	6.46	2070	108.88	2017	103.54	1265	27.65
31	1012	2.12	2125	114.43	2072	109.09	1012	2.12
32	1081	9.08	2030	104.85	2013	103.13	1260	27.15
33	1111	12.11	1752	76.79	1830	84.67	1194	20.49
34	1194	20.49	2159	117.86	1590	60.45	2107	112.62
35	1074	8.38	1891	90.82	1809	82.55	1108	11.81

(*Contd*. Table 1)

Instance	MACS VRPMTW $\beta = 20, \nu = 2$	Relative error	Nearest Neighbor	Relative error	MACS VRPMTW $\beta = 0, \nu = 2$	Relative error	MACS VRPMTW $\beta = 20, \nu = 0$	Relative error
36	1189	19.98	1869	88.60	1816	83.25	1202	21.29
37	1235	24.62	1741	75.68	1731	74.68	1731	74.68
38	1327	33.91	2095	111.41	2060	107.87	2060	107.87
39	1017	2.63	2066	108.48	2013	103.13	2013	103.13
40	1332	34.41	2214	123.41	2162	118.17	2162	118.17
41	1238	24.93	2016	103.43	1989	100.71	1989	100.71
42	1378	39.05	2325	134.62	1849	86.58	2272	129.27
43	1174	18.47	2063	108.18	2045	106.36	1635	64.99
44	1323	33.50	2052	107.07	1999	101.72	1999	101.72
45	1126	13.62	2180	119.98	1641	65.59	2128	114.74
46	1351	36.33	1940	95.77	1890	90.72	1890	90.72
47	1383	39.56	1944	96.17	1926	94.35	1926	94.35
48	1189	19.98	1894	91.12	1764	78.01	1764	78.01
49	1117	12.72	1464	47.73	1335	34.71	1295	30.68
50	1237	24.83	2015	103.33	1962	97.99	1592	60.65
51	1079	8.88	2188	120.79	2058	107.67	2058	107.67
52	1171	18.17	2162	118.17	1821	83.76	2032	105.05
53	1404	41.68	2115	113.42	1843	85.98	1574	58.83
54	1265	27.65	2049	106.76	1996	101.42	1996	101.42
55	1136	14.63	1777	79.32	1725	74.07	1725	74.07
56	1260	27.15	1807	82.34	1754	77.00	1754	77.00

Table 2

Relative error

71 customers	*MACS-VRPMTW* $\beta = 20, \nu = 2$	Nearest neighbor	*MACS-VRPMTW* $\beta = 0, \nu = 2$	*MACS-VRPMTW* $\beta = 20, \nu = 0$
Average relative error	34.02	102.13	91.88	78.22
Maximum relative error	48.59	125.66	124.19	117.68
Minimum relative error	12.75	75.31	23.83	12.75
44 customers	*MACS-VRPMTW* $\beta = 20, \nu = 2$	Nearest neighbor	*MACS-VRPMTW* $\beta = 0, \nu = 2$	*MACS-VRPMTW* $\beta = 20, \nu = 0$
Average relative error	21.36	102.01	89.90	72.82
Maximum relative error	41.68	134.62	118.17	129.27
Minimum relative error	2.12	47.73	34.71	2.12

References

[1] B. Bullnheimer, R. F. Hartl and C. Strauss, An improved ant system for the vehicle routing problem, *Annals of Operations Research*, Vol. 89 (1999), pp. 319–328.

[2] B. Bullnheimer, R. F. Hartl and C. Strauss, Applying the ant system to the vehicle routing problem, in *Meta-heuristics: Advances and Trends in Local Search for Optimization*, S. Voss, S. Martello, I. H. Osman and C. Roucairol (editors), Kluver Academic Publishers, Boston, 1999.

[3] A. Colorni, M. Dorigo and V. Maniezzo, An investigation of some properties of an ant algorithm, in *Proceedings of the Parallel Problem Solving form Nature Conference* (*PPSN92*), Elsevier Publishing, Bruxelles, 1992.

[4] J. F. Cordeau, G. Desaulniers, J. Desrosiers, M. M. Solomon and F. Soumis, The VRP with time windows, in *The Vehicle Routing Problem*, P. Toth and D. Vigo, *SIAM Monographs on Discrete Mathematics and Applications*, SIAM, Philadelpia, 2000.

[5] C. de Jong, G. Kant and A. van Vlient, On finding minimal route duration in the vehicle routing problem with multiple time windows, *Manuscript*, Department of Computer Science, Utrecht University, Netherlands, 1996.

[6] M. Desrochers, J. K. Lenstra, M. W. P. Savelsbrgh and F. Soumis, Vehicle routing with time windows: optimization and approximation, in *Vehicle Routing: Methods and Studies*, B. L. di Golden and A. A. Assad (editors), North-Holland, Amsterdam, 1988.

[7] M. Dorigo and L. M. Gambardella, Ant colonies for the traveling salesman problem, *BioSystems*, Vol. 43 (1997), pp. 73–81.

[8] M. Dorigo and L. M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, *IEEE Transaction on Evolutionary Computation*, Vol. 1 (1) (1997), pp. 53–66.

[9] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, MIT Press, 2004, Massachusetts Institute of Technology, Cambridge.

[10] D. Favaretto, E. Moretti and P. Pellegrini, Ant colony system for variants of traveling salesman problem with time windows, *Technical Report*, Applied Mathematics Department of Ca' Foscari University of Venice, No. 120/2004, 2004, Venice.

[11] M. L. Fisher, Optimal solution of vehicle routing problems using minimum *K*-trees, *Operations Research*, Vol. 42 (1994), pp. 626–642.

[12] L. M. Gambardella, E. Taillard and G. Agazzi, MACS-VRPTW: a multiple ant colony system for vehicle routing problem with time windows, in *New Idea in Optimization*, D. Corne, M. Dorigo and F. Glover (editors), McGraw-Hill, 1999, pp. 63–76.

[13] M. W. P. Savelsberg, Local search in routing problems with time windows, *Annals of Operations Research*, Vol. 4 (1985), pp. 285–305.

[14] R. W. Sinnott, Virtues of the Haversine, *Sky and Telescope*, Vol. 68 (2) (1984), p. 159.

[15] M. M. Solomon, Algorithms for the vehicle routing and scheduling problem with time window constraints, *Operation Research*, Vol. 35 (1987).

[16] T. Stutzle and M. Dorigo, AGO algorithms for the traveling salesman problem, in *Evolutionary Algorithms in Engineering and Computer Science: Recent Advances in Genetic Algorithms, Evolution Strategies, Evolutionary Programming and Industrial Applications*, K. Miettien, M. Mkel, P. Neittaanmki and J. Periaux (editors), John Wiley & Sons, 1999.

Received October, 2005