A Multiple Ant Colony System for the Electric Vehicle Routing Problem with Time Windows

Michalis Mavrovouniotis
KIOS Research and Innovation Center of Excellence
Department of Electrical and Computer Engineering
University of Cyprus
Nicosia, Cyprus.
mavrovouniotis.michalis@ucy.ac.cy

Georgios Ellinas
KIOS Research and Innovation Center of Excellence
Department of Electrical and Computer Engineering
University of Cyprus
Nicosia, Cyprus.
gellinas@ucy.ac.cy

Changhe Li
School of Automation
China University of Geosciences
Hubei Key Laboratory of Advanced Control and Intelligent Automation for Complex Systems
Wuhan, China.
changhe.lw@gmail.com

Marios Polycarpou
KIOS Research and Innovation Center of Excellence
Department of Electrical and Computer Engineering
University of Cyprus
Nicosia, Cyprus.
mpolycar@ucy.ac.cy

***Abstract*—Ant colony optimization (ACO) has been found to be useful on several vehicle routing problem variations. In this work, ACO is applied to the electric vehicle routing problem with time windows (E-VRPTW). The E-VRPTW has a hierarchical multiple objective function, which is to minimize the number of electric vehicles and the total distance traveled. A multiple ACO is applied to E-VRPTW in which two colonies cooperate to minimize the objectives in parallel. A local search is embedded in ACO to improve the quality of the output. The experimental results on a set of benchmark instances show that the multiple ACO is competitive with existing methods.**

***Index Terms*—Electric vehicle, vehicle routing problem with time windows, ant colony optimization**

I. Introduction

Ant colony optimization (ACO) is a metaheuristic inspired by the foraging behavior of real ant colonies [1]. ACO consists of a several artificial ants (or agents) that construct high-quality solutions for difficult optimization problems. The construction of solutions is guided by artificial pheromone trails and some heuristic information based on the optimization problem. The pheromone trails are updated according to the quality of the constructed solutions and they are accessible by all the ants of the colony when constructing solutions. ACO algorithms have shown promising performance in different variations of the vehicle routing problem (VRP), including the capacitated VRP [2], [3], VRP with time windows (VRPTW) [4], [5], dynamic VRP [6], and more recently electric versions of the VRP [7]–[10].

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 739551 (KIOS CoE - TEAMING) and from the Republic of Cyprus through the Deputy Ministry of Research, Innovation and Digital Policy.

In this work, the electric vehicle routing problem with time windows (E-VRPTW) [11] is addressed, which is defined as the problem of minimizing the operational costs for a fleet of EVs in the case where these vehicles must deliver goods to a set of customers, starting from and ending at a central depot. The E-VRPTW minimizes a multiple hierarchical objective function, as is the common practice in the literature when addressing VRPs with time windows constraints [12]. The first objective is to minimize the number of EVs and the second objective is to minimize the total distance traveled. A solution consisting of fewer EVs is always superior to a solution consisting of more EVs, even if the traveled distance of the former solution is higher than the latter solution [13]. In other words, the two objectives are conflicting, as the optimal solution in terms of total distance traveled can include higher number of EVs than the optimal solution in terms of the number of EVs used. In fact, this hierarchical objective is realistic, considering that the use of more EVs corresponds to higher personnel costs as well as higher operational or capital expenditures (since many companies are either renting/leasing EVs to perform deliveries or are acquiring EVs that have a higher cost compared to conventional fuel vehicles) [11].

Since the E-VRPTW is an $\mathcal{NP}$-hard combinatorial optimization problem arising from the use of EVs instead of fuel cars [14], several approximation methods have been proposed. Schneider et al. [11] developed a metaheuristic approach based on tabu search with a variable neighborhood search that consists of different local search operators to solve the E-VRPTW. Goeke and Schneider [15] used the same local search operators with a large neighborhood search with different repair and destroy operators. They addressed a variation of the E-VRPTW in which a mixed fleet of electric and fuel

vehicles is considered. For the same E-VRPTW variation, Hiermann et al. [16] developed another metaheuristic based on large neighborhood search with embedded local search and labeling procedures. Keskin and Catay [17] adopted a similar metaheuristic and addressed a different E-VRPTW variant in which partial recharging is also considered.

In this work, we adopt the multiple ant colony system (MACS) [4], [5] designed for the VRPTW and extend it to the basic E-VRPTW. The advantage of MACS is that the two objectives of the problem are optimized in parallel, whereas in the aforementioned approaches either sequential optimization is performed to first determine the minimum number of EVs and then optimize the total distance traveled [11], or the number of EVs is empirically determined without any optimization [15]. The CROSS exchange heuristic [18] designed for the VRPTW is also extended to the E-VRPTW and used as a local search with MACS. Experiments are conducted on a set of small and large-scale E-VRPTW benchmark instances, demonstrating that the extended CROSS exchange heuristic improves the quality of the MACS solution. Furthermore, MACS shows competitive performance against existing methods.

The rest of the paper is organized as follows. Section II describes the E-VRPTW, while Section III describes the MACS extended to the E-VRPTW together with the CROSS exchange heuristic. The experimental results obtained from the benchmark set are presented in Section IV. Finally, Section V presents the conclusions and follow-up research work.

II. The Electric Vehicle Routing Problem with Time Windows (E-VRPTW)

A. Problem Description

The E-VRPTW can be defined on a complete weighted graph $G = (N, A)$, where $N = \{0\} \cup I \cup F'$ is a set of nodes and $A = \{(i, j) \mid i, j \in N, i \neq j\}$ is a set of arcs connecting these nodes. Non-negative values d_{ij} and t_{ij} are associated with each arc which represent, respectively, the Euclidean distance and traveled time between nodes i and j. Node $\{0\}$ denotes the central depot and the set $I \subset N$ denotes the set of customers, where each customer $i \in I$ is assigned two positive values δ_i and s_i indicating the customer's delivery demand and customer's service time, respectively. For the central depot and charging stations the demand is set to zero ($\delta_i = 0, \forall i \notin I$). Moreover, a time window $[e_i, l_i]$ is associated with each node $i \in N$ (i.e., for the depot, customers, and recharging stations), and, thus, a node cannot be served before e_i or after l_i (but the service time may end later than l_i).

Three decision variables are associated with nodes to keep track of the status of each EV: (1) variable u_i ($0 \leq u_i \leq C$), where C is the maximum cargo load of an EV and u_i is the remaining cargo load upon arrival at node i; (2) variable z_i ($0 \leq z_i \leq l_0$), where l_0 is the latest service time of the depot and z_i is the time of arrival at node i; and (3) variable y_i ($0 \leq y_i \leq Q$), where Q is the maximum battery charge level of an EV and y_i is the remaining battery charge level upon arrival at node i. Each traveled arc consumes hd_{ij} of the battery's energy, where h denotes the energy consumption rate of an EV traversing that arc.

The set $F' \subset N$ denotes the set of β_i node copies of each charging station $i \in F$ (i.e., $|F'| = \sum_{i \in F} \beta_i$), which are used to permit multiple visits to each charging station $i \in F$ (if required) [19]. The upper bound on the number of node copies for each charging station is equal to $\beta_i = 2|I|$, because in the worst case an EV for each customer is needed and a visit to a charging station before and after serving the customer is required [20]. The service time (or recharging time) at the recharging station is defined by a constant rate g and the difference between the battery charge level y_i when arriving at the station and the maximum battery capacity.

B. Problem Formulation

A binary decision variable is defined, i.e., $x_{ij},\ \forall i, j \in N$, denoting whether arc (i, j) is traversed by an EV (i.e., $x_{ij} = 1$) or not (i.e., $x_{ij} = 0$). The mathematical model of E-VRPTW is formulated as [11]:

$$\min \sum_{i \in N, j \in N, i \neq j} d_{ij} x_{ij}, \tag{1}$$

s.t.

$$\sum_{j \in N, i \neq j} x_{ij} = 1, \quad \forall i \in I, \tag{2}$$

$$\sum_{j \in N, i \neq j} x_{ij} \leq 1, \quad \forall i \in F', \tag{3}$$

$$\sum_{j \in N, i \neq j} x_{ij} - \sum_{j \in N, i \neq j} x_{ji} = 0, \quad \forall i \in N, \tag{4}$$

$$z_i + (t_{ij} + s_i) x_{ij} - l_0 (1 - x_{ij}) \leq z_j, \quad \forall i, j \in N, i \neq j, \tag{5}$$

$$z_i + t_{ij} x_{ij} + g(Q - y_i) - (l_0 + qQ)(1 - x_{ij}) \leq z_j, \quad \forall i \in F', \forall j \in N, i \neq j, \tag{6}$$

$$e_j \leq z_j \leq l_j, \quad \forall j \in N, \tag{7}$$

$$0 \leq u_j \leq u_i - \delta_i x_{ij} + C(1 - x_{ij}),\ \forall i, j \in N, i \neq j, \tag{8}$$

$$0 \leq u_0 \leq C, \tag{9}$$

$$0 \leq y_j \leq y_i - (h d_{ij} x_{ij} + Q(1 - x_{ij}),\ \forall i, j \in I, i \neq j, \tag{10}$$

$$0 \leq y_j \leq Q - h d_{ij} x_{ij}, \quad \forall i \in F', \forall j \in N, i \neq j, \tag{11}$$

$$x_{ij} \in \{0, 1\},\ \forall i \in N,\ \forall j \in N, i \neq j, \tag{12}$$

where Eq. (1) defines the objective of minimizing the total traveled distance, Eq. (2) enforces the connectivity of customer visits, Eq. (3) addresses the connectivity of visits to recharging stations, Eq. (4) establishes the flow conservation by guaranteeing that at each node the number of incoming arcs is equal to the number of outgoing arcs, Eq. (5) guarantees time feasibility for arcs leaving customers and the depot, Eq. (6) guarantees time feasibility for arcs leaving recharging stations, Eq. (7) enforces that every node is visited within its time window, Eqs. (8) and (9) guarantee delivery demand fulfillment for all customers by assuring a non-negative cargo

Algorithm 1 MACS for the E-VRPTW

1: $T^{best} \leftarrow$ initial solution with unlimited number of EVs
2: $m \leftarrow$ determine the number of EVs of T^{best}
3: **while** termination condition not met **do**
4: $T^{dist} \leftarrow$ start ACS-DIST(T^{best}, m)
5: $T^{vei} \leftarrow$ start ACS-VEI(T^{best}, m)
6: **while** ACS-DIST and ACS-VEI are optimizing **do**
7: **if** T^{dist} is better than T^{best} **then**
8: $T^{best} \leftarrow T^{dist}$
9: **end if**
10: **if** T^{vei} is better than T^{best} **then**
11: $T^{best} \leftarrow T^{vei}$
12: **end if**
13: $m' \leftarrow$ determine the number of EVs of T^{best}
14: **if** $m' < m$ **then**
15: $m \leftarrow m'$
16: stop ACS-DIST and ACS-VEI
17: goto line 4
18: **end if**
19: **end while**
20: **end while**

load upon arrival at any node, Eqs. (10) and (11) ensure that the battery charge level never falls below 0, and Eq. (12) is the binary decision variable previously described.

III. The Multiple Ant Colony System

The Multiple Ant Colony System (MACS) [4] was initially designed to address the conventional VRPTW where a multiple hierarchical objective is optimized. In this work, MACS is applied to the E-VRPTW in which, two colonies, ACS-VEI and ACS-DIST, are used to minimize the number of EVs and the total distance traveled, respectively. The key idea is to allow the two colonies to minimize the two objectives in parallel by sharing their outputs during the optimization process as described in Alg. 1.

An initial feasible solution T^{best} is generated using the nearest neighbor heuristic [21] without any predefined number of EVs. Then, the number of EVs m of T^{best} is determined, in order for ACS-DIST to start constructing solutions with m EVs, while ACS-VEI tries to construct feasible solutions with $m-1$ EVs. Both ACS-DIST and ACS-VEI use the proposed method in Alg. 2 to construct feasible E-VRPTW solutions. The initial nearest neighbor solution T^{vei} of ACS-VEI in Alg. 3 may not be necessarily feasible, whereas the initial nearest neighbor solution T^{dist} of ACS-DIST in Alg. 4 is always feasible. When ACS-VEI discovers a feasible solution (recall, with one EV less), then the information is send to ACS-DIST to start constructing solutions with the updated number of EVs, while ACS-VEI will carry on searching for another feasible solution with one less EV.

A. Initialization

Both ACS-DIST and ACS-VEI colonies consist of multiple ants which are initially positioned at the central depot, i.e., node $\{0\}$. Each colony updates its own pheromone trail, since the two colonies are optimizing two different objectives. All the solution components (i.e., nodes) of the problem are associated with a pheromone trail value which is uniformly initialized at the start of the execution as follows:

$$\tau_{ij} \leftarrow \tau_0, \quad \forall (i,j) \in A, \tag{13}$$

where τ_{ij} is the pheromone trail value associated with arc (i,j) connecting nodes i and j, and τ_0 is the initial pheromone trail value.

Additionally, ACS-VEI assigns a value in_j for each node j to keep track of the number of times customer j has not been inserted in a solution. Initially, this value is set to zero for all customers, i.e., $in_j \leftarrow 0, \quad \forall j \in I$.

Algorithm 2 E-VRPTW Solution Construction

1: **input:** number of EVs, i.e., σ
2: $T^k \leftarrow$ insert depot $\{0\}$ at position 0
3: $step \leftarrow 1$
4: $\#vehicles \leftarrow \#vehicles + 1$
5: **while** $\#vehicles \leq \sigma$ **do**
6: $i \leftarrow$ select from T^k at position $step - 1$
7: $j \leftarrow$ select probabilistically from $\mathcal{N}_i^k$
8: $step \leftarrow step + 1$
9: **if** $\mathcal{N}_i^k = \emptyset$ **then**
10: $T^k \leftarrow$ insert depot $\{0\}$ at position $step$
11: $\#vehicles \leftarrow \#vehicles + 1$
12: **else if** not enough energy **then**
13: $s \leftarrow$ select closest station between i and j
14: $T^k \leftarrow$ insert s at position $step$
15: **else**
16: $T^k \leftarrow$ insert j at position $step$
17: $\#customers \leftarrow \#customers + 1$
18: **end if**
19: **end while**
20: **while** $\#customers \leq |I|$ **do**
21: **if** customer i is not visited **then**
22: $step \leftarrow step + 1$
23: $T^k \leftarrow$ customer i insertion at position $step$
24: $\#customers \leftarrow \#customers + 1$
25: **end if**
26: **end while**
27: $T^k \leftarrow$ insert depot $\{0\}$ at position $step + 1$
28: **output:** E-VRPTW solution T^k

B. Solution Construction

Both ACS-DIST and ACS-VEI use the same construction procedure to build feasible solutions. In particular, each ant k represents a complete E-VRPTW solution T^k (i.e., the routes of all EVs) and makes selections (node by node) biased by the existing pheromone trails and some heuristic information associated with the solution components of the problem, until all the components (i.e., customers) are selected.

Algorithm 3 ACS-VEI

input: T^{best} and m
$\sigma \leftarrow m - 1$
$T^{vei} \leftarrow$ nearest neighbor solution with σ vehicles
$C^{best} \leftarrow$ evaluate solution T^{best}
for each arc (i, j) **do**
  $\tau_{ij} \leftarrow \tau_0$
end for
for each customer j **do**
  $in_j \leftarrow 0$
end for
while termination condition not met **do**
  for each ant k **do**
    $T^k \leftarrow$ E-VRPTW solution with σ EVs (Alg. 2)
    $C^k \leftarrow$ evaluate solution T^k
    for each customer $j \notin T^k$ **do**
      $in_j \leftarrow in_j + 1$
    end for
  end for
  if $\exists k:\ T^k$ is feasible **then**
    $C^{vei} \leftarrow C^k$
    $T^{vei} \leftarrow T^k$
    for each customer j **do**
      $in_j \leftarrow 0$
    end for
  end if
  for each arc$(i, j) \in T^{vei}$ **do**
    $\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} + \rho/C^{vei}$
  end for
  for each arc$(i, j) \in T^{best}$ **do**
    $\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} + \rho/C^{best}$
  end for
end while
output: best-so-far solution T^{vei}

Ant k selects the next customer j, while at customer i, according to the probability distribution defined as follows:

$$p_{ij}^k = \begin{cases} \frac{[\tau_{ij}]^\alpha [\eta_{ij}^k]^\beta}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}]^\alpha [\eta_{il}^k]^\beta}, & \text{if } j \in \mathcal{N}_i^k, \\ 0, & \text{otherwise,} \end{cases} \tag{14}$$

where τ_{ij} and η_{ij}^k are, respectively, the existing pheromone trail and the dynamic heuristic information available a priori between components i and j. The η_{ij}^k value is different for each ant k (more details are given later on). Parameters α and β determine the relative influence of τ_{ij} and η_{ij}^k, respectively. The neighborhood set $\mathcal{N}_i^k$ is the set of unvisited customers adjacent to component i for the k–th ant, that do not violate the capacity, energy, and time window constraints.

Note that the depot and charging station components are not included in the $\mathcal{N}_i^k$ set. These components are selected when some predefined conditions occur. Specifically, when $\mathcal{N}_i^k$ is empty, then the depot component is selected denoting the return of the EV to the central depot. In case, $\mathcal{N}_i^k$ is not empty and a customer violates the energy constraint, then the closest energy recharging station s from the current solution component i to the next customer j is selected as follows:

$$s = \arg\min_{l \in F} \{d_{il} + d_{lj}\}, \tag{15}$$

and it is added along with customer j to satisfy the energy constraint. A lookahead policy [7] is used in which customer j must have at least one charging station (including the central depot) within its energy range. In this way, EVs will always have enough energy to travel to a charging station or return back to the depot from any customer.

Furthermore, with probability q_0 ant k selects the next customer j while at customer i that has the highest probability, i.e., $j = \arg\max_{l \in \mathcal{N}_i^k} \{\tau_{il}[\eta_{il}^k]^\beta\}$, whereas with probability $(1 - q_0)$ ant k selects the next customer j probabilistically as defined in Eq. (14), where $q_0 \in [0, 1]$.

Since both colonies are constructing solutions with a predetermined number of EVs, some solutions may be incomplete (i.e., some customers may remain unvisited). Therefore, all the unvisited customers are finally added to the position that causes the smallest increase in the solution quality while satisfying the capacity, energy, and time windows constraints. Note that some unvisited customers may be added together with a recharging station (if required) or some existing customers may be relocated to satisfy the constraints [15].

The dynamic heuristic information η_{ij}^k depends on the delivery time dt_j of ant k when traveling to customer j, i.e., $dt_j \leftarrow \max(ct^k + t_{ij}, e_j)$, where ct^k is the accumulated travel time of ant k, t_{ij} is the travel time when traveling from

Algorithm 4 ACS-DIST

input: T^{best} and m
$\sigma \leftarrow m$
$T^{dist} \leftarrow T^{best}$
$C^{dist} \leftarrow$ evaluate solution T^{dist}
for each arc (i, j) **do**
  $\tau_{ij} \leftarrow \tau_0$
end for
while termination condition not met **do**
  for each ant k **do**
    $T^k \leftarrow$ E-VRPTW solution with σ EVs (Alg. 2)
    $T^k \leftarrow$ apply local search
    $C^k \leftarrow$ evaluate solution T^k
  end for
  if $\exists k:\ C^k$ is better than C^{dist} **then**
    $C^{dist} \leftarrow C^k$
    $T^{dist} \leftarrow T^k$
  end if
  for each arc$(i, j) \in T^{dist}$ **do**
    $\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} + \rho/C^{dist}$
  end for
end while
output: best-so-far solution T^{dist}

customer i to customer j and e_j is the earliest time window of customer j. For ACS-DIST η_{ij}^k is defined as follows:

$$\eta_{ij}^k = \frac{1}{\max(1, (dt_j - ct^k)(l_j - ct^k))}, \quad (16)$$

whereas for ACS-VEI it is defined as follows:

$$\eta_{ij}^k = \frac{1}{\max(1, (dt_j - ct^k)(l_j - ct^k) - in_j)}, \quad (17)$$

where l_j is the latest time windows of customer j and in_j is the number of times customer j has not been inserted in a solution.

C. Local Search

All solutions constructed by ACS-VEI undergo local search improvements using the CROSS exchange heuristic [18]. The particular local search heuristic was designed for VRPs with time windows because it preserves the orientation (defined by the time window constraints) on each route. The CROSS exchange heuristic generalizes the exchange neighborhood, in which two nodes are swapped, and the reallocate neighborhood, in which a node is moved to a different location [22].

Suppose that a CROSS exchange move is applied to the segment from node i to node j for one route and to the segment from node h to node q for another route, as shown in Fig. 1. This will cause arcs $(i, i+1)$, $(j, j+1)$, $(h, h+1)$, and $(q, q+1)$ to be removed, and arcs $(i, h+1)$, $(h, i+1)$, $(j, q+1)$, and $(q, j+1)$ to be inserted. In case the selected nodes j and q are directly connected to the depot (i.e., $j+1$ is the depot of one route and $q+1$ is the depot of the other route), then a *2–opt** [23] move is performed which is special case of the CROSS exchange heuristic as shown in Fig 2(a). It is also possible to select an empty segment from a route to be exchanged with a non-empty segment from another route. As a result, an *Or–opt* [24] move is performed as shown in Fig. 2(b), in which a segment will be reallocated from one route to another route. Suppose that the selected nodes h and q of one route are the same (i.e., $h = q$), this will cause arcs $(i, i+1)$, $(j, j+1)$, and $(h, h+1)$ to be removed and arcs $(i, j+1)$, $(h, i+1)$, $(i+1, j)$, and $(j, h+1)$ to be inserted. In few cases, this *Or–opt* move can create empty routes resulting in an E-VRPTW solution with fewer EVs. For example, in Fig 2(b), if i and $j+1$ are the first and last nodes of the first route, respectively, then the entire route is reallocated between nodes h and $h+1$ in the other route.

In addition to the described inter-route moves (i.e., involving two routes), the CROSS exchange heuristic also performs intra-route moves (i.e., involving a single route). Specifically, *Or–opt* moves are allowed in which nodes are reallocated in another location within the same route.

Note that, only inter-route moves that do not violate the capacity constraint are allowed. On the contrary, some moves may violate the energy constraint, since the charging stations may change positions, or the current position of the stations is affected due to the changes caused in the order of the customers. In that case, the affected routes of the E-VRPTW solution are repaired using the stationInRe operator [11] by inserting new station(s) to satisfy the energy constraint and removing unnecessary station(s).

The CROSS exchange heuristic is applied until no further improvement is possible. It must be noted that the pheromone trails are updated after the local search improvements to mark them in the pheromone trails so they can be exploited in the following iterations (line 11 in Alg. 4).

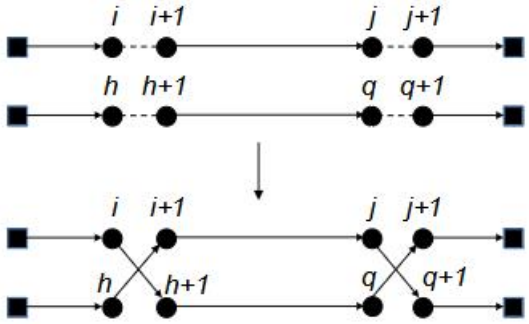
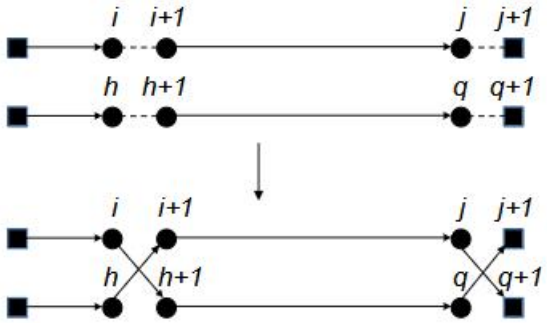
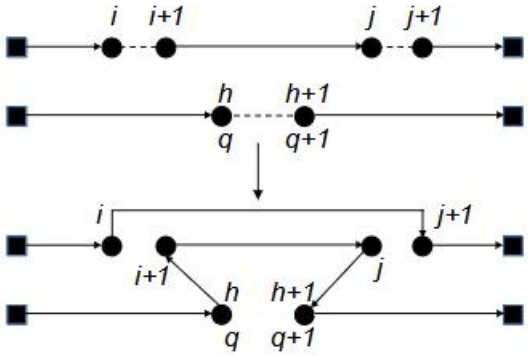


Fig. 1: CROSS exchange move. The square represents the central depot and the circles the customers.



(a)



(b)

Fig. 2: Special cases of the CROSS exchange move (a) *2–opt** and (b) *Or–opt*. The square represents the central depot and the circles the customers.

D. Pheromone Update

The two colonies follow the Ant Colony System (ACS) [1] variant, which is one of the most efficient ACO variants. In particular, only the pheromone trails of the best-so-far ant are globally updated as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho/C^{bs}, \quad \forall (i,j) \in T^{bs}, \tag{18}$$

where $\rho \in (0,1]$ is the evaporation rate and C^{bs} is the quality of the best-so-far solution T^{bs}, which for ACS-DIST $T^{bs} = T^{dist}$ and for ACS-VEI $T^{bs} = T^{vei}$. Note that T^{vei} is the unfeasible solution with the highest number of visited customers and, thus, the pheromone trails of unvisited customers will not be updated. To address this issue, ACS-VEI also uses T^{best}, which is the feasible solution (i.e., all customers are visited) with the lowest number of vehicles so far, to update pheromone trails (lines 29–31 in Alg. 3).

In addition, while ants construct solutions (described in the previous subsection), for both ACS-VEI and ACS-DIST, and move from node i to node j, the amount of pheromone trails associated with arc (i,j) are locally updated as follows:

$$\tau_{ij} \leftarrow (1-\xi)\tau_{ij} + \xi\tau_0, \tag{19}$$

where $\xi \in (0,1)$ is the deduction rate and τ_0 is the initial pheromone trail.

IV. Experimental Results

A. Experimental Setup

Ten independent executions (with different random seeds), are performed. The total distance traveled of the solution with the best quality and minimum number of EVs are recorded.

1) Parameter Tuning: The colony sizes for ACS-VEI and ACS-DIST are set to 10 ants each. The initial pheromone trail value is set to $\tau_0 = 1/nC^{nn}$, where n is the problem size and C^{nn} is the quality of the solution generated by the nearest neighbor heuristic. The evaporation rates are set to $\rho = \xi = 0.1$ and the three parameters of the decision rule are set to typical values as follows: $\alpha = 1$, $\beta = 2$, and $q_0 = 0.9$.

2) Benchmark Problem Instances: For the experiments, the benchmark set generated in [11] is used. It consists of 36 small-scale problem instances with 5, 10, and 15 customers and 56 large instances with 100 customers. The benchmark uses the well-known Solomon's VRPTW problem instances [25] as the base to generate the E-VRPTW problem instances by adding recharging stations (i.e., every customer can be reached from the depot using at most two different recharging stations). The small-scale problem instances contain 2 to 8 recharging stations whereas the large-scale problem instances contain 21 recharging stations. The instances are classified depending on the distribution of the customer locations as follows: random distribution (R), clustered distribution (C), and a combination of both (RC).

TABLE I: Mean solution quality and standard deviation (averaged over 10 runs) of MACS with and without local search.

	MACS+ls		MACS−ls	
E-VRPTW	mean	stdev	mean	stdev
C101-5	257.75	0.00	257.75	0.00
C103-5	**176.05**	0.00	184.50	0.00
C206-5	242.55	0.00	242.55	0.00
C208-5	158.48	0.00	158.48	0.00
R104-5	**136.68**	0.00	140.28	7.50
R105-5	**156.08**	0.00	168.30	0.27
R202-5	128.78	0.00	128.78	0.00
R203-5	**179.06**	0.00	194.57	1.20
RC105-5	241.30	0.00	241.30	0.00
RC108-5	253.93	0.00	253.93	0.00
RC204-5	185.15	0.00	185.16	0.00
RC208-5	167.98	0.00	167.98	0.00
C101-10	**393.76**	0.00	411.05	4.00
C104-10	**273.93**	0.00	278.02	1.32
C202-10	**304.05**	0.00	322.29	0.00
C205-10	**228.28**	0.00	266.18	11.01
R102-10	249.19	0.00	249.19	0.00
R103-10	**207.05**	0.00	212.14	0.00
R201-10	**241.51**	0.00	267.97	0.00
R203-10	**219.54**	2.13	232.68	0.00
RC102-10	**423.51**	0.00	452.56	15.32
RC108-10	**345.92**	0.00	362.95	6.53
RC201-10	412.86	0.00	412.86	0.00
RC205-10	**328.78**	8.88	430.74	4.46
C103-15	**384.29**	0.00	425.71	2.05
C106-15	**275.13**	0.00	348.52	0.81
C202-15	**383.62**	0.00	512.77	6.25
C208-15	**300.55**	0.00	305.37	0.30
R102-15	**413.93**	0.00	429.64	3.02
R105-15	336.15	0.00	336.15	0.00
R202-15	**358.59**	1.23	457.54	24.94
R209-15	**318.92**	6.12	411.35	4.82
RC103-15	**397.67**	0.00	403.56	3.61
RC108-15	**371.05**	2.56	390.70	4.31
RC202-15	**394.39**	0.00	486.85	0.00
RC204-15	**391.04**	1.93	412.06	3.90

Best values are indicated in **bold**.

B. Effect of Local Search

The experimental results regarding the mean and standard deviation of the total distance traveled (averaged over 10 runs) of MACS with (MACS+ls) and without (MACS−ls) the CROSS exchange local search heuristic (presented in Section III-C) for small-scale problem instances (i.e., 5–customer, 10–customer, and 15–customer instances) are given in Table I. Recall that the local search operator is only used in ACS-DIST of MACS. The termination condition for both MACS algorithms is set to 5×10^5 function evaluations. Note that the partial evaluations required when applying the CROSS exchange local search are added to the total function evaluations of MACS+ls.

From Table I, it can be observed that the mean solution quality of MACS+ls is better (or equal) than MACS−ls. In addition, the standard deviation results of MACS+ls is lower than MACS−ls, indicating small variation from the mean. These results are expected, due to several reasons. First, typically, a local search heuristic is not terminated until a local optimum solution is found in the neighborhood, and hence, the solution quality of MACS+ls is more likely to be

TABLE II: Best solution quality, number of vehicles m, and CPU time (in seconds) results obtained from CPLEX [11] and MACS for small-scale E-VRPTW problem instances. $\Delta\%$ is the solution quality deviation of MACS from CPLEX (in percentage).

	CPLEX			MACS			
E-VRPTW	m	best	secs	m	best	secs	$\Delta\%$
C101-5	2	257.75	81	2	257.75	0.0002	0.00
C103-5	1	176.05	5	1	176.05	0.21	0.00
C206-5	1	242.55	518	1	242.55	0.008	0.00
C208-5	1	158.48	15	1	158.48	0.0002	0.00
R104-5	2	136.69	1	2	136.69	0.005	0.00
R105-5	2	156.08	3	2	156.08	0.001	0.00
R202-5	1	128.78	1	1	128.78	0.08	0.00
R203-5	1	179.06	5	1	179.06	1.11	0.00
RC105-5	2	241.30	764	2	241.30	2.37	0.00
RC108-5	2	253.93	311	1	253.93	0.002	0.00
RC204-5	1	176.39	54	1	176.39	0.001	0.00
RC208-5	1	167.98	21	1	167.98	0.003	0.00
C101-10	3	393.76	171	3	393.76	4.53	0.00
C104-10	2	273.93	360	2	273.93	24.1	0.00
C202-10	1	304.06	300	1	304.06	2.85	0.00
C205-10	2	228.28	4	2	228.28	20.70	0.00
R102-10	3	249.19	389	3	249.19	1.57	0.00
R103-10	2	207.05	119	2	207.05	13.50	0.00
R201-10	1	241.51	177	1	241.51	1.14	0.00
R203-10	1	218.21	573	1	218.21	15.45	0.00
RC102-10	4	423.51	810	4	423.51	11.85	0.00
RC108-10	3	345.93	39	3	345.93	7.99	0.00
RC201-10	1	412.86	7200	1	412.86	0.02	0.00
RC205-10	2	325.98	399	2	325.98	25.27	0.00
C103-15	3	384.29	7200	3	384.29	24.36	0.00
C106-15	3	275.13	17	3	275.13	21.88	0.00
C202-15	2	383.62	7200	2	383.62	59.46	0.00
C208-15	2	300.55	5060	2	300.55	44.1	0.00
R102-15	5	413.93	7200	5	413.93	25.84	0.00
R105-15	4	336.15	7200	4	336.15	13.42	0.00
R202-15	2	358.00	7200	2	358.00	7.32	0.00
R209-15	1	313.24	7200	1	313.24	9.01	0.00
RC103-15	4	397.67	7200	4	397.67	24.52	0.00
RC108-15	3	370.25	7200	3	370.25	26.96	0.00
RC202-15	2	394.39	7200	2	394.39	73.38	0.00
RC204-15	1	407.45	7200	1	382.22	15.51	-6.19

CPLEX values with 7200 secs are upper bounds, whereas the remaining CPLEX values are optimal.

similar (i.e., having a small standard deviation). Second, the solution construction of ACO uses a different neighborhood than the CROSS exchange heuristic, and thus, the combination of MACS+ls has more chances to improve the solution quality. Third, MACS−ls performs only global optimization, and thus, only large steps in the search space are made during the optimization process.

C. Performance of MACS on Small-Scale Problem Instances

The experimental results regarding the total distance traveled, the number of EVs (m), and the computation time (secs) for small-scale problem instances of MACS [1] and CPLEX [2] are given in Table II. For the CPLEX approach the "best" and m values correspond to either the optimal solution or the upper bound found within 7200 seconds [11]. For the MACS approach the "best" and m values correspond to the best solution found in 10 runs, and $\Delta\%$ denotes the traveled distance deviation percentage of MACS from CPLEX.

[1]Intel Core i5 processor clocked at 2.67GHz with 8GB RAM, running 64-bit Linux.

[2]Intel Core i5 processor clocked at 2.67GHz with 4GB RAM, running Windows 7 Professional.

TABLE III: Best solution quality and number of vehicles (m) results obtained from MACS in comparison with the TS [11] approach for large-scale E-VRPTW problem instances.

	TS		MACS	
E-VRPTW	m	best	m	best
C101	12	1053.83	12	1053.83
C102	11	1069.35	11	**1051.38**
C103	10	1134.36	10	**1034.86**
C104	10	979.63	10	**961.88**
C105	11	1079.69	11	**1075.37**
C106	11	1057.87	11	**1057.65**
C107	11	1033.08	11	**1031.56**
C108	11	**1015.73**	10	1095.66
C109	10	1051.36	10	**1033.67**
C201	4	645.16	4	645.16
C202	4	645.16	4	645.16
C203	4	644.98	4	644.98
C204	4	636.43	4	636.43
C205	4	641.13	4	641.13
C206	4	638.17	4	638.17
C207	4	638.17	4	638.17
C208	4	638.17	4	638.17
R101	18	1670.80	18	**1663.04**
R102	16	1495.31	16	**1487.41**
R103	13	1348.25	13	**1271.35**
R104	11	1097.09	11	**1088.43**
R105	14	1514.36	14	**1442.35**
R106	13	1369.55	13	**1324.10**
R107	12	1162.90	12	**1150.95**
R108	11	1056.84	11	**1050.04**
R109	12	1308.62	12	**1261.31**
R110	11	1126.74	11	**1119.50**
R111	12	1123.96	12	**1106.19**
R112	11	1047.92	11	**1016.63**
R201	3	1264.82	3	1264.82
R202	3	1052.32	3	1052.32
R203	3	914.10	3	**895.54**
R204	2	790.68	2	**779.49**
R205	3	997.15	3	**987.36**
R206	3	928.26	3	**922.19**
R207	2	855.99	2	**845.26**
R208	2	741.44	2	**736.12**
R209	3	874.74	3	**867.05**
R210	3	848.44	3	**846.20**
R211	2	861.17	2	**827.89**
RC101	16	1753.35	16	**1726.91**
RC102	15	1559.95	14	**1552.08**
RC103	13	1355.36	13	**1350.09**
RC104	11	1280.82	11	**1227.25**
RC105	14	1479.56	14	**1475.31**
RC106	13	1437.96	13	**1427.21**
RC107	12	1284.47	12	**1274.89**
RC108	11	1209.61	11	**1197.83**
RC201	4	1446.03	4	**1444.94**
RC202	3	1425.17	3	**1410.74**
RC203	3	1084.66	3	**1055.19**
RC204	3	889.22	3	**884.80**
RC205	3	1360.39	3	**1273.55**
RC206	3	1207.77	3	**1188.63**
RC207	3	1010.66	3	**985.03**
RC208	3	838.03	3	**836.29**

Best values are indicated in **bold**.

From Table II it can be clearly observed that MACS is able to solve small-scale E-VRRTW instances to optimality in a few seconds. For most 15-customer instances, CPLEX fails to find the optimal solution (since the termination condition is reached) and provides an upper bound solution. MACS is able to find a solution equal to the upper bound provided by CPLEX. A better solution is provided by MACS for `RC204-15`. Thus, these results demonstrate the ability of MACS to provide high-quality solutions efficiently.

D. Performance of MACS on Large-Scale Problem Instances

The experimental results regarding the total distance traveled and the number of EVs (m) for large-scale problem instances of MACS approach are given in Table III and are compared with the Tabu Search (TS) approach [11]. The "best" and m values correspond to the best solution found in 10 runs. Note that CPLEX fails to provide a solution within an acceptable amount of time for large-scale problem instances.

From the comparisons in Table III, it can be observed that MACS performs better than (or equal to) TS in most problem instances, except in `C108`. Although both approaches explore similar local neighborhood structures, their main difference is that MACS is searching in multiple points of the search space in parallel (i.e., one search point for each artificial ant) whereas TS is searching on a single point of the search space. Therefore, in large and complex search spaces like the E-VRPTW, MACS has more chances to discover solutions with better quality.

V. Conclusions

In this work, we adopt the MACS algorithm to the E-VRPTW which consists of two objectives, i.e., total distance traveled and number of EVs used. MACS uses two different colonies, one for each objective, to optimize them in parallel. The two colonies cooperate by exchanging information concerning the hierarchy of the two objectives. The CROSS exchange local search heuristic is also extended to the E-VRPTW. The experimental results on a set of small-scale and large-scale problem instances demonstrate the effect of the local search and shows competitive performance of the proposed MACS with existing methods.

For future work, it would be interesting to investigate different customer and station insertion methods in the E-VRPTW solution construction since the greedy approach used in this work may not always be the optimal one [9], [10]. In addition, comparing MACS with other peer algorithms designed for the E-VRPTW is another future work.

References

[1] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, April 1997.

[2] B. Bullnheimer, R. Hartl, and C. Strauss, "Applying the ant system to the vehicle routing problem," in *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, S. Voß, S. Martello, I. Osman, and C. Roucairol, Eds. Kluwer Academic, 1997, pp. 285–296.

[3] M. Mavrovouniotis and S. Yang, "An ant system with direct communication for the capacitated vehicle routing problem," in *Computational Intelligence (UKCI), 2010 UK Workshop on*, 2011, pp. 14–19.

[4] L. M. Gambardella, E. D. Taillard, and C. Agazzi, "MACS-VRPTW: A multicolony ant colony system for vehicle routing problems with time windows," in *New Ideas in Optimization*, 1999, pp. 63–76.

[5] A. E. Rizzoli, R. Montemanni, E. Lucibello, and L. M. Gambardella, "Ant colony optimization for real-world vehicle routing problems," *Swarm Intelligence*, vol. 1, no. 2, pp. 135–151, Dec 2007.

[6] M. Mavrovouniotis and S. Yang, "Ant algorithms with immigrants schemes for the dynamic vehicle routing problem," *Information Sciences*, vol. 294, pp. 456–477, 2015.

[7] M. Mavrovouniotis, G. Ellinas, and M. Polycarpou, "Ant colony optimization for the electric vehicle routing problem," in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2018, pp. 1234–1241.

[8] M. Mavrovouniotis, C. Li, G. Ellinas, and M. Polycarpou, "Parallel ant colony optimization for the electric vehicle routing problem," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 1660–1667.

[9] Y.-H. Jia, Y. Mei, and M. Zhang, "Confidence-based ant colony optimization for capacitated electric vehicle routing problem with comparison of different encoding schemes," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2022.

[10] ——, "A bilevel ant colony optimization algorithm for capacitated electric vehicle routing problem," *IEEE Transactions on Cybernetics*, pp. 1–14, 2021.

[11] M. Schneider, A. Stenger, and D. Goeke, "The electric vehicle-routing problem with time windows and recharging stations," *Transportation Science*, vol. 48, no. 4, pp. 500–520, 2014.

[12] O. Bräysy and M. Gendreau, "Vehicle routing problem with time windows, Part II: Metaheuristics," *Transportation Science*, vol. 39, no. 1, pp. 119–139, 2005.

[13] N. Kohl, J. Desosiers, O. B. G. Madsen, M. M. Solomon, and F. Soumis, "2-path cuts for the vehicle routing problem with time windows," *Transportation Science*, vol. 33, no. 1, pp. 101–116, 1999.

[14] M. Garey and D. Johnson, *Computer and intractability: A guide to the theory of $\mathcal{NP}$-completeness*. San Francisco: Freeman, 1979.

[15] D. Goeke and M. Schneider, "Routing a mixed fleet of electric and conventional vehicles," *European Journal of Operational Research*, vol. 245, no. 1, pp. 81–99, 2015.

[16] G. Hiermann, J. Puchinger, S. Ropke, and R. F. Hartl, "The electric fleet size and mix vehicle routing problem with time windows and recharging stations," *European Journal of Operational Research*, vol. 252, no. 3, pp. 995–1018, 2016.

[17] M. Keskin and B. Çatay, "Partial recharge strategies for the electric vehicle routing problem with time windows," *Transportation Research Part C: Emerging Technologies*, vol. 65, pp. 111–127, 2016.

[18] É. Taillard, P. Badeau, M. Gendreau, F. Guertin, and J.-Y. Potvin, "A tabu search heuristic for the vehicle routing problem with soft time windows," *Transportation Science*, vol. 31, no. 2, pp. 170–186, 1997.

[19] S. Erdoğan and E. Miller-Hooks, "A green vehicle routing problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 48, no. 1, pp. 100–114, 2012.

[20] A. Froger, J. E. Mendoza, O. Jabali, and G. Laporte, "Improved formulations and algorithmic components for the electric vehicle routing problem with nonlinear charging functions," *Computers & Operations Research*, vol. 104, pp. 256–294, 2019.

[21] M. M. Flood, "The traveling-salesman problem," *Operations Research*, vol. 4, no. 1, pp. 61–75, 1956.

[22] I. Osman, "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem," *Annals of Operations Research*, vol. 41, pp. 421–451, 1993.

[23] J.-Y. Potvin and J.-M. Rousseau, "An exchange heuristic for routeing problems with time windows," *The Journal of the Operational Research Society*, vol. 46, no. 12, pp. 1433–1446, 1995.

[24] I. Or, "Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking," Ph.D. dissertation, Northwestern University, Illinois, 1967.

[25] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations Research*, vol. 35, no. 2, pp. 254–265, 1987.