Parallel Ant Colony Optimization for the Electric Vehicle Routing Problem

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

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

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

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*—Parallelizing metaheuristics has become a common practice considering the computation power and resources available nowadays. The aim of parallelizing a metaheuristic is either to increase the quality of the generated output, given a fixed computation time, or to reduce the required time in generating an output. In this work, we parallelize one of the best-performing ant colony optimization (ACO) algorithms and apply it to the electric vehicle routing problem (EVRP). EVRP is more challenging than the conventional vehicle routing problem, as with the consideration of electric vehicles additional hard constraints arise within the EVRP due to their limited driving range (e.g., the consideration whether electric vehicles need to visit a charging station during their daily operation). The proposed parallel ACO algorithm with several colonies also uses a migration policy to allow communication between the different colonies. From the simulation studies it is shown that parallelizing ACO algorithms, both with and without a migration policy, is highly effective.**

***Index Terms*—Ant colony optimization, electric vehicle, vehicle routing problem**

I. Introduction

The use of metaheuristics to solve difficult optimization problems (i.e., $\mathcal{NP}$–hard) usually requires huge computational effort [1]. For this reason, there is a growing interest for the parallelization of these algorithms to take advantage of the increased availability of powerful computing resources and parallel architectures. In this way, the computation time of a single execution of the algorithm is often reduced significantly [2], [3].

Parallel population-based metaheuristics, such as evolutionary algorithms [4]–[6] and ant colony optimization (ACO) algorithms [7]–[9], typically adopt the *island model*, in which the communication – known as *migration* – among the parallel populations (for evolutionary algorithms) or colonies (for ACO) usually plays a significant role.

There are several opportunities for parallelization within the ACO framework, including ants constructing solutions independently from other ants in a single colony [10], [11] or in multiple colonies [3]. In fact, parallel ACO algorithms with multiple (independent) colonies have been successfully applied to several combinatorial optimization problems, including the travelling salesman problem [3], [8], the capacitated vehicle routing problem [12], and the quadratic assignment problem [13]. The reader is referred to the comprehensive survey in [14] for more applications.

In this work, we apply parallel ACO to a more challenging optimization problem, that is, the electric vehicle routing problem (EVRP) [15]. EVRP has recently attracted significant attention by the research community due to the interest of logistic companies and government authorities to reduce the carbon dioxide (CO_2) emissions of their vehicle fleets [16]–[18]. In particular, we focus in parallelizing (with multiple colonies) a high-performing ACO algorithm, that is, the $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System ($\mathcal{MM}$AS [24]) that has been previously applied to the EVRP [19]. Previous research studies with the $\mathcal{MM}$AS algorithm (on a different optimization problem) investigated the effect of parallel independent runs (i.e., when no migration policy is utilized) [20]. Other research studies, investigated the effect of different migration schemes [8]. However, these migration schemes had a negative effect because: i) the pheromone re-initialization feature of the $\mathcal{MM}$AS was not used, and ii) the colonies were exchanging information (typically the best-so-far solution [21], [22]) very frequently (e.g., every 25 iterations). As a result, the parallel colonies were most probably stagnating at the same solution

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 739551 (KIOS CoE) and from the Government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development.

instead of exploring the search space in different areas. Later research studies, improved the migration schemes by utilizing the pheromone re-initialization of $\mathcal{MM}$AS and a variable scheduling for exchanging the best-so-far solution [3], [9].

The contributions of this work are: i) the parallelization of the $\mathcal{MM}$AS algorithm for the EVRP, and ii) the utilization of a larger number of colonies (previously 2 colonies in [9] and 8 colonies in [3], [8] were used) to further investigate the effect of parallelism for the $\mathcal{MM}$AS algorithm. Our simulation studies show the positive effects that parallelism has on the solution quality of $\mathcal{MM}$AS and the positive effect that migration has in parallel $\mathcal{MM}$AS that results in shortening the time to discover the best output.

The rest of the paper is organized as follows. Section II describes the EVRP model used in this work. Section III describes the parallel $\mathcal{MM}$AS algorithm for EVRP, while Section IV presents the experimental results and analysis of parallel $\mathcal{MM}$AS. The effect of the number of colonies and the migration policy are also analyzed. Finally, Section V offers some concluding remarks and discusses directions for future work.

II. Electric Vehicle Routing Problem

Transportation has been the main contributor to CO_2 emissions. Due to global warming, pollution and climate changes, logistic companies such as FedEx, UPS, DHL and TNT have became more sensitive to the environment and they are investing in ways to reduce the CO_2 emissions that result as part of their daily operations. There is no doubt that using electric vehicles (EVs) instead of conventional vehicles will significantly contribute to the reduction of CO_2 emissions [18].

A. Problem Formulation

With the growing interest of logistic companies in utilizing EVs for their daily operations, a problem of routing a fleet of EVs has emerged, namely the EVRP. The EVRP aims to find the best possible routes for EVs in order to serve a set of customers by typically taking into account different constraints. Several variations of the EVRP have been introduced that mainly differ on their constraints (refer to the comprehensive survey in [15] for further details). The common constraint of all EVRP variations, is the consideration of visiting a charging station due to the limited driving range of EVs. The EVRP variation we consider in this work also takes into account the following constraints: vehicle load capacity and service time (typical constraints in the conventional vehicle routing problem).

An EVRP instance is modelled by a fully connected weighted graph $G = (N \cup F, L)$, where $N = \{1, \ldots, n\}$ is a set of n customers (nodes), $F = \{n+1, \ldots, n+s\} \cup \{0\}$ is a set of s energy recharging stations and a central depot 0 which is also a recharging station and $L = \{(i, j) \mid i, j \in N\}$ is a set of links (arcs) used to interconnect the network of nodes $N \cup F$. Note that multiple visits are permitted to each node in the F set of stations whereas a single visit is permitted to each node in the N set of customers.

Each arc $(i, j) \in L$ is associated with a non-negative value $t_{ij} = \mathbb{R}^+$ which represents the travel time between customers i and j and can be defined as:

$$t_{ij} = d_{ij}/s_{ij}, \tag{1}$$

where d_{ij} and s_{ij} are the distance (km) and average speed (km/h)[1] associated with arc $(i, j) \in L$, respectively. Each customer $i \in N$ is associated with a non-negative demand δ_i of some goods that need to be delivered by a fleet of m vehicles as well as a service time σ_i required to unload the goods from the vehicle. Note that σ_i is proportional to the demand of the customer $i \in N$ (e.g., the higher the demand δ_i, the more the service time required). The vehicle load of an EV, say k, on arc (i, j), is given by l^k_{ij} (i.e., $0 \leq l^k_{ij} \leq Q$), where Q is the maximum vehicle capacity (which may be different for each EV). The maximum service time for each EV is set to J minutes (which is the same for all drivers of an EV and basically defines the maximum working hours of the drivers). Each recharging station $i \in F$ is associated with a non-negative time w_i that represents the possible waiting time at the recharging station. The battery recharging rate r in all charging stations is the same and is constant.

The k–th EV has a battery level e^k_i that determines the current energy when reaching customer or recharging station $i \in N \cup F$ which satisfies $0 \leq e^k_i \leq B_C$, where B_C is the maximum battery capacity. The charging time c^k_i of the k–th EV at recharging station $i \in F$ is determined by its current energy level e^k_i and the charging rate of the station such that $c^k_i = (B_C - e^k_i)/r$, assuming that the EV will be fully charged (i.e., when $e^k_i = B_C$). Note that all EVs are fully charged and loaded when they leave the central depot at the start of the day.

The objective of the problem is to find the minimum set of $M = \{1, \ldots, m\}$ EVs visiting all customers once and only once satisfying their demands, all starting from and ending at the depot, such that the total travel time (including driving time, and charging and waiting time at the energy recharging station(s)) is minimized. A complete EVRP solution π is defined by a permutation of nodes (both customers and energy recharging stations) and consists of the complete routes of all the EVs. An EVRP solution is evaluated as follows:

$$\min \phi(\pi) = \sum_{k=1}^{m} \left(\sum_{i=0}^{n} \sum_{j=0}^{n} (t_{ij} + \sigma_i) x^k_{ij} + (c^k_i + w_i) y^k_i \right), \tag{2}$$

s.t

$$\sum_{j=0}^{n} \sum_{k=1}^{m} x^k_{ij} = 1, \ \forall i \in N, \tag{3}$$

$$\sum_{i=0}^{n} \sum_{k=1}^{m} x^k_{i0} = m, \tag{4}$$

[1]Note that the average speed is based only on the speed limit of the roads. It may also be affected by other uncertainties such as traffic, road conditions, etc.

$$\sum_{j=0}^{n}\sum_{k=1}^{m} x_{0j}^{k} = m, \tag{5}$$

$$\sum_{i=0}^{n} x_{il}^{k} - \sum_{j=0}^{n} x_{lj}^{k} = 0, \forall l \in N \cup F, \forall k \in M, \tag{6}$$

$$\sum_{i=1}^{n}\sum_{j=0}^{n} \delta_i x_{ij}^{k} \leq Q, \forall k \in M, \tag{7}$$

$$\sum_{i=0}^{n}\sum_{j=0}^{n} (t_{ij} + \sigma_i) x_{ij}^{k} + (c_i^k + w_i) y_i^k \leq J, \forall k \in M \tag{8}$$

$$\begin{gathered} e_j^k \leq e_i^k - b_{ij}^k x_{ij}^k + B_C(1 - x_{ij}^k), \\ \forall i \in N \cup F, \forall j \in N, \forall k \in M, \end{gathered} \tag{9}$$

$$e_i^k \geq \min\{b_{i0}^k, (b_{ij}^k + b_{j0}^k)\}, \forall i \in N, \forall j \in F, \forall k \in M, \tag{10}$$

$$x_{ij}^k \in \{0,1\},\ \forall i,j \in N \cup F, \forall k \in M, \tag{11}$$

$$y_i^k \in \{0,1\},\ \forall i \in F, \forall k \in M, \tag{12}$$

where Eq. (2) defines the EVRP objective function (output in minutes), Eq. (3) ensures that each customer is visited exactly once, Eqs. (4) and (5) ensure that all EVs leave and return to the central depot, Eq. (6) ensures that when an EV visits a customer or a charging station it also leaves from it, Eq. (7) ensures that no EV is overloaded, Eq. (8) ensures that the maximum service time of an EV is respected, Eq. (9) ensures that the battery level is reduced by b_{ij}^k (described in more details in Eq. (15) later on) when visiting customer j from customer i, Eq. (10) ensures that there is enough energy to either return to the depot or visit a recharging station, Eq. (11) is a binary decision variable defined as follows:

$$x_{ij}^k = \begin{cases} 1, & \text{if EV } k \text{ visited customer } j \text{ immediately after } i, \\ 0, & \text{otherwise,} \end{cases} \tag{13}$$

and Eq. (12) is another binary decision variable defined as follows:

$$y_i^k = \begin{cases} 1, & \text{if EV } k \text{ recharged at station } i, \\ 0, & \text{otherwise,} \end{cases} \tag{14}$$

B. Energy Consumption

The energy consumption of the k-th EV between customers i and j is calculated as follows [23]:

$$b_{ij}^k = \left(a_{ij}(w + l_{ij}^k)d_{ij} + z(s_{ij})^2 d_{ij}\right)/E_F \tag{15}$$

where $a_{ij} = a + g\sin\theta_{ij} + gC_R\cos\theta_{ij}$ is an arc specific constant, $z = 0.5C_D A D$ is a vehicle specific constant, E_F is the engine efficiency, w is the vehicle curb weight (kg), a is the acceleration (m/s^2), g is the gravitational constant (m/s^2), θ_{ij} is the road angle (degree) associated with arc (i,j), A is the frontal surface area of the vehicle (m^2), D is the air density (kg/m^3), C_R is the coefficient of rolling resistance and C_D is the coefficient of rolling drag. The parameter values used in the EVRP model for our simulation studies (in Section IV) are shown in Table I.

TABLE I
EVRP PARAMETER VALUES USED IN THE SIMULATION STUDIES.

Parameter	Description	Value
E_F	Engine efficiency	70%
w	Vehicle curb weight	3.629 $tons$
a	Acceleration	0 m/s^2
g	Gravitational constant	9.81 m/s^2
A	Frontal surface area of the EV	5 m^2
θ_{ij}	Road angle	0 °
s_{ij}	Average EV speed	50 kph
D	Air density	1.2041 kg/m^3
r	Charging rate in public stations	40 kW
C_R	Coefficient of rolling resistance (unitless)	0.01
C_D	Coefficient of rolling drag (unitless)	0.7
B_C	EV battery capacity	120 kWh
J	Maximum service time of EV	480 $mins$
Q	Maximum EV capacity	3.871 $tons$

Note that the energy consumption calculated using Eq. (15) for different EVs (even if we have a homogeneous fleet of EVs) travelling on the same arc may be different because their current loads (i.e., l_{ij}^k) may differ. For example, an EV with a heavier load will consume more energy than an EV with a lighter load. Also, note that the load of an EV changes whenever a customer is served (unloading goods).

III. PARALLEL IMPLEMENTATION OF ACO

In this work, we use the $\mathcal{MM}$AS [24], [25]–one of the best performing and well studied ACO algorithms–for our parallel ACO implementation. The implementation of $\mathcal{MM}$AS is based on the publicly available ACOTSP code.[2]

The basic idea of the parallel $\mathcal{MM}$AS is to execute multiple colonies, where each colony uses independent pheromones, including independent updating of the pheromone trail values (discussed in Section III-C) and independent re-initialization of the pheromone trails when stagnation behavior is detected (described in Section III-D). Each colony records its own best-so-far[3] solution which is collected by the migration procedure (described in Section III-E) to exchange it among the colonies.

A. Constructing Solutions for the EVRP

Ants read pheromones to construct solutions and write pheromones to mark their constructed solutions (see Algorithm 1). Each ant h represents a complete EVRP solution (i.e., the

[2] http://www.aco-metaheuristic.org/aco-code/public-software.html

[3] The best-so-far ant is a special ant and may not necessarily belong to the colony of the current iteration. It is the best solution found among all the iterations.

Algorithm 1 Parallel ACO

```
t ← 0
τ_0 ← τ_max
InitializePheromoneTrails(τ_0)
while (termination condition is not satisfied) do
    ConstructSolutions
    π^ib ← FindIterationBest
    if (φ(π^ib) < φ(π^bs)) then
        π^bs ← π^ib
        Migration(π^bs)
    end if
    PheromoneUpdate
    if (stagnation behavior is detected) then
        InitializePheromoneTrails(τ_max)
    end if
    t ← t + 1
end while
OUTPUT: π^bs    %best EVRP solution
```

routes of all EVs). The probability distribution with which ant h selects customer j from customer i is defined as follows:

$$p_{ij}^{h} = \begin{cases} \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in \mathcal{N}_i^h} [\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}}, \text{if } j \in \mathcal{N}_i^h, \\ 0, \quad \text{otherwise} \end{cases} \tag{16}$$

where τ_{ij} and η_{ij} are, respectively, the existing pheromone trail and the heuristic information available a priori between customers i and j. The heuristic information is calculated as $\eta_{ij} = (1/t_{ij})$ where t_{ij} is defined as in Eq. (1). Parameters α and β are the two parameters which determine the relative influence of τ_{ij} and η_{ij}, respectively. $\mathcal{N}_i^h$ is the set of unvisited customers satisfying EV's capacity constraint in Eq. (7), the maximum service time constraint in Eq. (8) and the energy constraints in Eqs. (9) and (10) for the h-th ant adjacent to customer i.

In particular, if all the current unvisited customers violate the capacity or service constraints, then the central depot is added to the EVRP solution to close the route of the EV (i.e., denotes the return of the EV to the central depot). Note that if this return to the depot violates the energy constraint, then the EV visits an energy charging station prior to the return. In addition, if all the current unvisited customers violate the energy constraints, then the closest energy recharging station is added to the EVRP solution (i.e., denotes the visit to an energy recharging station).

B. Updating Pheromones

At the beginning all the pheromone trails are initialized as follows:

$$\tau_0 \leftarrow 1/\rho C^{nn}, \forall (i,j) \in L, \tag{17}$$

where ρ ($0 < \rho \leq 1$) is the pheromone evaporation rate and C^{nn} is the length of a tour generated by the nearest-neighbor heuristic[4]. Then, the pheromone trails are updated by applying evaporation as follows:

$$\tau_{ij} \leftarrow (1-\rho)\,\tau_{ij}, \forall (i,j) \in L, \tag{18}$$

where ρ is the pheromone evaporation rate and τ_{ij} is the existing pheromone value on arc (i,j). After evaporation, the best ant deposits pheromone proportional to its solution quality as follows:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^{best}, \forall (i,j) \in \pi^{best}, \tag{19}$$

where $\Delta\tau_{ij}^{best} = 1/C^{best}$ is the amount of pheromone that the best ant deposits and C^{best} is the cost of the best solution π^{best} (i.e., $C^{best} = \phi(\pi^{best})$). The "best" ant that is allowed to deposit pheromone may be either the best-so-far ant (π^{bs}), in which case $C^{best} = C^{bs}$, or the iteration-best ant (π^{ib}), in which case $C^{best} = C^{ib}$, where C^{bs} and C^{ib} are the tour costs of the best-so-far ant (i.e., $C^{bs} = \phi(\pi^{bs})$) and the iteration-best ant (i.e., $C^{ib} = \phi(\pi^{ib})$), respectively. These two types of ants are applied in an alternate way. More precisely, the iteration-best ant is allowed as a default to deposit pheromone and the best-so-far ant is used only after a fixed number of iterations (i.e., every 25 iterations, more details in [25]).

C. Pheromone Trail Limits

The lower and upper limits τ_{min} and τ_{max} of the pheromone trail values are imposed. The τ_{max} value is bounded by $1/(\rho C^{bs})$, where C^{bs} is initially the solution quality of an estimated optimal tour (i.e., C^{nn} defined in Eq. (17)), and later on is updated whenever a new best-so-far solution quality is discovered. The τ_{min} value is set to $\tau_{min} = \tau_{max}(1 - \sqrt[n]{0.05})/((avg-1)\sqrt[n]{0.05})$ where avg is the average number of different choices available to an ant at each solution construction step.

D. Pheromone Reinitialization

Since only the best-so-far and iteration best ants are allowed to deposit pheromone, the algorithm may reach stagnation behavior quickly. To address this issue, the pheromone trails are occasionally reinitialized. In particular, whenever the stagnation behavior occurs or when no improved solution is found for a given number of iterations, the pheromone trails are uniformly reinitialized to the τ_{max} value.

Stagnation behavior is detected using the λ-branching factor that measures the distribution of the pheromone trail values [26]. The λ-branching factor is given by the number of arcs incident to node i satisfying the following condition: $\tau_{ij} \geq \tau_{min}^{i} + \lambda(\tau_{max}^{i} - \tau_{min}^{i})$, where τ_{max}^{i} and τ_{min}^{i} are, respectively, the maximum and minimum pheromone values on arcs incident to node i, and $\lambda \in [0,1]$ is a constant. The average λ-branching factor from all nodes' λ-branching factors indicates whether the search has entered stagnation behavior or not.

[4]The tour may not be feasible in terms of energy because energy stations are not considered in the tour. However, it does not affect the performance of the algorithm since an estimation is enough to generate a good initial pheromone trail value.

TABLE II
DETAILS OF THE GENERATED EVRP INSTANCES

Instance Name	# of Customers	# of Depots	# of Charging Stations	Gross Vehicle Weight ating (GVWR)
F-n45-k4.evrp	45	1	5	7.5 $tons$
F-n72-k4.evrp	72	1	10	7.5 $tons$
F-n135-k7.evrp	135	1	15	7.5 $tons$

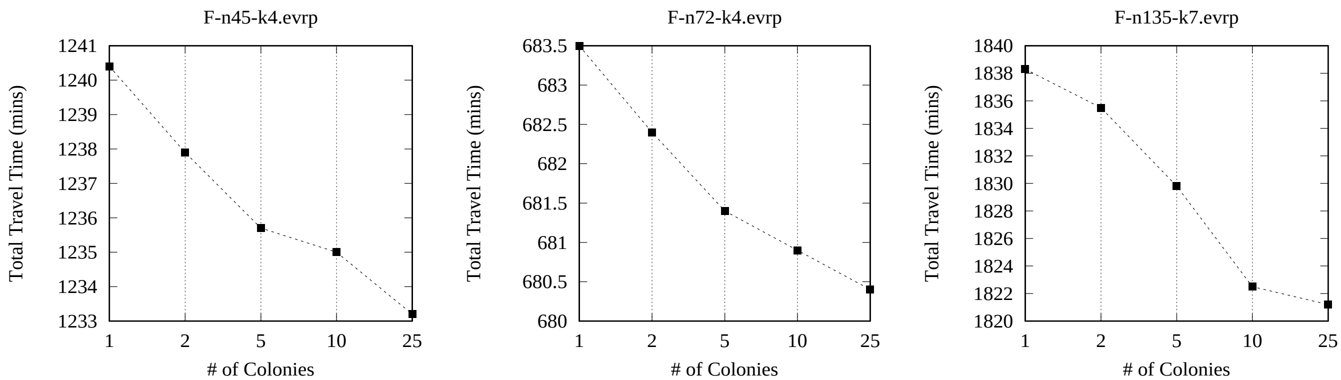

Fig. 1. Effect of increasing the number of colonies (given on x-axis) on the solution quality of the parallel $\mathcal{MM}$AS in minutes (given on y-axis).

E. Migration

There are two main concerns to address when designing migration policies: i) when to exchange information (i.e., migration schedule), and ii) where to exchange information (i.e., the topology of the colonies). Regarding the first concern, most policies exchange information after a fixed number of iterations. Regarding the second concern, the information is sent either to all the remaining colonies or to the worst-performing colony or to the nearest colonies (depending on the topology utilized [8]). Colonies typically exchange the best-so-far solution in migration policies [21], [22]. Furthermore, for all migration policies the exchange is accepted if and only if the received best-so-far solution is better than the colony's current best-so-far solution. Studies have shown that setting the frequency of exchange (in any topology) is challenging and significantly affects the quality of the output [3], [8].

In this work, we use a non-parametric migration policy that is not concerned in setting any of the two aforementioned concerns (since the aim of our studies is to investigate parallelism in $\mathcal{MM}$AS for EVRP and not to identify which migration policy performs best). The migration policy is defined as follows: whenever a new best-so-far solution is discovered by a colony, that colony will broadcast the best-so-far solution to the other colonies [9]. In this way, the other colonies will always receive a better best-so-far solution and the frequency of exchanging information will be selected dynamically. The topology of the colonies is in fact a star topology where the center colony is always the colony that has discovered the new best-so-far solution.

IV. SIMULATION STUDIES

A. Experimental Setup

1) Benchmark Instances: Three publicly available[5] EVRP benchmark instances F-n45-k4.evrp, F-n72-k4.evrp, and F-n135-k7.evrp previously used in [19] are utilized. The details of the problem instances are given in Table II. The gross vehicle weight rating (GVWR) is based on the specifications of Smith Newton's electric vans [23]. Considering that these EVs have a gross vehicle weight rating (GVWR) of 7.5 tons and their vehicle curb weight is 3.629 tons, their maximum cargo (or capacity) is estimated up to 3.871 tons (parameter Q in Eq. (7)).

2) Algorithmic Parameter Settings: The termination condition of the algorithms was set to $10e4$ iterations and 50 independent executions are performed on the same set of random seeds. The mean total travel time (in minutes) and the average number of iterations to find the best solution in an execution are recorded.

All ACO parameters used were set to the following values: $\alpha = 1, \beta = 5$ and $\rho = 0.8$. The ρ parameter was systematically tuned from these values $\rho \in \{0.02, 0.2, 0.4, 0.6, 0.8\}$ in a previous study [19]. Parallel $\mathcal{MM}$AS algorithms of 2, 5, 10 and 25 colonies of 50 ants each are developed to investigate the effect of increasing the number of parallel colonies. The experiments are presented in Fig 1. It must be noted that these parallel $\mathcal{MM}$AS algorithms are not utilizing the migration policy in this set of experiments.

[5]https://github.com/Mavrovouniotis/evrp-instances/

TABLE III
EXPERIMENTAL RESULTS REGARDING THE TOTAL TRAVEL TIME ($mins$) OF THE PARALLEL AND SEQUENTIAL VARIATIONS OF THE $\mathcal{MM}$AS ALGORITHM.

Instance & ACO	Best	Mean±Stdev	Worst	i_{avg}
F-n45-k4.evrp				
SEQ	1235.8	1240.4±2.9	1252.8	5374.6
SEQ2	1226.6	1236.1±2.8	1239.7	46339.7
PIR	1225.4	1235.0±3.3	1240.1	4441.2
MIG	1223.2	1234.4±3.6	1237.6	3986.5
F-n72-k4.evrp				
SEQ	679.9	683.5±2.9	690.7	4965.5
SEQ2	679.9	681.6±2.1	685.2	24713.6
PIR	679.9	680.9±1.6	685.1	2687.9
MIG	679.9	681.2±2.1	686.2	2618.7
F-n135-k7.evrp				
SEQ	1819.4	1838.3±8.4	1859.5	4832.4
SEQ2	1808.2	1824.4±8.3	1840.0	53409.6
PIR	1808.6	1822.5±5.8	1835.2	5571.3
MIG	1810.4	1822.4±6.3	1836.1	4015.6

TABLE IV
STATISTICAL TESTS REGARDING THE TOTAL TRAVEL TIME OF THE PARALLEL AND SEQUENTIAL VARIATIONS OF THE $\mathcal{MM}$AS ALGORITHM.

Comparisons	F-n45-k4.evrp	F-n72-k4.evrp	F-n135-k7.evrp
SEQ⇔SEQ2	−	−	−
PIR⇔SEQ	+	+	+
MIG⇔SEQ	+	+	+
PIR⇔SEQ2	+	∼	∼
MIG⇔SEQ2	+	∼	∼
PIR⇔MIG	∼	∼	∼

From Fig. 1 it can be observed that as the number of colonies increases the solution quality (in terms of total travel time) is improved. This is natural because the parallel variations are able to explore different areas of the search space simultaneously. Hence, these areas can be compared in order to select the one leading to the solution with the highest quality.

B. Results and Analysis

In our simulation studies, the parallel $\mathcal{MM}$AS algorithm with ten colonies using the migration policy described previously (MIG) is compared against the equivalent sequential $\mathcal{MM}$AS algorithm (SEQ) [24]. In order to investigate the effect of the migration between the colonies we develop a parallel variation of $\mathcal{MM}$AS, in which migration is not utilized. In other words, the colonies in this parallel $\mathcal{MM}$AS variation behave as the parallel independent runs (PIR) introduced in [20]. Also, another sequential $\mathcal{MM}$AS variation (SEQ2) is considered which runs the same overall time as the parallel variations, i.e., 10–times the computation time (or termination condition) of MIG or PIR with ten colonies [8].

The results of the aforementioned algorithms are given in Table III. "Best" and "Worst" are the minimum and maximum values, "Mean" and "Stdev" are the average and standard deviation, and "i_{avg}" is the average number of iterations required to find the best solution. Note that averages are taken over 50 executions. Pairwise comparisons using the Wilcoxon sum–rank statistical tests with confidence level of 95% are given in Table IV. The results are shown as "+", "−", and "∼" when the first algorithm is significantly better than the second one, when the second algorithm is significantly better than the first one, and when the two algorithms are not significantly different, respectively. The solution quality (in minutes) against the algorithmic iterations are plotted in Fig. 2. Note that the SEQ2 algorithm performs 10 times more iterations (which is not shown in the plots) than the rest of the algorithms. From Table III, Table IV, and Fig. 2 the following observations can be drawn.

1) Sequential vs Parallel: It can be clearly observed that MIG and PIR are significantly better than SEQ (see Table IV). This shows that $\mathcal{MM}$AS benefits from the parallelization of the colonies. This is because the different colonies used in MIG and PIR may search at different areas in the search space simultaneously promoting in this way exploration.

On the contrary, MIG and PIR are usually not significantly different than SEQ2 (see Table IV). This is to be expected because SEQ2 requires ten times more computation time than the two parallel algorithms in order to find a solution comparable to MIG and PIR. In fact, this becomes clear when observing the i_{avg} of SEQ2 in Table III which is significantly higher than PIR and MIG. The fact that the best solution is found after iteration $10e4$, which is the termination condition of the competing algorithms, confirms that SEQ2 requires much more time to locate the best output.

Moreover, from Fig. 2 it can be observed that up to iteration $10e4$, SEQ2 is basically identical to SEQ. Their difference is after the $10e4$ iteration, at which time the SEQ2 algorithm is able to find a better solution than the SEQ algorithm (see Table III). This is due to the pheromone reinitialization mechanism used in $\mathcal{MM}$AS acting as multiple restarts of the algorithm whenever stagnation behavior in the search is detected.

2) Independent vs Migration: It can be observed that MIG and PIR are not significantly different (see Table IV). This shows that the migration policy used in MIG to promote the cooperation between the colonies is not very effective. This is probably because broadcasting the best-so-far solution to the other colonies will most probably lead the colonies to search the same area in the search space, promoting in this way exploitation.

Although in terms of solution quality MIG is comparable to PIR, it is interesting to observe that it is better in terms of i_{avg} (see Table III). This shows that the cooperation among the colonies helps MIG in locating the best solution faster than the PIR which is not utilizing any migration policy. This can also be observed in Fig. 2, where MIG converges faster (during iterations 1000–3000) than PIR.

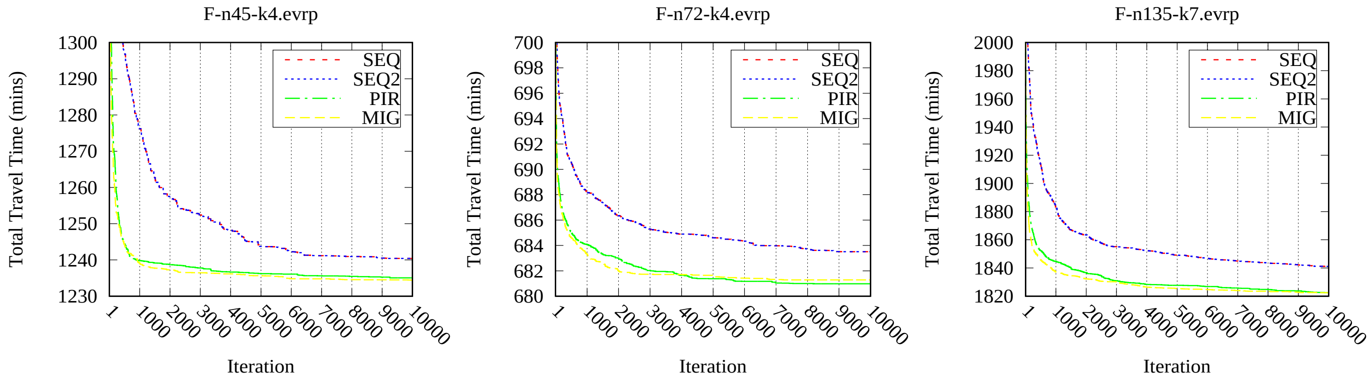

Fig. 2. Solution quality in minutes (given on y-axis) of the parallel and sequential variations of the $\mathcal{MM}$AS algorithm against the number of algorithmic iterations (given on x-axis).

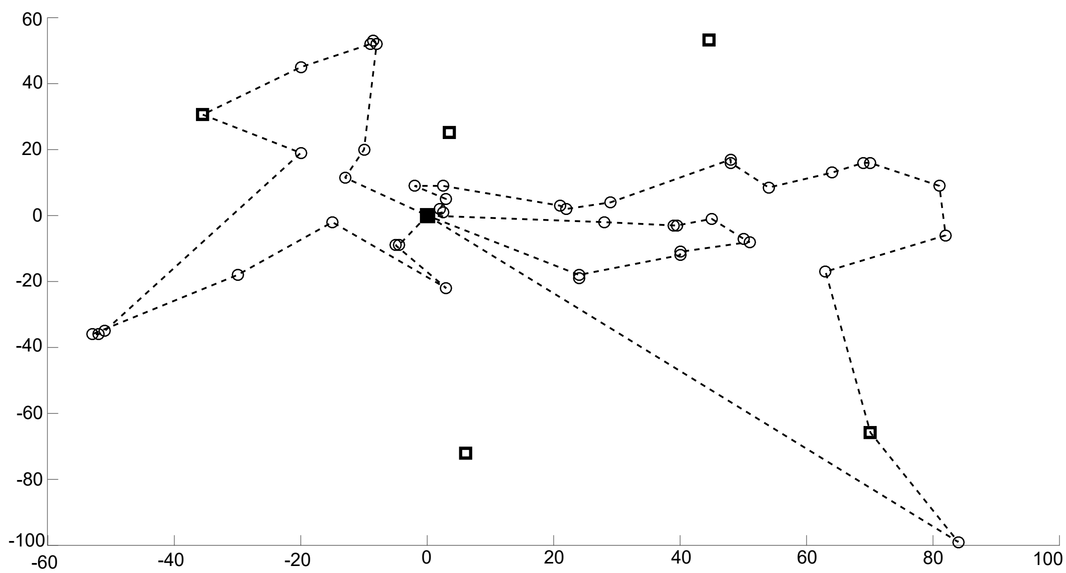

Fig. 3. Illustration of the solution generated by the `MIG` algorithm on the the F-n45-k4.evrp problem instance. The circles are the customers, white squares are the energy recharging stations and the black square is the central depot (which is also an energy recharging station).

The best solution (out of the 50 executions) generated by the `MIG` algorithm is plotted in Fig. 3, that consists of three routes operated by three EVs in which the two EVs have to visit a recharging station during their operation. The total travel time of the three generated routes is 1223.2 minutes.

V. Conclusions

In this work, a parallel ACO is applied to the EVRP to minimize the total travel time of a fleet of EVs. The parallel ACO uses several colonies that run independently. Each colony updates its own pheromone trails and maintains a separate search history. A non-parametric migration policy is used to allow the communication between the colonies. Our simulation studies demonstrate that: i) parallelism is beneficial when solving difficult optimization problems such as the EVRP, and ii) the communication between colonies contributes in locating the best output faster.

For future work, it will be interesting to consider migration policies that better balance exploration and exploitation. Also, parallel $\mathcal{MM}$AS algorithms may benefit from the use of heterogeneous colonies instead of homogeneous colonies [9], [27]. Finally, it will be interesting to convert the static EVRP model to a dynamic one. There are several dynamic factors that can affect the energy consumption of an EV such as the traffic conditions [28], the number of idle and acceleration situations [17], the angle of the roads [29], and the weather and cabin temperature [30].

References

[1] M. Garey and D. Johnson, *Computer and intractability: A guide to the theory of NP-completeness*. San Francisco: Freeman, 1979.

[2] E. Alba and M. Tomassini, “Parallelism and evolutionary algorithms,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 443–462, 2002.

[3] C. Twomey, T. Stützle, M. Dorigo, M. Manfrin, and M. Birattari, “An analysis of communication policies for homogeneous multi-colony ACO algorithms,” *Information Sciences*, vol. 180, no. 12, pp. 2390–2404, 2010.

[4] R. Tanese, “Parallel genetic algorithms for a hypercube,” in *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*. Hillsdale, NJ, USA: L. Erlbaum Associates Inc., 1987, pp. 177–183.

[5] D. Whitley, S. Rana, and R. B. Heckendorn, “The Island Model Genetic Algorithm: On Separability, Population Size and Convergence,” *Journal of Computing and Information Technology*, vol. 7, pp. 33–47, 1998.

[6] J. Apolloni, G. Leguizamón, J. García-Nieto, and E. Alba, “Island Based Distributed Differential Evolution: An Experimental Study on Hybrid Testbeds,” in *Proceedings of the IEEE International Conference on Hybrid Intelligent Systems*, 2008, pp. 696–701.

[7] R. Michel and M. Middendorf, “An island model based ant system with lookahead for the shortest supersequence problem,” in *Parallel Problem Solving from Nature - PPSN V*, ser. Lecture Notes in Computer Science, A. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, Eds. Springer Berlin Heidelberg, 1998, vol. 1498, pp. 692–701.

[8] M. Manfrin, M. Birattari, T. Stützle, and M. Dorigo, “Parallel ant colony optimization for the traveling salesman problem,” in *Ant Colony Optimization and Swarm Intelligence*, M. Dorigo, L. M. Gambardella, M. Birattari, A. Martinoli, R. Poli, and T. Stützle, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 224–234.

[9] M. Mavrovouniotis, S. Yang, and X. Yao, “Multi-colony ant algorithms for the dynamic travelling salesman problem,” in *2014 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*, Dec 2014, pp. 9–16.

[10] 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.

[11] A. Siemiński, “Using hyper populated ant colonies for solving the TSP,” *Vietnam Journal of Computer Science*, vol. 3, no. 2, pp. 103–117, May 2016.

[12] K. Doerner, R. Hartl, G. Kiechle, M. Lucka, and M. Reimann, “Parallel ant systems for the capacitated vehicle routing problem,” in *Evolutionary Computation in Combinatorial Optimization*, ser. Lecture Notes in Computer Science, J. Gottlieb and G. Raidl, Eds. Springer Berlin Heidelberg, 2004, vol. 3004, pp. 72–83.

[13] E.-G. Talbi, O. Roux, C. Fonlupt, and D. Robillard, “Parallel ant colonies for the quadratic assignment problem,” *Future Generation Computer Systems*, vol. 17, no. 4, pp. 441–449, 2001.

[14] M. Pedemonte, S. Nesmachnow, and H. Cancela, “A survey on parallel ant colony optimization,” *Applied Soft Computing*, vol. 11, no. 8, pp. 5181–5197, 2011.

[15] T. E. abd Tonci Carić, “A survey on the electric vehicle routing problem: variants and solution approaches,” *Journal of Advanced Transportation*, vol. 2019, 2019.

[16] A. Afroditi, M. Boile, S. Theofanis, E. Sdoukopoulos, and D. Margaritis, “Electric vehicle routing problem with industry constraints: Trends and insights for future research,” *Transportation Research Procedia*, vol. 3, pp. 452–459, 2014.

[17] R. Abousleiman and O. Rawashdeh, “Energy-efficient routing for electric vehicles using metaheuristic optimization frameworks,” in *2014 17th IEEE Mediterranean Electrotechnical Conference*, 2014, pp. 298–304.

[18] D. Kosmanos, L. Maglaras, M. Mavrovouniotis, S. Moschoyiannis, A. Argyriou, A. Maglaras, and H. Janicke, “Route optimization of electric vehicles based on dynamic wireless charging,” *IEEE Access*, vol. 6, pp. 42 551–42 565, 2018.

[19] 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)*, Nov 2018, pp. 1234–1241.

[20] T. Stützle, *Parallelization strategies for Ant Colony Optimization*. Springer Berlin Heidelberg, 1998, pp. 722–731.

[21] K. Doerner, R. Hardl, S. Benkner, and M. Lucka, “Parallel cooperative saving based ant colony optimization-multi search and decomposition approaches,” *Parallel Processing Letters*, vol. 16, no. 3, pp. 351–369, 2006.

[22] M. Middendorf, F. Reischle, and H. Schmeck, “Information exchange in multi colony ant algorithms,” in *Parallel and Distributed Processing*, ser. Lecture Notes in Computer Science, J. Rolim, Ed. Springer Berlin Heidelberg, 2000, vol. 1800, pp. 645–652.

[23] J. Lin, W. Zhou, and O. Wolfson, “Electric vehicle routing problem,” *Transportation Research Procedia*, vol. 12, pp. 508 – 521, 2016.

[24] T. Stützle and H. Hoos, “$\mathcal{MAX}$–$\mathcal{MIN}$ ant system and local search for the traveling salesman problem,” in *Proceedings of 1997 IEEE International Conference on Evolutionary Computation*, April 1997, pp. 309–314.

[25] T. Stützle and H. H. Hoos, “$\mathcal{MAX}$–$\mathcal{MIN}$ ant system,” *Future Generation Computer Systems*, vol. 16, no. 8, pp. 889–914, June 2000.

[26] L. M. Gambardella and M. Dorigo, “Ant-Q: A reinforcement learning approach to the traveling salesman problem,” in *Machine Learning Proceedings 1995*, A. Prieditis and S. Russell, Eds. San Francisco, CA: Morgan Kaufmann, 1995, pp. 252–260.

[27] A. Mambrini, D. Sudholt, and X. Yao, “Homogeneous and heterogeneous island models for the set cover problem,” in *Parallel Problem Solving from Nature - PPSN XII*, ser. Lecture Notes in Computer Science, C. Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, and M. Pavone, Eds. Springer Berlin Heidelberg, 2012, vol. 7491, pp. 11–20.

[28] S. Shao, W. Guan, B. Ran, Z. He, and J. Bi, “Electric vehicle routing problem with charging time and variable travel time,” *Mathematical Problems in Engineering*, vol. 2017, pp. 1–13, 2017.

[29] S. Shao, W. Guan, and J. Bi, “Electric vehicle-routing problem with charging demands and energy consumption,” *IET Intelligent Transport Systems*, vol. 12, no. 3, pp. 202–212, 2018.

[30] R. Abousleiman and O. Rawashdeh, “Electric vehicle modelling and energy-efficient routing using particle swarm optimisation,” *IET Intelligent Transport Systems*, vol. 10, no. 2, pp. 65–72, 2016.