A Bi-level Ant Colony Optimization Algorithm for Capacitated Electric Vehicle Routing Problem

Ya-Hui Jia, *Member, IEEE,* Yi Mei, *Senior Member, IEEE,* and Mengjie Zhang, *Fellow, IEEE*

***Abstract*—The development of electric vehicle techniques has led to a new vehicle routing problem called capacitated electric vehicle routing problem (CEVRP). Because of the limited number of charging stations and the limited cruising range of electric vehicles, not only the service order of customers but also the recharging schedules of electric vehicles should be considered. However, solving these two aspects of the problem together is very difficult. To address the above issue, we treat CEVRP as a bi-level optimization problem and propose a novel bi-level ant colony optimization algorithm in this paper, which divides CEVRP into two levels of sub-problem: 1) capacitated vehicle routing problem and 2) fixed route vehicle charging problem. For the upper-level sub-problem, the electricity constraint is ignored and an order-first split-second max-min ant system algorithm is designed to generate routes that fulfill the demands of customers. For the lower-level sub-problem, a new effective heuristic is designed to decide the charging schedule in the generated routes to satisfy the electricity constraint. The objective values of the resultant solutions are used to update the pheromone information for the ant system algorithm in the upper level. Through good orchestration of the two components, the proposed algorithm can significantly outperform state-of-the-art algorithms on a wide range of benchmark instances.**

***Index Terms*—Electric Vehicle Routing Problem, Capacitated Vehicle Routing Problem, Vehicle Charging Problem, Ant Colony Optimization, Combinatorial Optimization.**

I. Introduction

TRANSPORTATION has been one of the main sources of greenhouse gas emission for decades [1]. Recently, the rapid development of electric vehicles (EVs) has provided an applicable alternative to conventional fossil-fueled vehicles [2], [3]. According to [4], replacing conventional vehicles with EVs can contribute a lot to the reduction of greenhouse gas emissions. Considering the increasingly dire environmental problems and governmental policies [5], many logistics companies like DHL, JD, and FedEx have started using EVs instead of conventional fossil-fueled vehicles in their delivery businesses [6], [7].

From the academic perspective, the services provided by logistics companies are usually modeled as vehicle routing problems (VRPs). In the past several decades, there are many different variants of VRP that have been extensively studied [8]–[12], such as capacitated VRP (CVRP) [13], [14], VRP with time window [9], [15], VRP with pickup and delivery [12], multi-objective VRP [16], [17], and dynamic VRP [10]. Most of these variants are built based on conventional vehicles. Since the cruising range of conventional vehicles is quite long and gas stations are widespread in modern cities, the refueling problem is usually not considered in these traditional VRPs. However, the cruising range of EVs cannot match conventional vehicles yet and there are not many charging stations even in the urban area of a city at present [18]. These problems make it much harder to manage a fleet of EVs than conventional vehicles, leading to a new category of VRP called electric VRP (EVRP) [19]. In EVRP, two aspects should be considered: 1) the service order of customers and 2) the recharging schedule of each vehicle. Since these two aspects are highly interdependent, most of the existing methods for traditional VRPs cannot be directly applied to EVRPs.

EVRP is a quite new subject. Most of the existing studies focus on proposing and defining new problem models, such capacitated EVRP (CEVRP) [6], EVRP with time window (EVRPTW) [19], [20], EVRP with pickup and delivery [21], EVRP with non-linear charging [22], and EVRP with shared charging stations [23]. Compared with proposing problems, the effort that was spent on designing algorithms is relatively little. Therefore, most of these problems have not been extensively studied. Throughout the research history of VRP, we know that sufficient research of the fundamental CVRP will benefit the other variants a lot. Following the same path, CEVRP is studied in this paper as the basic EVRP model to contribute to the fundamental research of EVRP [6]. Following the definition of traditional CVRP, the objective of CEVRP is also defined as the minimization of the total travel distance of EVs. EVs depart from the depot to serve the customers. Every customer should be served exactly once. Each EV cannot be overloaded or run out of electricity during its journey. If a vehicle cannot finish the journey by only using the initial electricity, it can recharge its battery in charging stations. After serving all customers, all EVs should go back to the depot.

So far, most of the existing methods proposed to solve EVRPs fall into two categories. In the first category, scholars transferred the EVRPs into mixed integer linear programming (MILP) problems and applied commercial software like CPLEX [24] to solve them [25], [26]. The commercial software can generate very good solutions on small-scale problems, but the high computational complexity makes them inefficient on larger problems with more than 50 customers [22]. Individual-based meta-heuristic algorithms [27] are the second category of algorithms, including variable neighborhood search (VNS) [20], iterative local search (ILS) [28],

This work was supported in part by the Marsden Fund of New Zealand Government under Contracts VUW1509 and VUW1614, the Science for Technological Innovation Challenge (SfTI) fund under grant E3603/2903, and the MBIE SSIF Fund under Contract VUW RTVU1914.

The authors are all with School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand (email: jiayahui@ecs.vuw.ac.nz; yi.mei@ecs.vuw.ac.nz; mengjie.zhang@ecs.vuw.ac.nz). (*Corresponding author: Ya-Hui Jia*)

[29], and adaptive large neighborhood search (ALNS) [30], [31]. These algorithms can be efficient but the quality of the final solution highly depends on the perturbation and the local search operators that are applied. Also, since there is only one solution under operation, these algorithms are sensitive to the quality of the initial solution. If the applied operators are not appropriate or the initial solution is not good, the algorithm is easy to be trapped into poor local optima [27].

Ant colony optimization (ACO) algorithms have been proven effective for many routing problems like traveling salesman problem (TSP) [32], [33], VRPs [34], and arc routing problems (ARPs) [35]. As a population-based meta-heuristic algorithm, ACO has better adaptability than the individual-based meta-heuristic algorithms since it does not depend on specific perturbation or local search operators, and it is less sensitive to the initial solution [36], [37]. Among several ACO variants [38]–[40], the max-min ant system (MMAS) is adopted in this paper because of its better capability than the others [36].

However, directly using MMAS to generate a integrated solution containing both service orders and recharging schedules for CEVRP is difficult. Guo *et al.* [41] and Shao *et al.* [42] have tried to solve the two aspects of EVRP simultaneously by encoding the customer order and charging station order together in their genetic algorithms (GAs). However, this encoding scheme leads to a huge search space and also the traditional local search methods like 2-opt cannot be applied since they cannot handle the routes with charging stations. These drawbacks make the proposed GAs ineffective on large-scale EVRPs.

Considering these drawbacks, we solve CEVRP in a way of bi-level optimization in this paper. The main contributions are:

- A bi-level ACO (BACO) is designed by considering CEVRP as a bi-level optimization problem, in order to reduce the search space and focus more on the promising regions. The sub-problems of the two levels are CVRP and fixed route vehicle charging problem (FRVCP), corresponding to the service order of customers and the recharging schedules of vehicles.
- An order-first split-second MMAS (OS-MMAS) algorithm is proposed to generate capacity-feasible routes for the upper-level sub-problem CVRP, deciding the service order of customers.
- A new heuristic method called removal heuristic (RH) is designed to solve the lower-level sub-problem FRVCP, deciding the recharging schedules of vehicles. The fitness values of the integrated solutions after inserting charging stations will be used to update the pheromone information of OS-MMAS. In addition, based on RH, a restricted enumeration method is further proposed to refine the recharging schedule of the final solution.

The rest of this paper is organized as follows. Section II gives the formal description of CEVRP. Then, Section III shows the related works. The proposed algorithm BACO is explained in Section IV. Finally, the experimental studies are conducted in Section V and the conclusions are drawn in Section VI.

II. Problem Definition

Given a fleet of homogeneous EVs, the goal of CEVRP is to find an optimal set of routes that minimize the total traveling distance subject to several constraints. CEVRP can be formally defined on a fully connected weighted undirected graph $G = (V, E)$. $V = \{0\} \cup I \cup \hat{F}$ represents the set of nodes in the graph. 0 is the index of the depot. I represents the customers. Each customer i has a fixed cargo demand c_i. $\hat{F}$ is an extended set of charging stations that contains β_i copies of each charging station $i \in F$. The creation of the copies of charging stations allows each charging station to be visited multiple times [43]. β_i is set to $2|I|$ since in the worst case, each EV needs to visit each station once on both ways to serving a customer and returning to depot [44]. $E = \{(i, j)|i, j \in V, i \neq j\}$ is the set of arcs. Each arc (i, j) is associated with a weight representing the distance between i and j, denoted as d_{ij}. Each EV has a maximum capacity of cargo demand C and a maximum battery capacity Q. The consumption rate of the battery is denoted as h. For each arc (i, j), an EV will consume the amount $h \cdot d_{ij}$ of the battery to traverse it. Through introducing two more variables u_i and y_i that respectively represent the remaining carrying capacity and remaining battery level of an EV when it arrives at node $i \in V$, the mathematical definition of CEVRP is given in [6] as follows:

$$\min f(\mathbf{x}) = \sum_{i \in V, j \in V, i \neq j} d_{ij} x_{ij}, \tag{1}$$

s.t.

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

$$\sum_{j \in V, i \neq j} x_{ij} \leq 1, \forall i \in \hat{F}, \tag{3}$$

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

$$u_j \leq u_i - c_i x_{ij} + C(1 - x_{ij}), \forall i \in V, \forall j \in V, i \neq j, \tag{5}$$

$$0 \leq u_i \leq C, \forall i \in V, \tag{6}$$

$$y_j \leq y_i - h d_{ij} x_{ij} + Q(1 - x_{ij}), \forall i \in I, \forall j \in V, i \neq j, \tag{7}$$

$$y_j \leq Q - h d_{ij} x_{ij}, \forall i \in \hat{F} \cup \{0\}, \forall j \in V, i \neq j, \tag{8}$$

$$0 \leq y_i \leq Q, \forall i \in V, \tag{9}$$

$$x_{ij} \in \{0, 1\}, \forall i \in V, \forall j \in V, i \neq j. \tag{10}$$

The objective (1) of the problem is to minimize the total travel distance of all EVs. Constraint (2) stipulates that every customer should be served only once. Constraint (3) indicates that a charging station can be visited multiple times. Constraint (4) guarantees that every EV must leave each customer served by it. Constraints (5) and (6) stipulate that EVs cannot be overloaded during their journeys. We call this constraint "capacity constraint" in this paper. Constraints (7), (8), and (9) stipulate that EVs cannot run out of electricity during their journeys. Correspondingly, we call this constraint "electricity constraint". Constraint (10) defines the domain of x_{ij}. If the arc (i, j) is traveled by an EV, x_{ij} equals to one, otherwise, it equals to zero. Despite the explicit constraints displayed by the

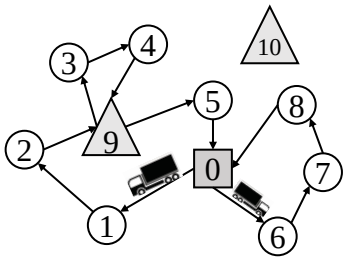


Fig. 1. An example solution of CEVRP with two routes.

formulas, the formulas also imply that all EVs should depart from and return to the depot. Also, the constraints (7), (8), and (9) assume that EVs are always fully charged in charging stations. It should be noted that the CEVRP studied in this paper is exactly the EVRP proposed in [6], but, to conform to the nomenclature of traditional VRPs, the word "capacitated" is added to the name representing the capacity constraint. Fig. 1 shows an example consisting of two routes. An EV visits the charging station twice during its journey, and another EV does not visit any charging station.

III. Related Works

In this section, the methods proposed to solved EVRPs are reviewed. The predecessor of EVRP is Green VRP (GVRP) [43], which is a routing model for alternative fuel vehicles. Based on GVRP, scholars in the operations research community and transportation research community replaced the alternative fuel vehicles with EVs and started to study EVRP.

A popular method to handle EVRPs is to model EVRPs as MILP problems and use commercial software such as CPLEX and Gurobi to solve them. Lin [25] used CPLEX to solve an EVRP with the influence of vehicle load on energy consumption. The objective was to minimize the total money cost of delivery including the driver's hourly wage and the charging cost. A scenario containing only 13 customers was tested. Montoya *et al.* [22] proposed a variant called EVRP with non-linear charging that adopted different kinds of charging stations. Also, the recharging process is modeled as a non-linear function. They tried to use Gurobi to solve the problem but found it could only solve the problems with less than 50 customers within the given time. Chen *et al.* [21] proposed a mixed-integer quadratically constrained programming model for the EVRP with pick and delivery and used CPLEX to solve the problem, but the test case only considered less than 20 customers. Xiao *et al.* [45] focused on developing the energy consumption model to an EVRPTW to minimize the total cost like [25]. Also, the experiments showed that only the cases with less than 30 customers could be solved by CPLEX in the given time. Although using mathematical programming methods to solve the problem is an easy way to get high-quality solutions, the experiments in the literature show that those methods are not efficient enough to solve the problems with more than 50 customers due to the high time complexity.

Individual-based meta-heuristic algorithms were also widely adopted to solve EVRPs, such as VNS, ILS, and ALNS. VNS is the first one that was applied to EVRP. In 2014, both Afroditi *et al.* [19] and Schneider *et al.* [20] proposed their EVRPTW models. Schneider proposed a VNS heuristic with a tabu search method to solve the problem. Following their researches, Bruglieri *et al.* [46], [47] changed the objective of EVRPTW from the total traveling distance to the total time. They combined VNS with a local branching method to solve the problem. Recently, Woller *et al.* proposed a VNS algorithm that won the IEEE WCCI2020 competition on EC for the EVRP [6]. Regarding ILS, Zhang *et al.* [29] modeled the energy consumption of an EV by considering many factors and proposed an ILS algorithm using ACO as the perturbation operator to solve the problem. Generally, ACO is recognized as a population-based meta-heuristic, but in their algorithm, there is only one ant. Thus, it is still an individual-based algorithm rather than a population-based algorithm. Besides using Gurobi, Montoya *et al.* [22] also proposed an ILS algorithm to solve the EVRP with non-linear charging. Following Montoya's work, Froger *et al.* [44] further improved the model and the heuristic algorithm for FRVCP, but still used the same routes generated by the previous ILS algorithm. ALNS is another popular individual-based meta-heuristic algorithm. Keskin and Çatay [31], [48] proposed their EVRPTW models by considering the partial recharging situation and different objectives. Both works applied the same ALNS algorithm. Koç *et al.* [23] considered a situation where several companies jointly invest in charging stations and proposed the problem EVRP with shared charging stations (EVRP-SCS). The EVRP-SCS includes not only the routing of EVs but also locating the charging stations. They used ALNS and CPLEX to solve different parts of the problems. Generally, those individual-based meta-heuristic algorithms typically solved the problems effectively, but the performance highly depends the applied perturbation/destruction operators and local search methods, which are problem-dependent. Designing these operators and methods needs high expertise. If the applied perturbation/destruction operators and local search methods are not appropriate, the algorithm would encounter the premature convergence problem frequently especially for the large and complex problems [27]. Also, the quality of the initial solution has a big influence to the final solution that should be carefully generated.

Considering the good adaptability of evolutionary computation (EC), several scholars also tried to used EC algorithms to solve EVRPs. Liu *et al.* [49] proposed a differential evolution algorithm to solve an EV charging scheduling problem, but different from the previous works, it mainly focused on the charging scheduling of EVs from point to point rather than serving a set of costumers. Guo [41] and Shao [42] proposed two GAs to solve the EVRPTW. To decide the customer order and the recharging order simultaneously, Guo encoded the customers and the stations together in GA and made an unrealistic assumption that each station can only be visited once. This assumption makes the algorithm only be applicable to some specific small-scale problems. Shao addressed this issue by setting copies to each station, but this behavior led to another severe issue of huge and redundant search space.

Overall, we have found that the state-of-the-art approaches to EVRPs still have limitations. The mathematical methods

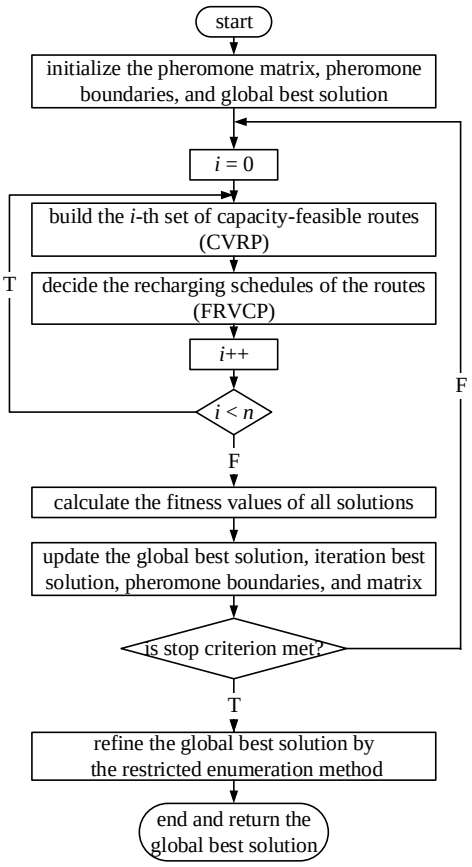


Fig. 2. Flowchart of BACO.

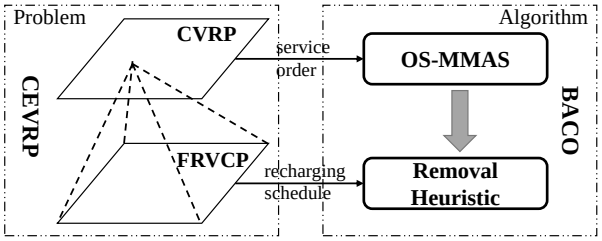


Fig. 3. Components of BACO and their relationships with the sub-problems.

are currently not efficient enough to solve EVRPs with more than 50 customers. Individual-based meta-heuristic algorithms are easy to be trapped into local optima if the perturbation/destruction operators, local search methods, and initial solution generation methods are not elaborately designed. EC algorithms have not incorporated a good way to handle both the customer order and recharging order simultaneously. Considering these drawbacks, in this paper, we propose a bi-level algorithm named BACO to solve the CEVRP.

IV. Bi-level Ant Colony Optimization

In this section, we first give the overall process of BACO. Then, the three components of BACO including the OS-MMAS for the upper-level CVRP, the RH for the lower-level FRVRP, and the final solution refinement procedure are described. Finally, the theoretical time complexity and the design philosophy of the algorithm are analyzed.

A. Process of BACO

The overall process of BACO is shown in Fig. 2. As the flowchart shows, the components and parameters of the algorithm are initialized in the first place. Then, in every generation, $n = |I| + 1$ ants build their solutions one by one. The generated solutions are evaluated to update the global best solution, the iteration best solution, the pheromone boundaries, and the pheromone matrix. If the stopping criterion is met, an restricted enumeration method is applied to the global best solution to make a final refinement. Finally, the whole algorithm ends and returns the global best solution.

Fig. 3 demonstrates the bi-level structure of CEVRP and the relationship with BACO. Corresponding to the two levels of sub-problems, it takes two steps to generate a feasible solution in BACO. First, the CEVRP is degraded to a corresponding CVRP by removing the electricity constraint and charging stations. Considering only depot and customers, the OS-MMAS algorithm is applied to generate the capacity-feasible routes of the corresponding CVRP. Then, given the capacity-feasible routes, we take the electricity constraint into consideration. The problem of finding the optimal recharging schedule for a fixed route is called FRVCP. To solve FRVCP effectively and efficiently, the RH algorithm is proposed.

After generating the solutions, the objective values are evaluated to update the global best solution and the pheromone matrix. RH is a deterministic algorithm. It provides a single-value mapping relationship between the routes of the corresponding CVRP and the fitness values of the CEVRP. Thus, we can directly use the fitness values of the solutions to update the pheromone matrix of OS-MMAS. The formulation of the two sub-problems and their relationships with CEVRP are further analyzed in the supplemental material to show how BACO treats the problem in a bi-level way.

B. OS-MMAS for Route Construction

Pheromone Setting: In OS-MMAS, a pheromone matrix is maintained to guide the ants to construct the routes. At first, we generate the routes for the corresponding CVRP without considering the charging stations. Thus, the size of the pheromone matrix Φ is $n \times n$ where $n = |I| + 1$ is the total number of customers plus the depot. Each element $\varphi_{ij} \in \Phi$ represents the pheromone value of traveling from i to j. Since the applied ACO algorithm is MMAS [40], there are two boundaries $\varphi_{\min}$ and $\varphi_{\max}$ of the pheromone values that are used to maintain the exploration ability of the algorithm, $\varphi_{\min} \leq \varphi_{ij} \leq \varphi_{\max}$. The values of these two boundaries are [40]:

$$\varphi_{\max} = \frac{1}{(1-\rho) \cdot f(\mathbf{x}^{\text{gb}})}, \tag{11}$$

$$\varphi_{\min} = \frac{\varphi_{\max}(1 - \sqrt[n]{pr})}{(n/2 - 1)\sqrt[n]{pr}}, \tag{12}$$

where ρ is the parameter used to control the pheromone evaporation speed. $\mathbf{x}^{\text{gb}}$ is the global best solution. pr is a parameter that is usually set to 0.05. To generate the initial $\mathbf{x}^{\text{gb}}$, in BACO, the convex hull insertion method is firstly applied to generate a giant tour. Then, this giant tour is split into a set

of capacity-feasible routes. Afterward, these routes are fixed to be an electricity-feasible solution by RH. The value of the initial pheromone φ_0 is set equal to the value of $\varphi_{\max}$.

Route Construction: The route construction process of OS-MMAS is shown in Algorithm 1, which can be divided into two sub-processes: the order-first sub-process to generate a giant tour (line 1-11) and the split-second sub-process to split the tour into capacity-feasible routes (line 12-30).

First, the giant tour $\mathbf{r}$ is initialized empty and a set $\hat{I}$ containing the depot and the customers is maintained (line 1). A random node is selected from $\hat{I}$ as the starting point. It is added into the tour $\mathbf{r}$ and removed from $\hat{I}$ (line 2-4). The remain nodes in $\hat{I}$ will be chosen one by one until the complete tour is constructed (line 5-10). In each step, the next node j is selected from $\hat{I}$ through the roulette wheel selection strategy (line 7). The probability of going to j from the current node i is [40]:

$$p_{ij} = \frac{\varphi_{ij}^{\alpha}/d_{ij}^{\beta}}{\sum_{l\in I_s}\varphi_{il}^{\alpha}/d_{il}^{\beta}}, \text{ if } j \in \hat{I} \tag{13}$$

where α and β are two parameters to balance the importance between the pheromone value and the distance value. From (13), we can know that the node with larger pheromone value and shorter distance from i is preferred. Afterward, this selected node j is added into the tour $\mathbf{r}$ and removed from $\hat{I}$ (line 8-9). When all customers and the depot have been selected, we rotate $\mathbf{r}$ to let the depot be the first node in the tour (line 11).

Then, the tour $\mathbf{r}$ is split into several capacity-feasible routes by using the split algorithm proposed in [50]. Given a tour $\mathbf{r}$ starting from depot 0, the splitting algorithm maintains two values V_i and P_i for each node $\mathbf{r}_i$ representing the minimum total distance of all generated routes until $\mathbf{r}_i$ and the index of its predecessor in the route (line 12). Through two nested loops (line 13-29), the algorithm finds the predecessor for each node that leads to the minimum overall distance which equals $V_{|I|}$. Then, from the last node in $\mathbf{r}$, all routes can be generated through a backtrace process (line 30). Essentially, the splitting algorithm enumerates all capacity-feasible routes and finds the best segmentation of the giant tour.

After generating all routes, the 2-opt algorithm is applied as the local search method to each route to further improve the solution (line 31).

Pheromone Updating: After obtaining the set of routes, the recharging schedules of the routes will be determined to obtain a feasible solution of the original CEVRP. The process of inserting charging stations into the routes will be introduced later. At the end of each iteration, these feasible solutions will be evaluated. The iteration best solution $\mathbf{x}^{\text{ib}}$ and the corresponding giant tour $\mathbf{r}^{\text{ib}}$ are recorded. If the iteration best $\mathbf{x}^{\text{ib}}$ is better than the global best $\mathbf{x}^{\text{gb}}$, $\mathbf{x}^{\text{gb}}$ will be updated. $\varphi_{\max}$ and $\varphi_{\min}$ will be also updated according to (11) and (12). The pheromone matrix is updated according to:

$$\varphi_{ij}(t+1) = \rho\cdot\varphi_{ij}(t) + \Delta\varphi_{ij}^{\text{best}}, \tag{14}$$

$$\Delta\varphi_{ij}^{\text{best}} = \begin{cases} 1/f(\mathbf{x}^{\text{ib}}) & \text{if } x_{ij} == 1, \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Algorithm 1 Route Construction

Input: pheromone matrix Φ, arc set E, customer set I, maximum capacity C.
Output: a set of capacity-feasible routes $\mathbf{\Gamma}$

1: initialize $\mathbf{r}$ empty; $\hat{I} = \{0\} \cup I$; //order-first
2: Randomly select a node i in $\hat{I}$;
3: append i to $\mathbf{r}$;
4: remove i from $\hat{I}$;
5: **while** $\hat{I}$ is not empty **do**
6: take the last node i in $\mathbf{r}$;
7: $j = roulette\_wheel\_selection(\hat{I}, i)$;
8: append j to $\mathbf{r}$;
9: remove j from $\hat{I}$;
10: **end while**
11: let $\mathbf{r}_0 = 0$ by rotating $\mathbf{r}$;
12: $V_0 = 0; V_{1\to|I|} = +\infty; P_{0\to|I|} = 0$; //split-second
13: **for** $i = 1 \to |I|$ **do**
14: $j = i; tc = 0; td = 0$;
15: **repeat**
16: $tc = tc + c_{\mathbf{r}_j}$;
17: **if** $i == j$ **then**
18: $td = d_{0,\mathbf{r}_j} + d_{\mathbf{r}_j,0}$;
19: **else**
20: $td = td - d_{0,\mathbf{r}_{j-1}} + d_{\mathbf{r}_{j-1},\mathbf{r}_j} + d_{\mathbf{r}_j,0}$;
21: **end if**
22: **if** $tc \le C$ **then**
23: $j = j + 1$;
24: **if** $V_{i-1} + td < V_j$ **then**
25: $V_j = V_{i-1} + td; P_j = i - 1$;
26: **end if**
27: **end if**
28: **until** $j > |I|$ or $tc > C$
29: **end for**
30: $\mathbf{\Gamma} = back\_trace(P_{0\to|I|}, \mathbf{r})$;
31: Do local search on each route $\Gamma \in \mathbf{\Gamma}$;
32: **return** $\mathbf{\Gamma}$;

C. Removal Heuristic for Fixed Route Vehicle Charging

After the route construction, some routes are not electricity-feasible. We need to insert charging stations into these routes to make them electricity-feasible. To this end, a new heuristic algorithm called RH is proposed to solve the lower-level sub-problem FRVCP, which is shown in Algorithm 2. It can be divided into two stages: inserting (line 1-6) and removing (line 7-23).

In the inserting stage, between each pair of successive nodes Γ_i and Γ_{i+1}, a station s that brings minimum extra traveling distance is found and inserted (line 2-6). The electricity constraint is considered so that the resultant route Γ is electricity-feasible after the inserting stage (line 3).

Then, in the removing stage, we remove redundant stations one by one. For each station s in Γ (line 9), we first judge whether Γ would be still electricity-feasible after removing s (line 11). If it can be removed without violating the electricity constraint, we record the distance saved by removing this station (line 12-15). After checking all the stations, the station

Algorithm 2 Removal-Heuristic

Input: a route $\Gamma \in \mathbf{\Gamma}$ starting and ending at 0 that is not electricity-feasible, arc set E, customer set I, charging station set F, maximum battery capacity Q, battery consumption rate h.

Output: an electricity-feasible route Γ

1: $D = Q/h$; $k = |\Gamma|$; $ld = D$;
2: **for** $i = 0 \rightarrow k-2$ **do**
3: $s = \text{argmin}_{l \in F} d_{\Gamma_i,l} + d_{l,\Gamma_{i+1}}$ s.t. $d_{\Gamma_i,l} < ld$;
4: insert s between Γ_i and Γ_{i+1};
5: $ld = D - d_{s,\Gamma_{i+1}}$;
6: **end for**
7: **while** Γ contains charging stations **do**
8: $ds' = 0$; $s' = 0$;
9: **for** each station s in Γ **do**
10: denote the index of s as ps;
11: **if** Γ is electricity-feasible after removing s **then**
12: $ds = d_{\Gamma_{ps-1},s} + d_{s,\Gamma_{ps+1}} - d_{\Gamma_{ps-1},\Gamma_{ps+1}}$;
13: **if** $ds' < ds$ **then**
14: $s' = s$; $ds' = ds$;
15: **end if**
16: **end if**
17: **end for**
18: **if** $ds' \neq 0$ **then**
19: remove s' from Γ;
20: **else**
21: break;
22: **end if**
23: **end while**
24: **return** Γ;

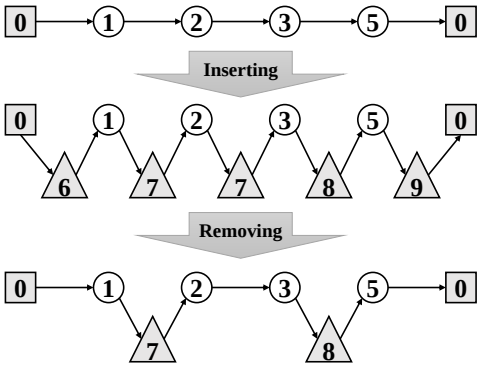


Fig. 4. Process of the removal heuristic.

that can be legally removed and bring maximal distance saving is removed (line 18-19). If there is no station that can be removed, the whole process stops and returns the electricity-feasible route Γ.

An example is shown in Fig. 4 to demonstrate RH. Given a route (0,1,2,3,5,0), in the inserting stage RH first inserts five charging stations $(6, 7, 7, 8, 9)$ to the route between each pair of successive nodes. Then, in the removing stage, three stations are removed and only two stations are reserved.

D. Restricted Enumeration Method

Since FRVCP is also a NP-hard problem [22], it is too expensive to enumerate all the recharging schedules. As a heuristic method, RH is efficient but cannot guarantee the optimal recharging schedule. To further improve the final global best solution, we design a restricted enumeration method based on the solution obtained by RH. Specifically, for each route, it is reasonable to consider that the optimal solution tends to have no more charging station visits than the solution obtained by RH, as more visits tends to cause extra cost. Therefore, the restricted enumeration method only examine the solutions that have no more charging stations than the solution obtained by RH. Given any pair of nodes i and j, we can further filter out some charging stations that are obviously not considered by the optimal solution. Borrowing the concept from multi-objective optimization, we define the "dominance" relationship between two stations as follow.

Definition 1. Given two nodes i and j, a station s_1 is said to *dominate* another station s_2, if and only if:

$$\begin{aligned}&((d_{i,s_1} \leq d_{i,s_2}) \wedge (d_{s_1,j} < d_{s_2,j})) \vee \\ &((d_{i,s_1} < d_{i,s_2}) \wedge (d_{s_1,j} \leq d_{s_2,j}))\end{aligned} \tag{16}$$

Based on the dominance relationship, we only need to consider the non-dominated stations between i and j. According to the above analysis, the restricted enumeration algorithm is given in Algorithm 3.

First, taking the electricity-feasible route Γ generated by RH, we count how many stations are in Γ, denoted as ub (line 1). Then, the non-dominated stations between each pair of successive nodes Γ_i and Γ_{i+1} are found and stored in Π_i (line 3-6). After getting the upper bound and the non-dominated station list, we use the function $enumerate(\Gamma', sn, \mathbf{\Pi})$ to enumerate all possible recharging schedules containing sn charging stations chosen from $\mathbf{\Pi}$ (line 7-8). Whenever a better solution is found, Γ is updated (line 9-11). Due to the page limit, the detailed pseudo code about the function $enumerate$ is given in the supplemental material.

E. Discussion

1) Time Complexity Analysis: The time complexity of BACO is analyzed in a top-down way. First, the complexity of the whole algorithm can be shown as $\text{O}(X \cdot N_{gen} + Y)$, where N_{gen}, $\text{O}(X)$, and $\text{O}(Y)$ represent the number of generations, the time complexity of one generation, and the time complexity of the restricted enumeration method, respectively.

From Fig. 2 we can know that there are three operations in each generation of BACO, 1) generating n sets of capacity-feasible routes, 2) deciding the recharging schedules for these routes, and 3) updating the pheromone matrix. The time complexity of the first operation is $\text{O}(n \cdot (n^2 + q \cdot n))$ where $\text{O}(n^2)$ is the time complexity of generating a giant tour and $\text{O}(q \cdot n)$ is the complexity of splitting [50], $q = C/(\sum_{i \in I} c_i/|I|)$. For the second operation, suppose there are $m = |F|$ stations (the copies of the stations are not considered in our algorithm), the time complexity is $\text{O}(n \cdot (m \cdot n + n^2))$ where $\text{O}(m \cdot n + n^2)$ is the time complexity of RH. The time complexity of the third

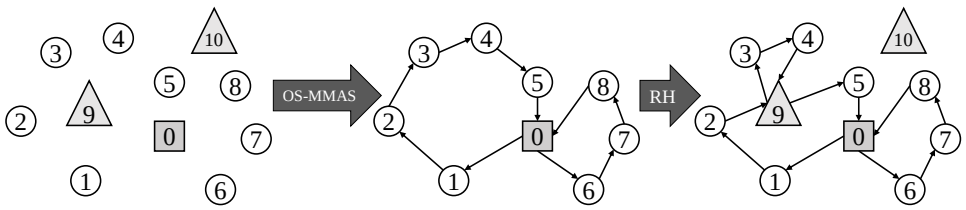


Fig. 5. Process of generating a solution in BACO.

Algorithm 3 Restricted Enumeration Method

Input: the global best solution $\mathbf{\Gamma}$, arc set E, customer set I, charging station set F, maximum battery capacity Q, battery consumption rate h.
Output: refined global best solution $\mathbf{\Gamma}$

```
for each Γ ∈ Γ do
    count how many stations are in Γ as ub;
    Γ' = Γ.remove_all_stations();
    k = |Γ'|; Π = {Π_0, ..., Π_{k-1}};
    for i = 0 → k − 1 do
        Π_i = find_non_dominated_stations(Γ'_i, Γ'_{i+1});
    end for
    for sn = 1 → ub do
        Γ'' = enumerate(Γ', sn, Π);
        if Γ'' is better than Γ then
            Γ = Γ'';
        end if
    end for
end for
return Γ;
```

operation is $\mathrm{O}(n^2)$. Adding these three components together, we can get $\mathrm{O}(X) = \mathrm{O}(n^2 \cdot (2n + q + m + 1))$.

Since for the worst case, the restricted enumeration method still needs to enumerate all the possible solutions, its theoretical time complexity is $\mathrm{O}((m \cdot n)^n)$ that is very high. However, practically, a vehicle will only recharge the battery less than five times in its journey and the number of non-dominated charging stations between two successive customers is usually less than or equal to three. Thus, the real time complexity of the restricted enumeration method is smaller than $\mathrm{O}(Y) = \mathrm{O}((3n)^4)$.

Overall, the time complexity of BACO is $\mathrm{O}(n^2 \cdot (2n + q + m + 1) \cdot N_{gen} + (3n)^4) = \mathrm{O}(n^2 \cdot (n + q + m) \cdot N_{gen} + n^4)$. Since in most cases $q \ll n$ and $m \ll n$, the time complexity of BACO is roughly $\mathrm{O}(n^3 \cdot N_{gen} + n^4)$.

2) Design Philosophy: The fundamental design philosophy of BACO is divide-and-conquer which leads to the bi-level structure as Fig. 5 showing. BACO divides the CEVRP into two sub-problems, i.e. CVRP and FRVCP, which are both NP-hard [22]. In BACO, the service order of customers is generated by OS-MMAS and the charging schedule is generated by RH. There are two reasons that we design BACO in this way.

- If a meta-heuristic algorithm is applied to solve FRVCP, there will be two meta-heuristic algorithms corresponding to the two levels of optimization, which will lead to an extremely high time complexity. Many studies show that not only for EVRP [22] but also for some other combinatorial optimization problems [51], [52], using meta-heuristic algorithms in both levels is too computationally expensive.
- Although both CVRP and FRVCP are NP-hard problem, the high-quality route set of CVRP is the prerequisite for the success of FRVCP. Also, the number of customers is usually larger than the number of charging stations which implies that the complexity of CVRP is higher than the complexity of FRVCP. Thus, we choose to use a meta-heuristic algorithm to solve CVRP and use a heuristic algorithm to solve FRVCP.

The design philosophy of BACO does not mean that FRVCP is not important. To get a good solution of CEVRP, both CVRP and FRVCP should be solved successfully.

Generally, by embodying the divide-and-conquer strategy, BACO has successfully decomposed the search space of CEVRP into two sub-spaces. For each level of optimization, the search space is reduced without losing much useful information, which makes it easier to find a good solution than directly searching in the whole space. Considering the complexities of the two sub-problems, the meta-heuristic algorithm OS-MMAS is used to generate good service orders of customers and defines the promising region of the lower-level search space. The heuristic algorithm RH decides the charging schedule of each vehicle, which helps BACO to generate good solutions without having a high computational complexity.

V. Experiments

A. Experimental Setup

1) Benchmark Instances: To investigate the performance of BACO, a newly-proposed benchmark for the IEEE WCCI2020 competition on EC for the EVRP is adopted [6]. The benchmark contains a group of small-scale instances and a group of large-scale instances. The small-scale group consists of seven instances, having up to 100 customers. The large-scale group consists of 10 instances, having up to 1000 customers. These two groups are derived from [53] and [54], respectively. The details of the benchmark are shown in Table I.

2) Algorithm Settings: To check the ability of BACO, we set a fixed execution time on each instance that allows BACO to converge:

$$ExeTime = \vartheta \cdot \frac{|I| + |F|}{100}(\text{hr}), \tag{17}$$

where ϑ equals 1, 2, and 3 for E22-E101, X143-X916, and X1001, respectively. Using fixed execution time as the termination condition is a common way in the research of combinatorial optimization [55]. BACO is implemented in

TABLE I
INFORMATION OF THE CEVRP BENCHMARK SET.

Name	Customer	Depot	Station	Route
E22	21	1	8	4
E23	22	1	9	3
E30	29	1	6	4
E33	32	1	6	4
E51	50	1	5	5
E76	75	1	7	7
E101	100	1	9	8
X143	142	1	4	7
X214	213	1	9	11
X352	351	1	35	40
X459	458	1	20	26
X573	572	1	6	30
X685	684	1	25	75
X749	748	1	30	98
X819	818	1	25	171
X916	915	1	9	207
X1001	1000	1	9	43

C++ and executed on Intel i7-6700 3.40Hz CPU with the Arch Linux system. On each instance, BACO is executed 30 times.

We compare BACO with five algorithms, VNS, simulated annealing (SA), GA, ILS and a single-level MMAS denoted as SLMMAS. The first three algorithms are the winners of the IEEE WCCI2020 competition on EC for the EVRP[1]. ILS is originally proposed to solve the EVRP with non-linear charging, but it can be easily used to solve CEVRP by changing the constraint of working time to the capacity constraint [22]. It uses an order-first split-second ILS to generate the capacity-feasible routes and uses a greedy-heuristic (GH) algorithm to insert charging stations. SLMMAS is implemented by ourselves to demonstrate the advantage of the bi-level structure of BACO. SLMMAS inserts charging stations during the construction of a route. Whenever the next selected customer cannot be reached due to the electricity constraint, one charging station that brings minimum extra distance is visited before serving the customer. Since the codes and papers of the three winners of the competition are not provided, we directly use the reported results for comparison including the min, mean, and standard deviation values. Also, the one-sample t-test is made between BACO and the three algorithms to check whether their performance is significantly different. As ILS and SLMMAS, they are also tested 30 independent times on each test case. Their performances are compared with BACO using Wilcoxon rank-sum test. The significance level is set to 0.05 with Bonferroni correction.

All the parameters of BACO follow the canonical settings of MMAS [40]. The population size of BACO is set to $n = |I|+1$. The pheromone evaporation speed ρ is set to 0.98. α and β are set to 1 and 2. For the instances with less than 500 customers, the giant tour is generated just like Algorithm 1. For the instances with more than 500 customers, a candidate list $\bar{I}_i$ is first generated in the initialization phase for each node. Each list $\bar{I}_i$ maintains the nearest 20 nodes to node i. When building the giant tour, the ants will choose the nodes from the candidate lists rather than the universal set. This method is widely adopted in ACO algorithms [39], [40]. We also investigated the influence of α and β. The experimental results show that the conventional setting $\alpha = 1, \beta = 2$ is a good choice for BACO. Due to the page limit, this experiment is provided in the supplemental material.

[1]The performance of the three algorithms can be found on https://mavrovouniotis.github.io/EVRPcompetition2020/

B. Comparisons on Small-scale Instances

We first compare these algorithms on the small-scale instances. The experimental results are shown in Table II. The best min and mean values are highlighted. From the results, we can get the following information:

- According to the statistic test, BACO is only worse than VNS on E33. On the other instances, BACO either has equivalent performance to the compared algorithms or significantly outperforms the other algorithms.
- According to the min objective values, VNS has found the best known solutions on all the seven instances. BACO found five out of seven, but on the two instances where BACO failed find the best known solutions, i.e. E33 and E101, the gaps between the solutions found by BACO and VNS are very small.
- The reason of the superiority of BACO on small-scale instances is that both OS-MMAS and RH are effective on small-scale instances. OS-MMAS does not rely on any pre-defined neighborhood structure to generate routes. For CEVRP, the neighborhood structure is hard to define since not only the service orders of customers but also the recharging schedules of vehicles should be considered. Using OS-MMAS can generate many different candidate routes for RH, and for small-scale instances, RH is very effective to find the optimal or a near-optimal recharging schedule. In contrast, using a pre-defined neighborhood structure in the individual-based meta-heuristic algorithms may miss some good solutions.
- Compared with the population-based meta-heuristic algorithms like SLMMAS and GA, the the advantage of the bi-level optimization structure of BACO is demonstrated in terms of effectiveness. By decomposing the decision space, we can use different effective methods to handle different levels of sub-problems (OS-MMAS for upper-level, RH for lower-level), and the integrated solutions generated by solving the two sub-problems are used as feedback to help OS-MMAS to generated better service orders of customers. Thus, the effectiveness of BACO is somewhat guaranteed. However, in SLMMAS, since the recharging schedules of vehicles are decided along with the construction of the routes, it may generate bad recharging schedules and may also cause bad influence to the decision of the service orders of customers, which leads to its poor performance.

Overall, the results on the small-scale instances show that BACO is very effective.

C. Comparisons on Large-scale Instances

Then, the algorithms are compared on the large-scale instances. The comparison results are shown in Table III. From the results, we can get the following observations:

TABLE II
COMPARISON BETWEEN BACO AND THE COMPARED ALGORITHMS ON 7 SMALL-SCALE INSTANCES.

Case	Index	BKS	SLMMAS	ILS	GA	SA	VNS	BACO
E22	min	384.67	385.44	**384.67**	**384.67**	**384.67**	**384.67**	**384.67**
	mean		385.44↑	385.69↑	**384.67**↕	**384.67**↕	**384.67**↕	**384.67**
	std		0.00	2.11	0.00	0.00	0.00	0.00
E23	min	571.94	582.61	590.35	**571.94**	**571.94**	**571.94**	**571.94**
	mean		582.94↑	592.05↑	**571.94**↕	**571.94**↕	**571.94**↕	**571.94**
	std		0.60	1.04	0.00	0.00	0.00	0.00
E30	min	509.47	514.42	**509.47**	**509.47**	**509.47**	**509.47**	**509.47**
	mean		516.11↑	**509.47**↕	**509.47**↕	**509.47**↕	**509.47**↕	**509.47**
	std		0.61	0.00	0.00	0.00	0.00	0.00
E33	min	840.14	859.25	840.57	844.25	840.57	**840.14**	840.57
	mean		860.08↑	844.07↑	845.62↑	854.07↑	**840.43**↓	842.30
	std		0.74	7.78	0.92	12.80	1.18	1.42
E51	min	529.90	549.81	**529.90**	**529.90**	533.66	**529.90**	**529.90**
	mean		564.93↑	539.03↑	542.08↑	533.66↑	543.26↑	**529.90**
	std		8.70	6.92	8.57	0.00	3.52	0.00
E76	min	692.64	724.24	694.64	697.27	701.03	**692.64**	**692.64**
	mean		762.09↑	704.24↑	717.30↑	712.17↑	697.89↑	**692.85**
	std		13.23	7.15	9.58	5.78	3.09	0.81
E101	min	839.29	891.77	841.02	852.69	845.84	**839.29**	840.25
	mean		904.62↑	851.62↑	872.69↑	852.48↑	853.34↑	**845.95**
	std		4.64	6.93	9.58	3.44	4.73	4.58
w/t/l	-	-	7/0/0	6/1/0	4/3/0	4/3/0	3/3/1	

↑ means BACO is significantly better than the compared algorithm. ↓ means BACO is significantly worse than the compared algorithm. ↕ means BACO is well-matched to the compared algorithm. 'w/t/l' means on how many cases BACO wins, ties, or loses to the compared algorithm. BKS represents the best known solution.

- According to the mean objective value and the statistic test, BACO has overwhelmed SLMMAS and GA on all the instances. Also, BACO has outperformed ILS on nine instances, outperformed SA on eight instances, and outperformed VNS on seven instances. According to the min objective value, BACO has updated the best know solutions of seven instances out of ten, except for X573, X819, and X916. This evidence shows that BACO is generally the best one among the compared algorithms.
- On X573, X819, and X916, BACO did not perform well. After checking the maps of these instances, we found that the customers in these three instances are clustered and the depot is far away from the customers. This kind of customer distribution may be the reason for the poor performance of BACO. In the experiment of parameter tuning that is shown in the supplemental material, we have found that adjusting the weight of heuristic may improve the performance of BACO on this kind of instances. We retain the adaptive parameter adjusting as a future research direction.

To further analyze the converging behavior of BACO, its convergence curves are drawn with SLMMAS and ILS, which are shown in Fig. 6. The convergence curves of other algorithms are not available because their codes and papers are not publicly available now. These figures show that:

- Basically, BACO has converged on most of the instances within the given execution time, except on the largest one X1001. Its converging speed is faster than ILS and SLMMAS. SLMMAS even did not work on some instances after initialization. This phenomenon is also shown in Table III that the values of the standard deviation of SLMMAS on some instances are zero.
- The advantage of the converging speed of BACO originates from its better global search ability than ILS on large-scale instances. ILS generates new solutions within a pre-defined local area each time based on some well-known local search and perturbation operators like 2-opt and random double-bridge that were used in [22]. When solving large-scale instances, the global search ability of ILS in the huge search space is not sufficient. It takes a longer time to jump out of local optima on large-scale instances than on small-scale instances. BACO uses a pheromone matrix to guide the construction of solutions. In the early stage of optimization when the difference among the pheromone values of different connections is not great, BACO has a stronger global search ability than ILS and is more able to jump out of local optima. Thus, in Fig. 6, we can see that BACO has a faster converging speed than ILS and finally gets better results within the given period of time.
- The comparison between BACO and SLMMAS on large-scale instances demonstrates the advantage of the bi-level optimization structure of BACO again in terms of both effectiveness and efficiency. By decomposing the decision space, the complexity of the problem is reduced and BACO can focus on the promising area which leads good converging speed. However, in SLMMAS, the search space is huge for large-scale instances, which makes it hard to find good solutions.
- Although BACO has shown good performance, it has two disadvantages in terms of the OS-MMAS algorithm. The first one is the sensitivity to the parameters α and β in (13). The experiment in the supplemental material shows that different parameter settings may prefer different kinds of instances. The second one is the relatively slow execution speed. In OS-MMAS, every next customer

TABLE III
COMPARISON BETWEEN BACO AND THE COMPARED ALGORITHMS ON 10 LARGE-SCALE INSTANCES.

Case	Index	BKS	SLMMAS	ILS	GA	SA	VNS	BACO
X143	min	16028.05	17238.76	16058.29	16488.60	16610.37	16028.05	**15901.23**
	mean		17985.91↑	16318.57↑	16911.50↑	17188.90↑	16459.31↑	**16031.46**
	std		415.20	160.07	282.30	170.44	242.59	262.47
X214	min	11674.96	11421.43	11323.56	11762.07	11404.44	11323.56	**11133.14**
	mean		12023.72↑	11537.58↑	12007.06↑	11680.35↑	11482.20↑	**11219.70**
	std		129.22	72.55	156.69	116.47	76.14	46.25
X352	min	27064.88	31927.71	27947.89	28008.09	27222.96	27064.88	**26478.34**
	mean		31927.71↑	28364.41↑	28336.07↑	27498.03↑	27217.77↑	**26593.18**
	std		0.00	142.04	205.29	155.62	86.20	72.86
X459	min	25370.80	28448.27	26511.28	26048.21	27222.96	25370.80	**24763.93**
	mean		29434.48↑	26726.69↑	26345.12↑	25809.47↑	25582.27↑	**24916.60**
	std		540.16	126.93	185.14	157.97	106.89	94.08
X573	min	51929.24	56609.94	53102.46	54189.62	**51929.24**	52181.51	53822.87
	mean		57170.35↑	53507.46↓	55327.62↑	52793.66↓	**52548.09**↓	54567.15
	std		199.81	275.76	548.05	577.24	278.85	231.05
X685	min	71345.40	84435.72	74409.65	73925.56	72549.90	71345.40	**70834.88**
	mean		84435.72↑	75087.58↑	74508.03↑	73124.98↑	71770.57↑	**71440.57**
	std		0.00	259.60	409.43	320.07	197.08	281.78
X749	min	81002.01	90441.6	84298.43	84034.73	81392.78	81002.01	**80299.76**
	mean		90441.36↑	84860.28↑	84759.79↑	81848.13↑	81327.39↑	**80694.54**
	std		0.00	287.26	376.10	275.26	176.19	223.91
X819	min	164289.95	171553.21	168651.19	170965.68	165069.77	**164289.95**	164720.80
	mean		172355.34↑	169837.06↑	172410.12↑	165895.78↑	**164926.41**↓	165565.79
	std		311.09	483.35	568.58	403.70	318.62	401.02
X916	min	341649.91	353046.07	348733.86	357391.57	342796.88	**341649.91**	342993.01
	mean		354262.15↑	350822.41↑	360269.94↑	343533.85↓	**342460.70**↓	344999.95
	std		464.68	1177.08	229.19	556.98	510.66	905.72
X1001	min	77476.36	89611.93	79493.37	78832.90	78053.86	77476.36	**76297.09**
	mean		89611.93↑	79928.29↑	79163.34↑	NA↑	77920.52↑	**77434.33**
	std		0.00	265.91	NA	306.27	234.73	719.8671
w/t/l	-	-	10/0/0	9/0/1	10/0/0	8/0/2	7/0/3	-

The mean value of SA and the standard deviation of GA on X1001 are not given.

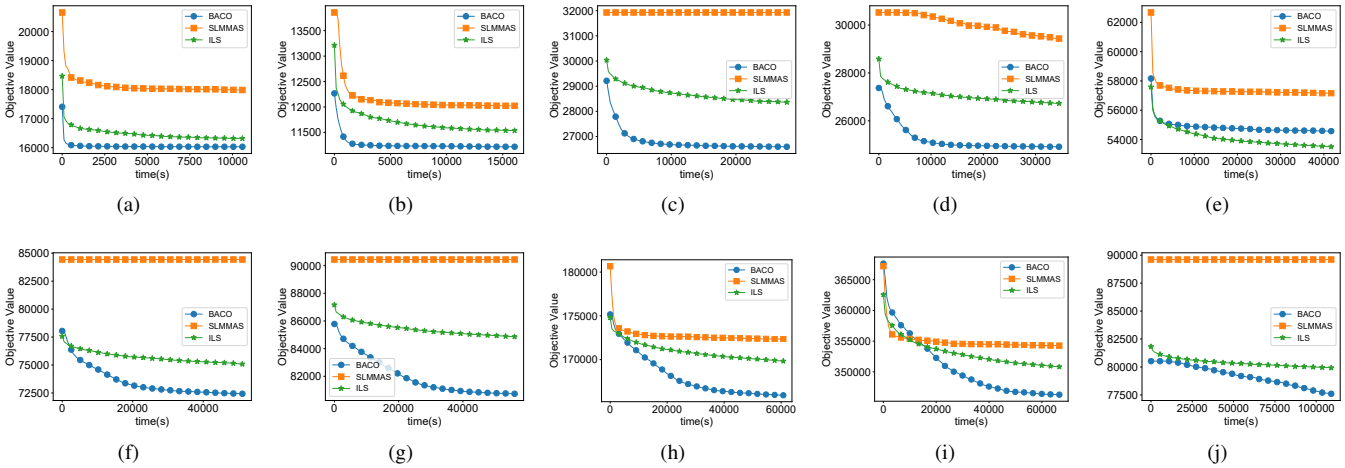


Fig. 6. Convergence curves of BACO, SLMMAS, and ILS on large-scale instances. (a) to (j) represent X143 to X1001, respectively. x-axis represents the execution time. y-axis represents the objective value.

is selected through a roulette wheel selection. As (13) shows, besides the addition operation, the division operation is also frequently used in a roulette wheel selection. Thus, compared with other methods that have similar time complexities but use addition and subtraction most of the time, OS-MMAS tasks a longer time to generate a tour. However, these two disadvantages are not fatal. As we can see from the experimental results, the traditional setting $\alpha = 1$ and $\beta = 2$ works well most of the time although it may be not the best setting for every instance. Fig. 6 shows that the convergence speed of BACO is not slow compared with ILS in terms of the real execution time. Overall, the results of Table III and Fig. 6 show that although BACO has not outperformed the state-of-the-art algorithms on every instance, generally it is both effective and efficient on the large-scale instances.

D. *Effectiveness of the Removal Heuristic*

In BACO, the pheromone matrix is update according to the solutions that are fixed by RH. The effectiveness of RH can greatly affect the pheromone update and thus the subsequent route construction. To further investigate the effectiveness of

using RH in BACO, we compare it with another two simple yet effective heuristic methods. The reason that we do not apply meta-heuristic algorithms to the lower-level sub-problem has been explained in Section IV.E, that is, meta-heuristic algorithms would be too time-consuming to solve the lower-level sub-problem [56].

The first compared heuristic is the greedy-heuristic (GH) algorithm that was applied in [22], [43]. The greedy-heuristic algorithm keeps inserting charging stations into a route that would lead to the minimum violation degree of the electricity constraint until the route is electricity-feasible. The second one is the forward-heuristic (FH). The forward-heuristic algorithm follows the track of the EV. Every time it finds that the EV does not have enough electricity to reach the next node in the route, it inserts a charging station. BACO is re-implemented with these two heuristic algorithms.

1) Comparison of the performance of BACO using different heuristic methods: BACO is tested on all the instances for 10 independent runs using different heuristic algorithms. Due to the page limit, Fig. 7 only shows the convergence curves on some of the instances. Full results can be found in the supplemental material.

- On the small-scale instances, the proposed RH algorithm either outperforms GH and FH or has equivalent performance with GH. Using different heuristic algorithms barely has influence to the converging speed of BACO.
- On the large-scale instances, compared with FH, RH has shown better performance on most of the instances but was beaten on some of them such as X916 and X1001. Compared with GH, RH is always better. Also, the figures show that the converging speed of BACO with RH and FH are much faster than using GH.

Generally, we can find that RH is efficient and effective on both the small-scale benchmark instances and the large-scale instances. GH can generate good solutions on small-scale instances but it is not efficient to handle large-scale instances. FH is not effective on small-scale instances but can be efficient on large-scale instances.

This phenomenon can be analyzed from the perspective of time complexity. Fig. 8 shows how many solutions are generated during the execution on the largest instance X1001. From Fig. 8, we can see that the rank of the time complexity of these three algorithms is GH>RH>FH. The reason that the simplest heuristic FH can get better results on some large-scale instances is that BACO with FH has generated more solutions than the other two algorithms. On these large-scale instances, BACO requires high speed to converge within the limited execution time. Thus, using a simple heuristic to generate more solutions within the limited time is a good choice. However, if the algorithm can fully converge within the limited execution time just like the situations on the instances with less than 300 customers, GH and RH can obtain better solutions than FH, since in this case, generating good solutions is more important than generating more solutions. Regarding the proposed RH in this paper, the experimental results show that it can generate very good solutions without bringing a high computational burden.

2) Comparison of the solutions generated by the three heuristic algorithms: To visually demonstrate the advantages and disadvantages of these three heuristic algorithms, we use the solution generated by BACO with these three heuristics on E-n23-k3 as an example. The charging stations in the solution are removed in the first place. Then, three different solutions are generated by using the three heuristic algorithms respectively which are shown in Fig. 9.

- Regarding FH, it is easy to understand that the positions to insert the charging stations may not be optimal since it only recharges the EV when it has to. Checking the left route of Fig. 9(a), we can find that the EV served two customers far away from the top-left station and then went back to the station to recharge. Fig. 9(b) and Fig. 9(c) shows that GH and RH let the EV recharge after serving the customer near to the station then went to the two customers, which is different from FH.
- Regarding GH, if the EV only recharges once during the journey, GH can always find the best position to insert the charging station. However, if the EV should recharge multiple times during the journey, it may fail in finding the best position to insert charging stations, because the goal of GH is to minimize the degree of violation rather than the objective value before the route becomes electricity-feasible. This may lead to the wrong positions of charging station insertion. From the right route of Fig. 9(b), we can see that it chose different charging stations from the stations that RH chose.
- Regarding RH, in most cases, given a route with the same customer order, it can generates better recharging schedules than GH and FH. However, there is a case that RH may insert more charging stations to a route than it really needs. Assume a scenario that a route can be fixed by either inserting a charging station in the very middle of the route or inserting two charging stations in the head and tail respectively, and the extra distance of inserting one charging station in the middle is longer than inserting each of the two charging stations individually but is shorter than inserting two charging stations together. In this situation, RH will choose the second plan to insert two charging stations which is worse than the first plan.

Overall, since FRVCP is essentially a NP-hard problem [22], all these three heuristic algorithms cannot guarantee the optimality of the recharging schedule, but from both the general performance and the case study, we can see that the proposed RH has clear advantages against FH and GH. Thus, it is the best choice of BACO.

E. Effectiveness of the Restricted Enumeration Method

To investigate the effectiveness of the restricted enumeration method, the solutions generated by BACO before the final refinement and after the final refinement are compared. On the seven small-scale instances, we found that the restricted enumeration method did not change any recharging schedules of the solutions. This result indicates again that RH is effective enough to generate good even optimal recharging schedules on the small-scale instances. Thus, only the comparisons on

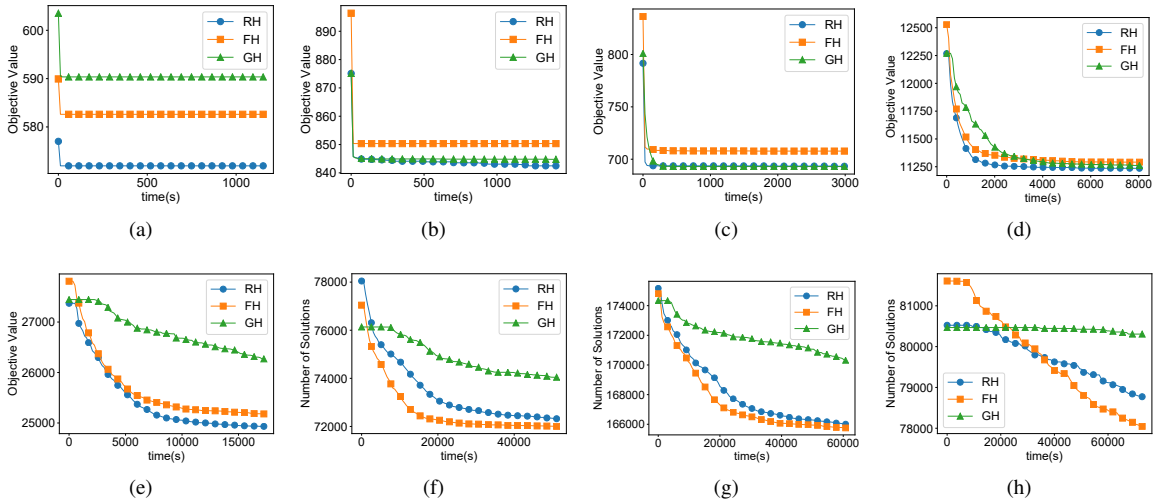


Fig. 7. Convergence curves of using GH, FH, and RH in BACO. (a) E23, (b) E33, (c) E76, (d) X214, (e) X459, (f) X685, (g) X819, (h) X1001. x-axis represents the execution time. y-axis represents the objective value.

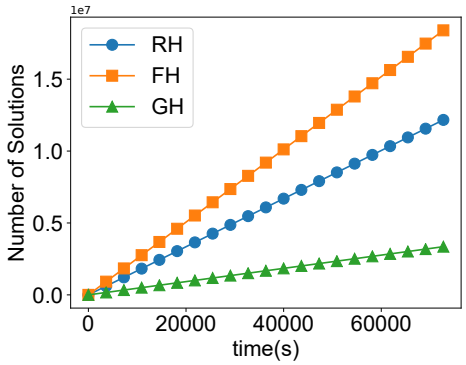


Fig. 8. The number of solutions generated during the execution by BACO using three different heuristic algorithms on X1001.

the large-scale instances are shown in Table IV. How much profit the restricted enumeration method has brought to the best and the average performance of BACO is shown. The averaged execution time of the restricted enumeration method is also given to show that it will not bring too much extra computational burden.

For the first five instances X143 to X573, the enumeration basically did not help to much, only improved the mean objective values for X352 and X573 a little. This fact implies that RH is already very effective to the problems with less than 600 customers. From X685, the restricted enumeration begins to show its effectiveness. It has brought different degrees of improvement to the performance of BACO on the last five instances. Especially on X685, it made over one percent improvement which is very large for a large-scale problem. Moreover, checking the execution time, we can see that although the theoretical time complexity of the restricted enumeration method is very high $O(n^4)$, its actual execution time is short that is usually less than three milliseconds.

Overall, the restricted enumeration method is effective to improve the quality of the final solution and efficient enough without bringing too much extra computational cost. Based on the results, we recommend using the restricted enumeration method to deal with large-scale problems having more than 600 customers.

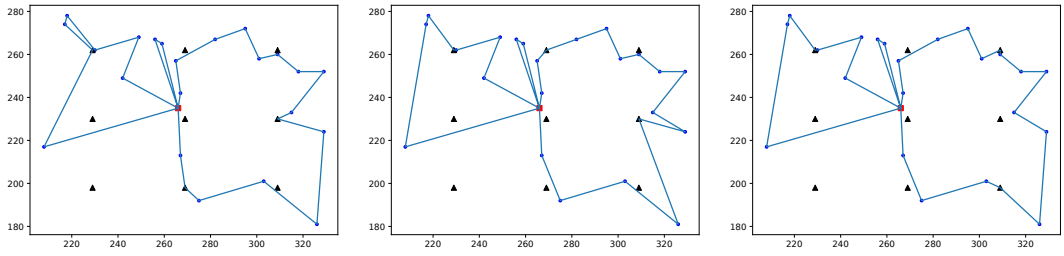
TABLE IV
THE OBJECTIVE VALUES BEFORE APPLYING THE RESTRICTED ENUMERATION METHOD AND AFTER APPLYING THE RESTRICTED ENUMERATION METHOD.

Case	Index	Before	After	Percent	Time(ms)
X143	min	15901.24	15901.24	0.00%	–
	mean	16031.46	16031.46	0.00%	0.43
X214	min	11133.14	11133.14	0.00%	–
	mean	11219.71	11219.71	0.00%	0.30
X352	min	26478.35	26478.35	0.00%	–
	mean	26593.49	26593.19	0.00%	0.63
X459	min	24763.93	24763.93	0.00%	–
	mean	24916.61	24916.61	0.00%	0.66
X573	min	53822.88	53822.88	0.00%	–
	mean	54574.41	54567.15	0.01%	0.87
X685	min	71817.18	70834.89	1.37%	–
	mean	72427.80	71440.58	1.36%	0.90
X749	min	80307.55	80299.76	0.01%	–
	mean	80737.13	80694.54	0.05%	1.45
X819	min	164867.95	164720.81	0.09%	–
	mean	165932.57	165565.80	0.22%	1.72
X916	min	344400.74	342993.01	0.41%	–
	mean	346278.33	344999.96	0.37%	1.69
X1001	min	76390.14	76297.09	0.12%	–
	mean	77603.25	77434.34	0.22%	2.18

[1] Percent = (Before - After) / Before.
[2] Time refers to the execution time of only the restricted enumeration method instead of the whole optimization process of BACO.

VI. Conclusions and Future Work

The goal of this paper was to solve CEVRP effectively. This goal has been successfully achieved by proposing a new algorithm called BACO. Specifically, BACO has shrunk the search space of CEVRP by decomposing it into two-level sub-problems CVRP and FRVCP. Then, orchestrating OS-MMAS and RH together corresponding to the two sub-problems, BACO has shown good capability in solving CEVRPs with different scales. Comparisons with state-of-the-art algorithms show that the general performance of BACO is significantly better on most of the instances and it has even updated the



(a) (b) (c)

Fig. 9. Solutions generated by FH, GH, and RH on E23. (a) FH, the objective value is 589.58. (b) GH, the objective value is 599.23. (c) RH, the objective value is 571.95.

best known solutions of seven instances. Further analyses also demonstrated the effectiveness and efficiency of the components of BACO, which is the reason that BACO can maintain a steady and fast optimizing speed on large-scale instances.

Although BACO has shown promising performance in this paper, there are still many open issues to investigate. First, the experimental results show that the ability of BACO in solving the instances with clustered customers is weaker than solving regular instances. Thus, how to incorporate orientation information in the algorithm to further improve its ability in solving the special cases is the main direction we are going to follow in the future work. Second, both empirical and theoretical analyses show that there is a trade-off between the complexity and the effectiveness of the heuristic algorithm in solving the lower-level sub-problem FRVCP. Thus, how to use different heuristic algorithms in different stages of BACO to fully take advantage of the computing resource is another promising direction to improve the performance of BACO.

REFERENCES

[1] Y. Van Fan, S. Perry, J. J. Klemeš, and C. T. Lee, "A review on air emissions assessment: Transportation," *J. Clean. Prod.*, vol. 194, pp. 673–684, 2018.

[2] X. Huang and J. Ge, "Electric vehicle development in beijing: An analysis of consumer purchase intention," *J. Clean. Prod.*, vol. 216, pp. 361–372, 2019.

[3] F. Un-Noor, S. Padmanaban, L. Mihet-Popa, M. N. Mollah, and E. Hossain, "A comprehensive study of key electric vehicle (ev) components, technologies, challenges, impacts, and future direction of development," *Energies*, vol. 10, no. 8, p. 1217, 2017.

[4] F. Knobloch, S. V. Hanssen, A. Lam, H. Pollitt, P. Salas, U. Chewpreecha, M. A. Huijbregts, and J.-F. Mercure, "Net emission reductions from electric cars and heat pumps in 59 world regions over time," *Nat. Sustain.*, pp. 1–11, 2020.

[5] B. Lin and W. Wu, "Why people want to buy electric vehicle: An empirical study in first-tier cities of china," *Energy Policy*, vol. 112, pp. 233–241, 2018.

[6] M. Mavrovouniotis, C. Menelaou, S. Timotheou, C. Panayiotou, G. Ellinas, and M. Polycarpou, "Benchmark set for the ieee wcci-2020 competition on evolutionary computation for the electric vehicle routing problem," 2020.

[7] D. Shicong, "Jd.com will introduce 1,000 logistics nevs in over 10 chinese cities," [Online], 2017, https://www.yicaiglobal.com/news/jdcom-will-introduce-1000-logistics-nevs-in-over-10-chinese-cities.

[8] S. N. Kumar and R. Panneerselvam, "A survey on the vehicle routing problem and its variants," 2012.

[9] M. M. Solomon and J. Desrosiers, "Survey paper—time window constrained routing and scheduling problems," *Transport. Sci.*, vol. 22, no. 1, pp. 1–13, 1988.

[10] U. Ritzinger, J. Puchinger, and R. F. Hartl, "A survey on dynamic and stochastic vehicle routing problems," *Int. J. Prod. Res.*, vol. 54, no. 1, pp. 215–231, 2016.

[11] S. Karakatič and V. Podgorelec, "A survey of genetic algorithms for solving multi depot vehicle routing problem," *Appl. Soft Comput.*, vol. 27, pp. 519–532, 2015.

[12] S. N. Parragh, K. F. Doerner, and R. F. Hartl, "A survey on pickup and delivery problems," *J. für Betriebswirtschaft*, vol. 58, no. 1, pp. 21–51, 2008.

[13] R. Fukasawa, H. Longo, J. Lysgaard, M. P. De Aragão, M. Reis, E. Uchoa, and R. F. Werneck, "Robust branch-and-cut-and-price for the capacitated vehicle routing problem," *Math. Program.*, vol. 106, no. 3, pp. 491–511, 2006.

[14] P. Toth and D. Vigo, "Models, relaxations and exact approaches for the capacitated vehicle routing problem," *Discrete Appl. Math.*, vol. 123, no. 1-3, pp. 487–512, 2002.

[15] J. Wang, T. Weng, and Q. Zhang, "A two-stage multiobjective evolutionary algorithm for multiobjective multidepot vehicle routing problem with time windows," *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2467–2478, 2018.

[16] K. C. Tan, Y. Chew, and L. H. Lee, "A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems," *Eur. J. Oper. Res.*, vol. 172, no. 3, pp. 855–885, 2006.

[17] K. C. Tan, C. Y. Cheong, and C. K. Goh, "Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation," *Eur. J. Oper. Res.*, vol. 177, no. 2, pp. 813–839, 2007.

[18] J. Bi, Y. Wang, Q. Sai, and C. Ding, "Estimating remaining driving range of battery electric vehicles based on real-world data: A case study of beijing, china," *Energy*, vol. 169, pp. 833–843, 2019.

[19] A. Afroditi, M. Boile, S. Theofanis, E. Sdoukopoulos, and D. Margaritis, "Electric vehicle routing problem with industry constraints: trends and insights for future research," *Transport. Res. Procedia*, vol. 3, pp. 452–459, 2014.

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

[21] T. Chen, B. Zhang, H. Pourbabak, A. Kavousi-Fard, and W. Su, "Optimal routing and charging of an electric vehicle fleet for high-efficiency dynamic transit systems," *IEEE Trans. on Smart Grid*, vol. 9, no. 4, pp. 3563–3572, 2016.

[22] A. Montoya, C. Guéret, J. E. Mendoza, and J. G. Villegas, "The electric vehicle routing problem with nonlinear charging function," *Transport. Res. B-Meth*, vol. 103, pp. 87–110, 2017.

[23] Ç. Koç, O. Jabali, J. E. Mendoza, and G. Laporte, "The electric vehicle routing problem with shared charging stations," *Int. Trans. Oper. Res.*, vol. 26, no. 4, pp. 1211–1243, 2019.

[24] I. I. Cplex, "V12. 1: User's manual for cplex," *IBM*, vol. 46, no. 53, p. 157, 2009.

[25] J. Lin, W. Zhou, and O. Wolfson, "Electric vehicle routing problem," *Transport. Res. Procedia*, vol. 12, no. Supplement C, pp. 508–521, 2016.

[26] J. P. Vielma, "Mixed integer linear programming formulation techniques," *Siam Review*, vol. 57, no. 1, pp. 3–57, 2015.

[27] F. W. Glover and G. A. Kochenberger, *Handbook of metaheuristics*. Springer Science & Business Media, 2006, vol. 57.

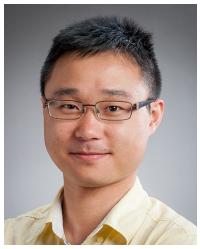
[28] D. S. Johnson and L. A. McGeoch, "The traveling salesman problem: A case study in local optimization," *Local Search Comb. Optim.*, vol. 1, no. 1, pp. 215–310, 1997.

[29] S. Zhang, Y. Gajpal, S. Appadoo, and M. Abdulkader, "Electric vehicle routing problem with recharging stations for minimizing energy consumption," *Int. J. Prod. Econ.*, vol. 203, pp. 404–413, 2018.

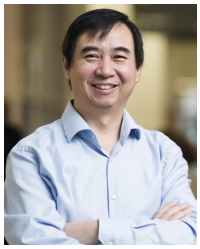
[30] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transport. Sci.*, vol. 40, no. 4, pp. 455–472, 2006.

[31] M. Keskin and B. Çatay, "A matheuristic method for the electric vehicle routing problem with time windows and fast chargers," *Comput. Oper. Res.*, vol. 100, pp. 172–188, 2018.
[32] M. Dorigo and L. M. Gambardella, "Ant colonies for the travelling salesman problem," *Biosystems*, vol. 43, no. 2, pp. 73–81, 1997.
[33] M. Mavrovouniotis, F. M. Müller, and S. Yang, "Ant colony optimization with local search for dynamic traveling salesman problems," *IEEE Trans. Cybern.*, vol. 47, no. 7, pp. 1743–1756, 2016.
[34] J. E. Bell and P. R. McMullen, "Ant colony optimization techniques for the vehicle routing problem," *Adv. Eng. Inform.*, vol. 18, no. 1, pp. 41–48, 2004.
[35] L. Santos, J. Coutinho-Rodrigues, and J. R. Current, "An improved ant colony optimization based algorithm for the capacitated arc routing problem," *Transport. Res. B-Meth*, vol. 44, no. 2, pp. 246–266, 2010.
[36] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theor. Comput. Sci.*, vol. 344, no. 2-3, pp. 243–278, 2005.
[37] W.-N. Chen, D.-Z. Tan, Q. Yang, T. Gu, and J. Zhang, "Ant colony optimization for the control of pollutant spreading on social networks," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 4053–4065, 2020.
[38] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Trans. Syst. Man Cybern. B*, vol. 26, no. 1, pp. 29–41, 1996.
[39] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 53–66, 1997.
[40] T. Stützle and H. H. Hoos, "Max–min ant system," *Future Gener. Comput. Syst.*, vol. 16, no. 8, pp. 889–914, 2000.
[41] Z. Guo, Y. Li, X. Jiang, and S. Gao, "The electric vehicle routing problem with time windows using genetic algorithm," in *Proc. IAEAC*. IEEE, 2017, pp. 635–639.
[42] S. Shao, W. Guan, B. Ran, Z. He, and J. Bi, "Electric vehicle routing problem with charging time and variable travel time," *Math. Probl. Eng.*, vol. 2017, 2017.
[43] S. Erdoğan and E. Miller-Hooks, "A green vehicle routing problem," *Transport. Res. E-Log*, vol. 48, no. 1, pp. 100–114, 2012.
[44] 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," *Comput. Oper. Res.*, vol. 104, pp. 256–294, 2019.
[45] Y. Xiao, X. Zuo, I. Kaku, S. Zhou, and X. Pan, "Development of energy consumption optimization model for the electric vehicle routing problem with time windows," *J. Clean. Prod.*, vol. 225, pp. 647–663, 2019.
[46] M. Bruglieri, F. Pezzella, O. Pisacane, S. Suraci *et al.*, "A variable neighborhood search branching for the electric vehicle routing problem with time windows." *Electron. Notes in Discrete Math.*, vol. 47, no. 15, pp. 221–228, 2015.
[47] M. Bruglieri, S. Mancini, F. Pezzella, O. Pisacane, and S. Suraci, "A three-phase matheuristic for the time-effective electric vehicle routing problem with partial recharges," *Electron. Notes in Discrete Math.*, vol. 58, pp. 95–102, 2017.
[48] M. Keskin and B. Çatay, "Partial recharge strategies for the electric vehicle routing problem with time windows," *Transpor. Res. C-Emer*, vol. 65, pp. 111–127, 2016.
[49] W.-L. Liu, Y.-J. Gong, W.-N. Chen, Z. Liu, H. Wang, and J. Zhang, "Coordinated charging scheduling of electric vehicles: A mixed-variable differential evolution approach," *IEEE Trans. Intell. Transp. Syst.*, 2019.
[50] C. Prins, "A simple and effective evolutionary algorithm for the vehicle routing problem," *Comput. Oper. Res.*, vol. 31, no. 12, pp. 1985–2002, 2004.
[51] E.-G. Talbi, "A taxonomy of metaheuristics for bi-level optimization," in *Metaheuristics for bi-level optimization*. Springer, 2013, pp. 1–39.
[52] W.-N. Chen, J. Zhang, H. S.-H. Chung, R.-Z. Huang, and O. Liu, "Optimizing discounted cash flows in project scheduling—an ant colony optimization approach," *IEEE Trans. Syst. Man Cybern. C*, vol. 40, no. 1, pp. 64–77, 2009.
[53] N. Christofides and S. Eilon, "An algorithm for the vehicle-dispatching problem," *J. Oper. Res. Soc.*, vol. 20, no. 3, pp. 309–318, 1969.
[54] E. Uchoa, D. Pecin, A. Pessoa, M. Poggi, T. Vidal, and A. Subramanian, "New benchmark instances for the capacitated vehicle routing problem," *Eur. J. Oper. Res.*, vol. 257, no. 3, pp. 845–858, 2017.
[55] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM Comput. Surv.*, vol. 35, no. 3, pp. 268–308, 2003.
[56] E. Kieffer, G. Danoy, M. R. Brust, P. Bouvry, and A. Nagih, "Tackling large-scale and combinatorial bi-level problems with a genetic programming hyper-heuristic," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 44–56, 2019.

Ya-Hui Jia (M'20) received the B.Eng. and Eng.D. degrees from Sun Yat-sen University, China, in 2013 and 2019, respectively. He is currently a postdoctoral research fellow at the School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand. His research interests include evolutionary computation, combinatorial optimization, software engineering, cloud computing, and intelligent transportation.



Yi Mei (M'09-SM'18) received the BSc and PhD degrees from the University of Science and Technology of China, Hefei, China, in 2005 and 2010, respectively. He is currently a Senior Lecturer at the School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand. His research interests include evolutionary scheduling and combinatorial optimisation, machine learning, genetic programming, and hyper-heuristics. He has over 100 fully referred publications, including the top journals in EC and Operations Research such as IEEE TEVC, IEEE TCYB, Evolutionary Computation Journal, European Journal of Operational Research, ACM Transactions on Mathematical Software. He serves as a Vice-Chair of the IEEE CIS Emergent Technologies Technical Committee, and a member of Intelligent Systems Applications Technical Committee. He is an Editorial Board Member/Associate Editor of three International Journals, and a guest editor of a special issue of the Genetic Programming Evolvable Machine journal. He serves as a reviewer of over 30 international journals.



Mengjie Zhang (M'04-SM'10-F'19) received the B.E. and M.E. degrees from Artificial Intelligence Research Center, Agricultural University of Hebei, Baoding, China, and the Ph.D. degree in computer science from RMIT University, Melbourne, VIC, Australia, in 1989, 1992, and 2000, respectively. He is currently Professor of Computer Science, Head of the Evolutionary Computation Research Group, and the Associate Dean (Research and Innovation) in the Faculty of Engineering. His current research interests include evolutionary computation, particularly genetic programming, particle swarm optimization, and learning classifier systems with application areas of image analysis, multi-objective optimization, feature selection and reduction, job shop scheduling, and transfer learning. He has published over 500 research papers in refereed international journals and conferences. Prof. Zhang is a Fellow of Royal Society of New Zealand, a Fellow of IEEE, and an IEEE Distinguished Lecturer. He was the chair of the IEEE CIS Intelligent Systems and Applications Technical Committee, the chair for the IEEE CIS Emergent Technologies Technical Committee, the chair of Evolutionary Computation Technical Committee, and a member of the IEEE CIS Award Committee. He is a vice-chair of the Task Force on Evolutionary Computer Vision and Image Processing, and the founding chair of the IEEE Computational Intelligence Chapter in New Zealand. He is also a committee member of the IEEE NZ Central Section.