Conflict-Free Genetic Algorithm with Nash Equilibrium Seeking for Game-based Battery Swapping Station Recommendation

Chang-Long Sun, Xin-Xin Xu (Co-First Author), Chun-Hua Chen (Corresponding Author), *Member*, *IEEE*, Jun Hong, Sam Kwong, *Fellow*, *IEEE*, and Zhi-Hui Zhan (Corresponding Author), *Fellow*, *IEEE*

***Abstract*— The rapid growth of electric vehicles (EVs) has led to significant challenges in providing efficient and sustainable charging solutions. This paper addresses the battery swapping station (BSS) recommendation problem by proposing a novel conflict-free genetic algorithm (CFGA) integrated with a Nash equilibrium seeking (NES) approach to identify optimal Nash equilibrium (ONE) solutions to such a non-cooperative optimization problem. The CFGA employs specialized crossover and mutation operators to generate offspring that satisfy the constraints of the problem, ensuring that each EV decides a unique battery swap strategy without conflict. Firstly, an order crossover operator is proposed to preserve the order of genes in the chromosomes. Secondly, a replacement and exchange mutation operator is proposed to enhance mutation diversity. The resulting optimal solution is then used as the initial strategy for the NES, which iteratively converges to the ONE. The proposed CFGA with NES algorithm is evaluated under both small-scale and large-scale cases, demonstrating its effectiveness in achieving a balance between costs for EVs and utilization for BSSs. The study's findings have practical implications for the smart grid and EV integration, offering a robust method for optimizing EV infrastructure and operations.**

Keywords*—Electric vehicles, battery swapping stations, optimal Nash equilibrium, evolutionary computation, order crossover, replacement and exchange mutation, conflict-free

I. INTRODUCTION

Faced with the global energy crisis and environmental pollution, electric vehicles (EVs) are increasingly recognized as a sustainable transportation solution [1]. However, EV development is hindered by the insufficient number of charging stations (CSs) compared to refueling demand [2]. Battery swapping, where customers exchange depleted batteries for charged ones at a battery swapping station (BSS) in minutes, is a promising solution [3]. It prevents issues associated with simultaneous EV charging [4] and extends battery life [5] offering benefits for power system operations [6]. The BSS recommendation problem [7] is a key research topic, focusing on recommending the most suitable BSS for EV to maximize BSS utilization or minimize costs for EV drivers.

The BSS recommendation problem is generally modeled as an optimization problem [8] or a non-cooperative game problem [9]. Therefore, the methods in the literatures that are proposed for solving the BSS recommendation problem can be mainly classified into two categories, one is the optimization-based methods while the other is the game-based methods.

The optimization-based methods generally aim to minimize the travel costs for EVs, reduce electricity costs, or optimize BSS operations. The problem can also be treated as a multi-objective problem [10] to obtain the Pareto front of the cost of each EV. Studies in [11] employs particle swarm optimization algorithm [12] to explore the optimal location of BSS to reduce power grid losses and enhance system reliability. To avoid BSS congestion and reduce the total travel costs for EVs, a bipartite method to address the BSS recommendation problem was employed in [13]. The work [14] introduced a real-time algorithm for the BSS recommendation problem to a group of EVs, with the goal of reducing the additional travel distance for EVs and preventing long waiting times. However, it should be noted that the individual choice of a BSS by one EV may affect the choices of other EVs. Consider such a scenario: although a BSS allocation has been recommended with the lowest total cost for all the EVs, there might be a BSS with a lower cost available for a particular EV. Then this EV would naturally change its strategy to choose the BSS with the lowest cost. This change could lead to increased costs for other EVs, prompting them to adjust their choices as well, potentially making the recommended plan unstable.

This work was supported in part by the National Key Research and Development Program of China under Grant 2022ZD0120001, in part by the National Natural Science Foundations of China (NSFC) under Grant 62176094 and Grant U23B2039, in part by the Tianjin Top Scientist Studio Project under Grant 24JRRCRC00030, and in part by the Fundamental Research Funds for the Central Universities, Nankai University (Grant 078-63243159).

C.-L. Sun and J. Hong are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China.

X.-X. Xu is with the School of Computer Science and Technology, Ocean University of China, Qingdao 266100, China.

C.-H Chen is with the School of Software Engineering, South China University of Technology, Guangzhou 510006, China.

S. Kwong is with the Department of Computing and Decision Science, Lingnan University, Hong Kong SAR.

Z.-H. Zhan is with the College of Artificial Intelligence, Nankai University, Tianjin 300350, China (e-mail: zhanapollo@163.com).

In contrast, the game-based methods focus more on the interactions between EVs, aiming to achieve a stable state where the strategies of all EVs remain unchanged, known as the Nash equilibrium [9]. In recent years, many researchers have used game theory to study BSS recommendation problems, as reflected in [15], [16], and [17]. An effective game-theoretic method to recommend CSs for EVs was proposed to minimize travel time and waiting time in [15]. In addition, the work in [16] has explored the optimal dispatch problem of EVs through a Stackelberg game model, emphasizing the role of electricity price control strategies in guiding EVs to suitable CSs. Similarly, the work in [17] has presented a framework for optimizing the selection of CSs for EVs based on an oligopolistic game-theoretic approach, utilizing differentiated product theory and conjectural variations to dynamically adjust prices and quantities. However, the Nash equilibrium represents a balance of benefit among EVs, and there may be a solution that yields a higher expected benefit for all EVs than a certain Nash equilibrium point.

To integrate the advantages of the optimization-based methods and game-based methods, this paper proposes modeling the BSS recommendation problem as a non-cooperative optimization problem [18] (NCOP) and the goal of NCOP is to find an optimal Nash equilibrium (ONE) solution. Note that in a non-cooperative game problem, there may be multiple NE solutions. However, in our NCOP, the goal of ONE is to find the optimal one (e.g., the lowest total cost one). To find the ONE, we firstly propose a novel conflict-free genetic algorithm (CFGA) to find an optimal solution to the NCOP. Then, the resulting optimal solution is used as initial strategy for a Nash equilibrium seeking (NES) approach [7], which will iteratively converge to the ONE. This way, the integrated CFGA-NES algorithm can find the ONE to the BSS recommendation problem.

The remaining part of this paper includes three sections. Section II provides background knowledge on related topics, including the definition of non-cooperative game problem, and the BSS recommendation system model. Section III presents the CFGA-NES algorithm to solve the BSS recommendation problem. The experimental results are presented in Section IV. The Section V provides some conclusions of this paper.

II. Background

A. Non-cooperative Game Problem

The non-cooperative game [9] refers to a process where a group of players, given their respective strategy spaces, select strategies with the aim of maximizing their expected utility, and ultimately, a set of outcomes is realized based on the strategies of all players involved.

Let the strategy space of player i be denoted as S_i, the strategy of player i be denoted as $x_i \in S_i$, and x_{-i} be the vector of strategies taken by all other players except i. the joint strategy of each player is denoted as $x = (x_i, x_{-i})$. The strategy x_i^* of player i is the best response of the opponent's strategy x_{-i} if

$$f_i\left(x_i^*, x_{-i}\right) \le f_i\left(x_i, x_{-i}\right), \forall x_i \in S_i \tag{1}$$

where f_i represents the cost of player i. A Nash Equilibrium refers to a strategy combination in which all players are the best response of the opponent's strategy. There are many algorithms for seeking a Nash equilibrium, such as the best response method [19], quasi-newton methods [20], iterative methods and so on. Among them, the NES based on iterative method [7] is adopted in this paper for its fast convergence speed and suitability for large-scale problems.

B. BSS Recommendation System Model

The BSS Recommendation System is designed to recommend suitable BSSs for EVs within a certain area, and its framework is shown in Fig. 1. The battery swapping station - recommendation center (BSS-RC) communicates with EVs and BSSs via a wireless network and is the core part of the algorithm's operation. Its functions include receiving the location and decisions of EVs, receiving the swapping prices from BSSs, and calculating strategies to reduce the cost for EVs and achieve coordinated utilization of BSSs.

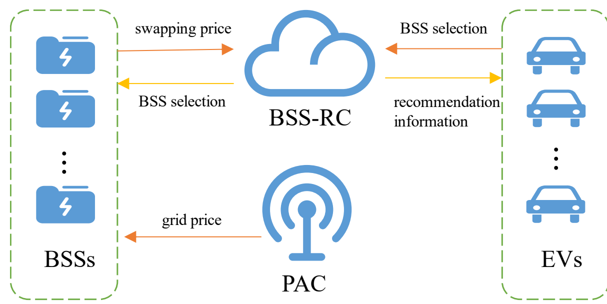

Fig. 1. Framework of BSS recommendation system model.

Assuming there are D EVs within a certain area and K BSSs, for a certain BSS k, it contains J_k batteries. A tuple vector of size $2 \times D$ is used to represent the joint strategy of each EV choosing a BSS and a battery as follows:

$$\boldsymbol{x} = \left[\left(k_1, j_{k_1}\right), \left(k_2, j_{k_2}\right), \ldots, \left(k_D, j_{k_D}\right)\right] \tag{2}$$

The strategy $x_d = (k_d, j_{k_d})$ corresponding to the EV d is to choose the battery j_{k_d} from the BSS k_d, and it must satisfy the condition

$$k_d \in [1, K] \wedge j_{k_d} \in \left[1, J_{k_d}\right] \tag{3}$$

During the selection process of each EV towards BSSs, a series of constraints must be met to simulate a real-world environment. The constraints that need to be satisfied are as follows:

1) Conflict-free Constraint

Different EVs can only choose different batteries at the same moment:

$$\forall d_1, d_2 \in [1, D], d_1 \ne d_2 \wedge \left(k_{d_1}, j_{k_{d_1}}\right) \ne \left(k_{d_2}, j_{k_{d_2}}\right) \tag{4}$$

2) Remaining Battery Charge Constraint

When an EV arrives at the BSS, it is required to retain a certain amount of battery charge. Assuming the average speed of the EVs is v, and the energy consumption per kilometer for the EV is ΔE_c, then the total energy consumption $E_{d,k}^{cost}$ from the EV d to the BSS k can be calculated as:

$$\Delta E_c = av^2 - bv + c \tag{5}$$

$$E_{d,k}^{cost} = s_{d,k}\Delta E_c \tag{6}$$

where a, b, and c are respectively 7.344×10^{-5}, 7.656×10^{-3}, and 0.3536, representing the coefficients that describe the relationship between the battery consumption per kilometer and the speed. The remaining battery charge of the vehicle and the corresponding constraint can be calculated as:

$$E_{d,k}^{res} = E_d^{init} - E_{d,k}^{cost} \tag{7}$$

$$E_{d,k}^{res} \geq \hat{E}_d \tag{8}$$

where E_d^{init} represents the initial battery charge of EV d, $E_{d,k}^{res}$ represents the remaining battery charge of EV d upon arrival at BSS k, and $\hat{E}_d$ represents the minimum threshold of the remaining battery charge that needs to be preserved.

3) Sufficient Battery Charge Constraint

A certain amount of electricity should be reserved when EVs arriving at the BSS. The corresponding constraint is as:

$$E_{d,k}^{j} \geq E_d^{thr} \tag{9}$$

where $E_{d,k}^{j}$ represents the amount of battery charge after the EV undergoes a battery swapping, and E_d^{thr} represents the minimum threshold of the battery charge required to leave the BSS.

After ensuring that the strategy chosen by each EV satisfies the constraints, the BSS adjusts the electricity prices accordingly, and the resulting battery swapping prices are as:

$$E_{d,k}^{swap} = E_{d,k}^{j} - E_{d,k}^{res} \tag{10}$$

$$\omega_k = p_{grid}\left(1 - \frac{\sum_{j=1}^{J_k} E_k^j + R_k - \sum_{d=1}^{D} \mathbb{I}_{\{k_d = k\}} E_{d,k_d}^{swap}}{J_k}\right) \tag{11}$$

$$p_k = p_{grid} + \omega_k \tag{12}$$

where $E_{d,k}^{swap}$ represents the amount of electricity swapped between EV d and BSS k, R_k represents the sum of newly added charging energy of all batteries, p_{grid} represents the grid price, $\mathbb{I}_{\{.\}}$ represents the indicator function that takes the value 1 if a particular condition is met and 0 otherwise, and ω_k represents the service price. If this problem is modeled as a non-cooperative game, the objective of each EV d is:

$$\begin{cases} \min\ f_d(x_d, x_{-d}) = \alpha\, p_{k_d} E_{d,k_d}^{swap} + \beta \tau_d E_{d,k}^{cost} \\ s.t.(4)-(9) \end{cases} \tag{13}$$

where α and β are the weighted coefficients used to adjust the battery swapping cost and travel distance cost of the EV, τ_d denotes the travel cost coefficient of EV d. If this problem is modeled as a NC-OP, the objective is:

$$\min obj = \sum_{d=1}^{D} (f_d(x^*) + \gamma_d)$$
$$s.t. \begin{cases} x_d^* = \arg\min\ f_d(x_d, \hat{x}_{-d}) \\ \gamma_d = \gamma \cdot \mathbb{I}_{\{x_d^* \notin \{x_d | (4)-(9)\}\}} \end{cases} \tag{14}$$

where γ represents the penalty coefficient for violating constraints.

C. NES Algorithm Based on Iterative Method

The calculation process of NES is briefly explained as follows: Firstly, initialize the parameters of the model. Next, randomly select initial the joint strategy of each EV that meet the constraints. Then, all BSSs adjust the swapping price based on the received EV selection strategies through (10)-(12). Afterward, for each EV d, calculate the cost to all BSSs based on the swapping price of BSS k and its distance to k, and select the BSS with the minimum cost to update its own strategy until all EVs have updated their strategies. Consequently, BSSs recalculate the swapping price based on the updated strategies of the EVs until the iterative rounds reach the maximum. The convergence of the NES algorithm has been fully proven in [7].

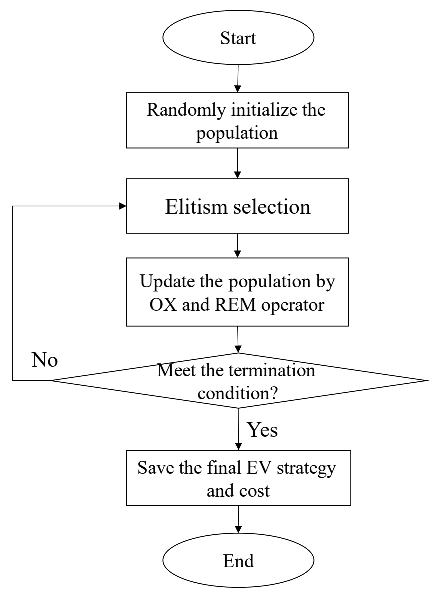

Fig. 2. Flowchart of the CFGA

III. Proposed CFGA-NES Algorithm

In this section, the CFGA-NES is proposed for solving the ONE solution of the NC-OP problem. First, the flowchart of CFGA is presented, with a brief explanation of its algorithm procedure. Secondly, the detailed algorithm flow of the CFGA that contains an order crossover (OX) operator and a replacement and exchange mutation (REM) operator is detailed introduced. Finally, the solution obtained from CFGA is used as the initial strategy for NES, and the entire CFGA-NES process is completed through iteration.

A. The Framework of CFGA

To solve the NC-OP problem, the CFGA is proposed. The CFGA employs special OX operator and REM operator that ensure the offspring generated must satisfy the constraint that an EV can only choose one BSS and Battery. The OX operator and the REM operator will be introduced later on.

Furthermore, to improve the convergence speed of the algorithm, CFGA employs an elitism strategy. Although the average fitness of the genetic algorithm population typically increases with the number of generations, there is always a possibility of losing the elite individuals of the current generation. Therefore, it is necessary to adopt an elitist strategy in order to ensure that the elite individuals reach the next generation. Before filling the population with offspring created through selection, crossover, and mutation, the top $size_{elite}$ individuals are copied into the next generation. The copied elite individuals still qualify for the selection process and can still serve as parents for new individuals.

The Elitism strategy has a significant positive impact on the performance of the algorithm because it avoids the potential waste of time required to rediscover excellent solutions lost during the evolutionary process, thus accelerating the convergence of the algorithm. The flowchart of the CFGA algorithm is shown in Fig. 2.

B. OX Operator

The goal of the OX operator [21] is to preserve the order information of the parent chromosomes while generating new offspring chromosomes. Besides, the OX operator ensures that each gene in a chromosome appears only once, preventing duplicate genes within a single chromosome through this crossover strategy. The details of the OX operator are presented in Algorithm 1.

Algorithm 1 OX

Input: p_1, p_2, D, $rate_{crossover}$
Output: c_1, c_2
Begin
If $rand\ (0,1) < rate_{crossover}$:
  Randomly select two different integers, r_1 and r_2 from 1 to D, such that $r_1 < r_2$
  $c_1 \leftarrow p_1[r_1 : r_2]$
  For each gene b in p_2:
    If $b \notin c_1$:
      $c_1 \leftarrow c_1 \cup \{b\}$
    End if
    If $|c_1| == D$:
      Break
    End if
  End for
  Similarly generating the second offspring c_2 by swapping the order of operations for p_1 and p_2
Else:
  $c_1 \leftarrow p_1$, $c_2 \leftarrow p_2$
End if
End

C. REM Operator

Similar to the OX operator, the REM operator can also ensure that each gene in a chromosome appears only once. The REM operator performs external replacement and internal exchange operations on genes within the chromosome, which increasing the diversity of mutation. The details of the REM algorithm are presented in Algorithm 2.

Algorithm 2 REM

Input: c, D, set_{all}, $rate_{mutation}$
Output: pop
Begin
The set of unchoosed elements $set_u = set_{all} - c$
For d=1 to D:
  If $rand\ (0,1) < rate_{mutation}$:
    Randomly choose an element e from set_u
    $c[d] \leftarrow e$
    $set_u \leftarrow \{e\}$
  End if
  If $rand\ (0,1) < rate_{mutation}$:
    Randomly exchange two elements of c
  End if
End for
End

D. Complete Procedures of CFGA-NES

After performing a complete CFGA, the output of the algorithm is merely an optimal total cost solution, not a ONE solution. It is necessary to use this optimal total cost solution as the initial solution for NES and iterate to ultimately find the NE, which is the ONE. The framework of the CFGA-NES algorithm is shown in Algorithm 3.

Algorithm 3 CFGA-NES

Input: $size_{elite}$, $size_{pop}$, D, $maxFEs$
Output: f_{opt}, x_{opt}
Begin
Initialize the population pop randomly
Calculate the fitness of pop
Calculate the set of all candidate elements set_{all}
$FEs = size_{pop}$
While $FEs < maxFEs$:
  Rank pop according to the fitness
  Elitism selection, Preserving the top $size_{elite}$ solutions as pop_{new}
  For $i = size_{elite}$ to $size_{pop}$
    Select two parents p_1, p_2 from pop by using roulette method
    Running Algorithm 1 on p_1, p_2 to obtain c_1, c_2
    Running Algorithm 2 on c_1, c_2
    $pop_{new} \leftarrow pop_{new} \cup \{c_1, c_2\}$
  End for
  $pop \leftarrow pop_{new}$
  Calculate the fitness of pop
  $FEs \leftarrow FEs + size_{pop}$
End while
Calculate the optimal fitness f_{opt} in the pop and its corresponding optimal solution x_{op}
Run NES with x_{op} as the initial strategy to obtain the ONE solution.
End

Firstly, the population is randomly initialized, and the fitness of each individual is calculated, ensuring that no elements within each individual in the population are repeated. Subsequently, a set containing all candidate elements, denoted as set_{all}, is identified for use in subsequent mutation operations. Afterward, an offspring population, denoted as pop_{new}, is created, preserving the elite individuals that rank in the top $size_{elite}$ according to their fitness. Using the roulette wheel algorithm, two parent individuals are randomly selected from the population pop, and they are successively subjected to crossover using the OX operator and mutation using the REM operator. The resulting offspring are added to pop_{new} until pop_{new} reaches its maximum capacity, $size_{pop}$. After the CFGA iteration is completed, the obtained optimal total cost solution is used as the initial solution to run the NES algorithm, ultimately yielding the ONE solution.

IV. Experimental Results

The specific experimental parameters of the BSS recommendation problem simulation environment have been detailed in [7]. In the experiment, we also include greedy selection (GS) algorithm that each EV greedily selects the BSS closest to itself as a comparative algorithm. Note that the GS algorithm does not guarantee that all EVs will meet the constraints, and these EVs will not be included in the cost calculation. Both small-scale-case with a small number of EVs and BSSs and large-scale-case with a larger number of EVs and BSSs are tested. At the same time, we conducted multiple independent repeated experiments to compare the average total cost among these algorithms. In addition to reducing the total cost of all EVs, another goal of the BSS recommendation problem is to increase the utilization rate of the BSS, i.e., the ratio of successfully swapped batteries in each BSS to the total number of BSS batteries. Follow-up experiments also carried out research on this metric.

A. Small-Scale Case -20 EVs and 5 BSSs

In the Small-Scale Case, consider an area with 5 BSSs, each equipped with a certain number of batteries. At a certain moment *t*, there are 20 EVs that require battery swapping. Fig. 3 illustrates the trend of the total cost of all EVs with during the iterative process for the GS, CFGA, NES, and CFGA-NES algorithms. It can be observed that when CFGA converges, the costs for each EV will remain constant, which implies the corresponding BSSs chosen by each EV will also remain unchanged. However, NES is an approximate NE seeking algorithm based on best response. Since during the iterative process, EVs can only decide their best response for this round based on the joint strategy information from the last iteration, and the change in their own strategy may affect the strategy of other EVs in the next iteration, the strategy of each EV may remain unchanged as the number of iterations increases, meaning that the joint strategy of the EVs has already reached a Nash equilibrium. But there may also be mutual interference, leading to fluctuations in the strategies of EVs. But this fluctuation is periodic, so it is also considered that the algorithm ultimately converges to a stable approximate NE. Note that in the CFGA-NES, the total cost of EVs converges quickly after only changed in the first few iterations, indicating that the optimal total cost solution obtained by CFGA is very close to the ONE solution, and sometimes it is indeed the ONE solution.

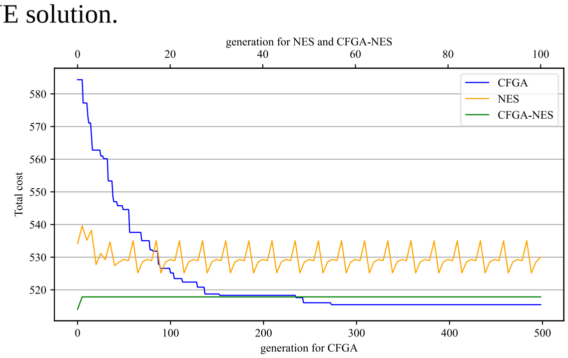

Fig. 3. Trend of total cost of 20 EVs by CFGA, NES and CFGA-NES under the small-scale case.

The best strategies of each EV under the small-scale case are shown in Fig. 4.

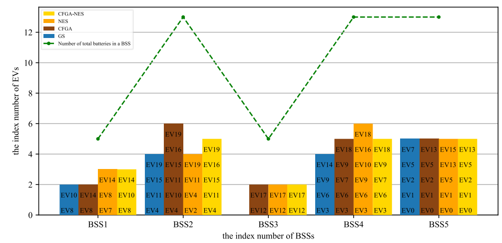

Fig. 4. BSSs selection of 20 EVs by CFGA, NES, and CFGA-NES under the small-scale case.

B. Large-Scale Case -120 EVs and 15 BSSs

In the Large-Scale Case, the trend of the total cost of all EVs with GS, CFGA, NES, and CFGA-NES are shown in Fig. 5. The curve represented by NES reveals that NES does not guarantee that the total cost of the final convergent strategy will be better than that of the initial strategy. This is because the goal of NES is to find a NE, even if the total cost of its initial strategy is better than the final strategy, the NE nature of the initial strategy cannot be guaranteed. NES will start from the initial strategy and use a smaller amplitude of change to find a NE. This is also the reason why the proposed CFGA-NES is effective. The results in Fig. 5 show that CFGA-NES is able to obtain a NE with a total cost not much different from the optimal total cost solution obtained by CFGA. Additionally, as the problem scale increases, the number of iterative rounds and the population size required by CFGA-NES also increase, which will greatly increase the time for algorithm solving. Therefore, although CFGA-NES is able to obtain the optimal NE, that is, the ONE solution, there is still room for optimization in its algorithm performance.

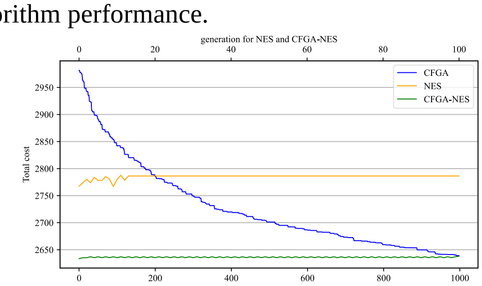

Fig. 5. Trend of total cost of 120 EVs by CFGA, NES, and CFGA-NES under the large-scale case.

The best strategies of each EV obtained by CFGA, NES, and CFGA-NES under the large-scale case are shown in Fig. 6, and similar conclusions can be drawn as those in Section IV-A. This indicates that CFGA still maintains good performance under large-scale-case.

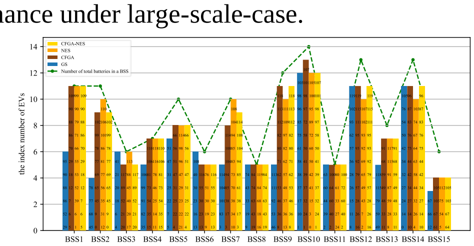

Fig. 6. BSSs selection of 120 EVs by CFGA, NES, and CFGA-NES under the large -scale case.

C. Algorithm Comparison with Several Metrics by Independent Repeated Experiment

Twenty independent repeated experiments were conducted for the two metrics for algorithm evaluation: the mean average total cost, represents the mean of the average costs calculated over each independent repeated experiments; the mean average BSS utilization ratio, represents the mean of the average BSS utilization ratio calculated over each independent repeated experiments; with the results shown in TABLE I. It shows that GS performs the worst in both metrics. CFGA has the lowest mean average cost in both cases, and the performance of CFGA-NES is very close to CFGA. NES has the highest Mean average BSS utilization ratio in both cases, and the performance of CFGA-NES is very close to NES. The results indicate that the total cost from CFGA-NES is the optimal (guaranteed by CFGA) NE (guaranteed by NES) solution, i.e., the ONE solution.

TABLE I. ALGORITHM EVALUATION WITH TWO METRICS

Algorithm	Mean average cost		Mean average BSS utilization ratio	
	Small scale	Large scale	Small scale	Large scale
GS	26.73±0.00	22.79±0.00	28.00±0.00	72.76±0.00
CFGA	**25.88±0.10**	**22.03±0.05**	41.60±1.21	88.26±0.33
NES	26.99±0.43	23.30±0.14	42.59±1.67	**88.85±0.30**
CFGA-NES	26.06±0.19	22.05±0.06	**42.59±1.26**	88.71±0.27

V. CONCLUSION

Our research highlights the algorithm's ability to balance EV costs and BSS utilization. Experiments across scales demonstrate the CFGA-NES algorithm's convergence to strategies minimizing EV costs and ensuring equitable EV distribution across BSSs, achieving an ONE. Despite its effectiveness, the algorithm can be further enhanced for computational efficiency and scalability, with matrix-based operators [22], learning-aid techiques [23][24], and distributed techiques [25][26]. Moreover, future research also aim at improving performance on larger instances and exploring applications within the broader smart grid ecosystem.

REFERENCES

[1] N. I. Nimalsiri, C. P. Mediwaththe, E. L. Ratnam, M. Shaw, D. B. Smith, and S. K. Halgamuge, "A survey of algorithms for distributed charging control of electric vehicles in smart grid," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4497–4515, Nov. 2019.

[2] J. Yang, W. Wang, K. Ma, B. Yang, and C. Dou, "Optimization and self-adaptive dispatching strategy for multiple shared battery stations of electric vehicles," *IEEE Trans. Ind. Informat*., vol. 17, no. 2, pp. 1363–1374, Feb. 2021.

[3] X. Chen, K. Xing, F. Ni, Y. Wu, and Y. Xia, "An electric vehicle battery swapping system: Concept, architectures, and implementations," *IEEE Intell. Transp. Syst. Mag.*, vol. 14, no. 5, pp. 175–194, Sep. 2022.

[4] H. Ko, S. Pack, and V. C. Leung, "An optimal battery charging algorithm in electric vehicle-assisted battery swapping environments," *IEEE Trans. Intell. Transp. Syst*., vol. 23, no. 5, pp. 3985–3994, May 2020.

[5] F. Ahmad, M. S. Alam, I. S. Alsaidan, and S. M. Shariff, "Battery swapping station for electric vehicles: Opportunities and challenges," *IET Smart Grid*, vol. 3, no. 3, pp. 280–286, Jun. 2020.

[6] M. R. Sarker, H. Pandžic, and M. A. Ortega-Vazquez, "Optimal operation and services scheduling for an electric vehicle battery swapping station," *IEEE Trans. Power Syst.*, vol. 30, no. 2, pp. 901–910, Mar. 2015.

[7] L. Ran, Y. Wan, J. Qin, W. Fu, D. Zhang and Y. Kang, "A game-based battery swapping station recommendation approach for electric vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 9, pp. 9849 - 9860, Sep. 2023.

[8] S. Muthuraman and V. P. Venkatesan, "A Comprehensive Study on Hybrid Meta-Heuristic Approaches Used for Solving Combinatorial Optimization Problems," in *World Congress on Computing and Communication Technologies*, 2017, pp. 185-190

[9] Nash, John. "Non-Cooperative Games." *Ann. Math*, vol. 54, no. 2, pp. 286–295, Sep. 1951.

[10] H. Gao, S. Kwong, J. Yang, J. Cao. "Particle swarm optimization based on intermediate disturbance strategy algorithm and its application in multi-threshold image segmentation," *Information Sciences*, vol. 250, pp. 82-112, Nov. 2013.

[11] S. CHANDROL and V. ANAND, "Optimal Location of Battery Swapping Station for EV Using PSO," *The 3rd International Conference on Intelligent Sustainable Systems*, 2020, pp. 1531-1536.

[12] K. Li, S. Kwong, K. Deb. "A dual-population paradigm for evolutionary multiobjective optimization," *Information Sciences*, vol. 309, pp. 50-72, July. 2015.

[13] P. You, J. Z. Pang, M. Chen, S. H. Low, and Y. Sun, "Battery swapping assignment for electric vehicles: A bipartite matching approach," in *Proc. IEEE Annu. Conf. Decis. Control*, 2017, pp. 1421–1426.

[14] P. You, S. H. Low, Z. Yang, Y. Zhang, and L. Fu, "Real-time recommendation algorithm of battery swapping stations for electric taxis," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, 2016, pp. 1–5.

[15] T. Guo, P. You, and Z. Yang, "Recommendation of geographic distributed charging stations for electric vehicles: A game theoretical approach," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, 2017, pp. 1–5.

[16] J. J. Escudero-Garzas and G. Seco-Granados, "Charging station selection optimization for plug-in electric vehicles: An oligopolistic game theoretic framework," in *Proc. IEEE PES Innov. Smart Grid Technol.*, 2012, pp. 1–8.

[17] H. Zhou, C. Liu, B. Yang, and X. Guan, "Optimal dispatch of electric taxis and price making of charging stations using Stackelberg game," in *Proc. Annu. Conf. IEEE Ind. Electron. Soc.*, 2015, pp. 4929–4934.

[18] Monfared, Mohammadali Saniee, Sayyed Ehsan Monabbati, and Atefeh Rajabi Kafshgar. "Pareto-optimal equilibrium points in non-cooperative multi-objective optimization problems," *Expert Systems with Applications*, vol. 178, no. 2021, pp. 114995, Sep. 2021.

[19] M. Fazli, A. Maazallahi, J. Habibi and M. Habibi, "Controlling Best Response Dynamics for Network Games," *IEEE Trans. Network Science and Engineering*, vol. 6, no. 3, pp. 225-236, Jul. 2019

[20] Tolwinski, Boleslaw. "Newton-type methods for stochastic games," in *Differential Games and Applications*. Berlin Heidelberg, Springer, 1989, pp. 128-144.

[21] Deep, Kusum, and Hadush Mebrahtu. "New variations of order crossover for travelling salesman problem." *Int. J. of Combinatorial Optimization Problems and Informatics*, vol. 2, no. 1, pp. 2-13, 2011.

[22] Z. H. Zhan, *et al.*, "Matrix-based evolutionary computation," *IEEE Trans. Emerging Topics in Comput. Intell.*, vol. 6, no. 2, pp. 315-328, April 2022.

[23] Y. Jiang *et al.*, "Knowledge learning for evolutionary computation," *IEEE Trans. Evol. Comput.*, DOI: 10.1109/TEVC.2023.3278132. May 2023.

[24] Z. H. Zhan, *et al.*, "Learning-aid evolution for optimization," *IEEE Trans. Evol. Comput.*, vol. 27, no. 6, pp. 1794–1808, Dec. 2023.

[25] J. Y. Li *et al.*, "Distributed differential evolution with adaptive resource allocation," *IEEE Trans. Cybern.*, vol. 53, no. 5, pp. 2791-2804, May 2023.

[26] Z. H. Zhan, *et al.*, "Cloudde: A heterogeneous differential evolution algorithm and its distributed cloud version," *IEEE Trans. Parallel and Distributed Systems*, vol. 28, no. 3, pp. 704-716, March. 2017.