This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



# A Many-objective Ensemble Optimization Algorithm for the Edge Cloud Resource Scheduling Problem

Jiangjiang Zhang, Zhenhu Ning, Raja Hashim Ali, Muhammad Waqas, *Senior Member, IEEE,* Shanshan Tu, *Member, IEEE,* Iftekhar Ahmad, *Senior Member, IEEE,*

***Abstract*—An edge cloud architecture plays a key role in improving the user task computing service system by combining the powerful data processing capability of cloud centres with the low latency of edge computing. Existing methods for maximizing the efficiency of an edge cloud architecture take into account time and task parameters but ignore other factors such as load balancing, cost, and user satisfaction when scheduling resources. In this work, we propose a many-objective resource scheduling model for optimizing the performance of an edge cloud architecture, which takes into account the time spent on task, cost, load balance, user satisfaction, and trust measurement. The resource scheduling model converges to the optimal solution using a novel many-objective ensemble optimization algorithm based on a dynamic selection mechanism. The study also explores the support set convergence of eight evolutionary operators using the ensemble algorithm. The model solutions are dynamically updated with the help of the dynamic integration probability, and then a selection criteria is used to pick the best solutions from the pool of generated solutions. Two simulations on a benchmark dataset are used to verify the usefulness and performance of the designed algorithm. Our approach was able to locate more than half of the best solutions on the benchmark functions, and it also showed to be a better model solution than the some of the popular many-objective algorithms for dealing with the edge cloud resource scheduling problem, according to the results obtained from the simulations.**

***Index Terms*—Cloud computing, mobile edge computing, ensemble learning and resource scheduling.**

## I. Introduction

THE fields of cloud computing, big data, and the Internet of Things (IoT) have gained prominence and attention as a result of recent advancements in architecture, algorithms, and technology [1], [2]. The integration of these disciplines has been successfully deployed in healthcare [3], [4], smart metering [5], [6], smart urban surveillance [7], [8], smart transportation [9], [10], smart factories [11], [12], and smart homes [13], [14], among other applications. The rapid progress and popularity of internet of things and cloud computing for applications in many fields have lead to an exponential growth in the amount of data generated and passed over to the cloud for processing. In addition, the recent advancements in networks and communication technology are another reason for the exponential growth in the demand of computational power for the internet of everything (IoE) and the cloud [15]. Hence, in modern times, efficient resource scheduling is one of the core components of cloud computing algorithms for provision of service in a reasonable amount of time [16] and low latency resulting from efficient resource scheduling has become a necessary characteristic for cloud computing systems, in particular, for real time systems [17], [18].

In a typical traditional cloud computing architecture, the IoT devices collect and transfer the data over the network to the cloud, where it is stored, processed, and results are transferred to the IoT device [19]. In such a system, the IoT device has no computational abilities and the complete data processing is done by the cloud. Note that traditional cloud computing approaches may result in accumulation of substantial amount of unprocessed and processed data due to difference in speed of processing in the cloud and the rapid transfer of data from the device to the cloud [20]. This leads to considerable latency in the traditional cloud computing architectures, wastage of resources on the cloud, network and IoT devices, and places a significant strain on bandwidth of cross-domain link [21].

As an alternate to traditional cloud computing architecture, edge computing architecture have been proposed in recent times, which deploy a distributed computing paradigm where edge computing extends cloud computing from the centralized computing to the edge of the network [22]. Essentially, the computing devices for edge computing are typically positioned near the data source, on the edge side of cloud computing, so that fast access to services can be achieved and a distributed, low-delay, and continuous work properties are available [12], [23]. Such a mechanism leads to data forwarding, storage, computing and analysis being processed on the edge devices, deployed in the closest position to the user's network [24], and plays an important role in reducing the response delay, cloud pressure, bandwidth cost, and providing a timely service response to the users [9], [25]. By using cloud edge collaborative

J. Zhang, Z. Ning and S. Tu are with the Faculty of Information Technology, Beijing University of Technology, 100124 Beijing, China, (e-mail: zhangjiangjiang@emails.bjut.edu.cn, ningzhenhu@bjut.edu.cn, sstu@bjut.edu.cn ).

R. H. Ali is with the Faculty of Computer Science and Engineering, Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Topi 23460, Pakistan (e-mail: hashim.ali@giki.edu.pk).

M. Waqas is with the Computer Engineering Department, College of Information Technology, University of Bahrain, 32038, Bahrain and School of Engineering, Edith Cowan University, Perth WA 6027, Australia (e-mail: engr.waqas2079@gmail.com).

I. Ahmad is with the School of Engineering, Edith Cowan University, Perth WA 6027, Australia (e-mail: i.ahmad@ecu.edu.au)

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



computing, the network function management layer strategy can be distributed to the center cloud and edge cloud to realize the flexible deployment of network functions and intelligent management of cloud server resources, where the controller is responsible for the efficient forwarding configuration of the underlying data plane and data stream [26]. Based on the system's current load and resource status, a unified computing task scheduling and resource optimization configuration strategy is employed to execute tasks on the deployed edge nodes and the central cloud. When the request for user's task arrives, the collaborative scheduling of computing tasks is realized on the premise of guaranteeing the user requirements [3], [5]. However, due to the physical limitations of computing facilities, the computing capacity of edge cloud computing is limited [19], [27], which may cause uneven allocation of resources, among other issues. As a result of the uneven allocation of resources, there will be a load imbalance, resulting in a backlog of "hot spots" in specific computer clusters [20], [24]. Therefore, it is critical to investigate how to improve the task computing service system by combining the cloud data center's robust data processing capabilities with the low latency of edge clusters [5], [28].

The demand for flexibility in edge cloud resource scheduling strategies, particularly for real-time large applications, has skyrocketed in recent years and further optimization of the resource scheduling strategy is required to satisfy the expectations of users [5], [8]. Several significant contributions have been made to address this challenge. Liu *et al.* [29] presented the joint optimization of resource allocation of computing resources and wireless bandwidth in a cloudlets and clouds coexist system, and a multi-dimensional resource optimization strategy employing a general-purpose linear programming solution and expressing the optimization problem as a semi-Markov decision process (SMDP) [30] was proposed. The simulation in [29] resulted in a significant reduction in energy consumption for the scheduling tasks system between micro-cloud and cloud servers. By integrating edge association and resource allocation, Luo *et al.* [21] proposed a novel hierarchical federated edge learning to minimise the core network's transmission cost while ensuring data privacy security [31]. By concurrently scheduling network resources in cloud radio access networks and computing resources in edge computing, Wang *et al.* [1] established a unified mobile service provider performance trade-off framework to maximise mobile service providers' profit [32]. Sotiriadis *et al.* [33] established a cross-cloud task scheduling framework to maximize the performance of the participating cloud, which optimized the scheduling indicators of task execution, completion time, and turn-around time based on the design of the message switch showing noticeable improvement in the performance of single cloud scheduling performance.

Although only a limited few factor(s) affect edge cloud resource scheduling (ECRS), a variety of criteria should be balanced in the ECRS system in order to produce good results, including job completion time, cost, load balancing, user satisfaction, and trust measurement [10], [33], [34]. Many-Objective Optimization Problems (MaOPs) feature more than three objective functions, posing a significant challenge to the algorithm's convergence and diversity (CaD) maintenance [35], [36]. According to Cheung *et al.* [37], Many-Objective Evolutionary Algorithms (MaOEAs) can be employed to solve MaOPs, where the various functions of an MaOEA are not the consequence of a single element, but of the integration and coordination of interdependent and interacting factors via appropriate mechanisms [36]. Designing an efficient ensemble of various many-objective evolutionary algorithms not only maintains MaOEA's advantages in solving problems of high complexity, but also purposefully selects different operators to improve the efficiency of solving these problems based on their characteristics, thereby improving the algorithm's overall evolution efficiency. When addressing a MaOP, the population's early evolution should focus on the convergence of solutions, while the late evolution should focus on the diversity of solutions [35], [36]. One of the most challenging aspect in building a high performance ensemble of many-objective evolutionary algorithms is considering the importance of CaD at various phases of development [38]. Many existing solutions in the literature consider a selected time and task factors for finding an optimal solution for the ECRS problem. In this paper, we view the ECRS problem as a MaOP [10], and we propose an efficient ensemble of MaOEA to solve the ECRS problem utilising five factors (the complete time spent on task, cost, load balance, user satisfaction, and trust measurement).

Following is a list of the notable contributions made in this study.

- A many-objective edge cloud resource scheduling model is designed to describe the problem in detail, with five objectives to be optimized: time for task completion, cost of server completion, degree of server load balancing, degree of user satisfaction, and trust measurement between task and server factors.
- A novel MaOEA based on dynamic probabilistic selection ensemble (MaOEA-DPS) is devised to find the model solution, where the MaOEA-DPS initially looks at the support set convergence of eight evolutionary operators and then the evolutionary operators of the ensemble algorithm are picked from three popular evolutionary algorithms. Through mutual coordination and collaborative action, each evolutionary strategy can contribute its different benefits and generate more excellent solutions for selection under the dynamic probability ensemble mechanism. In addition, a selection mechanism is employed to select and store the excellent solutions, which will overcome the problem of declining selection pressure in the later stage of evolution and help achieve a "strong combination" and superior performance in balancing CaD.
- Two substantial validation tests are conducted on the benchmark function and the ECRS problem, where the suggested ensemble method produces more than half of the good solutions on the benchmark function when compared to other advanced MaOEAs. In addition, as

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



compared to the involved algorithm in addressing the ECRS problem, a superior model solution is obtained.

The rest of this study is structured in the following manner. A detailed discussion of the ECRS problem and model design is presented in Section II. Then, in Section III, the notion of developing a novel many-objective ensemble method based on dynamic probability selection is explained. Two extensive validation tests are then performed in Section IV. Finally, Section V brings the paper to a close.

## II. Edge Cloud Resource Scheduling Problem

The volume of task data provided by users is growing exponentially [39] in tandem with major advancements [30], [40] in the field of artificial intelligence [9], [41]. In the traditional cloud computing paradigm, the data for these tasks is sent to the cloud data centre, which performs centralized processing and the devices are solely utilized for data collection and transmission to the cloud. The edge cloud computing paradigm, on the other hand, is based on the availability of computing capabilities on both the data centre (called the cloud in IoE) and the data collection device (called the edge in IoE) [1], as depicted in Figure 1. The edge cloud computing architecture essentially creates a flexible cloud platform with numerous important options, such as edge computing, network, storage, security, and so on. This concept requires and forms a technical framework for "cloud-edge-collaboration" between the cloud and the edge, where part of the decision making from data analysis and processing tasks can be handled on the edge to reduce response delay, cloud pressure, and bandwidth costs [42]. In addition to these capabilities, the configurable cloud platform of the edge cloud computing paradigm can provide the same services as traditional cloud computing, such as network scheduling and computing power distribution.

### A. Problem Description

A central cloud and numerous edge clouds make up an edge cloud resource scheduling system in general. The central cloud is in charge of controlling and monitoring each edge cloud platform, as well as coordinating resource allocation among platforms to satisfy the demands of users' tasks [1], [7]. For each edge cloud platform, the user-side equipment delivers tasks to the server, and the edge server decomposes the tasks into many sub-tasks, which are sent to different edge devices (EDs) for computing [43]. Large tasks are scheduled to the cloud computing center for processing due to the limited computing capability of EDs. The data is then fed back to the user equipment layer by layer via the cloud computing center. The entire resource scheduling procedure of the system can be summarized as depicted in Figure 1. The $n$ tasks in set $Task = \{task_1, task_2, \cdots, task_n\}$ are scheduled reasonably on the $m$ EDs computing nodes in the ED computing node-set $ED = \{ed_1, ed_2, \cdots, ed_m\}$ (where, $m < n$), in order to maximize the efficiency of task completion [44]. The matrix $D$ (shown in Equation 1) represents the corresponding relationship between the user task and the ED computing node. A row in $D$ denotes the $m$ ED computing node queue, a column in $D$ denotes the $n$ task queue, and $d_{ij}$ denotes the $task_j$ is assigned for completion.

$$D = \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mn} \end{pmatrix}. \tag{1}$$

In most cases, an edge cloud task is assigned to finish the calculation on one of the EDs that can handle several tasks. Therefore, $d_{ij} \in (0, 1)$, $\sum_{i=1}^{m} d_{ij} = 1 (j = 1, 2, \cdots, n)$. According to the corresponding relation matrix, the completion time matrix of ED is constructed as shown in Equation 2, where $tet_{ij}$ denotes the completion time of $task_j$ on $ed_i$.

$$TET = \begin{pmatrix} tet_{11} & tet_{12} & \cdots & tet_{1n} \\ tet_{21} & tet_{22} & \cdots & tet_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ tet_{m1} & tet_{m2} & \cdots & tet_{mn} \end{pmatrix}. \tag{2}$$

In addition, certain attributes of tasks and edge computing node for cloud resource scheduling system are described in Table I [35].

TABLE I
Description of edge node and task related attributes

| Type | Attribute | Description |
|---|---|---|
| $ed_i$ | $ed_i^{bandwidth}$ | bandwidth of $ed_i$ |
| | $ed_i^{CPU}$ | CPU of $ed_i$ |
| | $ed_i^{mips}$ | mips of $ed_i$ |
| $Task_j$ | $Task_j^{filesize}$ | bandwidth of $ed_i$ |
| | $Task_j^{outputsize}$ | outputsize of $Task_j$ |
| | $Task_j^{length}$ | length of $Task_j$ |
| | $Task_j^{waittime}$ | waittime of $Task_j$ |
| | $Task_{ed_i}^{deadline}$ | deadline of $Task_j$ |
| | $Task_j^{budget}$ | budget of $Task_j$ |

Some notable desired characteristics of an optimal ECRS algorithm in an edge cloud environment that impacts user satisfaction positively are security of the system, lowered total execution time, reduced costs, and avoiding uneven resource scheduling. Since the system is unable to differentiate between malicious fraudulent service and normal service during resource scheduling because of the virtualization characteristics of the edge cloud environment, hence the security of the entire scheduling mechanism is of utmost importance for user in the edge cloud environment [45]. Similarly, the total execution time for the task has a direct impact on user satisfaction with the computing service. Additionally , edge cloud service operators should minimize idle machines in the edge-cloud platform to reduce costs. In addition, uneven resource scheduling results in poor load balancing of the edge

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



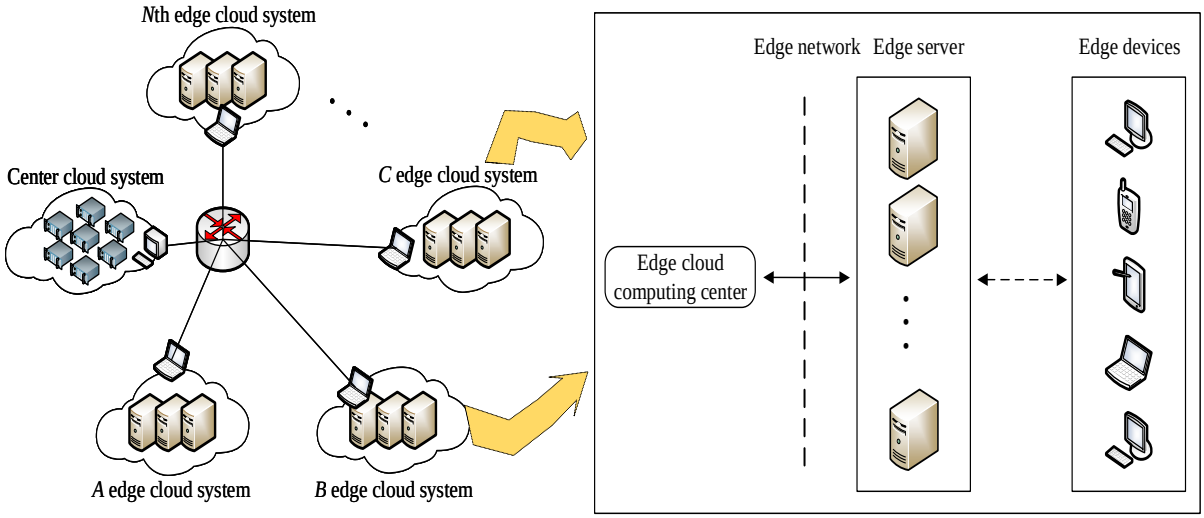



Fig. 1. **A Typical Edge Cloud Resource Scheduling System** - The figure displays a typical edge cloud resource scheduling system, with a center cloud system and N edge cloud systems. Each edge cloud system has computing capability, with an edge network connecting the edge cloud computing center to the edge servers which are then connected to the edge devices. The tasks need to be coordinated and managed at both the center cloud system as well as edge servers to get the most optimal solution often with mutually conflicting factors.

cloud server, which in turn leads to reduced resource utilization of the corresponding edge cloud and higher operation and consumption costs [9]. Note that the factors that affect the ECRS problem are restricted, with conflicted goals, and influenced, and thus the ECRS problem can be regarded as a Many Objective Problem (MaOP). The objective function formulated below is then used to express the many-objective edge cloud resource scheduling model.

### B. Objective Function

*1) Completion time:* Let $A_1$, $A_2$, and $A_3$ represent the million instructions per second (MIPs) for performance, bandwidth and CPU of the $ed_i$. Then, the transmission time of $task_j$ from edge cloud to $ed_i$ is represented by $T_{ed_i}^{transmission}$ and shown in Equation 3.

$$T_{ed_i}^{transmission} = \frac{Task_j^{filesize} + Task_j^{outputsize}}{ed_i^{A_2}}. \tag{3}$$

Then, the computing time of $task_j$ on $ed_i$ is represented by $T_{ed_i}^{computing}$ and shown in Equation 4.

$$T_{ed_i}^{computing} = \frac{Task_j^{length}}{ed_i^{A_3} \cdot ed_i^{A_1}}. \tag{4}$$

Also, note that when a large number of tasks are to be scheduled given the limited computing resources of EDs, the waiting time of some tasks (shown in Equation 5) may be non-zero and is described by $T_{ed_i}^{waittime}$ [35].

$$T_{ed_i}^{waittime} = Task_j^{waittime}. \tag{5}$$

Then the completion time $tet_{ed_i}$ of each $task_j$ can be calculated as shown in Equation 6.

$$tet_{ed_i} = T_{ed_i}^{transmission} + T_{ed_i}^{computing} + T_{ed_i}^{waittime}. \tag{6}$$

In the resource scheduling decision-making scheme of the whole ECRS system, the task with the maximum task completion time (represented by $T_{ed_i}$ and shown in Equation 7) in the task execution queue directly affects the execution efficiency.

$$T_{ed_i} = \sum_{j=1}^{n} tet_{ed_i} \cdot d_{ij}. \tag{7}$$

Note that typically assignment tasks take the shortest time to complete. Suppose $x$ represents an edge cloud task resource scheduling scheme, then the service completion time for $x$ represented by $T_{ed_i}(x)$ is shown in Equation 8.

$$T_{ed_i}(x) = \max(T_{ed_1}, T_{ed_2}, \cdots, T_{ed_m}). \tag{8}$$

Since the minimum service completion time is preferred, the completion time of resource scheduling decision-making scheme $f_1(x)$ is shown in Equation 9.

$$f_1(x) = \min(T_{ed_i}(x)). \tag{9}$$

*2) Completion Cost:* In general, a high-quality service of user tasks should be obtained at the lowest possible cost. Therefore, resource providers need to explore reducing service charges based on typical service needs of the user. Since the investment cost of leasing edge cloud servers represents the majority cost investment of the provided service in edge cloud computing systems, we limit the rental cost of $ed_i$ to the cost of the million instructions per second (MIPS) of CPU of $ed_i$, network bandwidth, and memory space [46].

Suppose $Mony_{A_{1,2,3}}$ denotes three types of cost in the scheduling process of using $ed_i$, and $Y$ describes whether the

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



$task_j$ is assigned to the $ed_i$, where $Y$=0 denotes the $task_j$ is not assigned to the $ed_i$ and $Y$=1 denotes the $task_j$ is assigned to the $ed_i$. Then, the total cost of the entire task can be calculated as follows, where $f_2(x)$ represents the completion cost of the best resource scheduling decision-making system and is shown in Equation 12.

$$Kind = \frac{Task_j^{length}}{ed_i^{A_1}} + \frac{Task_j^{filesize} + Task_j^{outputsize}}{ed_i^{A_2}} + \frac{Task_j^{length}}{ed_i^{A_3}}, \tag{10}$$

$$Task_{Total}^{\text{Cost}} = \sum_{i=1}^{m}\sum_{j=1}^{n}(ed_i^{A_{1,2,3}} \cdot Mony_{A_{1,2,3}} \cdot Kind \cdot Y), \tag{11}$$

$$f_2(x) = \min(Task_{Total}^{\text{Cost}}). \tag{12}$$

*3) Degree of user satisfaction:* User satisfaction can be calculated from the computing power and service level of the edge cloud resource scheduling system, and has a direct impact on service providers' resource scheduling decisions. If the user satisfaction level is too low, for example, a high number of tasks submitted by the user will be unable to be finished on time [35]. As a result, whether from the perspective of users' or service providers' interests, the degree of user satisfaction ($f_3(x)$) should be considered in the resource scheduling process, and is expressed in Equation 13.

$$f_3(x) = \max((\sum_{j=1}^{n}(\frac{T_{ed_i}}{Task_{ed_i}^{deadline}} + \frac{Task_j^{\text{cost}}}{Task_j^{budget}}))/n). \tag{13}$$

*4) Degree of load balance:* The edge cloud server displays varied resource scheduling service levels due to the differences in resource scheduling tasks in the edge cloud system. The idle phenomenon of available computer resources is observed due to non-scheduling of all tasks on some resources, because of difference in performance optimality among various edge devices of the ECRS system. Edge cloud server resource load balancing is connected to whether or not the service providers of ECRS system are completely and systematically exploited, with a direct impact on the time it takes to complete a task, the cost, and user satisfaction. A standard deviation technique is used to determine the degree of load balancing. The lower the standard deviation coefficient, the more load-balanced resource scheduling jobs can be assigned to edge devices. The specific load balancing degree ($f_4(x)$) can be seen in Equation 14.

$$f_4(x) = \min Load_{ed_i} = \min \frac{\sum_{j=1}^{n}(\frac{(tet_{ed_i} - Load_{ed_i}^{mean})^2}{N})}{Load_{ed_i}^{mean}}, \tag{14}$$

where $Load_{ed_i}^{mean} = \frac{T_{ed_i}}{N}$.

*5) Trust Measurement:* Because of virtualization features, edge cloud resource scheduling systems are insecure, making it difficult for cloud users to identify their service capabilities and meet their service requirements. Furthermore, because of malicious and fraudulent cloud services, the trust measurement value of services may be lowered. As a result, cloud resource scheduling service trustworthiness measurement is critical in the whole ECRS system [6], [47]. In general, there are two forms of trust measurement: direct and indirect trust measurement. The value of a direct trust can be calculated using the probability of a direct transaction's success and failure to perform the assigned task. Suppose $N_{Success}$ is the number of tasks successfully completed the assignment schedule between $ed_i$ and $Task_j$, $N_{Failure}$ is the failure number of completing the assigned task due to the provision of poor service or fraudulent actions by the other party, and $\mu$ denotes a penalty factor and $\mu \geqslant 1$, which is mainly used to encourage task nodes to provide real and effective information rather than false information. Then the direct trust for $ed_i$ and $Task_j$ (represented by $Trust_{direct}(ed_i, Task_j)$)is shown in Equation 15.

$$Trust_{direct}(ed_i, Task_j) = \begin{cases} \frac{N_{Success}}{N_{Success} + \mu N_{Failure}}, & N_{Success} > N_{Failure} \\ 0, & others \end{cases} \tag{15}$$

Similarly, indirect trust between edge server node and task node is measured through the recommendation of a third-party edge server nodes coming from the friends and strangers. Suppose $N_{ed_j}$ is the direct interaction number between task node $Task_k$ and friend node $ed_j$, and $\lambda$ denotes the weight of recommendation information from friend nodes $ed_j$. Then Equation 16 represents the indirect trust measurement from the friends with a direct interaction with task nodes.

$$\text{Re}\, com_{friend}(ed_i, Task_j) = \sum_{k=1}^{N_{ed_j}} \lambda \cdot Trust_{direct}(ed_j, Task_k). \tag{16}$$

Suppose that $NS_{ed_j}$ is the indirect interaction number between task node $Task_k$ and stranger node $ed_j$ for $ed_i$; $Trust_{direct}(ed_j, Task_k)$ describes the recommendation trust value between task node $Task_k$ and stranger nodes $ed_j$ for $ed_i$. Then Equation 17 represents the indirect trust measurement from the strangers with an indirect interaction with task nodes.

$$\text{Re}\, com_{stranger}(ed_i, Task_j) = \frac{\sum_{k=1}^{NS_{ed_j}} Trust_{direct}(ed_j, Task_k)}{NS_{ed_j}}. \tag{17}$$

The task node $Task_k$ and stranger nodes $ed_j$ for $ed_i$ have direct interaction history. Suppose that $\alpha$ and $\beta$ are the weight adjustment factor of direct and indirect trust measurement, respectively, and $\alpha$+$\beta$ = 1, then the indirect trust measure considers the recommended trust values of friends and strangers, and is shown in Equation 18.

$$Trust_{recom}(ed_i, Task_j) = \alpha \cdot \text{Re}\, com_{friend}(ed_i, Task_j) + \beta \cdot \text{Re}\, com_{stranger}(ed_i, Task_j). \tag{18}$$

In addition, to meet the user's satisfaction with task computing service, the reliability of edge cloud server and user nodes shown in Equation 19 should be considered. Suppose $Task_m$

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



represents the task node in the same trust domain that has directly interacted with node. $NT_{ed_i}$ is the interaction number of task node $Task_j$ and node $ed_i$, then

$$\mathrm{Re}\,liab(ed_i, Task_j) = \frac{Trust_{direct}(ed_i, Task_j)}{\sum_{m=1}^{NT_{ed_i}} Trust_{direct}(ed_i, Task_m)}. \quad (19)$$

In calculating comprehensive trust measurement, the weighted average method considers the direct and indirect trust measurement. Suppose that $\delta$, $\theta$ and $\gamma$ are three weight factors to adjust the proportion of factors influencing trust measurement, respectively. Then the comprehensive trust measurement with the task node and the node are expressed in Equation 20.

$$Trust(ed_i, Task_j) = \delta \cdot (\theta \cdot Trust_{direct}(ed_i, Task_m) + (1-\theta) \cdot Trust_{recom}(ed_i, Task_j)) + \gamma \cdot \mathrm{Re}\,liab(ed_i, Task_j). \quad (20)$$

The trade-off between trust value and service reliability requirement needs to be quantified, where the comprehensive satisfaction of the user with the quality of service should screen out the resource nodes to meet the comprehensive needs of the user. Equation 21 represents the trust factor.

$$f_5(x) = \max(Trust(ed_i, Task_j)). \quad (21)$$

### C. Model Constraints

The influence of several environmental or equipment factors make it quite challenging to perfect the ECRS system [28], and constraining the many-objective ECRS model may help in bridging the gap betwen reality and the abstract model. For example, the completion time of task can not be greater than the deadline required by the task in the ECRS process, described as $T_{ed_i} \leqslant Task_{ed_i}^{deadline}$. It is necessary to pay a certain amount of money in the design of an edge cloud resource scheduling system, whether to repair a damaged server or to perform routine maintenance on the system, in order to ensure the system's normal operation and meet the needs of users. Note that such type of input cost, on the other hand, is not limitless, N and the cost of task consumption should be within the service provider's budget in a standard resource scheduling system. Otherwise, the system will crash and the task will not be completed. Therefore, whether it is friend node or stranger node for the trust measurement between task and server, the trust measurement value should be between 0 and 1, i.e., $Task_j^{\cos t} \leqslant Task_j^{budget}$, $0 \leqslant Trust_{direct}(ed_j, Task_k) \leqslant 1$ and $0 \leqslant \mathrm{Re}\,com_{stranger}(ed_i, Task_j) \leqslant 1$.

## III. Designed Algorithm

According to the no-free lunch (NFL) theorem [48], it is impossible to design an algorithm that is more effective than all other algorithms when solving optimization problems with different characteristics [36], [49]. And it is very time-consuming to select an effective algorithm from thousands of candidate algorithms in the face of practical problems [50], [51]. On the basis of the existing optimization algorithms, we explore that if we can make use of the proposed operators, consider the advantages of each operator in different time segments of the algorithm, and organically integrate a series of improvement strategies of these operators to make them coordinate and work together, the performance of the algorithm can be greatly improved [50], [51]. In the literature, several ensemble techniques, such as parameter ensemble [52], [53] and evolutionary variable ensemble [54], [55], are proposed from numerous perspectives and are based on the many aspects of the ECRS problem. These studies mostly focus on addressing Single-objective Optimization Problems (SOPs) [49], [56] or on Multi-objective Optimization Problems (MOPs) [57], [58], and perform poorly when it comes to solving many-objective Optimization Problems (MaOPs)(with the number of objective exceeds three) - the problem at the core of ECRS systems [10]. To address this issue, a technique based on a probabilistic selection ensemble has been developed in this study. Through mutual coordination and joint action, each evolutionary strategy gives various advantages and achieves a "strong combination" and superior performance under the ensemble mechanism [49]. The detailed framework including selection of evolutionary operators, ensemble mechanism, selection mechanism, algorithmic framework, and complexity analysis of our approach is described below.

### A. Selection of evolutionary operators

The classical optimization method converges slowly and is not easy to obtain the optimal solution or local optimal solution. In addition, its robustness is poor, and the design of optimization rules is often only for a specific objective environment. Thus, an evolutionary strategy has been proved to have the characteristics of fast convergence and strong robustness, and can avoid the problems of classical optimization methods.

*1) Common evolutionary strategies:* Genetic algorithm (GA) [59], [60], differential evolutionary (DE) [52], [61], [62], particle swarm optimization (PSO) [38], [63], cuckoo search (CS) [11], [64], and pigeon-inspired optimization (PIO) [65], [66] are some of the popular evolutionary algorithms used in optimization problems. However, the search modes and performances of these works are limited [49], [50]. As a result, the global and local search patterns of evolutionary operators in distinct evolutionary algorithms differ from one another in terms of search patterns. Hence, each evolutionary algorithm offers a unique set of benefits while dealing with optimization problems of various types. We now discuss in detail the eight common evolutionary strategies (shown in Table 2) used in optimization problems.

**Strategy 1:** Random search (DE/rand/1)
The essential concept behind DE is that two individuals are chosen at random from the population, and offspring are produced by recombination between selected individuals [52] [61]. The optimal individual to be retained in the population is determined by comparing the fitness values of the two individuals and guiding the population to evolve in a better direction. Let $F$ be the contraction factor used to control

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



TABLE II
DESCRIPTION OF EDGE NODE AND TASK RELATED ATTRIBUTES

| Strategy | Evolutionary Operator | Evolutionary Strategy Name |
|---|---|---|
| 1 | DE | Random search (DE/rand/1) |
| 2 | DE | Random search (many-objective DE, MaOE/best/2) |
| 3 | GA | Single point crossover |
| 4 | GA | Many-objective Single point crossover |
| 5 | GA | Two-point crossover |
| 6 | PSO | Flying in the direction of global optimal individual |
| 7 | PSO | Many-objective flying in the direction of relatively better individual |
| 8 | CS | Flying in the direction of global optimal individual |

the influence of the difference vector, and $x_{id}(t)$, $x_{jd}(t)$ and $x_{kd}(t)$ be the individuals randomly selected from the current population. Then, the common DE random strategy (DE / rand / 1, where 1 is the number of difference vectors) is shown in Equation 22.

$$x_{id}(t) = x_{id}(t) + F \cdot (x_{jd}(t) - x_{kd}(t)). \tag{22}$$

**Strategy 2:** Random search (Many-objective DE, MaDE/best/2)
The relationship between solutions is non-dominated in solving MaOPs for the standard DE / rand / 1 strategy described earlier. Therefore, the optimal individual can not be obtained by comparing fitness values only leading to the inability of better evolution of the existing population beyond a certain point [62], [63]. For an effective solution, the evolutionary operator DE/rand/1 is improved to MaDE/best/2 (represented by $x_{id}(t)$ and shown in Equation 23).

$$x_{id}(t) = \begin{cases} x_{id}(t) + F \cdot (x_{jd}(t) - x_{id}(t)) + F \cdot (x_{kd}(t) - x_{id}(t)) & if\ rand \leqslant CR \\ x_{jd}(t) \ \ or \ \ x_{kd}(t) & otherwise \end{cases} \tag{23}$$

**Strategy 3:** GA with single-point crossover
GA is a simple optimization technique that works by imitating the natural evolutionary search process to deal with a practical problem. Selection, crossover, and mutation are the three most important GA operators. Simulated binary crossover (SBX) [38], [67] is a type of crossover, which mimics the evolutionary approach of the population using a binary string and causes the offspring to maintain the relevant pattern information of the parent chromosome.

Let $\eta$ be a custom parameter for defining distribution factor, whose size is positively correlated with the probability of the offspring approaching the parent individual. Then the parameter $\mu$ is defined using $\eta$ as shown in Equation 24.

$$\mu = \begin{cases} (2 \cdot r)^{\frac{1}{\eta+1}} & if\ r \leqslant 0.5, \\ [1/(2-r)]^{\frac{1}{\eta+1}} & otherwise\ . \end{cases} \tag{24}$$

Let $x_{id}$ and $x_{jd}$ are two parents randomly selected from the evolutionary population for the single point crossover evolutionary mode of SBX operator. Then the generation of two offspring $C_{id}$ and $C_{jd}$ is shown in Equation 25.

$$\begin{cases} C_{id} = 0.5 \cdot [(1+\mu)x_{id} + (1-\mu)x_{jd}], \\ C_{jd} = 0.5 \cdot [(1-\mu)x_{id} + (1+\mu)x_{jd}], \end{cases} \tag{25}$$

**Strategy 4:** GA with many-objective single point crossover SBX operator is known to be an effective method in solving MaOPs [38], [63]. Let $w_d$ and $v_d$ represent the maximum and minimum values of $X_{id}$ and $X_{jd}$, respectively ($d = (1, 2, \cdots D)$, which is the $d^{th}$ component), and $r_k$ be a random number within [0,1]. Then $\mu_k (k = 1, 2)$ parameter is defined in Equation 26.

$$\mu_k = \begin{cases} [r_k.a_k]^{\frac{1}{\eta+1}} & if\ r_k \leqslant 1/a_k, \\ [1/(2-r_k.a_k)]^{\frac{1}{\eta+1}} & otherwise\ , \end{cases} \tag{26}$$

Let $w_d \neq v_d$, $l_d$ and $u_d$ be the two boundary values of the d decision variable respectively, and $\eta$ be the crossover distribution index, then $a_k (k = 1, 2)$ is calculated in Equation 27.

$$a_k = \begin{cases} 2 - [1 + 2(v_d - l_d)/(w_d - v_d)]^{-(\eta+1)} & k = 1, \\ 2 - [1 + 2(u_d - w_d)/(w_d - v_d)]^{-(\eta+1)} & k = 2. \end{cases} \tag{27}$$

Let $X_{id}$ and $X_{jd}$ be the two individuals randomly selected from the evolutionary population, and let $C_{id}$ and $C_{jd}$ be the two offsprings. Then the offspring are generated from the parents as shown in Equation 28.

$$\begin{aligned} C_{id} &= 0.5 \cdot [(w_d + v_d) - \mu_1(w_d - v_d)], \\ C_{jd} &= 0.5 \cdot [(w_d + v_d) - \mu_2(w_d - v_d)]. \end{aligned} \tag{28}$$

By employing the polynomial mutation (PM) approach [38], [63] to determine the objective value of new offspring, a random number $k$ is created to select which offspring solution ($Cid$ or $Cjd$) will generate new solutions for further replacement. The relatively superior individuals of each generation can be retained by replacing the least fit individuals in the current population. Given the importance of SBX in solving MaOPs, the parent selection should be tweaked: $X_{id}$ is randomly selected from the current evolutionary population while $X_{jd}$ is randomly selected from the top 10% individuals of the external archive. The selection from the external archive can be used to improve the diversity and convergence of the population towards a solution.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



**Strategy 5:** GA with two-point crossover strategy
GA can also be performed while performing crossover at multiple points [59], [60]. This strategy is shown in Equation 29.

$$x_{id}(t+1) = \varepsilon \cdot x_{id}(t) + (1-\varepsilon) \cdot x_{jd}(t). \tag{29}$$

**Strategy 6:** PSO with flying in the direction of optimal global individual
PSO is one of the most widely used metaheuristic algorithms for problem solving. The most important feature of PSO is the possession of a velocity vector for each particle and a position vector reflecting the decision variables. The velocity vector is used to update the position, and the velocity is updated based on the individual's optimal position as well as the optimal global position [38], [63].

Let $w$ be the inertia value, $c1$ and $c2$ be two learning factors, $r1$ and $r2$ be two random numbers with uniform distribution in [0,1], and $pbest_{id}$ and $gbest_d$ be the positions of local optimal particles and global optimal particles respectively. Each particle moves towards the direction of global optimal solution, as explained by the evolution equation of velocity and position in PSO [38] and shown in Equation 30.

$$\begin{cases} v_{id}(t+1) = wv_{id}(t) + c_1 r_1(pbest_{id} - x_{id}(t)) + \\ \qquad\qquad c_2 r_2(gbest_d - x_{id}(t)), \\ x_{id}(t+1) = x_{id}(t) + v_{id}(t+1). \end{cases} \tag{30}$$

By substituting the above formula, it can be written as follows.

$v_{id}(t) = x_{id}(t) - x_{id}(t-1)$
$x_{id}(t+1) = x_{id}(t) + w(x_{id}(t) - x_{id}(t-1)) + c_1 r_1(pbest_{id} - x_{id}(t)) + c_2 r_2(gbest_d - x_{id}(t))$

Thus,

$x_{id}(t) = x_{id}(t-1) + w(x_{id}(t-1) - x_{id}(t-2)) + c_1 r_1(pbest_{id} - x_{id}(t-1)) + c_2 r_2(gbest_d - x_{id}(t-1))$

**Strategy 7:** PSO with many-objective flying in the direction of relatively better individual
Many variants of PSO have been proposed to solve MaOPs since the proposal of PSO, including NMPSO [38], MOPSO [68], dMOPSO [69], MPSO-D [70], and MOPSO-CD [71]. However, due to the peculiarities of the PSO operator, the new generation tends to gravitate toward the optimal solution, which diminishes population diversity to a certain extent [63]. As a result, we alter the particle velocity update search strategy of the original PSO operator to make it more suitable for solving MaOPs, in order to better guide the evolution of the population.

Let $t$ be the number of iterations, $w$ be the inertia value, $c1$, $c2$ and $c3$ be three learning factors, $r1$, $r2$ and $r3$ be three random numbers with uniform distribution in [0,1], and $pbest_{id}$ and $gbest_d$ be the positions of local optimal particles and global optimal particles respectively. Then the specific particle velocity and position update strategies are defined in Equation 31.

$$\begin{cases} v_{id}(t+1) = wv_{id}(t) + c_1 r_1(pbest_{id} - x_{id}(t)) + c_2 r_2(gbest_d \\ \qquad\qquad - x_{id}(t)) + c_3 r_3(gbest_d - pbest_{id}) \\ x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \end{cases} \tag{31}$$

The local and global optimal solutions are randomly selected from the current evolutionary population and the top 10% of the external solution set and ranked in descending order by using the evaluation mechanism. Through this strategy, the population can be guaranteed to evolve in a better direction, and the evolution can quickly converge towards the global optima in solving MaOPs. Hence Equation 31 can be re-written as follows.
$v_{id}(t) = x_{id}(t) - x_{id}(t-1)$
$x_{id}(t+1) = x_{id}(t) + w(x_{id}(t) - x_{id}(t-1)) + c_1 r_1(pbest_{id} - x_{id}(t)) + c_2 r_2(gbest_d - x_{id}(t)) + c_3 r_3(gbest_d - pbest_{id})$
Thus,

$x_{id}(t) = x_{id}(t-1) + w(x_{id}(t-1) - x_{id}(t-2)) + c_1 r_1(pbest_{id} - x_{id}(t-1)) + c_2 r_2(gbest_d - x_{id}(t-1)) + c_3 r_3(gbest_d - pbest_{id})$

**Strategy 8:** Cuckoo Search with convergence towards optimal global individual
Cuckoo Search is a recently proposed optimization algorithm, which is based on simulation of cuckoo's behavior of searching for a nest, laying eggs and brooding in nature. At the same time, it takes into account the cuckoo's distinctive Levy flight mode, and the individual flies towards the optimal global individual [11], [64].

Let $\lambda$=1.5, then $\phi$ can be calculated as

$$\phi = \left( \frac{\Gamma(1+\lambda) \cdot \sin(\frac{\lambda\pi}{2})}{\Gamma(\frac{1+\lambda}{2}) \cdot \lambda \cdot 2^{\frac{\lambda-1}{2}}} \right)^{\frac{1}{\lambda}}. \tag{32}$$

Let $gbest_d$ be the global optimal individual of the current population, $x_{jd}(t)$ be the individual randomly selected from the current population, then the unique flight mode of cuckoo $levy(\lambda)$ is defined as

$$levy(\lambda) \sim \phi \cdot u/|v|^{1/\lambda}, \tag{33}$$

where $u$ and $v$ obey the standard Gaussian distribution.

Let $r$ be a random number of [0,1], and $\alpha$ be the random step size (usually set to 1). The position updating equation in CS is shown in Equation 34.

$$x_{id}(t+1) = x_{id}(t) + \alpha \cdot 0.01 \cdot levy(\lambda) \cdot r \cdot (gbest_d - x_{jd}(t)). \tag{34}$$

*2) Convergence analysis:* Convergence performance is an important factor in selecting the appropriate evolutionary strategy as the evolutionary operator of any ensemble algorithm [49]. Therefore, the global convergence performance of eight common evolutionary strategies was analysed from the perspective of the support set, where the specific analysis is explained here.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



**Strategy 1:** The support set $x_{id}(t)$ of $M_{id}^t$ can be an interval. Due to the arbitrariness of the individual population, there are some differences, i.e., $-\max\left\{\left|x_{jd}(t)-x_{kd}(t)\right|\right\} \le x_{jd}(t)-x_{kd}(t) \le \max\left\{\left|x_{jd}(t)-x_{kd}(t)\right|\right\}$.

Therefore, the value range of $x_{id}(t)$ is

$$\begin{aligned}\Psi &= \max\left\{\left|x_{jd}(t)-x_{kd}(t)\right|\right\}\cdot F\\ x_{id}(t) &\in [x_{id}(t)-\Psi, x_{id}(t)+\Psi].\end{aligned} \tag{35}$$

Assumed that $x_{d,\max}^t$ and $x_{d,\min}^t$ represent the largest and smallest d-dimensional ($d = 1, 2, \ldots, D$) components of all individuals in the $t^{th}$ generation population, respectively. The $\Delta_d^t = x_{d,\max}^t - x_{d,\min}^t$, and $x_{id}(t) \in [x_{id}(t) - \Delta_d^t \cdot F, x_{id}(t) + \Delta_d^t \cdot F]$. The length of the interval can be calculated as $2\Delta_d^t \cdot F$. Therefore, the support set size of $x_i(t)$ is $M_{id}^t = \prod_{d=1}^{D} 2\Delta_d^t \cdot F = 2^D F^D \prod_{d=1}^{D} \Delta_d^t$.

**Strategy 2:** Similar to strategy 1, the support set $M_{id}^t$ of $x_{id}(t)$ is an interval, and the value interval of $x_{id}(t)$ is $x_{id}(t) \in \left[x_{id}(t) - 2\cdot F\cdot(x_{d,\min}^t - x_{id}(t)), x_{id}(t) - 2\cdot F\cdot(x_{d,\max}^t - x_{id}(t))\right]$. The interval length is calculated as $2\Delta_d^t \cdot F$.

As a conclusion, the support set size of $x_i(t)$ is $M_{id}^t = \prod_{d=1}^{D} 2\Delta_d^t \cdot F = 2^D F^D \prod_{d=1}^{D} \Delta_d^t$.

**Strategy 3:** It can be found from the Eq. (16) that $X_{id}$ and $X_{jd}$ have the same value space, and the value space of $x_{id}(t)$ can be calculated as $x_{id}(t) \in [0.5\cdot[(1+\mu)x_{d,\min}^t + (1-\mu)x_{d,\min}^t], 0.5\cdot[(1+\mu)x_{d,\max}^t + (1-\mu)x_{d,\max}^t]] \Rightarrow x_{id}(t) = [x_{d,\min}^t, x_{d,\max}^t]$. The support interval length of $a$ is expressed as $x_{d,\max}^t - x_{d,\min}^t = \Delta_d^t$. Hence, the support set size of $x_i(t)$ is $M_{id}^t = \prod_{d=1}^{D} \Delta_d^t$.

**Strategy 4:** Encouraged by strategy 3, the length of support set interval of $x_i(t)$ is still $x_{d,\max}^t - x_{d,\min}^t = \Delta_d^t$. The support set size of $x_i(t)$ is $M_{id}^t = \prod_{d=1}^{D} \Delta_d^t$.

**Strategy 5:** The $x_{id}(t) = \varepsilon \cdot x_{id}(t-1) + (1-\varepsilon)\cdot x_{jd}(t-1)$, and $x_{id}(t) \in [\varepsilon x_{id}(t-1) + (1-\varepsilon)x_{d,\min}^t, \varepsilon x_{id}(t-1) + (1-\varepsilon)x_{d,\max}^t]$. Therefore, the length of the interval is shown as $(1-\varepsilon)\cdot(x_{d,\max}^t - x_{d,\min}^t)$ is $M_{id}^t = \prod_{d=1}^{D}(1-\varepsilon)\cdot(x_{d,\max}^t - x_{d,\min}^t) = (1-\varepsilon)^D \cdot \prod_{d=1}^{D}\Delta_d^t$.

**Strategy 6:** If $\varphi_1 = c_1 r_1$, $\varphi_2 = c_2 r_2$ are set in strategy 6, then because $0 \le \varphi_1 \le c_1$, $0 \le \varphi_2 \le c_2$ vertices $\varphi_1 = \varphi_2 = 0$ and $\varphi_1 = c_1$, $\varphi_2 = c_2$ will form a hyperrectangular body.

When $\varphi_1 = \varphi_2 = 0$, the support set of $x_i(t)$ is $M_{id}^t = x_{id}(t-1) + w(x_{id}(t-1) - x_{id}(t-2))$.

When $\varphi_1 = c_1$, $\varphi_2 = c_2$, the support set of $x_{id}(t)$ is $M_{id}^t = x_{id}(t-1) + w(x_{id}(t-1) - x_{id}(t-2)) + c_1(pbest_{id} - x_{id}(t-1)) + c_2(gbest_d - x_{id}(t-1))$.

Let $A = x_{ik}(t-1) + w(x_{ik}(t-1) - x_{ik}(t-2))$, then $x_{id}(t) \in [A + c_1(pbest_{id} - x_{d,\max}^t) + c_2(gbest_d - x_{d,\max}^t), A + c_1(pbest_{id} - x_{d,\min}^t) + c_2(gbest_d - x_{d,\min}^t)]$.

Then interval length of $x_{id}(t)$ support set is $[A + c_1(pbest_{id} - x_{d,\min}^t) + c_2(gbest_d - x_{d,\min}^t)] - [A + c_1(pbest_{id} - x_{d,\max}^t) + c_2(gbest_d - x_{d,\max}^t)]$
$= (c_1 + c_2)(x_{d,\max}^t - x_{d,\min}^t)$
$= (c_1 + c_2)\Delta_d^t$

Therefore, the support set of point $x_{id}(t)$ is $M_{id}^t = \prod_{d=1}^{D}\Delta_d^t(c_1+c_2) = (c_1+c_2)^D \prod_{d=1}^{D}\Delta_d^t$.

**Strategy 7:** If $\varphi_1 = c_1 r_1$, $\varphi_2 = c_2 r_2$, $\varphi_3 = c_3 r_3$ are set in strategy 7, then because $0 \le \varphi_1 \le c_1$, $0 \le \varphi_2 \le c_2$, $0 \le \varphi_3 \le c_3$, vertices $\varphi_1 = \varphi_2 = \varphi_3 = 0$ and $\varphi_1 = c_1$, $\varphi_2 = c_2$, $\varphi_3 = c_3$ form a hyperrectangular body.

When $\varphi_1 = \varphi_2 = \varphi_3 = 0$, the support set of $x_i(t)$ can be expressed as $M_{id}^t = x_{id}(t-1) + w(x_{id}(t-1) - x_{id}(t-2))$.

When $\varphi_1 = c_1$, $\varphi_2 = c_2$, $\varphi_3 = c_3$, the support set of can be expressed as $M_{id}^t = x_{id}(t-1) + w(x_{id}(t-1) - x_{id}(t-2)) + c_1(pbest_{id} - x_{id}(t-1)) + c_2(gbest_d - x_{id}(t-1)) + c_3(gbest_d - pbest_{id})$.

Let $A = x_{ik}(t-1) + w(x_{ik}(t-1) - x_{ik}(t-2))$, then $x_{id}(t) \in [A + c_1(pbest_{id} - x_{d,\max}^t) + c_2(gbest_d - x_{d,\max}^t) + c_3(gbest_d - pbest_{id}), A + c_1(pbest_{id} - x_{d,\min}^t) + c_2(gbest_d - x_{d,\min}^t) + c_3(gbest_d - pbest_{id})]$

Therefore, the interval length of $x_{id}(t)$ support set is $[A + c_1(pbest_{id} - x_{d,\min}^t) + c_2(gbest_d - x_{d,\min}^t) + c_3(gbest_d - pbest_{id})] - [A + c_1(pbest_{id} - x_{d,\max}^t) + c_2(gbest_d - x_{d,\max}^t) + c_3(gbest_d - pbest_{id})]$
$= (c_1 + c_2)(x_{d,\max}^t - x_{d,\min}^t)$
$= (c_1 + c_2)\Delta_d^t$

And the support set of $x_{id}(t)$ is $M_{id}^t = \prod_{d=1}^{D}\Delta_d^t(c_1+c_2) = (c_1+c_2)^D \prod_{d=1}^{D}\Delta_d^t$.

**Strategy 8:** If $B = \alpha\cdot 0.01\cdot levy(\lambda)\cdot r, (B \in [B_{\min}, B_{\max}])$ in strategy 8, then $x_{id}(t) = x_{id}(t-1) + \alpha\cdot 0.01\cdot levy(\lambda)\cdot r\cdot(gbest_d - x_{jd}(t-1)), \Rightarrow x_{id}(t) = x_{id}(t-1) + B\cdot(gbest_d - x_{jd}(t-1))$. It can be seen that the support set of $x_i(t)$ is an interval, and because of the arbitrariness of cuckoo $x_{jd}(t)$, then $x_{id}(t) \in [x_{id}(t-1) + (gbest_d - x_{d,\max}^t)\cdot B_{\max}, x_{id}(t-1) + (gbest_d - x_{d,\min}^t)\cdot B_{\max}]$. Therefore, the interval length of support set of $x_i(t)$ is shown as $[x_{id}(t-1) + (gbest_d - x_{d,\min}^t)\cdot B_{\max}] - [x_{id}(t-1) + (gbest_d - x_{d,\max}^t)\cdot B_{\max}] = B_{\max}\Delta_d^t$. And the support set of $x_i(t)$ is $M_{id}^t = \prod_{d=1}^{D} B_{\max}\Delta_d^t = B_{\max}^D \prod_{d=1}^{D}\Delta_d^t$.

To summarize, the search performance of these eight techniques is compared based on the size of their support sets. Both strategy 1 and strategy 2 have the same support set sizes. Furthermore, strategies 3 and 4 are the same size, as are strategies 6 and 7. Specifically, when the random individuals are not in the range of $x_{id}(t)$, strategy 1 and 2 are unable to attain the global extremum. As a result, they are unable to guarantee global convergence. The same is true for strategies

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



5, 6, 7, and 8, none of which can guarantee convergence with probability 1. In addition, $c_1 + c_2 \geq 1$, $0 \leq B_{\max} < 1$ and $0 < 1-\varepsilon < 1$. $c_1+c_2 \geq 1-\varepsilon$, $c_1+c_2 \geq B_{\max}$ and $c_1+c_2 \geq 2 \cdot F$ are obtained. Therefore, the support sets of strategies 6 and 7 are greater than those of strategies 5 and 8, and they are also larger than those of strategies 1 and 2. The support set of strategy 5 is usually $B_{\max} \leq 1-\varepsilon$ and $2 \cdot F \geq 1-\varepsilon$, which is larger than that of strategy 8. Note that the support sets of strategies 1 and 2 are larger than that of strategy 5 and the support sets of strategies 3 and 4 are smaller than those of the other six strategies because their support sets do not involve other variable parameters.

### B. Ensemble mechanism

The design of the ensemble mechanism helps in dynamically selecting the operator to search for the optimal solution according to the characteristics of the problem. For solving MaOPs, we propose the integration strategy of probabilistic selection to build a strategy pool and store the solutions obtained by executing different evolutionary strategies in an abstract "pool" to form an alternative solution policy pool, where the pool can integrate multiple excellent operators. Under their mutual coordination and joint action, the population can obtain a wider range of alternative solutions in the evolution process. As one of the most commonly used strategies in algorithm mechanisms, the design and application of probability integration strategy have significant impact on performance of the algorithm. This paper introduces two commonly used strategies of equal probability ensemble mechanism (EPEM) and dynamic probability ensemble mechanism (DPEM) as shown in Figure 2.

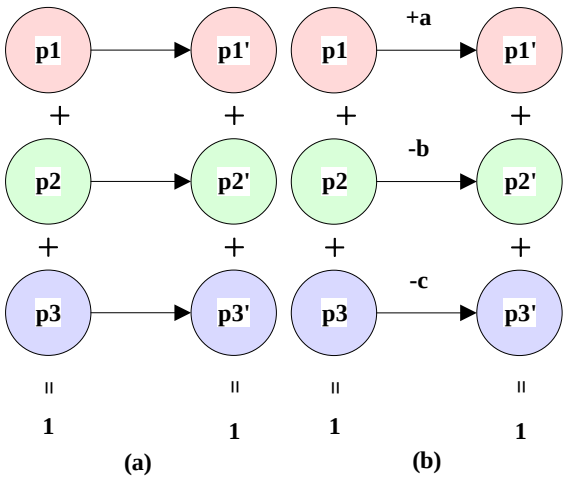



Fig. 2. **Strategies of Probability Ensemble Mechanism** - The figure displays the two strategies for probability integration. Panel a) displays the diagram of equal probabiolity ensemble mechanism (EPEM), where the probability of selecting an operator does not change with time. Panel b) displays the diagram of dynamic probability ensemble mechanism, where the probability of all operators is updated with every iteration under defined constraints for probability.

*1) Equal probability ensemble mechanism (EPEM):* EPEM strategies are chosen so that they all have the same chance of succeeding. The selection probability of operators does not vary during the process of population iteration EPEM is suitable for tackling real problems requiring a stronger fairness mechanism, in general. It will, however, be inapplicable when dealing with problems that are easily influenced by unknown factors. Figure 2(a) depicts the selected probability change process of three operators in the population iteration process to help you understand the EPEM. The selection probabilities of the three operators in the initial population are $p1$, $p2$, and $p3$, respectively, where $p1 = p2 = p3$, and $p1+p2+p3 = 1$. In the iterative procedure, the selection probability of each strategy is recorded as $p1'$, $p2'$, and $p3'$. The EPEM stipulates that the probability of selecting an operator does not change with time, i.e., $p1 = p1'$, $p2 = p2'$, $p3 = p3'$, and $p1' + p2' + p3' = 1$.

*2) Dynamic probability ensemble mechanism (DPEM):* The parameter setup is the most important aspect of DPEM. Designers frequently desire to adjust the parameter values of variables to affect the algorithm's performance in order to indirectly change the selection probability of different strategies. The DPEM is frequently used to select a strategy that varies with the population's evolutionary stage. When a strategy performs well across multiple generations of assessment metrics, the strategy's selection probability should be suitably enhanced, while the selection probability of other strategies should be kept the same or reduced. Every several generations, the population should be assessed and evaluated after evolution. The likelihood of three strategies being chosen should be listed in sequence, with the likelihood of the first option being chosen being enhanced. For the latter two operators, the likelihood of getting chosen can be separated into two categories.

**Case 1:** The likelihood of selecting the second strategy remains unchanged, while the probability of selecting the third strategy lowers, and the decreased probability value should be equal to the increase in probability value of the first operator.

**Case 2:** The second and third operators' probability values are decreased, and the sum of the reduced probability values of the two strategies should equal the increase in probability value of first strategy.

Before proceeding with the aforementioned process, additional metrics are employed to evaluate the amount of change in the probability value of the second and third-ranked strategies. Case 2 is used in this study since it is more realistic in terms of practical application. Figure 2(b) depicts the population being measured and evaluated after multiple generations of evolution. The chance of each of the three tactics being chosen is $p1 > p2 > p3$. Then, by satisfying $a = b + c$, the probability of the strategy being picked in the next initial iteration should be altered to $p1' = p1 + a$ (where $a$ denotes the enhanced probability value), and the probability values of $p2$ and $p3$ should be decreased to $p2' = p2 - b, p3' = p3 - c$. Furthermore, the probability of a technique being chosen will continue to increase or decrease in actual problems. As a result, the upper and lower bounds of

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



the probability value should be set to prevent the probability of the strategy being chosen from being larger than 1 or negative.

### C. Selection mechanism

When determining a selection mechanism for ECRS problem, two factors should be taken into account: convergence and distribution. Convergence assumes that solving MaOP is a continuous process of reaching the Pareto optimal solution set. Distribution requires non-dominated solutions to be distributed equally and widely in the objective space. As a result, the designed selection mechanism should ensure a good CaD for the generated solution as the measurement criteria of mechanism design. In this study, the optimal solution obtained in the process of population evolution is stored using a balanced fitness estimation (BFE) method based on an external solution set [38]. In addition, the BFE method is proved to be an effective method to overcome the limitation of the Pareto sorting and decomposition method, where the BFE method considers the factors of equilibrium CaD in the objective space.

Suppose that population $P = \{p_1, p_2, ..., p_N\}$ contains $N$ individuals, $Discon(p_i, P)$ be the convergence distance, and $Disdiv(p_i, P)$ be the diversity distance of $p_i$. Then $DIV(p_i)$ is the nearest neighbor individual distance value, which can be calculated as shown in Equation 36.

$$DIV(p_i) = \min_{p_j \in P, j \neq i} \sqrt{\sum_{m=1}^{M} div(f_m(p_i), f_m(p_j))^2}, \tag{36}$$

where $div(*) = \begin{cases} f_n(p_j) - f_n(p_i), if\ f_n(p_j) > f_n(p_i) \\ 0 \qquad otherwise \end{cases}$; $f_n(p_i)$ is the standardized fitness value of individual $p_i$ at the mth objective.

Let $DIV_{\min}$ and $DIV_{\max}$ represent the minimum and maximum of shift-based density estimation (SDE) values calculated by density transfer function, respectively, then the diversity distance of $p_i$ ($Disdiv(p_i, P)$) can be calculated as shown in Equation 37.

$$Disdiv(p_i, P) = 1 - \sqrt{\frac{\sum_{m=1}^{M} f_m(p_i)^2}{M}}. \tag{37}$$

On the other hand, the convergence distance $Discon(p_i, P)$ can be calculated by using the SDE as shown in Equation 38.

$$Discon(p_i, P) = \frac{DIV(p_i) - DIV_{\min}}{DIV_{\max} - DIV_{\min}}. \tag{38}$$

Let $\alpha$ and $\beta$ be the weight factors to adjust the influence of the two distances and to dynamically balance the distance of CaD. Then the BFE method ($fitness(p_i, P)$) is shown in Equation 39.

$$fitness(p_i, P) = w_1 * Discon(p_i, P) + w_2 * Disdiv(p_i, P). \tag{39}$$

Note that the fitness values for different individuals is adaptable with the distance between them, and the specific adjustment principles for adjusting the weight factors $w_1$ and $w_2$ can be found in literature [38]. When computing the BFE value, the maximum and lowest values of the relevant objectives should also be considered for normalization. In high-dimensional space, this type of normalized reduction objective will vibrate strongly. The truncated selection mechanism should be employed in the selection mechanism to avoid the size of the external solution set from surpassing the population size barrier. The BFE selection mechanism can effectively steer the population's search direction to approach the true Pareto-optimal Front (PF) by evaluating the CaD fully. Algorithm 1 lists the main steps of the selection mechanism, where $A$ is the external archive to store the optimal solution generated in each iteration and $A_j$ is the $j$th solution in $A$. $B$ is the offspring population obtained by executing the evolutionary strategy and $B_i$ is the $i$th solution in $B$.

**Algorithm 1:** Selection mechanism (A, B)

```
1  Begin
2    For i=1 to |B|
3      For j=1 to |A|
4        Check domination relationship between A_j and
5          B_i;
6        If nondominated solution in A
7          Keep the solution A_j in A;
8        End if
9        If nondominated solution in B
10         Add the solution B_i into A;
11       End if
12     End for
13     If |A| > |N|
14       Calculate the BFE fitness;
15       Delete the last worst solution;
16     End if
17   End for
18   Output A;
19 End
```

The Pareto dominance relationship between individuals in the external archive $A$ and the newly formed population $B$ is validated in the selection process (line 4), and the corresponding individual is then added to the external archive $A$ if the non-dominated solution is in $B$. If the non-dominated solution is in the external archive $A$, on the other hand, the individual is temporarily retained (lines 6-11). When the solution size of $A$ is larger than the population size, the truncation mechanism is used to sort the individuals in the population using the BFE method value (lines 13-16). Finally, until $A$ reaches the required size, the worst individuals in the population will be removed in an iterative manner.

### D. Algorithmic framework

Algorithm 2 - MaOEA-DPS algorithm describes the framework of the proposed MaOEA-DPS algorithm, where $Q$ and $A*$ represent the offspring generated by different evolutionary strategies, $P$ is the initial population, $A$ is the external archive, and $gen$ and $Maxgen$ are the current and maximum number of iterations.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



**Algorithm 2:** MaOEA-DPS Framework

```
1  Begin
2    Initialize populations P and the related parameters;
3    Initial external archive A;
4    while gen < Maxgen do
5      Employ evolutionary strategies to generate
6        offspring Q;
7      A = Selection mechanism (A, Q)
8      A* = SBX-PM(A);
9      A = Selection mechanism (A, A*);
10   End while
11   Output A;
12 End
```

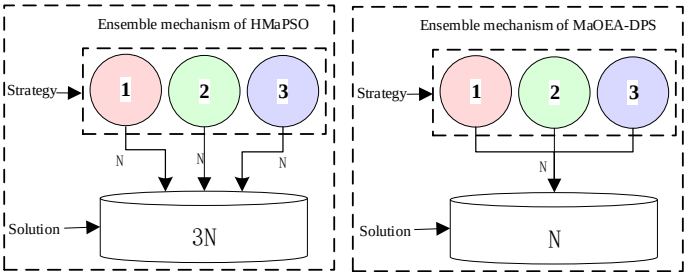



Fig. 3. **Comparison of Computational Complexity for Ensemble Mechanisms** - The figure compares the computational complexity of ensemble method of HMaPSO algorithm and that of the ensemble method of MaOEA-DPS algorithm. The figure shows that the computational complexity of both the methods is same, yet the number of steps deployed in MaOEA-DPS algorithm are 1/3rd of those deployed in HMaPSO algoithm.

The initial population $P$ is randomly generated in MaOEA-DPS algorithm (line 2) while the external archive $A$ is filled with individuals selected in the first layer of the non-dominated sorting method for the population $P$ (line 3). Furthermore, an alternate evolutionary strategy pool is constructed by merging strategies 2, 4 and 7. The offspring solutions in population $Q$ are derived from the offspring solutions obtained from the three evolutionary strategies (lines 5-6). The selection mechanism is activated, and the better offspring solutions from the evolution strategy pool are selected and stored, which will overcome the problem of declining selection pressure in the later stage of evolution and help achieve a "strong combination" and superior performance in balancing CaD (lines 7-9). The iterations continue until the stop condition (line 4) is met.

### E. Complexity analysis

The algorithm terminates when the number of iterations reaches the predefined limit. In contrast to the ensemble mechanism employed in hybrid many-objective particle swarm optimization (HMaPSO) algorithm [63], the probabilistic selection in MaOEA-DPS algorithm is incorporated into the evolutionary strategy's ensemble mechanism. The two ensemble strategies are depicted in Figure 3 to intuitively convey their differences. The three HMaPSO evolutionary techniques should obviously be implemented every generation, however MaOEA-DPS only uses one evolutionary approach every iteration. As a result, the computational complexity has been lowered to an extent from initial $3MN^2$ steps to $MN^2$. The selection mechanism and nondominated sorting steps consume the majority of the remaining computational complexity. The maximum number of iterations for a particular constant is significantly less than the number of nondominated solutions. As a result, the number of nondominated solutions can be used to calculate the computational complexity. To summarise, the time complexity of the algorithm is $O(MN^2)$.

## IV. Simulation Experiment

The efficacy of the design algorithm was verified through extensive experimentation. The major step was to verify the performance of our algorithm on a benchmark function, while the proposed algorithm optimizes the ECRS problem and achieves model solution. It is noted that the proposed algorithm outperforms other sophisticated MaOEAs in terms of performance. The detailed description and experiment conditions are described below.

### A. Experimental setup

*1) Function settings:* MaFs are fifteen benchmark functions, proposed by Chen *et. al* [72], to facilitate research on MaOPs. MaFs have shown a range of performance advantages in testing multimodality, preference, connectivity, and concavity [72], therefore they were adopted to test these features of our proposed approach. Furthermore, values for critical parameters for each algorithm, including MaOEA-R&D [73], VaEA [74], and RVEA [75], are determined by selecting the highest performing values in literature as discussed in Parameter settings. VaEA and RVEA, for example, use a double-layer reference point system, with SBX and PM probability of 1 and 1/D, respectively (where D represents the dimension of decision variables). The maximum number of iterations is considered the algorithm's stop condition, and the precise values for the parameters of MaFs are listed in Table III.

TABLE III
Parameter settings of involved algorithm

| Parameter name | Value |
|---|---|
| Population size | 240 |
| Objective Number | 5/10/15 |
| Number of evaluations | 10000 |
| Number of independent runs | 20 |

*2) Parameter settings:* In this paper, we set the recommended parameters according to the relevant literature that has successfully proved its effectiveness [10], [47], [76]. The total number of tasks for the ECRS problem is set to 400 [10], and the created model contains a lot of parameters. For the two-weight adjustment factors of direct and indirect trust measurement, $\alpha$ and $\beta$ are modified to 0.6 and 0.4, respectively [10]. To change the percentage of factors influencing trust measurement, the three weight factors $\delta$, $\theta$, and

$\gamma$ that quantify the trade-off between trust value and service reliability need are altered to 0.6, 0.7, and 0.4, respectively [47]. Meanwhile, the penalty factor *mu* is set to 2 when computing the procedure for direct trust measurement. For $\lambda$ of recommendation information from friend nodes, the weight factor is set to 1. For the cost of using the MIPs performance, bandwidth and CPU of the $ed_i$, $Mony_{A_{1,2,3}}$ are set to 0.05, 0.1 and 0.1, respectively [76]. The remaining parameters are shown in Table IV.

TABLE IV
PARAMETER SETTINGS OF ECRS PROBLEM

| Related properties | Value |
|---|---|
| Edge Cloud | $ed_i^{bandwidth}$=500<br>$ed_i^{CPU}$=2<br>$ed_i^{mips}$=250 |
| Task | $Task_j^{filesize}$=300<br>$Task_j^{outputsize}$=300<br>$Task_j^{length}$=5000 |
| Center Cloud | Number of edge cloud is 5<br>MIPs = 2500<br>Number of ED is set to 50<br>Host memory ram is 2048 (MB)<br>Host storage is set to 1000000<br>Bandwidth of center cloud is 10000 |

### B. Performance index

For the performance of MaOEA, CaD is an important aspect. Hypervolume (HV) [77], as a comprehensive evaluation indicator, has been widely used to test the CaD of the algorithm. In addition, it does not need to presuppose the true ideal Pareto frontier in advance. For convenience, the HV is selected to measure the performance of the evaluation algorithm. In general, HV is the volume covered by PF in the objective space, which is defined as the hypervolume between the front surface and the reference vector $r = (r_1, r_2, ..., r_m)^T$. Where $i = \{1, \cdots, m\}$ and m is the number of objectives. Let $f_i(x)$ denotes the $i$th fitness value of solution $x$ in the front surface solution set $X$, and $[f_i(x), r_i]$ denotes the hypercube, which can be constructed with the reference vector and the fitness value as two diagonal corners of the hypercube. Therefore, the HV is defined as follows:

$$HV(X, r) = VOL(\bigcup_{x \in X} [f_1(x), r_1] \times \cdots \times [f_m(x), r_m]) \quad (40)$$

where $VOL(*)$ is the Lebesgue metric, which calculates the hypervolume of all the objectives' hypercubes. Monte Carlo estimation mechanism [77] is included to help evaluate the HV metric in dealing with MaOPs [74], [75], where a higher HV metric indicates a better performance.

### C. Result analysis

In this section, the simulation results are presented in two subsections. Firstly, the designed MaOEA-DPS is compared and analyzed with the existing three advance MaOEAs on MaFs functions. And they are applied to address the ECRS problem. The specific results are expressed as follows.

*1) Performance analysis on benchmark function:* The HV values produced on MaF by MaOEA-DPS, MaOEA-R&D, VaEA, and RVEA are shown in Table V. The comparison results are indicated when compared to the MaOEAs involved for various test scenarios. Based on Wilcoxon's rank test and Friedman statistical test [63], the labels "+", "-" and "≈" respectively indicate that the results obtained by different algorithms are higher, lower or equal than the existing algorithms.

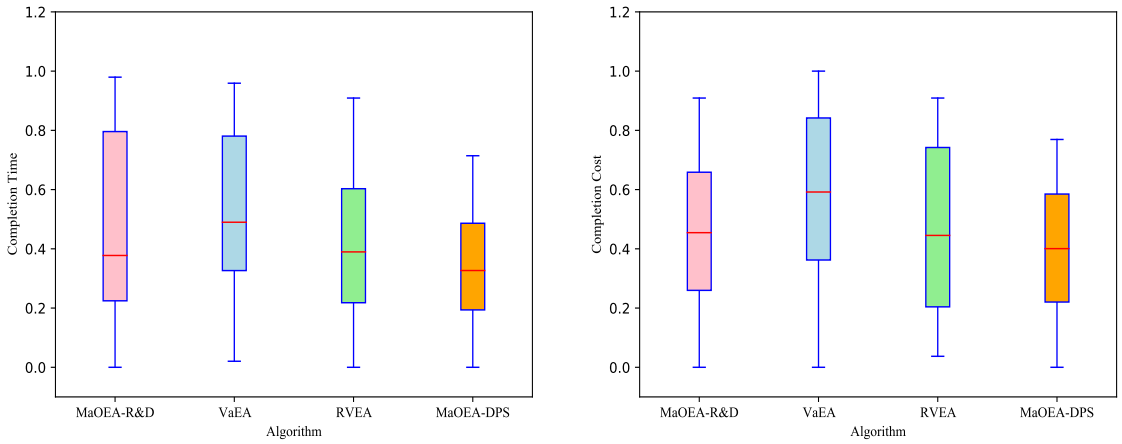
MaOEA-DPS had the most notable results on MaFs functions overall. Specifically, when each algorithm is compared individually, MaOEA-R&D and VaEA are not as good as MaOEA-DPS, with the exception that MaOEA-R&D outperforms MaOEA-DPS on 15 MaF1 objectives. When compared to MaOEA-DPS, the RVEA achieves higher performance on MaF8. The reason for this is that the reference vector selection process selects the better-distributed solution with more accuracy. MaOEA-DPS, on the other hand, outperformed the remainder of RVEA's MaF results, demonstrating that the ensemble selection process based on dynamic probability is capable of finding solutions with improved CaD performance. As a result, MaOEA-DPS outperforms the other algorithms in the MaF test function.

*2) Performance analysis on ECRS problem:* Fig. 4 illustrates the performance comparison box for several algorithms on different objectives to intuitively compare the performance of different algorithms for dealing with the ECRS problem. All four algorithms are very competitive in terms of completion time, completion cost, and load balance degree, based on the median and quartile value of model solutions. Fig. 4 (a), (b) and (d) show that the median values of the four algorithms are significantly different, with the median value of MaOEA-DPS clearly smaller. The ranking of all algorithm-based median values is followed to better discern the performance difference. For example, the ranking of median value is shown for completion time and completion cost: MaOEA-DPS<MaOEA-R&D≈RVEA<VaEA. And the objective of the load balance degree has a different landscape, MaOEA-DPS<VaEA<MaOEA-R&D≈RVEA is followed based on the median value. In terms of solution distribution, MaOEA-DPS in Fig. 4 (a), (b) and (d) is tighter and more concentrated than the other three algorithms. And it is clear to observe that the MaOEA-DPS obtains smaller upper and lower quartile values, which means that MaOEA-DPS has better performance than other algorithms in obtaining the objective solution of completion time, completion cost, and load balance degree.

Fig. 4 (c) and (e) demonstrate the reciprocal of user satisfaction and trust measurement owing to the minimal value of each objective, respectively. Other algorithms show the same behavior as MaOEA-DPS; they reach close to 0 for the solution value, indicating that the answers are identified with reasonable satisfaction and reliable measurement. VaEA, RVEA, and MaOEA-DPS have lower values in the objective of trust measurement, similar to user satisfaction, implying that

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064

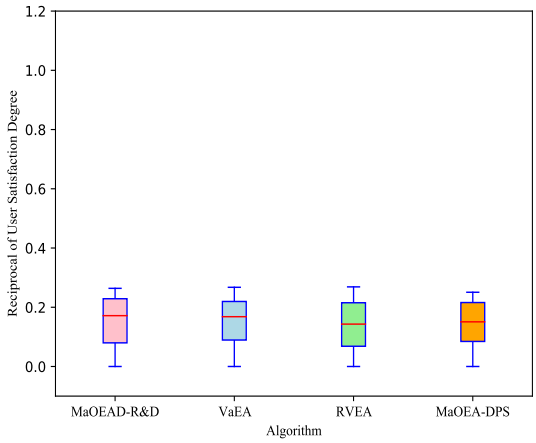




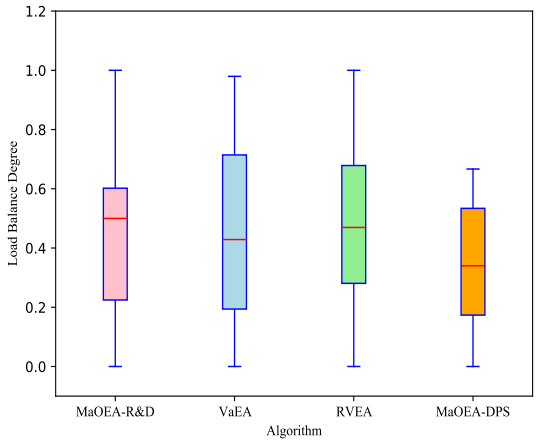

(a) Completion Time



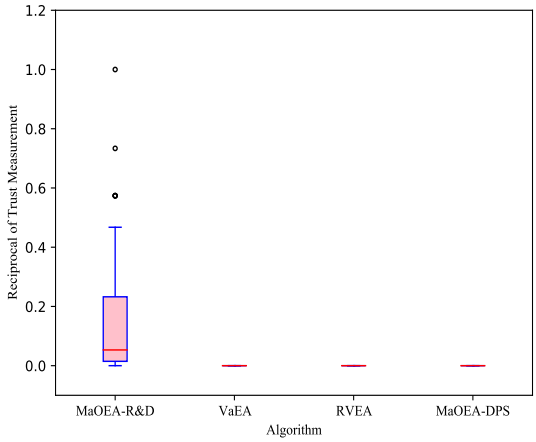

(b) Completion Cost




(c) Reciprocal of User Satisfaction degree




(d) Load Balance degree




(e) Reciprocal of Trust Measurement

Fig. 4. **Performance comparison box for different algorithms on different objectives** - The figure displays the boxplot for comparison of performance of MaOEA-R&D, VaEA, RVEA, and MaOEA-DPS algorithms on completion time in Panel a), completion cost in Panel b), reciprocal of degree of user satisfaction in Panel c), degree of load balance in Panel d), and reciprocal of trust management in Panel e).

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3235064



TABLE V
HV VALUE OF DIFFERENT ALGORITHMS ON THE MAF

| Problem | M | MaOEA-R&D | VaEA | RVEA | MaOEA-DPS |
|---|---|---|---|---|---|
| MaF1 | 5 | 4.4730e-3 (2.25e-3) − | **1.4114e-2 (5.74e-4)** ≈ | 6.6497e-3 (6.37e-4) − | 1.3859e-2 (1.16e-3) |
| | 10 | 5.9817e-8 (3.39e-8) ≈ | **3.2907e-7 (5.99e-7)** ≈ | 2.5310e-8 (1.86e-8) ≈ | 9.2618e-8 (1.06e-7) |
| | 15 | **5.5707e-13 (3.20e-13)** + | 0.0000e+0 (0.00e+0) ≈ | 2.0014e-13 (2.22e-13) + | 0.0000e+0 (0.00e+0) |
| MaF2 | 5 | 4.0036e-2 (1.77e-3) − | 4.8457e-2 (5.93e-4) − | 4.7877e-2 (4.52e-4) − | **5.6420e-2 (1.61e-4)** |
| | 10 | 5.8943e-3 (3.39e-4) − | 7.1777e-3 (1.20e-4) − | 6.3648e-3 (3.13e-4) − | **8.5940e-3 (8.66e-5)** |
| | 15 | 4.2899e-5 (4.93e-6) − | 5.1488e-5 (1.92e-6) − | 2.8873e-5 (2.09e-6) − | **7.7446e-5 (1.59e-6)** |
| MaF5 | 5 | 3.9780e+4 (2.71e+2) − | 3.8495e+4 (6.97e+2) − | 3.8617e+4 (1.49e+3) − | **4.1206e+4 (3.33e+3)** |
| | 10 | 8.4274e+16 (1.47e+15) − | 4.1725e+16 (7.29e+15) − | 7.5382e+16 (5.14e+15) − | **8.9263e+16 (1.49e+15)** |
| | 15 | 5.2448e+36 (6.60e+34) − | 2.3344e+36 (5.03e+35) − | 3.9622e+36 (7.57e+35) − | **5.4705e+36 (2.94e+34)** |
| MaF6 | 5 | 5.5253e-6 (1.23e-5) − | 5.8077e-3 (1.12e-3) − | 2.7754e-3 (2.43e-3) − | **8.4973e-3 (4.75e-4)** |
| | 10 | 2.7517e-16 (9.94e-16) ≈ | 1.1359e-9 (5.08e-9) ≈ | 1.1457e-8 (2.00e-8) − | **1.4225e-8 (2.53e-8)** |
| | 15 | 5.9627e-33 (2.67e-32) ≈ | 0.0000e+0 (0.00e+0) ≈ | **6.8755e-17 (2.03e-17)** + | 1.6464e-20 (7.36e-20) |
| MaF7 | 5 | 3.1382e-1 (1.68e-1) − | 8.5723e-1 (1.44e-1) − | 6.7882e-1 (2.31e-1) − | **2.1882e+0 (2.12e-1)** |
| | 10 | 8.1883e-5 (1.66e-4) − | 6.2266e-9 (2.69e-8) − | 1.7746e-3 (5.28e-3) − | **2.2289e+0 (2.63e-1)** |
| | 15 | 2.8499e-8 (1.23e-7) − | 0.0000e+0 (0.00e+0) − | 4.7690e-4 (9.39e-4) − | **1.3984e+0 (6.73e-2)** |
| MaF8 | 5 | 4.8212e-1 (8.44e-1) − | 3.4795e-1 (6.94e-1) − | **2.7666e+0 (3.57e-1)** ≈ | 1.7632e+0 (1.57e+0) |
| | 10 | 8.0942e-1 (2.86e+0) − | 2.8735e+0 (4.26e+0) ≈ | **1.1933e+1 (1.02e+0)** + | 3.3384e+0 (4.52e+0) |
| | 15 | 4.5464e-5 (2.03e-4) − | 1.4401e+0 (3.62e+0) − | **1.8766e+1 (4.21e+0)** + | 4.1131e+0 (7.82e+0) |
| MaF9 | 5 | 3.8103e-1 (1.20e+0) − | 3.5155e-1 (8.39e-1) − | 4.6887e+0 (1.19e+0) − | **6.4718e+0 (2.14e+0)** |
| | 10 | 5.8662e-1 (1.42e+0) − | 3.7769e+0 (4.30e+0) − | 3.2205e+0 (3.63e+0) − | **2.3542e+1 (4.95e+0)** |
| | 15 | 2.2483e+0 (5.72e+0) − | 2.4792e-1 (1.11e+0) − | 8.2772e+0 (1.01e+1) ≈ | **3.7107e+1 (4.29e+1)** |
| MaF10 | 5 | 1.7583e+3 (1.38e+2) − | 1.8675e+3 (8.01e+1) − | 1.9681e+3 (1.24e+2) − | **5.4145e+3 (2.09e+2)** |
| | 10 | 2.0308e+9 (1.01e+8) − | 1.9632e+9 (7.20e+7) − | 2.0972e+9 (1.54e+8) − | **7.6136e+9 (6.29e+8)** |
| | 15 | 2.6734e+16 (1.32e+15) − | 2.5528e+16 (2.45e+14) − | 2.7492e+16 (1.85e+15) − | **1.1713e+17 (1.29e+16)** |
| MaF11 | 5 | 5.4806e+3 (6.59e+1) − | 5.5874e+3 (5.65e+1) − | 5.5247e+3 (7.86e+1) − | **5.6871e+3 (9.05e+1)** |
| | 10 | 7.5098e+9 (2.31e+8) − | 7.9002e+9 (1.10e+8) − | 7.7752e+9 (1.87e+8) − | **8.4747e+9 (3.60e+7)** |
| | 15 | 1.1392e+17 (5.25e+15) − | 1.2849e+17 (1.96e+15) − | 1.1820e+17 (5.73e+15) − | **1.3633e+17 (5.17e+14)** |
| MaF12 | 5 | 3.2370e+3 (1.65e+2) − | 3.7097e+3 (1.19e+2) − | 3.9249e+3 (1.19e+2) − | **4.7992e+3 (1.28e+1)** |
| | 10 | 4.2153e+9 (4.57e+8) − | 5.8097e+9 (3.21e+8) − | 5.7978e+9 (4.64e+8) − | **8.5122e+9 (5.15e+7)** |
| | 15 | 7.7633e+16 (5.50e+15) − | 1.0610e+17 (4.24e+15) − | 8.3871e+16 (1.15e+16) − | **1.6210e+17 (1.18e+15)** |
| MaF13 | 5 | 1.9275e-1 (4.45e-2) − | 3.8198e-1 (2.01e-2) − | 2.3449e-1 (4.08e-2) − | **4.3288e-1 (1.90e-2)** |
| | 10 | 1.0518e-1 (1.13e-1) − | 2.8261e-1 (3.39e-2) − | 1.5750e-1 (6.95e-2) − | **3.5362e-1 (7.71e-3)** |
| | 15 | 1.2591e-1 (1.03e-1) − | 2.8117e-1 (4.29e-2) − | 1.6504e-1 (8.37e-2) − | **3.6330e-1 (3.61e-2)** |
| +/−/≈ | | 1/29/3 | 0/27/6 | 4/26/3 | |

MaOEA-DPS can effectively judge the ECRS system's trusted measurement by applying the ensemble selection mechanism based on dynamic probability. Due to the objective space reduction selection method, the MaOEA-R&D has a lower competition with high value in the objective of trust measurement than the other three algorithms.

It can be shown from the preceding analysis that MaOEA-DPS can provide generally consistent and excellent results on each objective.

## V. CONCLUSION

The ECRS problem is seen as a MaOP in this study, and a many-objective edge cloud resource scheduling model is created with five objectives to be optimised: task completion time, server cost, server load balance degree, user satisfaction degree, and task-server trust measurement. The MaOEA-DPS is designed in order to acquire a model solution. Using the notion of support set, the MaOEA-DPS examines the global convergence of eight evolutionary operators. As evolutionary operators of the ensemble algorithm, evolutionary algorithms are chosen. Through mutual coordination and joint action, the dynamic probability ensemble mechanism allows each evolutionary strategy to play to its unique advantages, generating better solutions for selection under the dynamic probability ensemble mechanism. A selection mechanism is employed to select and store the excellent solutions with superior CaD performance. This plays an important role in overcoming the problem of declining selection pressure in the later stage of evolution and realizing a "strong combination" performance. Meanwhile, MaFs and ECRS problems are subjected to two thorough validation tests. MaOEA-R&D and VaEA are not as good as MaOEA-DPS on MaFs when compared to other advanced MaOEAs, except that MaOEA-R&D outperforms MaOEA-DPS on 15 MaF1 objectives. It performs better on

MaF8 than MaOEA-DPS for the RVEA. MaOEA-DPS, on the other hand, outperforms the remainder of RVEA's MaF results, demonstrating that the ensemble selection process based on dynamic probability is effective in picking solutions with improved CaD performance. In addition, when compared to the included algorithms in dealing with the ECRS problem, a relatively good model solution is obtained. The ECRS problem, however, does not include any fuzzy components. More fuzzy factors, such as completion time and trust measurement, should be studied and added into the many-objective model in the future. Furthermore, more efficient MaOEAs could be constructed to improve the solution for the proposed model.

## ACKNOWLEDGEMENT

This work was supported by the Natural Science Foundation of China (61801008), The China National Key R&D Program (No. 2018YFB0803600), Scientific Research Common Program of Beijing Municipal Commission of Education (No. KM201910005025) and Chinese Postdoctoral Science Foundation (No. 2020M670074).

## REFERENCES

[1] X. Wang, K. Wang, S. Wu, S. Di, H. Jin, K. Yang, and S. Ou, "Dynamic resource scheduling in mobile edge cloud with cloud radio access network," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 11, pp. 2429–2445, 2018.

[2] A. Badshah, M. Waqas, F. Muhammad, G. Abbas, and Z. H. Abbas, "A novel framework for smart systems using blockchain-enabled internet of things," *IT Professional*, vol. 24, no. 3, pp. 73–80, 2022.

[3] Y. Li, W. Dai, X. Gan, H. Jin, L. Fu, H. Ma, and X. Wang, "Cooperative service placement and scheduling in edge clouds: A deadline-driven approach," *IEEE Transactions on Mobile Computing*, 2021.

[4] G. Aceto, V. Persico, and A. Pescapé, "Industry 4.0 and health: Internet of things, big data, and cloud computing for healthcare 4.0," *Journal of Industrial Information Integration*, vol. 18, p. 100129, 2020.

[5] K. Cao, L. Li, Y. Cui, T. Wei, and S. Hu, "Exploring placement of heterogeneous edge servers for response time minimization in mobile edge-cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 1, pp. 494–503, 2021.

[6] F. Bai, T. Shen, Z. Yu, K. Zeng, and B. Gong, "Trustworthy blockchain-empowered collaborative edge computing-as-a-service scheduling and data sharing in the iioe," *IEEE Internet of Things Journal*, vol. 9, no. 16, pp. 14 752–14 766, 2022.

[7] J. Fang and A. Ma, "IoT application modules placement and dynamic task processing in edge-cloud computing," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 475–488, 2018.

[8] K. Zhang, Y. Zhu, S. Maharjan, and Y. Zhang, "Edge intelligence and blockchain empowered 5G beyond for the industrial internet of things," *IEEE Network*, vol. 33, no. 5, pp. 12–19, 2019.

[9] I. Sorkhoh, D. Ebrahimi, R. Atallah, and C. Assi, "Workload scheduling in vehicular networks with edge cloud capabilities," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8472–8486, 2019.

[10] B. Cao, J. Zhang, X. Liu, Z. Sun, W. Cao, R. M. Nowak, and Z. Lv, "Edge-cloud resource scheduling in space-air-ground integrated networks for internet of vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 5765–5772, 2022.

[11] Z. Cao, C. Lin, and M. Zhou, "A knowledge-based cuckoo search algorithm to schedule a flexible job shop with sequencing flexibility," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 1, pp. 56–69, 2021.

[12] S. Hu and G. Li, "Dynamic request scheduling optimization in mobile edge computing for iot applications," *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 1426–1437, 2019.

[13] W. Choi, J. Kim, S. Lee, and E. Park, "Smart home and internet of things: A bibliometric study," *Journal of Cleaner Production*, vol. 301, p. 126908, 2021.

[14] M. S. Aliero, K. N. Qureshi, M. F. Pasha, and G. Jeon, "Smart home energy management systems in internet of things networks for green cities demands and services," *Environmental Technology & Innovation*, vol. 22, p. 101443, 2021.

[15] H. Zhang, N. Shlezinger, F. Guidi, D. Dardari, M. F. Imani, and Y. C. Eldar, "Near-field wireless power transfer for 6g internet of everything mobile networks: Opportunities and challenges," *IEEE Communications Magazine*, vol. 60, no. 3, pp. 12–18, 2022.

[16] M. Ghobaei-Arani, A. Souri, and A. A. Rahmanian, "Resource management approaches in fog computing: a comprehensive review," *Journal of Grid Computing*, vol. 18, no. 1, pp. 1–42, 2020.

[17] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 5031–5044, 2019.

[18] Z. Kuang, Z. Ma, Z. Li, and X. Deng, "Cooperative computation offloading and resource allocation for delay minimization in mobile edge computing," *Journal of Systems Architecture*, vol. 118, p. 102167, 2021.

[19] C. Pham, D. T. Nguyen, Y. Njah, N. H. Tran, K. K. Nguyen, and M. Cheriet, "Share-to-run iot services in edge cloud computing," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 497–509, 2022.

[20] Y. Ding, K. Li, C. Liu, and K. Li, "A potential game theoretic approach to computation offloading strategy optimization in end-edge-cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 6, pp. 1503–1519, 2022.

[21] S. Luo, X. Chen, Q. Wu, Z. Zhou, and S. Yu, "HFEL: Joint edge association and resource allocation for cost-efficient hierarchical federated edge learning," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6535–6548, 2020.

[22] Q. Luo, S. Hu, C. Li, G. Li, and W. Shi, "Resource scheduling in edge computing: A survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2131–2165, 2021.

[23] Z. Tong, X. Deng, J. Mei, B. Liu, and K. Li, "Response time and energy consumption co-offloading with slrta algorithm in cloud-edge collaborative computing," *Future Generation Computer Systems*, vol. 129, pp. 64–76, 2022.

[24] E. Ahmed, A. Ahmed, I. Yaqoob, J. Shuja, A. Gani, M. Imran, and M. Shoaib, "Bringing computation closer toward the user network: Is edge computing the solution?" *IEEE Communications Magazine*, vol. 55, no. 11, pp. 138–144, 2017.

[25] C. Kai, H. Zhou, Y. Yi, and W. Huang, "Collaborative cloud-edge-end task offloading in mobile-edge computing networks with limited communication capability," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 2, pp. 624–634, 2021.

[26] Z. Tong, X. Deng, J. Mei, B. Liu, and K. Li, "Response time and energy consumption co-offloading with slrta algorithm in cloud–edge collaborative computing," *Future Generation Computer Systems*, vol. 129, pp. 64–76, 2022.

[27] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *Journal of Systems Architecture*, vol. 98, pp. 289–330, 2019.

[28] S. Tuli, S. Ilager, K. Ramamohanarao, and R. Buyya, "Dynamic scheduling for stochastic edge-cloud computing environments using a3c learning and residual recurrent neural networks," *IEEE Transactions on Mobile Computing*, vol. 21, no. 3, pp. 940–954, 2022.

[29] Y. Liu, M. J. Lee, and Y. Zheng, "Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system," *IEEE Transactions on Mobile Computing*, vol. 15, no. 10, pp. 2398–2410, 2015.

[30] G. Nguyen, S. Dlugolinsky, M. Bobák, V. Tran, Á. L. García, I. Heredia, P. Malík, and L. Hluchỳ, "Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 77–124, 2019.

[31] M. Haus, M. Waqas, A. Y. Ding, Y. Li, S. Tarkoma, and J. Ott, "Security and privacy in device-to-device (d2d) communication: A review," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1054–1079, 2017.

[32] M. Waqas, Y. Niu, Y. Li, M. Ahmed, D. Jin, S. Chen, and Z. Han, "A comprehensive survey on mobility-aware d2d communications: Principles, practice and challenges," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1863–1886, 2020.

[33] S. Sotiriadis, N. Bessis, A. Anjum, and R. Buyya, "An inter-cloud meta-scheduling (icms) simulation framework: Architecture and evaluation," *IEEE Transactions on Services Computing*, vol. 11, no. 1, pp. 5–19, 2015.

[34] M. Waqas, Y. Niu, M. Ahmed, Y. Li, D. Jin, and Z. Han, "Mobility-aware fog computing in dynamic environments: Understandings and implementation," *IEEE Access*, vol. 7, pp. 38 867–38 879, 2019.

[35] J. Xu, Z. Zhang, Z. Hu, L. Du, and X. Cai, "A many-objective optimized task allocation scheduling model in cloud computing," *Applied Intelligence*, vol. 51, no. 6, pp. 3293–3310, 2021.

[36] X. Cai, J. Zhang, Z. Ning, Z. Cui, and J. Chen, "A many-objective multistage optimization-based fuzzy decision-making model for coal production prediction," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 12, pp. 3665–3675, 2021.

[37] Y.-m. Cheung, F. Gu, and H.-L. Liu, "Objective extraction for many-objective optimization problems: Algorithm and test problems," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 755–772, 2016.

[38] Q. Lin, S. Liu, Q. Zhu, C. Tang, R. Song, J. Chen, C. A. C. Coello, K.-C. Wong, and J. Zhang, "Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 32–46, 2016.

[39] Y. Zhai, Y.-S. Ong, and I. W. Tsang, "The emerging" big dimensionality"," *IEEE Computational Intelligence Magazine*, vol. 9, no. 3, pp. 14–26, 2014.

[40] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[41] M. Waqas, S. Tu, Z. Halim, S. U. Rehman, G. Abbas, and Z. H. Abbas, "The role of artificial intelligence and machine learning in wireless networks security: principle, practice and challenges," *Artificial Intelligence Review*, pp. 1–47, 2022.

[42] B. Yang, W. K. Chai, Z. Xu, K. V. Katsaros, and G. Pavlou, "Cost-efficient NFV-enabled mobile edge-cloud for low latency mobile applications," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 12 771–12 781, 2021.

[43] N. Fernando, S. W. Loke, and W. Rahayu, "Computing with nearby mobile devices: a work sharing algorithm for mobile edge-clouds," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 329–343, 2016.

[44] S. Ma, S. Guo, K. Wang, W. Jia, and M. Guo, "A cyclic game for service-oriented resource allocation in edge computing," *IEEE Transactions on Services Computing*, vol. 13, no. 4, pp. 723–734, 2020.

[45] Z. Zhou, S. Yang, L. Pu, and S. Yu, "CEFL: online admission control, data scheduling, and accuracy tuning for cost-efficient federated learning across edge nodes," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9341–9356, 2020.

[46] K. Kaur, S. Garg, G. Kaddoum, S. H. Ahmed, and M. Atiquzzaman, "KEIDS: Kubernetes-based energy and interference driven scheduler for industrial iot in edge-cloud ecosystem," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4228–4237, 2019.

[47] F. Qiao, J. Wu, J. Li, A. K. Bashir, S. Mumtaz, and U. Tariq, "Trustworthy edge storage orchestration in intelligent transportation systems using reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4443–4456, 2021.

[48] N. Moniz and H. Monteiro, "No free lunch in imbalanced learning," *Knowledge-Based Systems*, vol. 227, p. 107222, 2021.

[49] X. Cai, J. Zhang, H. Liang, L. Wang, and Q. Wu, "An ensemble bat algorithm for large-scale optimization," *International Journal of Machine Learning and Cybernetics*, vol. 10, no. 11, pp. 3099–3113, 2019.

[50] Q. Gu, X. Zhang, L. Chen, and N. Xiong, "An improved bagging ensemble surrogate-assisted evolutionary algorithm for expensive many-objective optimization," *Applied Intelligence*, vol. 52, no. 6, pp. 5949–5965, 2022.

[51] H. Peng, W. Xiao, Y. Han, A. Jiang, Z. Xu, M. Li, and Z. Wu, "Multi-strategy firefly algorithm with selective ensemble for complex engineering optimization problems," *Applied Soft Computing*, vol. 120, p. 108634, 2022.

[52] Q. Fan and X. Yan, "Differential evolution algorithm with self-adaptive strategy and control parameters for p-xylene oxidation process optimization," *Soft Computing*, vol. 19, no. 5, pp. 1363–1391, 2015.

[53] A. Hashemi, M. Bagher Dowlatshahi, and H. Nezamabadi-pour, "Ensemble of feature selection algorithms: a multi-criteria decision-making approach," *International Journal of Machine Learning and Cybernetics*, vol. 13, no. 1, pp. 49–69, 2022.

[54] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE transactions on evolutionary computation*, vol. 15, no. 1, pp. 55–66, 2011.

[55] M. Yu, J. Liang, Z. Wu, and Z. Yang, "A twofold infill criterion-driven heterogeneous ensemble surrogate-assisted evolutionary algorithm for computationally expensive problems," *Knowledge-Based Systems*, vol. 236, p. 107747, 2022.

[56] R. Jiao, S. Zeng, J. S. Alkasassbeh, and C. Li, "Dynamic multi-objective evolutionary algorithms for single-objective optimization," *Applied Soft Computing*, vol. 61, pp. 793–805, 2017.

[57] Y. Sun, F. Lin, and H. Xu, "Multi-objective optimization of resource scheduling in fog computing using an improved nsga-ii," *Wireless Personal Communications*, vol. 102, no. 2, pp. 1369–1385, 2018.

[58] Z. Peng, J. Lin, D. Cui, Q. Li, and J. He, "A multi-objective trade-off framework for cloud resource scheduling based on the deep q-network algorithm," *Cluster Computing*, pp. 1–15, 2020.

[59] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[60] S. Shi and H. Xiong, "A hybrid immune genetic algorithm with tabu search for minimizing the tool switch times in cnc milling batch-processing," *Applied Intelligence*, vol. 52, no. 7, pp. 7793–7807, 2022.

[61] K. Qiao, J. Liang, K. Yu, M. Yuan, B. Qu, and C. Yue, "Self-adaptive resources allocation-based differential evolution for constrained evolutionary optimization," *Knowledge-Based Systems*, vol. 235, p. 107653, 2022.

[62] Z. Liang, H. Dong, C. Liu, W. Liang, and Z. Zhu, "Evolutionary multitasking for multiobjective optimization with subspace alignment and adaptive differential evolution," *IEEE Transactions on Cybernetics*, vol. 52, no. 4, pp. 2096–2109, 2022.

[63] Z. Cui, J. Zhang, D. Wu, X. Cai, H. Wang, W. Zhang, and J. Chen, "Hybrid many-objective particle swarm optimization algorithm for green coal production problem," *Information Sciences*, vol. 518, pp. 256–271, 2020.

[64] C. Tang, S. Song, J. Ji, Y. Tang, Z. Tang, and Y. Todo, "A cuckoo search algorithm with scale-free population topology," *Expert Systems with Applications*, vol. 188, p. 116049, 2022.

[65] Z. Cui, J. Zhang, Y. Wang, Y. Cao, X. Cai, W. Zhang, and J. Chen, "A pigeon-inspired optimization algorithm for many-objective optimization problems," *Science China Information Sciences*, vol. 62, no. 7, pp. 1–3, 2019.

[66] Z. Zhao, M. Zhang, Z. Zhang, Y. Wang, R. Cheng, J. Guo, P. Yang, C. S. Lai, P. Li, Lai, and L. Lei, "Hierarchical pigeon-inspired optimization-based mppt method for photovoltaic systems under complex partial shading conditions," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 10, pp. 10 129–10 143, 2022.

[67] K. Deb, R. B. Agrawal *et al.*, "Simulated binary crossover for continuous search space," *Complex systems*, vol. 9, no. 2, pp. 115–148, 1995.

[68] C. C. Coello and M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)*, vol. 2. IEEE, 2002, pp. 1051–1056.

[69] S. Zapotecas Martínez and C. A. Coello Coello, "A multi-objective particle swarm optimizer based on decomposition," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, 2011, pp. 69–76.

[70] C. Dai, Y. Wang, and M. Ye, "A new multi-objective particle swarm optimization algorithm based on decomposition," *Information Sciences*, vol. 325, pp. 541–557, 2015.

[71] C. R. Raquel and P. C. Naval Jr, "An effective use of crowding distance in multiobjective particle swarm optimization," in *Proceedings of the 7th Annual conference on Genetic and Evolutionary Computation*, 2005, pp. 257–264.

[72] R. Cheng, M. Li, Y. Tian, X. Zhang, S. Yang, Y. Jin, and X. Yao, "A benchmark test suite for evolutionary many-objective optimization," *Complex and Intelligent Systems*, vol. 3, p. pages67–81, 2017.

[73] Z. He and G. G. Yen, "Many-objective evolutionary algorithm: Objective space reduction and diversity improvement," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 145–160, 2015.

[74] Y. Xiang, Y. Zhou, M. Li, and Z. Chen, "A vector angle-based evolutionary algorithm for unconstrained many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 1, pp. 131–152, 2016.

[75] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 773–791, 2016.

[76] X. Ma, S. Wang, S. Zhang, P. Yang, C. Lin, and X. S. Shen, "Cost-efficient resource provisioning for dynamic requests in cloud assisted mobile edge computing," *IEEE Transactions on Cloud Computing*, vol. 9, no. 3, pp. 968 – 980, July-Sept. 1 2021.

[77] E. Z. Johannes Bader, "Hype: An algorithm for fast hypervolume-based many-objective optimization," *Evolutionary Computation*, vol. 19, no. 1, pp. 45 – 76, 2011.