Hindawi
Security and Communication Networks
Volume 2022, Article ID 3810670, 11 pages
https://doi.org/10.1155/2022/3810670

WILEY | Hindawi

Research Article

Edge Computing Server Placement Strategy Based on SPEA2 in Power Internet of Things

Yongling Lu, Zhen Wang, Chengbo Hu, Ziquan Liu, and Xueqiong Zhu

Electric Power Science Research Institute, State Grid Jiangsu Electric Power Co., Ltd., Nanjing 211103, China

Correspondence should be addressed to Yongling Lu; yongling_lu@hotmail.com

Received 10 May 2022; Revised 24 June 2022; Accepted 20 July 2022; Published 24 August 2022

Academic Editor: Qi Li

Copyright © 2022 Yongling Lu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In order to meet the edge services placement demand for multiobjective optimization of Power Internet of Things, an edge services placement strategy based on an improved strength Pareto evolutionary algorithm (SPEA2) is proposed in this paper. Firstly, we model the delay, resource utilization, and energy consumption. Then, a multiobjective optimization is proposed. Finally, an enhanced genetic algorithm is used to derive the decision candidate set. Moreover, the optimal solution in the candidate set is selected to be utilized in the iteration of the multicriteria decision and the superior-inferior solution distance method. Numerical results and analysis show that the proposed strategy is more effective in reducing system delay, improving resource utilization, and saving energy consumption than the other two benchmark algorithms.

1. Introduction

With the rapid development of the Power Internet of Things (IoT), the IoT nodes of the power supply terminal, including smart devices and emerging applications, show explosive growth, which leads to massive heterogeneity and complex processing of the data [1, 2]. In the power industry, cloud computing architecture is usually used to upload terminal data to the cloud platform for centralized processing. However, the traditional cloud computing center is far away from the power grid equipment, and uploading data to the cloud platform can lead to large time delays [3]. In addition, centralizing data in the cloud platform can cause a burden on network communication and computing resources, resulting in transmission interruption or link congestion. Therefore, it is difficult for the cloud computing architecture to meet the service requirements of terminal equipment in the Power Internet of Things [4, 5].

Edge computing improves the service capability of the network by deploying the edge servers at radio access network side to provide power grid equipment with powerful computing and storage capabilities. Nowadays, edge computing has been widely used in many fields, such as mobile big data analytics and Power Internet of Things [6–8]. In addition, for the current power grid equipment, the deployment of edge computing could relieve the challenge caused by the lack of power and computing capacity. However, the proper edge service placement strategy needs to be designed to optimize other parameters such as energy consumption and resource utilization while simultaneously providing high-performance services to power grid equipment.

A lot of researches have been done on the service placement of edge computing and some constructive solutions have been proposed. In [9], to address the problem of edge computing service placement under resource constrained conditions, the authors have regarded that the ubiquitous MEC could implement service migration in mobile networks with highly dynamic characteristics by supporting multiserver collaboration. To maximize the system utility of the system, the optimization problem has been formulated by joint considering the constraints of server storage capacity and service delay. Firstly, the long-term optimization problem has been decomposed into a series of immediate optimization by the Lyapunov method. Then, a stochastic algorithm based on sample average approximation is proposed to approximate future expected system utility values. Next, the distributed Markov

approximation algorithm is used to determine the service placement policy. For addressing cost and energy consumption in edge computing power scenarios, in [10], the authors have considered that the energy consumption of servers is an important part of service cost in edge computing systems. Thus, the energy-aware edge computing application service placement problem has been designed; then the problem has been modeled as a multistage stochastic programming problem. The objective is to maximize the Quality of Service (QoS), under the energy budget constraints of the edge computing server. Finally, a novel sampling average approximation algorithm has been designed to solve the problem.

To address the problem of multiobjective optimization requirements for the placement of edge computing service, in [11], the authors have considered that one of the main challenges of edge computing is to consider service load variations and determine multiobjective performance optimization to make service placement decisions. The optimal service placement problem has been solved by further considering how to allocate the service loads placed at different locations. And a dynamic predictive service combined with load allocation strategy has been proposed by estimating the performance-cost tradeoff for service migration. The strategy has utilized the small amount of predictive processing to reduce the impact of load fluctuations. In [12], the authors have defined a network entity with a flexible allocation of communication, computing, and storage capabilities so that the resource constrained devices could use the communication and computation resources required for the service. In addition, the spectrum-aware service placement in edge computing has been investigated. The authors have formulated the service placement as a stochastic optimization problem. Then, the authors have jointed the optimized service placement, traffic routing, and spectrum allocation. Based on those, an enhanced coarse-grained service placement algorithm has been proposed.

Edge computing service architectures have recently attracted a lot of attention. In [13], the authors consider the multidimension of task requirements in mobile crowd sensing and propose a task-oriented user selection incentive mechanism to achieve higher task completion rate and maximize resource utilization. In order to solve the problem of insufficient accuracy of the medium-edge service model in the Industrial Internet of Things, a new smart contract was constructed in [14] to encourage multiple marginal service users to participate, thereby improving the model accuracy. In addition, a scale weighted aggregation strategy was proposed to verify the model parameters to improve the accuracy of the model. In [15], the Graph theory was introduced into the edge caching network architecture to reduce the processing complexity. Considering the physical attributes and social attributes, a cache solution based on physical-social weighted direction is proposed to minimize the average download latency of all edge users within a macrocell.

The improved genetic algorithm used in this paper has been partially explored by some scholars in the field of service placement using genetic algorithms. In [16], the authors have proposed an algorithm that combined the genetic algorithm with Monte Carlo simulations. The algorithm can greatly improve the efficiency of exhaustive search service placement strategy. First, an optimization model has been developed for the genetic algorithm; the main body of the model has been the QoS objective function, cost objective function, and the resource utilization objective function. Then, the FogTorch Monte Carlo framework has been utilized to address the problem. The proposed algorithm could minimize the resource consumption and service placement cost in the fog while guaranteeing QoS. By defining a representation of an application placement in a biased-random-key chromosome and using a fault-tolerance distributed pool model, the GRECO algorithm was proposed in [17] to solve the application placement problem in constrained hybrid cloud environment. In this paper, the multiobjective problem is also optimized using a genetic algorithm but different from [16, 17]. We utilize multicriteria decision and superior-inferior solution distance method to combine three fitness functions into a single meritocratic function to assist the search.

In this paper, we focus on multiobjective optimization requirements in power scenarios. To address the above issues, we develop an improved genetic algorithm based edge service placement (IGA-ESP) strategy, which optimizes the delay, energy consumption, and resource utilization parameters. The main contributions of this paper are summarized as follows: (1) We study the edge service placement problem in the Power Internet of Things and establish a multiobjective optimization problem under the constraint of the edge cloud capacity and a single service request per time slot. The objectives of this problem include minimizing service delay and energy consumption while maximizing resource utilization. (2) We propose the IGA-ESP strategy to solve this multiobjective optimization problem. Firstly, the decision candidate set is obtained by using the improved genetic algorithm. Then, the optimal solution in the candidate set is filtered using the multicriteria decision and the superior-inferior solution distance method. (3) Simulation results show that the proposed strategy can effectively reduce system delay, improve resource utilization, and save energy consumption. Furthermore, compared with TS algorithm and Greedy algorithm, IGA-ESP strategy can reduce the average end-to-end delay of power grid equipment by 7.8% and 16.7%, respectively.

2. System Model

In Power Internet of Things, edge computing networks can provide powerful infrastructure resource and value-added service capabilities for power grid terminal applications with insufficient power, computing power, and storage resource, such as remote monitoring, smart home, and VR. The guarantee of low-latency services requires a reasonable edge service placement strategy. In this section, the edge computing servers are deployed at the edge of the network closer to the power grid equipment. Besides, the computing power of edge computing is utilized to process service requests from power grid equipment, close to the edge of the

network. In this section, we first present the network model. Then, the placement model of each edge server is proposed. Finally, we present the wireless communication model between the grid equipment and the edge cloud.

2.1. Network Model. The network architecture is shown in Figure 1. There are multiple Small Base Stations (SBS) in the coverage area of Macro Base Station (MBS). In addition, all the SBS with edge computing server enhancements are called edge cloud (EC), and it is expressed as $E = \{1, 2, 3, \ldots, i, \ldots, N\}$. To improve service quality for power grid equipment and reduce service deployment costs for Application Service Provider (ASP), ASPs deploy a limited number of power popular application services, such as intelligence operations and video surveillance service in each EC by MBS. The set of all the service types of the system is represented by $\kappa = \{1, 2, 3, \ldots, k, \ldots, K\}$. In this model, the power grid equipment first uploads the service request to the local EC through the wireless channel. If the requested service already exists in the local EC, the power grid equipment will be served by the EC; otherwise the service request of the power grid equipment will be uploaded to the MBS via the local EC and forwarded to the cloud server of the ASP by the MBS.

2.2. Service Placement Model. The service placement model is implemented using containers, which are configured to allocate resource to power grid equipment to provide edge services [18].

It is assumed that each edge server has a certain number of unit containers and each unit container has a fixed amount of storage and compute resources. In addition, all edge servers use the same size unit container but are differences in the number of unit containers. Each container occupies an integer number of unit container resource. ASP places the service k on the EC i at the slot t denoted by $y_i^k(t) = 1$; otherwise, $y_i^k(t) = 0$. We define the number of containers per unit that the storage service k needs to occupy as r_k. The limited number of unit containers of EC results in the number of containers per unit occupied by the hosted service of EC i which needs to meet the following constrains at the slot t.

$$\sum_{k=1}^{K} r_k y_i^k(t) \le R_i, \tag{1}$$

where r_k denotes the number of containers per unit that the service k needs to occupy. R_i denotes the total number of unit containers owned by EC i.

2.3. Wireless Communication Model. The power grid equipment uploads the service request to the local EC which would incur a wireless communication delay; thus, we assume that the data to be uploaded after the service requested by the power grid equipment has been processed as $d_{i,j}^k(t)$, and the uplink channel gain between the power grid equipment j and the base station i is $H_{i,j}^k(t)$. The effect on the channel gain is negligible because the change of power grid equipment location within a time slot is very small. Therefore, the channel gain between the power grid equipment and the base station is assumed to be constant within a time slot. The transmission power of the power grid equipment is represented by $P_j(t)$; thus, the uplink transmission bandwidth between the power grid equipment j and the EC i can be expressed according to the Shannon channel capacity formula as

$$c_{i,j}^k(t) = B \log 2\left(1 + \frac{P_j(t) H_{i,j}(t)}{N_0 B + I}\right), \tag{2}$$

where B represents the channel bandwidth. N_0 is bilateral power spectral density of additive white Gaussian noise. I denotes the stochastic noise power.

3. Problem Formulation and Analysis

The edge computing networks can provide distributed computing resources and low-latency services for power grid equipment with insufficient battery capacity and computing resources. A reasonable edge service placement strategy can enhance the power grid equipment service quality in the edge computing networks. The resource constraints reduced QoS of delay-sensitive tasks and heavy traffic load applications. Thus, we deploy the edge computing server at the Power Internet of Things network edge closer to the power grid equipment and utilize the computing capacity of the edge computing to process power grid equipment service requests close to the network edge. In this section, we first introduce the heterogeneous service network architecture of the edge computing network. Then we calculate the service placement model and communication model according to the power grid equipment access network conditions.

3.1. Service Delay Model. In the edge computing network, $U_i, i \in E$, represents the number of power grid equipment served by EC i. Considering the limited computing resources and battery capacity of power grid equipment, we upload the power grid equipment's service requests to the covered edge cloud for computing and processing [19, 20].

$x_{i,j}^k(t), k \in \kappa$, denotes the service type requested by power grid equipment j in edge cloud i at time t. $x_{i,j}^k(t) = 1$ represents the k-th service required by power grid equipment j in edge cloud i; otherwise $x_{i,j}^k(t) = 0$. We assume that power grid equipment j served by edge cloud i at any time t can only request a type of service, and $x_{i,j}^k(t)$ is expressed as

$$\sum_{k=1}^{K} x_{i,j}^k(t) = 1, \quad \forall i \in E,\ 1 \le j \le U_i. \tag{3}$$

In addition, in Power Internet of Things, considering the limited computing and storage resources of edge servers, ASP can only deploy limited services in each edge server, so ECs in service hotspot areas are prone to overload. In this regard, if the associated EC of a power grid equipment does not host the service required by the power grid equipment,

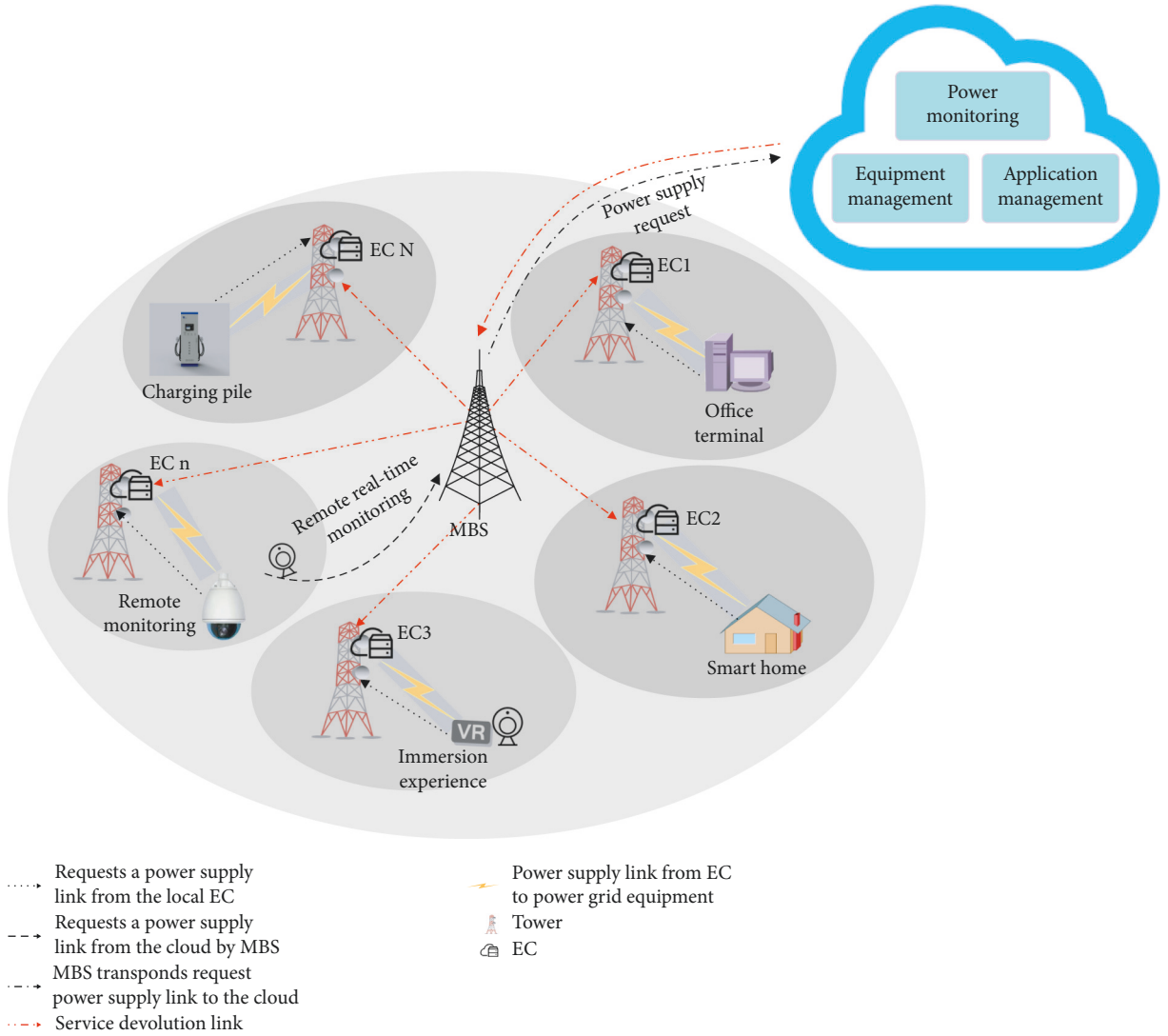

Figure 1: Edge computing service placement architecture.

the power grid equipment can only upload the service request to the ASP cloud server hosting all services through the EC. Therefore, the calculation of the power grid equipment's uplink transmission delay mainly includes two cases.

(i) When there is the k-th service requested by power grid equipment j in EC i, the uplink transmission delay of power grid equipment j served by EC i can be calculated as

$$T_{i,j}^{k,e}(t) = \frac{d_{i,j}^{k}(t)}{c_{i,j}^{k}(t)}. \tag{4}$$

(ii) When there is no k-th service requested by power grid equipment j EC i, the requested service can only be uploaded to the MBS through the EC and then forwarded to the cloud center through the core network. Thus, the uplink transmission delay of power grid equipment j served by EC i can be derived as

$$T_{i,j}^{k,c}(t) = d_{i,j}^{k}(t)\left\{\frac{1}{c_{i,j}^{k}(t)} + \frac{1}{c_{i}^{\text{cloud}}} + \frac{1}{c^{\text{core}}}\right\}, \tag{5}$$

where c_i^{cloud} represents the backhaul link bandwidth between EC i and MBS. c^{core} denotes the data transmission rate of the core network. Therefore, the final uplink transmission delay of power grid equipment j served by EC i can be expressed as

$$T_{i,j}^{k}(t) = x_{i,j}^{k}(t)\left[y_{i}^{k}(t)T_{i,j}^{k,e}(t) + \left(1 - y_{i}^{k}(t)\right)T_{i,j}^{k,c}(t)\right]. \tag{6}$$

Subsequently, we model the calculation delay. Considering the limited computing capacity of EC and the cloud computing center, there will be a certain computing delay when the service requested by power grid equipment is completed. c_k denotes the number of central processing unit (CPU) cycles required to complete the k-th service in edge cloud, and the unit is CPU cycle. Simultaneously, the computing capacity of the unit container is expressed as F^{rb}, and the unit is CPU cycle/s.

Thus, the calculation delay of power grid equipment j served by EC i can be calculated as

$$T_{i,j}^{k,ec}(t) = \frac{x_{i,j}^{k}(t) y_i^k(t) c_k}{F^{rb} r_k}. \tag{7}$$

It is worth noting that the power grid equipment's service request can only be uploaded to the cloud server of ASP through the Power Internet of Things local EC when the power grid equipment's service request cannot be responded to by the local edge cloud. As the cloud server has a strong computing capacity, it is assumed that the cloud server will always provide the services for each power grid equipment with F^c computing capacity, and the unit is CPU cycle. Thus, when the service request of power grid equipment j served by EC i is responded to by the cloud server, the calculation delay can be derived as

$$T_{i,j}^{k,cc}(t) = \frac{x_{i,j}^{k}(t)\left(1 - y_i^k(t)\right) c_k}{F^c}. \tag{8}$$

For the condition of cloud server storage service, we assume that there are all types of services in the cloud server of ASP [21].

To sum up, the total computing delay of power grid equipment j served by EC i is expressed as

$$T_{i,j}^{\text{comp}}(t) = T_{i,j}^{k,ec}(t) + T_{i,j}^{k,cc}(t). \tag{9}$$

Therefore, the end-to-end delay of all power grid equipment requesting services in all ECs at time t can be calculated as

$$D(t) = \sum_{i=1}^{N} \sum_{j=1}^{U_i} \sum_{k=1}^{K} T_{i,j}^{k}(t) + T_{i,j}^{\text{comp}}(t). \tag{10}$$

3.2. Resource Utilization Model. We allocate containers for power grid equipment services to provide services [22], and each container occupies a certain unit container. The resources of the cloud center are relatively sufficient, so the resource utilization only refers to the utilization of the edge cloud unit container, and the resource utilization of the edge cloud is expressed as

$$r_i^{\text{ratio}}(t) = \frac{1}{R_i} \sum_{k=1}^{K} r_k \cdot x_{i,j}^{k}(t). \tag{11}$$

Then, the resource utilization of all participating edge clouds is expressed as

$$\text{RU}(t) = \frac{1}{N} \sum_{i=1}^{N} r_i^{\text{ratio}}(t). \tag{12}$$

3.3. Energy Consumption Model. The edge computing server energy consumption in the Power Internet of Things networks is mainly divided into two categories: (1) the basic energy consumption to ensure the operation of the edge cloud and the cloud center, and (2) the computing unit container energy consumption used by the edge cloud and the cloud center to provide services.

The key factor that affects the basic energy consumption of edge cloud is the service duration t_{si} of edge cloud n, which is determined by the service that provides power grid equipment with the longest service time among all services and is given as

$$t_{si} = \max_{k=1}^{K} x_{i,j}^{k}(t) y_i^k(t) \cdot T_{i,j}^{k,ec}(t), \tag{13}$$

where $T_{i,j}^{k,ec}(t)$ denotes the execution time of service k in edge cloud i.

The basic energy consumption of all edge clouds is expressed as

$$P_b^e(t) = \sum_{i=1}^{N} t_{si} P_{\text{EC}}. \tag{14}$$

P_{EC} represents the operating basic power of the edge cloud. To facilitate representation, we assume that the operating power of all edge clouds is stable and constant, and all edge clouds operate at the same basic power.

The cloud center service duration is determined by the service with the longest execution time in the cloud, calculated as

$$t_{s0} = \max_{k=1}^{K} x_{i,j}^{k}(t)\left(1 - y_i^k(t)\right) \cdot T_{0,j}^{k,\text{ec}}(t), \tag{15}$$

where $T_{0,j}^{k,\text{ec}}(t)$ denotes the execution time of service k in the cloud center.

The basic energy consumption of the cloud center is calculated as

$$P_b^c(t) = t_{s0} \cdot P_c, \tag{16}$$

where P_c is the operating basic power of the cloud center. For the convenience of representation, we assume that the operating power of the cloud center is unvarying.

The total energy consumption of all unit containers required to request services from the edge is expressed as

$$P_s^e(t) = \sum_{i=1}^{N} \sum_{j=1}^{U_i} \sum_{k=1}^{K} x_{i,j}^{k}(t) y_i^k(t) \cdot T_{i,j}^{k,\text{ec}} \cdot r_k \cdot \gamma, \tag{17}$$

where γ represents the average operating power of the unit container. For the convenience of manifestation, it is assumed that the average running power per unit container required for all deployment services is stable and unchangeable.

The energy consumption of computing resources required to request services from the cloud center is calculated as

$$P_s^c(t) = \sum_{k=1}^{K} x_{i,j}^{k}(t)\left(1 - y_i^k(t)\right) \cdot T_{0,j}^{k,\text{cc}} \cdot \zeta. \tag{18}$$

The computing resources allocated by the cloud center are always stable, so it is assumed that the energy consumption per unit time generated by allocating computing resources in the cloud is ζ.

Therefore, after the service placement decision is made at time t in the system, its total energy consumption is expressed as

$$P(t) = P_b^e(t) + P_b^c(t) + P_s^e(t) + P_s^c(t). \tag{19}$$

3.4. Problem Formulation. To sum up, for any edge cloud EC i and power grid equipment j, the goal of this paper is to minimize the power grid equipment-aware end-to-end service delay $D(t)$ and the energy consumption $P(t)$ of edge cloud in the Power Internet of Things and maximize the utilization of edge cloud resources $RU(t)$, that is, to minimize energy consumption while improving overall service performance, and the problem is formulated as

$$\begin{aligned} &P1 \min D(t), P(t) \\ &\max RU(t) \\ &C1 \sum_{k=1}^{K} r_k y_i^k(t) \le R_i \\ &C2 \sum_{k=1}^{K} x_{i,j}^k(t) = 1, \forall i \in E,\ 1 \le j \le U_i \end{aligned} \tag{20}$$

where $C1$ indicates that the number of unit containers requesting services cannot exceed the total number of edge cloud unit containers. $C2$ means that each power grid equipment requests only one service at a time t. Because the problem $P1$ is a multiobjective optimization problem and NP-hard, it is difficult to solve the problem directly. The heuristic algorithm has strong robustness and global search ability and is widely used in various optimization problems. In addition, the heuristic algorithm is more efficient than the traditional search algorithm and can obtain the approximate global optimal solution in a short time. Therefore, in this paper, we consider using the improved genetic algorithm to solve the problem.

4. Edge Service Placement Strategy Based on Improved Genetic Algorithm

Genetic algorithm is an adaptive heuristic intelligent search algorithm that simulates the evolutionary process in nature to solve optimization problems [23]. The algorithm updates individuals through selection, crossover, and mutation operations and obtains an approximate optimal solution after several generations of evolution. Moreover, it can automatically adjust the search direction according to the population selection, so that it has a better global optimization ability. Compared with other heuristic algorithms, genetic algorithm avoids falling into a local optimum in the solution process through gene mutation, making the solution closer to the global optimum. Therefore, it is more suitable for solving complex nonlinear optimization problems [24].

In this section, we propose an edge computing service placement strategy based on IGA-ESP, which achieves multiobjective optimization. In this strategy, population chromosomes can be used to represent candidate solutions to the problem. If a solution satisfies the constraints of the problem, it is feasible; otherwise, it is infeasible. The proposed improved genetic algorithm uses chromosome representation, crossover, and mutation operators according to the needs of the problem, which are used to penalize infeasible solutions, so that these infeasible solutions have a smaller selection (or survival) probability. Specifically, we first use the IGA-ESP algorithm to select the candidate solutions. Then, we use the distance method of superior and inferior solutions and the multicriteria decision selection genetic algorithm to generate the optimal placement decision among the candidate decisions.

4.1. Service Placement Strategy Based on Improved Genetic Algorithm

4.1.1. Chromosomes and Chromosome Codes. For all genetic algorithms, what is considered firstly is how the chromosomes are encoded. In this paper, each chromosome is represented by an integer array, and each chromosome represents a complete service placement strategy; thus all solutions of the problem space can be expressed as the designed genotype, and any genotype corresponds to a possible solution, in which the array of each element corresponds to a service placement decision parameter for each service placement strategy. Actually, the service placement decision of each EC is an array. The algorithm proposed in this paper defines the chromosome as the set of all array service placement strategies. We first number the ECs, then put the service placement decision parameters into the array in order, and finally get an array with a length of NK. Numbers 1 and 0 in the array indicate that the service is placed in and not placed in the edge cloud, respectively.

4.1.2. Initialization. The initial population size in genetic algorithms has a critical influence on searchability. If the size is small, searchability is limited and rapid convergence occurs in early runs. Conversely, a larger initial population size leads to dispersion of solutions, which affects the efficiency and effectiveness of the algorithm. In addition, the crossover probability, mutation probability, and the number of iterations need to be set. The setting of these parameters requires extensive experimental exploration.

4.1.3. Selection Operator. In this paper, we adopt a binary tournament selection operator to select individuals with better performance, thereby enhancing the performance of the algorithm.

The binary tournament selection operator compares two individuals. If the matching pool is sufficient, a pruning process is performed to remove individuals with poor fitness. If the match pool is insufficient, the selection process continues until the match pool is sufficient. The selection operator can gradually eliminate inferior genes, so the

performance of the algorithm can be promoted in the continuous iterative process.

4.1.4. Crossover and Mutation Operator. We exploit the single-point crossover operator to randomly select crossover points from 1 to the number of genes per chromosome. The single-point crossover operator refers to first selecting a crossover point in genes with a certain random probability and then exchanging the gene codes located in the same position to generate new individuals. The mutation operator alters partial genes in a single chromosome with a certain probability, resulting in better chromosomes and preventing rapid convergence. Inappropriate mutation probability will have a malign influence on the algorithm results. The confirmation of the mutation operator requires repeated attempts, and the optimal mutation operator is selected by exploring the actual effect of the algorithm obtained by multiple mutation operators within a reasonable range. The mutation operator ensures the diversity of the population and has a very critical impact on the local search ability of the genetic algorithm.

4.1.5. Fitness Function and Constraints. The fitness function is utilized to calculate the environmental fitness of each individual. Representing solutions as individuals, all solutions constitute a population. The fitness function needs to be divided into three types: power grid equipment-aware delay fitness function, resource utilization fitness function, and energy consumption fitness function. Considering there is only one function for the fitness function judgment of the genetic algorithm, the selection function of the optimal solution will be obtained after processing these three functions through the multiattribute decision in the multicriteria decision and the superior-inferior solution distance method.

4.2. Optimal Strategy Using Superior-Inferior Solution Distance and Multicriteria Decision. Among all the strategies generated by the improved SPEA2 algorithm, the optimal placement strategy is obtained by using the superior-inferior solution distance method and the multicriteria decision method. In the superior and inferior solution distance method, the solutions are sorted according to the Euclidean distance between the candidate solution and the superior and inferior solution, and the superior solution is defined as the object closest to the ideal solution and furthest away from the negative ideal solution. Similarly, the inferior solution is defined as the object closest to the negative ideal solution and farthest from the ideal solution.

We assume that SPEA2 obtains H strategies to wait for the next step after analyzing all the strategies. Each strategy includes delay, resource utilization, and power consumption, which can be denoted as $D_{\text{attribute}} = [D_1, D_2, D_{31}, \ldots, D_H]$, $R_{\text{attribute}} = R_1, R_2, R_3, \ldots, R_H$, and $E_{\text{attribute}} = E_1, E_2, E_3, \ldots, E_H$, respectively.

The normalized delay can be expressed as

$$V_h^D = \frac{D_h}{\sqrt{\sum_{h=1}^{H} D_h^2}}. \tag{21}$$

The normalized resource utilization can be expressed as

$$V_h^R = \frac{R_h}{\sqrt{\sum_{h=1}^{H} R_h^2}}. \tag{22}$$

The normalized energy consumption can be expressed as

$$V_h^E = \frac{E_h}{\sqrt{\sum_{h=1}^{H} E_h^2}}. \tag{23}$$

The weight values of delay, resource utilization, and energy consumption are ω_D, ω_R, and ω_E, respectively. Therefore, their weighted normalized values are defined as

$$\begin{aligned} W_h^D &= \omega_D \cdot V_h^D, \\ W_h^R &= \omega_R \cdot V_h^R, \\ W_h^E &= \omega_E \cdot V_h^E. \end{aligned} \tag{24}$$

We aim to maximize resource utilization and delay and minimize the energy consumption in the Power Internet of Things through the analysis of the problem. Therefore, we define the resource utilization as the ideal solution, while delay and energy consumption are the nonideal solution. $W_{\max}^R$, $W_{\max}^D$, and $W_{\max}^E$ are the maximum values of the three targets, while $W_{\min}^R$, $W_{\min}^D$, and $W_{\min}^E$ are the minimum values.

The distance between the ideal solution and alternative solution can be denoted as

$$D_h^{\text{IS}} = \sqrt{\left(W_h^R - W_{\max}^R\right)^2 + \left(W_h^D - W_{\max}^D\right)^2 + \left(W_h^E - W_{\max}^E\right)^2}. \tag{25}$$

The distance between the nonideal solution and alternative solution is given by

$$D_h^{\text{NS}} = \sqrt{\left(W_h^R - W_{\min}^R\right)^2 + \left(W_h^D - W_{\min}^D\right)^2 + \left(W_h^E - W_{\min}^E\right)^2}. \tag{26}$$

The proximity between the ideal solution and alternative solution can be represented as

$$C_h^{\text{IS}} = \frac{D_h^{\text{NS}}}{D_h^{\text{NS}} + D_h^{\text{IS}}}. \tag{27}$$

According to the proximity of alternative solutions, the superior solution can be expressed as

$$\begin{aligned} &OP = \max_{h=1}^{H} C_h^{\text{IS}}, \\ &\text{s.t. } \omega_D + \omega_R + \omega_E = 1, \\ &\omega_D, \omega_R, \omega_E \in [0, 1], \end{aligned} \tag{28}$$

where the constraints indicate that the weights of delay, resource utilization, and energy consumption are 0 to 1, and the sum of the three weight factors is equal to 1.

The specific process of IGA-ESP is summarized in Algorithm 1. The workflow of IGA-ESP is shown in Algorithm 1. The input of the algorithm is the iteration times I, and the output is the optimal service placement strategy OP. The algorithm obtains the service set and performs crossover and mutation operations firstly, then performs the calculation of fitness function and selects the best individual for the next generation, next calculates the proximity of service placement strategy, and selects the placement strategy with the maximum proximity. The above process is repeated until the maximum number of iterations is reached to output the best strategy.

5. Numerical Results and Analysis

5.1. Simulation Parameter Settings. Matlab platform is very suitable for the simulation of complex systems because of its powerful computing ability. To shorten the simulation time, Huawei FusionServer Pro rack servers with strong computing performance are used for cloud computing, virtualization, high-performance computing, databases, and SAP HANA computation-intensive scenarios. In addition, the integrated high reliability design of the whole process, BSST system startup accelerated storage, DEMT smart energy efficiency, FDM smart diagnosis, and other technologies can further improve system performance.

In Power Internet of Things, we assume that the number of edge clouds within the coverage area of MBS is 3, and user power grid equipment is distributed within each EC. This paper assumes that the number of edge clouds within the coverage area of MBS is 3, and user power grid equipment is evenly distributed within each EC uniformly. The uplink transmission bandwidth allocated by the edge cloud for each power grid equipment is 10 Mbps. The number of service requests in the Power Internet of Things system is 4. Each edge cloud has a unit container range of 50 to 200. The storage capacity of the unit container is 1 GB, and the computing capacity of the unit container is 1 GHz. Other simulation parameter settings are shown in Table 1.

In this section, we compare the IGA-ESP algorithm with other two benchmark algorithms. The first one is the Tabu Search (TS) algorithm [25]; the algorithm has considered the cost optimal service placement problem and proposed a delay aware service placement strategy based on the placement cost, which can guarantee the minimum QoS requirements of the service and balance the delay performance and deployment cost. The other one is the Greedy algorithm [26]; the algorithm can meet the requirements of load balancing and delay performance and reduce the problem of QoS degradation caused by edge computing resource constraints.

The power grid equipment-aware delay is an important parameter to determine the network performance. Resource utilization and energy consumption are mainly considered to reduce costs, which must be based on the delay performance. The tradeoff among lower delay value, higher resource utilization, and lower energy consumption can be achieved by adjusting weight parameters. Since the weight parameters of delay, resource utilization, and energy consumption are determined by ASP in power scenarios; in this simulation, the weight parameter of delay is set to 2/3, and the weight parameters of resource utilization and energy consumption are set to 1/6.

5.2. Simulation Results Analysis. In Figure 2, it depicts the relationship between the number of power grid equipment and the average end-to-end delay of power grid equipment. It can be found that the IGA-ESP algorithm can achieve the best performance. Compared with TS algorithm and Greedy algorithm, IGA-ESP algorithm reduces the average end-to-end delay by 7.8% and 16.7% under different power grid equipment. The average delay of the three algorithms increases with the increase of the number of power grid equipment. As the number of power grid equipment increases, edge servers cannot deal with all tasks locally, and some services need to be transmitted to the cloud center through MBS, which will increase delay. Moreover, with the increase of power grid equipment, the types of requested services also increase. The edge server cannot store services that meet all requests of power grid equipment. A large number of power grid equipment need to request services from the cloud server, which results in an obvious increase in delay when the number of power grid equipment changes from 9 to 12. The average end-to-end delay of power grid equipment gradually slows down when the number of power grid equipment continues to increase. It is because, with the increase of power grid equipment, the edge servers are nearly full, and some services begin to turn to the cloud center. Even if the number of power grid equipment continues to increase, the growth trend is not obvious when the edge servers are not full. Further, it can be seen that the TS algorithm minimizes the service placement cost while meeting the power grid equipment QoS. Therefore, more services are placed in the cloud, resulting in higher delay. Greedy algorithm considers delay and load balance; its delay is close to the IGA-ESP algorithm at first; however, with the increase of the number of power grid equipment, its delay is higher and higher.

Figure 3 describes computing power per container versus average delay of power grid equipment. When the computing capacity of unit container increases gradually, the average end-to-end delay shows a decreasing trend, and the IGA-ESP algorithm always has the lowest delay. It can be seen that when the computing ability of the unit container is 0.25 GHz, the edge cloud computing ability is too weak. Even if the transmission delay of cloud computing is long, the performance is still better than that of the service placement of edge computing. Therefore, these three algorithms choose to place all services in the cloud for processing at the initial time. When the unit container computing ability begins to increase, IGA-ESP and Greedy algorithm move some services that were not significant to the edge, which is dependent on the computing power, and the average end-to-end delay was reduced by 0.125 s. As the computing ability per container continues to increase, the delay performance of the IGA-ESP algorithm is gradually better than that of the Greedy algorithm and much

Input: maximum number of iterations I
Output: optimal service placement strategy OP
getting service set
for s = 1 to S **do**
  $i = 1$
  while $i < I$ **do**
    mutating and crossover
    for all individuals in the population **do**
      calculating $D(t)$ using (10)
      calculating $RU(t)$ using (12)
      calculating $P(t)$ using (19)
    end for
    selecting and confirming the offspring
    $i = i + 1$
  end while
  estimating the relative proximity C_h^{IS} according to (26)
  selecting the best service placement strategy OP according to (27)
end for
return OP

Algorithm 1: IGA-ESP.

Table 1: Simulation parameter setting.

Parameter	Value
Storage capacity required by each service	20~80 Gb
Number of CPU cycles required by each service	10~50 gigacycles
Computing ability provided by the cloud server	50 Ghz
Data size of each service request	0.1~0.3 MB
Power of edge cloud	300 W
Operating power of edge cloud	5 W
Basic power of cloud center	1500 W
Computing power of cloud center	800 W
Number of iterations	500
Probability of crossover	0.9
Probability of mutation	0.007

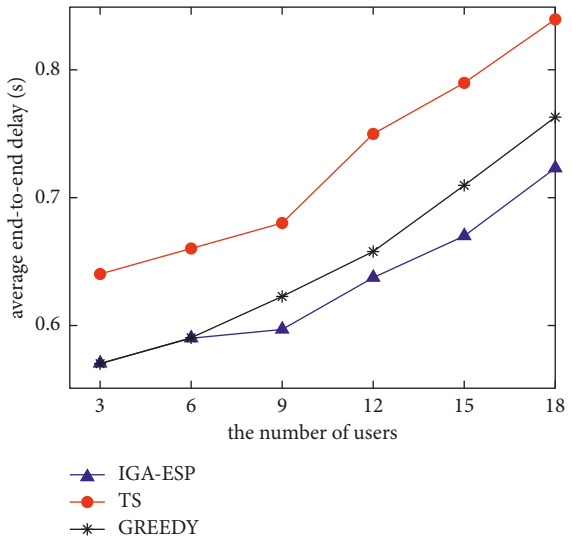

Figure 2: The number of power grid equipment versus average end-to-end delay.

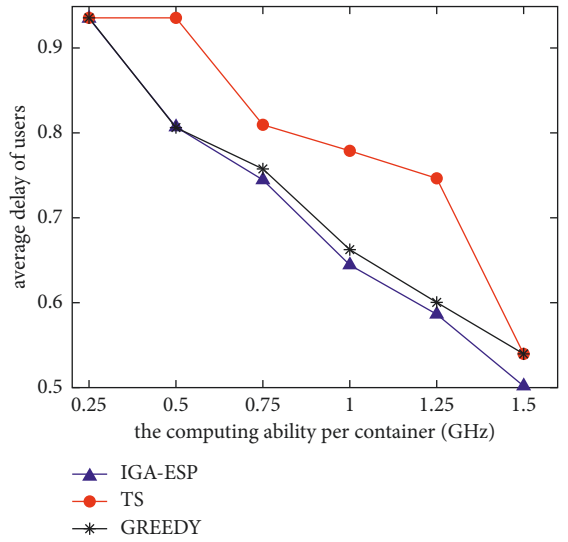

Figure 3: The computing ability per container versus average delay of power grid equipment.

lower than that of the TS algorithm. It indicates that the cost considered in TS algorithm has a great impact on the delay performance. When the computing ability of unit container increases, the performance of the two algorithms is basically consistent and worse than the IGA-ESP algorithm.

Figure 4 shows the unit container computing ability versus resource utilization. It can be seen that, with the increase of the computing ability of the unit container, the resource utilization shows an upward trend. When the computing ability of the unit container reaches 1 GHz, the resource utilization of IGA-ESP and TS algorithm reaches the saturation state under the current parameters. TS algorithm prefers to directly request services from remote cloud center; to consider the cost of service placement, its

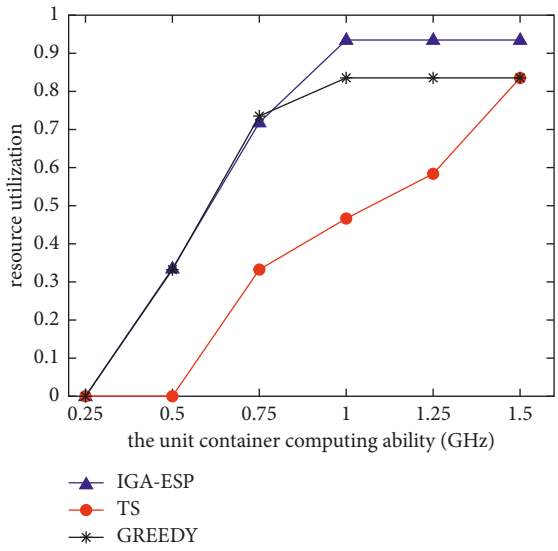

Figure 4: The unit container computing ability versus resource utilization.

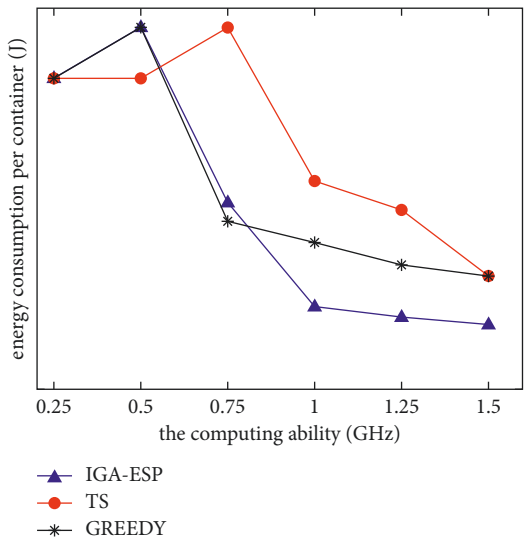

Figure 5: The computing ability versus energy consumption per container.

edge resource utilization has always maintained a low value. However, when the edge computing ability is very strong and reaches 1.5 GHz, its resource utilization suddenly reaches 0.83 together with TS algorithm. It indicates that the cost-centered TS algorithm only considers to place the service to the edge cloud when the edge performance is extremely strong and the delay is extremely low. The utilization of edge resource reflects the cost effect of the algorithm to some extent. After the edge server is established, the efficiency of ASP can be improved by deploying more services to the edge.

Figure 5 illustrates the computing ability versus energy consumption per container. As can be seen, the total energy consumption of the three algorithms has a trend of rising first and then decreasing and gradually flattening. The reason for the increase in energy consumption is the addition of the basic power consumption of edge cloud. Although the basic power consumption and operating power consumption of cloud computing are both high when only cloud computing is used to provide services at the beginning, there is no edge server, so the basic power consumption of three edge servers is 900 W. When more services are placed in the edge cloud, the total energy consumption of the system drops sharply. This is because the execution power of edge computing is far less than that of cloud center. When the execution power of edge computing is distributed among various services, the influence on the total energy consumption curve of the system becomes smaller. Another reason is that the increase of edge cloud computing ability will lead to a continuous decrease in computing ability. The total energy consumption will also decrease under the same power consumption. Finally, when a large number of services are deployed to the edge cloud, the low power characteristic of edge computing gradually flattens out. Even if more services are placed to the edge for hosting, they cannot be lower than the threshold limit of system energy consumption. It also implies that the IGA-ESP algorithm has the characteristics of low power consumption, which can reduce the total energy consumption of the system.

6. Conclusions

This paper studies the problem of edge computing service placement multiobjective optimization in Power Internet of Things system. Considering that the transmission distance of mobile cloud service is too long to guarantee the delay and the energy consumption, the edge server with limited resources is deployed at the power grid equipment edge of the network side to realize the nearby service firstly. Secondly, the service delay, resource utilization, and energy consumption are modeled. Finally, an edge service placement strategy based on SPEA2 algorithm is proposed, which can improve the overall performance of EC while optimizing multiple objectives and control the cost by reducing energy consumption. Simulation results show that, compared with the other two benchmark algorithms, the proposed strategy can effectively reduce system delay, improve resource utilization, and save energy consumption.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the Foundations of State Grid Corporation of China, under Grant no. J2021207.

References

[1] J. Liu, X. Zhao, P. Qin, S. Geng, and S. Meng, "Joint dynamic task offloading and resource scheduling for WPT enabled space-air-ground power Internet of things," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 2, pp. 660–677, 2022.

[2] J. Franco, A. Aris, B. Canberk, and A. S. Uluagac, "A survey of honeypots and honeynets for Internet of things, industrial Internet of things, and cyber-physical systems," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2351–2383, 2021.

[3] M. Rohith, A. Sunil, and Mohana, "Comparative analysis of edge computing and edge devices: key technology in IoT and computer vision applications," in *Proceedings of the 2021 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*, pp. 722–727, Bangalore, India, August 2021.

[4] X. Li, L. Huang, H. Wang, S. Bi, and Y.-J. A. Zhang, "An integrated optimization-learning framework for online combinatorial computation offloading in MEC networks," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 170–177, February 2022.

[5] S. Gong, M. Li, S. Wu, H. Cheng, and X. Yin, "Intelligent networking model at the edge of the power Internet of Things," in *Proceedings of the 2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*, pp. 841–844, Xi'an, China, October 2021.

[6] M. Babar, M. A. Jan, X. He, M. U. Tariq, S. Mastorakis, and R. Alturki, "An optimized IoT-enabled big data analytics architecture for edge-cloud computing," *IEEE Internet of Things Journal*, p. 1, 2022.

[7] G. Huang, G. Chen, J. Yi, M. Huang, and Y. Zhang, "Workload modelling method of edge computing terminals for distribution service under power Internet of things," in *Proceedings of the 2021 6th Asia Conference on Power and Electrical Engineering (ACPEE)*, pp. 430–435, Chongqing, China, April 2021.

[8] H. Wei, H. Weng, and M. Zhai, "Research on the application of 5G edge computing technology in the power Internet of things," in *Proceedings of the 2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*, pp. 600–605, Xi'an, China, October 2021.

[9] Z. Ning, P. Dong, X. Wang et al., "Distributed and dynamic service placement in pervasive edge computing networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 6, pp. 1277–1292, 2021.

[10] H. Badri, T. Bahreini, D. Grosu, and K. Yang, "Energy-aware application placement in mobile edge computing: a stochastic optimization approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 4, pp. 909–922, 1 April 2020.

[11] A. M. Maia, Y. Ghamri-Doudane, D. Vieira, and M. F. de Castro, "Dynamic service placement and load distribution in edge computing," in *Proceedings of the 2020 16th International Conference on Network and Service Management (CNSM)*, pp. 1–9, Izmir, Turkey, November 2020.

[12] H. Ding, Y. Guo, X. Li, and Y. Fang, "Beef up the edge: spectrum-aware placement of edge computing services for the Internet of things," *IEEE Transactions on Mobile Computing*, vol. 18, no. 12, pp. 2783–2795, 2019.

[13] J. Xiong, X. Chen, Q. Yang, L. Chen, and Z. Yao, "A task-oriented user selection incentive mechanism in edge-aided mobile crowdsensing," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2347–2360, 2020.

[14] Y. Tian, T. Li, J. Xiong, M. Z. A. Bhuiyan, J. Ma, and C. Peng, "A blockchain-based machine learning framework for edge services in IIoT," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 1918–1929, 2022.

[15] D. Wu, J. Li, P. He, Y. Cui, and R. Wang, "Graph-based edge-user collaborative caching with social attributes," in *Proceedings of the 2021 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, Madrid, Spain, December 2021.

[16] A. Brogi, S. Forti, C. Guerrero, and I. Lera, "Meet genetic algorithms in Monte Carlo: optimised placement of multi-service applications in the fog," in *Proceedings of the 2019 IEEE International Conference on Edge Computing (EDGE)*, pp. 13–17, Milan, Italy, July 2019.

[17] R. Mennes, B. Spinnewyn, S. Latré, and J. F. Botero, "GRECO: a distributed genetic algorithm for reliable application placement in hybrid clouds," in *Proceedings of the 2016 5th IEEE International Conference on Cloud Networking (Cloudnet)*, pp. 14–20, Pisa, Italy, October 2016.

[18] Vmware, *vSphere Single Host Management -VMware Host Client*, Vmware, Palo Alto, CA, USA, 2020, https://docs.vmware.com/cn/VMware-vSphere/5.5/vsphere-html-host-client-12-guide.pdf.

[19] X. Xu, X. Liu, Z. Xu, F. Dai, X. Zhang, and L. Qi, "Trust-oriented IoT service placement for smart cities in edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4084–4091, May 2020.

[20] P. Bellavista, A. Corradi, L. Foschini, and D. Scotece, "Differentiated service/data migration for edge services leveraging container characteristics," *IEEE Access*, vol. 7, Article ID 139746, 2019.

[21] 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, May 2019.

[22] L. Wang, L. Jiao, T. He, J. Li, and H. Bal, "Service placement for collaborative edge applications," *IEEE/ACM Transactions on Networking*, vol. 29, no. 1, pp. 34–47, 2021.

[23] K. Saadallah, V. Gustavo, W. Nannan, X. Wang, and P. Palacharla, "Service placement for real-time applications: rate-adaptation and load-balancing at the network edge," in *Proceedings of the 2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)*, pp. 207–215, New York, NY, USA, August 2020.

[24] 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.

[25] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: a survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681.

[26] C. E. F. Caetano, A. B. Lima, J. O. S. Paulino, W. C. Boaventura, I. J. S. Lopes, and E. N. Cardoso, "A conductor arrangement that overcomes the effective length issue in transmission line grounding," *Electric Power Systems Research*, vol. 46, no. 5, pp. 159–162, 2018.