Large-Scale Many-Objective Deployment Optimization of Edge Servers

Bin Cao, *Member, IEEE*, Shanshan Fan, Jianwei Zhao, Shan Tian, Zihao Zheng, Yanlong Yan, and Peng Yang

***Abstract*— The development of the Internet of Vehicles (IoV) has made transportation systems into intelligent networks. However, with the increase in vehicles, an increasing number of data need to be analyzed and processed. Roadside units (RSUs) can offload the data collected from vehicles to remote cloud servers for processing, but they cause significant network latency and are unfriendly to applications that require real-time information. Edge computing (EC) brings low service latency to users. There are many studies on computing offloading strategies for vehicles or other mobile devices to edge servers (ESs), and the deployment of ESs cannot be ignored. In this paper, the placement problem of ESs in the IoV is studied, and the six-objective ES deployment optimization model is constructed by simultaneously considering transmission delay, workload balancing, energy consumption, deployment costs, network reliability, and ES quantity. In addition, the deployment problem of ESs is optimized by a many-objective evolutionary algorithm. By comparing with the state-of-the-art methods, the effectiveness of the algorithm and model is verified.**

***Index Terms*— Internet of vehicles (IoV), many-objective, ES deployment, edge computing (EC).**

I. Introduction

RECENTLY, with the rapid increase in the number of vehicles, the development of 5G technologies and software-defined network (SDN) technologies [1]–[3] has created great opportunities and challenges to the Internet of Vehicles (IoV) [4]–[6]. The IoV realizes the network connection between vehicles, and vehicles and roadside units (RSUs) through advanced communication technologies, enabling drivers to obtain real-time traffic information, which not only provides users with comfortable driving experience but also contributes to the development of intelligent traffic systems [4].

Traditional vehicles have almost no data processing and storage capabilities [7], so running vehicles need to offload computing tasks to RSUs, and then RSUs transfer tasks to the cloud server for processing. However, cloud servers are usually deployed in remote locations, resulting in serious time consumption for data transmission between vehicles in the city center and cloud servers [8]. Therefore, edge computing (EC) was introduced. EC provides users with computing, storage, local offload and other functions, and EC servers usually exist close to users. Therefore, by deploying edge servers (ESs) on roadsides, the communication delay for processing vehicle computing tasks can be greatly reduced.

Most scholars focus on studying the user task offloading strategy under the assumption that ESs have been placed [6], [7], [9], [10] and less on the deployment of ESs. Placing ESs in proper positions can effectively reduce the access delay of RSUs and balance the workloads of ESs. In addition, the deployment of ESs is also related to the cost of server deployment [11]. Considering the possibility that the ESs may fail to respond to the service request from RSUs, the reliability of the network cannot be ignored. Therefore, it is urgent to study the ES deployment problem. Some studies formulate the ES deployment problem as an optimization model. When the number of objectives is no more than 3, it can be called a multiobjective optimization problem (MOP), and the problem with 4 or more objectives can be called a many-objective optimization problem (MaOP) [12], [13]. Most research focuses on one to three objectives. To solve the problem more comprehensively, we will simultaneously consider six objectives. Therefore, we comprehensively consider the deployment of ESs from the perspective of quality of experience (QoE) (latency, workload balancing and network reliability) and the perspective of server providers (deployment costs, number of ESs and power consumption).

In addition, some scholars use integer linear programming (ILP), clustering or multiobjective evolutionary algorithms (EAs) for ES deployment optimization problems. Jia *et al.* [14] proposed a density-based clustering placement method (DBC). Wang *et al.* [15] used mixed integer programming (MIP) to convert the ES placement problem into a single-objective optimization problem. Chin *et al.* [16] proposed the K-clustering algorithm, but the K must be known. Santoyo-Gonzalez *et al.* [17] proposed a hybrid simulated annealing placement strategy (Hybrid-SA) incorporating a tabu search. Li *et al.* [18] redefined the encoding scheme and operators of the PSO algorithm to solve the multiobjective ES deployment problem. Xu *et al.* [19] estimated the approximate

Manuscript received June 30, 2020; revised December 7, 2020; accepted January 15, 2021. Date of publication March 4, 2021; date of current version June 2, 2021. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61976242 and in part by the Opening Project of Guangdong Province Key Laboratory of Computational Science at the Sun Yat-sen University under Grant 2018002. The Associate Editor for this article was Z. Lv. *(Corresponding authors: Bin Cao; Jianwei Zhao.)*

The authors are with the State Key Laboratory of Reliability and Intelligence of Electrical Equipment, Hebei University of Technology, Tianjin 300130, China, and also with the School of Artificial Intelligence, Hebei University of Technology, Tianjin 300401, China (e-mail: caobin@scse.hebut.edu.cn; 201832102004@stu.hebut.edu.cn; 201422102003@stu.hebut.edu.cn; yangp@hebut.edu.cn).

Digital Object Identifier 10.1109/TITS.2021.3059455

number of ESs using canopy and k-medoid clustering and then used the nondominated sorting genetic algorithm III (NSGA-III) [20] to optimize the proposed model.

However, due to the delay sensitivity of RSUs, communication range limitations and the large number of deployed ESs, the real-life ES deployment problem is more complicated. The high-dimensional objective space in MaOPs makes it difficult for the traditional multiobjective EA to choose the optimal solutions among too many nondominated solutions, which creates considerable challenges in solving MaOPs. In some works, ε-dominance [21], L-optimality [22], fuzzy-dominance [23] and other new advantage relationships were proposed to replace the original Pareto-dominance to improve the optimization performance. There were also some works utilizing index-based algorithms to solve MaOPs [24]–[26]. In addition, decomposition-based algorithms [27], [28] are constantly being proposed, which decompose an MaOP into several MOPs or single-objective problems.

In practical applications, not only high-dimensional objectives but also large-scale decision space should be considered. For large-scale optimization problems (the number of variables is greater than 100), variable grouping and cooperative coevolution (CC) [29] can be used to improve optimization performance. Numerous variables are decomposed into multiple groups through the grouping strategy, and then the variables of each group are optimized using the CC framework so that the initial large-scale problem is decomposed into small-scale optimization problems. Research on grouping strategies is increasing gradually. Ma *et al.* [30] proposed the multi-objective evolutionary algorithm based on decision variable analyses (MOEA/DVA), which analyzes the basic attributes of variables and then uses a grouping algorithm to decompose the high-dimensional optimization problem into several low-dimensional subproblems. Zhang *et al.* [31] proposed the decision variable clustering-based evolutionary algorithm (LMEA) for large-scale many-objective optimization based on MOEA/DVA. Cao *et al.* proposed an improved distributed parallel cooperative coevolutionary multiobjective large-scale immune algorithm via adopting adaptive differential evolution (PCMLIA-ADE) [32] to optimize the fuzzy rough neural network and proposed a new many-objective evolutionary algorithm (MaOEA) [33] for the many-objective recommendation optimization model.

To deploy ESs more accurately, we proposed a clustering method in which the clustering radius changes with the density of RSUs to estimate the number of ESs that need to be deployed and then proposed an improved algorithm (PCMaLIA) based on PCMLIA-ADE to optimize the ES deployment model. Our contributions are mainly as follows:

1) Traditional ES deployment optimization studies have considered no greater than three objectives. This paper comprehensively considers six-objective optimization models when deploying ESs, namely, latency, workload balanced among ESs, energy consumption, deployment costs, transmission reliability and ES quantity.
2) A clustering method in which the clustering radius varies with the density of RSUs is used. To balance the workloads among ESs, the clustering radius is inversely proportional to the density of RSUs of the region. When selecting the locations of cluster heads (i.e., the locations of ESs), the workloads and density are simultaneously considered to generate more balanced clusters.
3) To accelerate the optimization efficiency, the clustering algorithm is employed to initialize the populations of a many-objective evolutionary algorithm, which is then utilized to optimize the proposed many-objective deployment model to achieve a tradeoff between conflicting objectives, and it is finally compared with three other state-of-the-art evolutionary algorithms with the same population initialization strategy.

The remainder of this paper is arranged as follows. The related work is provided in Section II. The optimization model of the deployment of the ESs and the algorithm are presented in Section III. The experimental study is given in Section IV. Finally, Section V summarizes this article.

II. Related Work

A. Many-Objective Optimization Problem (MaOP)

Multiobjective optimization problems with more than 3 objectives can be called MaOPs. In general, a minimization MaOP with n decision variables and m objective variables can be described as:

$$\begin{aligned} &\text{minimize } \mathrm{F(x)} = (f_1(x), f_2(x), \cdots, f_m(x)) \\ &\text{subject to } : \mathrm{x} \in \Omega \end{aligned} \tag{1}$$

where $\mathrm{x} = \{x_1, \cdots, x_n\} \in \Omega \subset R^n$ is the n-dimensional decision vector in the decision space Ω. $\mathrm{F(x)} = (f_1(x), f_2(x), \cdots, f_m(x)) \in R^m$ is located in the m-dimensional objective space, $f_i(x)$ represents an objective function, and it is assumed that $\mathrm{m} \geq 4$.

B. The Deployment Optimization Research of ESs

The deployment of ESs is mainly used to address the transmission delay caused by the long distance between the vehicles and the cloud servers. Therefore, the most commonly considered objective is the transmission delay between the users and the ESs. Wang *et al.* [15] optimized access delay and workload balancing of ESs using mixed integer programming methods when deploying ESs. Xu *et al.* [19] used the NSGA-III algorithm to search for a set of ES locations with the minimized ES quantity, low latency and maximized workload balancing. Liang *et al.* [34] proposed a cloudlet deployment algorithm based on k-means to reduce the network latency and the number of service instances of the edge cloud servers. Chen *et al.* [35] proposed the QoS-guaranteed efficient cloudlet deployment problem (QUICK) and divided QUICK into two subproblems according to the two objectives of minimizing the average access delay of mobile users and minimizing the number of cloudlets. An improved greedy algorithm and clustering algorithm were used to solve the quality of experience-oriented cloudlet placement (QOECP) problem and the delay bounded optimal cloudlet placement and user association (DBOCP) problem. Fan *et al.* [36], [37] proposed the cost-aware cloudlet placement strategy

(CAPABLE) in mobile EC aiming at optimizing the average end-to-end delay and deployment costs.

The deployment of ESs can reduce the cloud server workload. The computing and storage capacity of ESs are not as good as those of cloud servers. Therefore, for regions where RSUs are dense, more data will be offloaded to ESs, while for remote regions with fewer RSUs, ESs may be idle. Therefore, workload balanced among the ESs needs to be considered when deploying the ESs. Otherwise, it will cause time delays between RSUs and ESs due to queue time. Chin *et al.* [16] proposed the K-clustering algorithm and iteratively selected the placement of ESs to minimize the total traffic load. To balance the workload among the cloudlets in wireless metropolitan area networks, Jia *et al.* [14] proposed a density-based clustering placement method. Wang *et al.* [15] and Xu *et al.* [19] also considered the workload balanced among ESs.

In addition, some studies have considered the costs of deploying ESs from the perspective of server providers. The costs are mainly divided into fixed costs and dynamic costs [36]. Dynamic costs are related to the capacity and number of deployed servers. Santoyo-Gonzalez *et al.* [38] proposed a framework called EdgeON, which aimed to minimize the overall costs and number of deployed edge nodes (ENs) and maximize the capacity usage of each EN. Yin *et al.* [39] proposed a Tentacle framework for deploying ESs for online service providers, which can improve the efficiency of edge configuration and reduce the deployment costs. Zeng *et al.* [11] considered the problem of ES deployment with on-demand capacity allocation and with the same capacity and proposed a greedy boundary server capacity constraint algorithm to minimize the number and deployment costs of ESs. Mohan *et al.* [40] proposed the Anveshak framework to solve the ES placement problem by considering users' application requirements and the costs of deployment and operation.

The deployment of a large number of ESs can bring considerable power costs, so the energy consumption of ESs also needs to be considered. Badri *et al.* [41] proposed a multistage stochastic programming method to solve the ES placement problem and maximize the total QoS of the system while considering the energy budget of ESs. Li *et al.* [18] proposed an energy-aware ES layout algorithm based on the particle swarm optimization algorithm to optimize the total energy consumption of ESs. Li *et al.* [42] considered the deployment of ESs in the service area partition and used the task time and energy expenditure of each service area to measure the deployment performance of ESs. Gelenbe *et al.* [43] optimized the energy consumption and service quality of local and remote clouds.

To avoid the waste of computing resources, it is necessary to maximize the utilization rate of the server. Hu *et al.* [44] proposed a greedy algorithm for the placement of fog nodes to maximize the number of mobile nodes covered by the fog nodes. Xiang *et al.* [45] solved the problem of adaptive cloudlet placement by using GPS positioning and the k-means algorithm and considered maximizing the total number of devices covered by the cloudlet. The EdgeON framework proposed by Santoyo-Gonzalez *et al.* [38] improved the average capacity usage of edge nodes. In addition, Guerrero *et al.* [46] summarized the optimization objectives (namely, costs, latency, response time, QoS, energy consumption, load balancing, usage) considered when solving the placement problem of fog servers and proposed a decentralized optimization strategy for service configuration in fog computing. Guerrero *et al.* [47] also compared three commonly evolutionary algorithms for optimizing fog server placement: weighted sum genetic algorithm (WSGA), nondominated sorting genetic algorithm II (NSGA-II) [48], and multiobjective evolutionary algorithm based on decomposition (MOEA/D) [49].

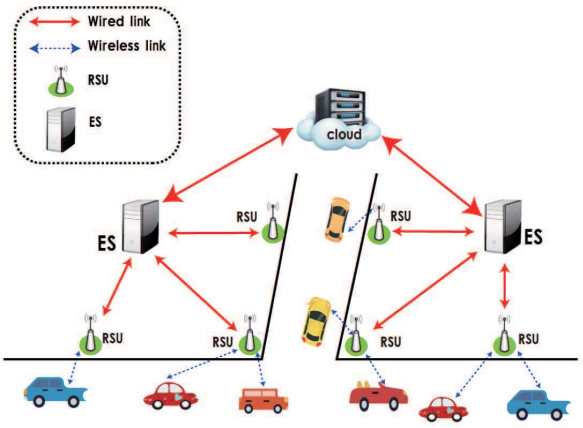

Fig. 1. The system model.

III. The Proposed Model and Algorithm

A. Problem Description

The network we considered is modeled as a fully connected undirected graph $G(R, L)$, where $R = \{r_1, r_2, \cdots, r_N\}$ ($|R| = N$) is the set of N RSUs in the network, and L is the set of links connecting these RSUs, from which K RSUs are selected to deploy ESs. The set of K ESs is $\mathrm{E} = \{e_1, e_2, \cdots, e_K\}$. When K ESs are deployed, the RSUS are divided into K subsets, namely $\mathrm{PE} = \{PE_1, PE_2, \cdots, PE_K\}$, where K subsets do not have intersections. The RSUs of the set $PE_i = \{r_{i1}, r_{i2}, \cdots, r_{iN_i}\}$ $(1 \le i \le K)$ are assigned to ES e_i, which means that these RSUs offload data to e_i. The network also includes remote cloud servers. The vehicles offload the data to the RSUs, which then transmit the data to ESs, realizing request processing locally. If ES is overloaded, it will also offload a portion of the incoming task to cloud servers. We assume that the closer to the city center, the denser the distribution of RSUs and the more workloads to be processed. It is assumed that the RSUs' task arrival rate set $\mathrm{DS} = \{\lambda_1, \lambda_2, \cdots, \lambda_j\}$. λ_j is the task arrival rate for RSU r_j. c is used to represent the number of processors in ES. The distance between ES and RSU is represented as follows [19]:

$$\mathrm{d}\left(e_i, r_j\right) = \sqrt{\left(x_i - x_j'\right)^2 + \left(y_i - y_j'\right)^2} \tag{2}$$

where (x_i, y_i) and $\left(x_j', y_j'\right)$ represent the latitude and longitude of the i-th ES and j-th RSU, respectively. Related notations are explained in detail in Table I.

TABLE I
PARAMETER SYMBOLS OF MODEL

Symbol	Meaning
$R=\{r_1,r_2,\ldots,r_N\}$	Set of all RSUs in the network.
$E=\{e_1,e_2,\ldots,e_K\}$	Set of all ESs in the network.
$D=\{\lambda_1,\lambda_2,\ldots,\lambda_N\}$	Set of the task arrive rate of RSUs.
$\mathrm{PE}=\{PE_1,PE_2,\cdots,PE_K\}$	K disjoint sets of RSUs.
$PE_i=\{r_{i1},r_{i2},\cdots,r_{iN_i}\}\,(1\le i\le K)$	RSU set assigned to ES e_i
K	Number of ESs.
N	Number of RUSs.
L	Set of links among RUSs.
e_i	The i-th ES.
r_j	The j-th RSU.
c	Number of processors in ES.
R_c	Coverage radius of ES.
(x_i,y_i)	Longitude and latitude of e_i.
$\left(x'_j,y'_j\right)$	Longitude and latitude of r_j.
T_i^j	The delay between r_j and e_i.
T_{cloud}^j	The delay between r_j and the cloud.
PE_i	Set of RSUs assigned to ES e_i.
r_{iN_i}	The RSUs assigned to ES e_i.
μ	The service rate of each processor.
λ_j	The request arrival rate of r_j.
λ_{e_i}	Total request arrival rate of ES e_i.
$\lambda_{\max}$	Maximum load threshold of an ES.
T_j^{trans}	Transmission delay of r_j.
T_{ij}^{prop}	Propagation delay between r_j and e_i.
T_{ij}^{queue}	Queuing delay of r_j when assigned to e_i.
T_{cloudj}^{prop}	Propagation delay between r_j and cloud.
$d\left(e_i,r_j\right)$	Distance between r_j and e_i.
λ_{trans}	Transmission rate.
λ_{prop}	Propagation rate.
λ_{mean}	The average workload of all ESs.
W_{e_i}	Workload of ES e_i.
E_i	Energy consumption of ES e_i.
$P_i(t)$	Power consumption of e_i at time t.
P_{idle}	Power of ES in idle state.
$P_{\max}$	Power of ES in full workload state.
β	Maximum number of processors in an ES.
ε_j^i	Whether r_j is covered by e_i.
x_j^i	Whether ES e_i is deployed on RSU r_j.
m	Price of a processor.
f_i	Fixed costs of ES e_i.

B. System Model

1) Transmission Delay Model Between RSU and ES: Network latency is mainly caused by transmission, propagation, processing and queuing. Processing delay is generally considered negligible [19]. Therefore, the total delay T_i^j of data transmission from r_j to e_i can be expressed as [19]:

$$T_i^j = T_j^{\text{trans}} + T_{ij}^{\text{prop}} + T_{ij}^{\text{queue}} \tag{3}$$

$$T_j^{\text{trans}} = \frac{\lambda_j}{\lambda_{\text{trans}}} \tag{4}$$

$$T_{ij}^{prop} = \frac{d\left(e_i,r_j\right)}{\lambda_{prop}} \tag{5}$$

where T_j^{trans} is the transmission delay, T_{ij}^{prop} is the propagation delay, T_{ij}^{queue} is the queuing delay, $d\left(e_i,r_j\right)$ is the distance between ES e_i and RSU r_j, the transmission rate is λ_{trans}, and the propagation rate is λ_{prop}.

The ES is modeled as an M/M/c queue, where ES e_i consists of c processors with fixed service rate μ [14]. It is assumed that each RSU has a stream of tasks that can be offloaded. According to the Poisson distribution, each RSU r_j offloads the task to ES randomly with the task arrival rate λ_j; then, the total request rate of ES e_i is expressed as $\lambda_{e_i}=\sum_{r_j\in PE_i}\lambda_j$. The maximum load of an ES is $\lambda_{\max}$. If the ESs overload, they will offload part of the arriving task to the cloud servers. The proportion φ_i of tasks processed by e_i is as follows [14]:

$$\varphi_i=\begin{cases}1 & \text{if } \lambda_{\max}>\lambda_{e_i}\\ \dfrac{\lambda_{\max}}{\lambda_{e_i}} & \text{otherwise}\end{cases} \tag{6}$$

The wait time in e_i is as follows:

$$T_{ij}^{queue}=\frac{C\left(c,\frac{\lambda_{e_i}\times\varphi_i}{\mu}\right)}{c\times\mu-\lambda_{e_i}\times\varphi_i}+\frac{1}{\mu} \tag{7}$$

$$C(a,b)=\frac{\frac{b^a}{a!}}{\frac{b^a}{a!}+\left(1-\frac{b}{a}\right)\sum_{k=0}^{a-1}\frac{b^k}{k!}} \tag{8}$$

Because remote cloud servers have enough resources to handle the requested tasks, the queuing time can be negligible. The delay can be calculated as follows:

$$\begin{aligned}T_{\text{cloud}}^j &= T_j^{\text{trans}}+T_{\text{cloud}j}^{\text{prop}}+\frac{1}{\mu}\\ &=\frac{\lambda_j}{\lambda_{\text{trans}}}+\frac{d\left(\text{cloud},r_j\right)}{\lambda_{\text{prop}}}+\frac{1}{\mu}\end{aligned} \tag{9}$$

We minimize the average delay from RSU to ES:

$$\min\overline{T}=\frac{\sum_{i=1}^{K}\sum_{r_j\in PE_i}\left(T_i^j\times\varphi_i+(1-\varphi_i)\,T_{\text{cloud}}^j\right)}{N} \tag{10}$$

2) ES Workload Balance Model: When some ESs are overloaded and others are idle, it will never be the optimal solution. We should allocate uniform workload to each server. The workload balanced among ESs can be measured by the standard deviation of ESs' workload [50] as follows:

$$\min f_{\text{load}}=\frac{\sqrt{\sum_{i=1}^{K}\left(\lambda_{e_i}-\lambda_{\text{mean}}\right)^2/K}}{\lambda_{\text{mean}}} \tag{11}$$

where

$$\lambda_{\text{mean}}=\frac{\sum_{i=1}^{K}\lambda_{e_i}}{K} \tag{12}$$

where λ_{mean} represents the average workload of the ESs, and λ_{e_i} represents the total request rate of ES e_i. The workload of e_i can be calculated as follows:

$$\lambda_{e_i}=\sum_{r_j\in PE_i}\lambda_j \tag{13}$$

3) Energy Consumption Model of ESs: The important factor affecting power consumption is CPU utilization. Therefore, the energy consumption of the server can be obtained indirectly according to the CPU utilization. The energy consumption of ES e_i is modeled as follows [18]:

$$E_i=\int_{t_1}^{t_2}P_i(t)dt \tag{14}$$

$$P_i(t)=\begin{cases}P_{\text{idle}}+(P_{\max}-P_{\text{idle}})\times\dfrac{\lambda_{e_i}}{\lambda_{\max}} & \text{if } \lambda_{\max}>\lambda_{e_i}\\ P_{\text{idle}}+(P_{\max}-P_{\text{idle}})\times 1 & \text{otherwise}\end{cases} \tag{15}$$

where E_i is the energy consumption of e_i, $\lambda_{\max}$ is the maximum workload threshold, and $\frac{\lambda_{e_i}}{\lambda_{\max}}$ is the utilization rate of ES e_i. The overall energy consumption with respect to all ESs can be minimized as:

$$\min E_{\text{total}} = \sum_{i=1}^{K} E_i \tag{16}$$

4) Network Reliability: It is assumed that the coverage radius of each ES is the same, denoted by R_c. If the RSU is within the coverage radius of an ES, the RSU can communicate with the ES. The event of RSU r_j covered by ES e_i can be expressed as ε_j^i:

$$\varepsilon_j^i = \begin{cases} 1 & \text{if } d\left(e_i, r_j\right) \le R_c \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

When the ES to which the RSU belongs is damaged, the RSU can select the nearest ES with which it can communicate to offload the data, thus ensuring the reliability of the task offloading. Correspondingly, the reliability objective can be expressed as follows:

$$\max f_{Rel} = \frac{\sum_{j=1}^{N}\sum_{i=1}^{K}\varepsilon_j^i}{N^2} \tag{18}$$

where f_{Rel} represents the network reliability.

5) Costs Model: The costs of deploying ESs mainly include three factors, namely, site rent, computing requirements and wire transmission distance. When server providers decide to deploy ESs, they must rent sites and then install basic equipment. The cost depends on the geographic location. Therefore, this part is regarded as a fixed cost, determined by the location of the ES. In the case of a given server price, the processor cost of an ES is considered a dynamic value depending on the number of processors. The dynamic wire transmission cost depends on the lengths of the transmission lines. Therefore, the costs of placing ESs can be expressed as follows [37]:

$$\min\ f_{\text{cost}} = \sum_{i=1}^{K}\sum_{j=1}^{N} f_i x_j^i + \sum_{i=1}^{K}\sum_{j=1}^{N} cm x_j^i + \sum_{i=1}^{K}\sum_{r_j \in PE_i} d\left(e_i, r_j\right) \tag{19}$$

where x_j^i represents whether ES e_i is deployed on RSU r_j. If so, $x_j^i = 1$ is obtained, otherwise $x_j^i = 0$. f_i is the fixed cost of ES e_i, c is the number of processors in the ES, and m is the price of each processor.

6) Deployment Model: In summary, the six-objective optimization model for deploying ESs is as follows:

$$\min F(x) = \left\{\overline{T}, f_{\text{load}}, E_{\text{total}}, -f_{\text{Rel}}, f_{\text{cost}}, K\right\} \tag{20}$$

$$\text{s.t. } PE_i \cap PE_j = \emptyset \tag{21}$$

$$\cup_{s\in K}\, PE_s = R \tag{22}$$

$$\sum_{i=1}^{K} x_j^i = 1 \tag{23}$$

$$\sum_{i=1}^{K} \varepsilon_j^i \ge 1 \quad (j = 1, 2, \cdots, N) \tag{24}$$

$$x_j^i \in \{0, 1\} \tag{25}$$

The above objectives are to minimize the average delay, minimize the standard deviation of the workload, minimize the total energy consumption, maximize the network reliability, minimize the deployment cost and minimize the number of ESs. Eqs. (21) and (22) indicate that each RSU is assigned to an ES, and two or more ESs cannot simultaneously manage an RSU. Eq. (23) indicates that only one ES can be placed at an RSU node position. Eq. (24) indicates that each RSU is covered by at least one ES.

C. The Clustering Algorithm

Obtaining a reasonable number of clusters becomes the key to designing a clustering strategy. If the number of clusters is too large, many ES deployment costs will be incurred; otherwise, the number of nodes in each cluster will be too large, and the ES workload will also increase. We design a clustering method in which the clustering radius varies with the density, that is, the larger the density is, the smaller the clustering radius to balance the workload, while in the sparse areas, the clustering radius will be set to the maximum to include more RSUs.

We calculate the number of RSUs n_{r_j} in a circle with R_c as the radius of each RSU r_j. n_{r_j} also represents the density of r_j. The clustering radius of r_j is calculated as follows:

$$R_{r_j} = \begin{cases} \dfrac{\overline{n_r}}{n_{r_j}} \times R_c & \text{if } n_{r_j} \ge \overline{n_r} \\ R_c & \text{otherwise} \end{cases} \tag{26}$$

$$\overline{n_r} = \frac{\sum_{j=0}^{N} n_{rj}}{N} \tag{27}$$

where R_{r_j} is the clustering radius, and $\overline{n_r}$ is the average density of all RSUs.

After the clustering radius of each RSU is obtained, the RSU with the highest weight is selected as the cluster head within its clustering radius to place ES. The weight as follows:

$$w_{r_j} = \frac{\sum_{r_i \in S_{r_j}} \lambda_{r_i}}{\pi R_{r_j}^2} \tag{28}$$

where λ_{r_i} is the workload of r_i, S_{r_j} is the set of RSUs in the circle where r_j takes R_{r_j} as the radius, and w_{r_j} is the weight determined by the workload and area of the circle with radius R_{r_j}, which considers the density and workload to generate more balanced clusters.

After the cluster head r_c is selected, within the circle with R_{r_c} as the radius, the RSUs closest to r_c and whose weight is not higher than r_c are successively added to the cluster with cluster head r_c until the average density is reached. After adding, the clustered RSUs are deleted. Then, a new round of cluster head selection proceeds until all RSUs have been allocated.

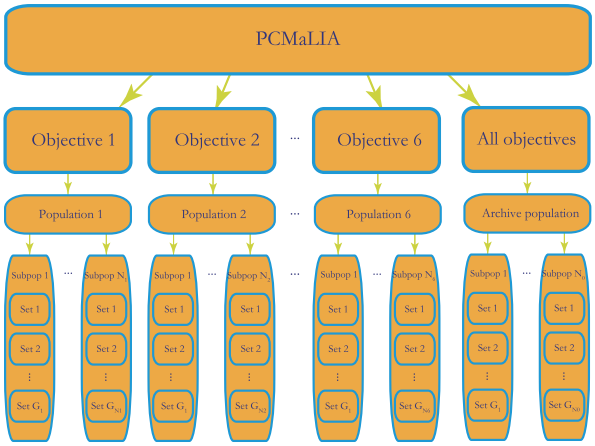

Fig. 2. Parallel Structure.

D. The Improved Many-Objective Optimization Method

After initializing the population using the above method, the estimated number K and positions of ESs are obtained. Then the improved algorithm PCMaLIA based on PCMLIA-ADE [32] is utilized to optimize the number and positions of ESs. PCMaLIA has a similar parallel structure to PCMLIA-ADE, which forms a specific population for each objective to better explore the solution space, and an extra archive population is utilized to simultaneously optimize all objectives. Next, based on variable grouping, each population can be divided into several subpopulations. Finally, individuals in each subpopulation can be further divided. Correspondingly, a three-level parallel structure is constructed based on the objective, variable, and subpopulation decompositions.

Although PCMLIA-ADE is proposed for an MOP with two objectives, it utilizes a population to explore each objective and an extra archive population to explore all objectives simultaneously, which has great potential to address MaOPs. So we proposed PCMaLIA on this basis. The parallel structure of PCMaLIA is shown in Fig. 2. The steps to optimize ES deployment using PCMaLIA are as follows.

1) Individual Encoding: Since the locations of RSUs are the potential positions of ESs and the number of RSUs is N, the chromosome is represented by an N-bit real encoding. If the value of the i-th bit is equal to or greater than the threshold (the midvalue), then the i-th RSU is selected as the position to place an ES; otherwise, it is not selected.

2) Grouping Strategy: As there are numerous variables, the ES deployment problem can be considered a large-scale MaOP. Therefore, PCMaLIA separates the variables into several groups to divide the original large-scale MaOP into several small-scale problems. In the large-scale many-objective ES deployment problem, each variable represents an RSU, and the properties of all variables can be regarded the same. Based on this prior knowledge, in PCMaLIA, all variables are randomly and uniformly divided into four groups.

3) Population Initialization: To accelerate the optimization process, the clustering algorithm described in Section III-C is first employed to cluster all RSUs to several groups. Then, similar to [19], in initializing the population, the clustering result is utilized. Specifically, each individual selects one RSU in each cluster to deploy an ES. Additionally, to improve the population diversity, except for the first individual, some clusters initialize no or more than one ES.

Fig. 3. The distribution of RSUs before clustering.

IV. Experiment Evaluation

In this section, we use the proposed algorithm to optimize the proposed many-objective optimization model. To verify the performance of the proposed algorithm, we compared our proposed method with several other algorithms. The experimental results show that the proposed method is effective and efficient.

A. Experimental Setup

We conduct experiments on the RSU distribution in the Binhai New Area of Tianjin. The region we considered in the experiments contains about six thousand RSUs, and we use the k-means to select 1,500 RSUs. To verify the performance of the proposed method, a visual comparison is made between the proposed algorithms and two methods (k-mediods and random selection) commonly used to solve ES deployment. Then, the optimization results of the DPCCMOLSEA-MP-I [51], NSGA-III, PCMaLIA and MOEA/DVA algorithms are evaluated according to the HyperVolume (HV) [52] indicator.

In the experiments, we set μ, $\lambda_{\max}$, P_{idle}, P_{max} and m to 10, 45, 300, 495 and 50, respectively. Our clustering algorithm divides the RSUs into 47 clusters, so we set K to 47. Random selection randomly selects K RSUs to place ESs. The k-medoids algorithm divides the RSUs into K clusters and then selects the RSU closest to the centroid of each cluster as the position of the ES for each cluster. For PCMaLIA, we also selected a solution where K is 47.

Four optimization algorithms are used to solve the ES deployment optimization problem. Each algorithm is run 20 times, the evaluation times of the objective function are set to 10^6, the population size is 126, while there are 128 individuals in NSGA-III as the population size should be divisible by 4. In addition, MOEA/DVA and NSGA-III use SBX and

(a) Clustering described in Section III-C (b) PCMaLIA (c) K-mediods (d) Random selection

Fig. 4. The distribution of ESs.

TABLE II
HV INDICATOR VALUES

	0%	4%	8%	12%	16%	20%	24%	28%	32%	36%	40%	44%	48%
MOEA/DVA	5.94E-05	2.70E-04	4.24E-04	5.19E-04	8.39E-04	1.18E-03	1.35E-03	1.77E-03	2.17E-03	2.32E-03	2.67E-03	2.84E-03	2.95E-03
NSGA-III	6.04E-05	8.19E-04	1.80E-03	2.66E-03	3.16E-03	3.48E-03	3.60E-03	3.64E-03	3.72E-03	3.77E-03	3.79E-03	3.82E-03	3.82E-03
DPCCMOLSEA-MP-I	6.58E-05	1.72E-03	2.64E-03	3.05E-03	3.65E-03	3.92E-03	4.41E-03	6.20E-03	6.35E-03	6.68E-03	6.76E-03	6.99E-03	7.76E-03
PCMaLIA	1.47E-04	2.04E-03	2.97E-03	3.76E-03	4.40E-03	5.15E-03	5.81E-03	6.97E-03	7.39E-03	7.80E-03	8.10E-03	8.38E-03	8.70E-03
	52%	**56%**	**60%**	**64%**	**68%**	**72%**	**76%**	**80%**	**84%**	**88%**	**92%**	**96%**	**100%**
MOEA/DVA	3.11E-03	3.21E-03	3.29E-03	3.36E-03	3.44E-03	3.46E-03	3.52E-03	3.59E-03	3.58E-03	3.63E-03	3.65E-03	3.69E-03	3.70E-03
NSGA-III	3.80E-03	3.81E-03	3.82E-03	3.83E-03	3.86E-03	3.87E-03	3.89E-03	3.91E-03	3.92E-03	3.92E-03	3.93E-03	3.95E-03	3.96E-03
DPCCMOLSEA-MP-I	8.12E-03	8.29E-03	8.44E-03	8.84E-03	8.96E-03	9.21E-03	9.26E-03	9.34E-03	9.45E-03	9.52E-03	9.59E-03	9.61E-03	9.64E-03
PCMaLIA	8.92E-03	9.27E-03	9.42E-03	9.54E-03	9.68E-03	9.75E-03	9.85E-03	1.00E-02	1.02E-02	1.03E-02	1.03E-02	1.04E-02	1.04E-02

polynomial mutations. The SBX crossover probability and distribution index are 1 and 20, respectively, and the polynomial mutation probability and distribution index are $1.0/N$ and 20, respectively. The experimental platform used for simulation is the Tianhe-2 supercomputer.

B. Performance Measurement

To compare the performance of the three optimization algorithms, we use the HV indicator to measure the convergence and diversity of the solution set. The HV calculation equation is as follows:

$$HV(Z, pop) = \text{volume}\left(\bigcup_{p\in pop}\prod_{i=1}^{m}[p_i, z_i]\right) \quad (29)$$

where pop is the optimal solution set obtained by the optimization algorithm, and Z is the reference point of the objective space. All the experiments are independently tested 20 times, and the average value of the 20 tests is taken as the final value. The higher the HV value is, the better the algorithm performance.

C. Experimental Analysis

In the visualization experiments, RSUs are marked as green dots and ESs are marked as red dots. Fig. 3 shows the initial RSU distribution. Fig. 4a shows the result of placing ESs using the clustering algorithm described in Section III-C, and Fig. 4b shows the ES placement results optimized by PCMaLIA. Fig. 4c and Fig. 4d are the results of placing ESs with k-mediods and random selection, respectively. The ES positions obtained by k-mediods are relatively uniform, while the proposed method will place more ESs in dense areas of RSUs to balance the ES workload. Among them, the clustering algorithm mainly considers the density factor, while PCMaLIA considers 6 objectives. In addition, the HV values of PCMaLIA, MOEA/DVA, NSGA-III and DPCCMOLSEA-MP-I are 0.0104, 0.003705, 0.003965 and 0.009644, respectively. The comparison of HV values are shown in Fig. 5. The corresponding HV indicator values are listed in Table II. It can be concluded that PCMaLIA is superior to the other three algorithms.

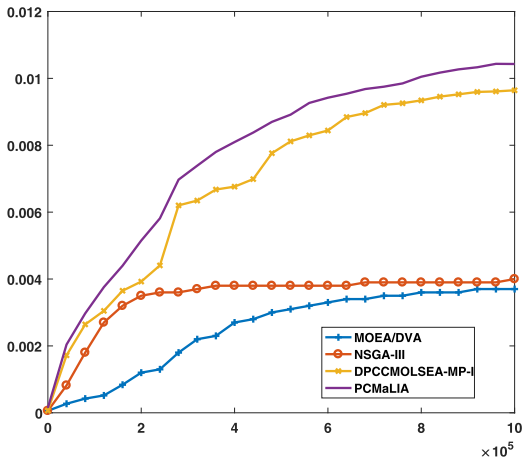

Fig. 5. The evolutionary curves of HV indicator values.

V. CONCLUSION

We studied the deployment of ESs in the IoV from the perspective of users and server providers. We considered the impact of the transmission delay between RSUs and ESs and the unreliability of RSU data transmission due to ES damage, and we attempted to avoid the unbalanced workload

distribution among ESs. The deployment costs and quantity of the ESs were also considered. In addition, we described a clustering algorithm in which the clustering radius varies with the density. Finally, we proposed an improved many-objective optimization algorithm, PCMaLIA, to optimize the locations and number of ESs. In future work, when deploying ESs, we consider that when an ES is overloaded, tasks can be offloaded to other ESs according to the length of the queue. In addition, RSUs may be susceptible to network attacks and have the risk of leaking user privacy, so research on RSU security protection will be considered.

ACKNOWLEDGMENT

The authors would like to thank Zhaopeng Song for his contributions to this article.

REFERENCES

[1] X. Ge, Z. Li, and S. Li, "5G software defined vehicular networks," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 87–93, 2017.

[2] L. Zhao, W. Sun, Y. Shi, and J. Liu, "Optimal placement of cloudlets for access delay minimization in SDN-based Internet of Things networks," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1334–1344, Apr. 2018.

[3] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani, "Software-defined networking for RSU clouds in support of the Internet of vehicles," *IEEE Internet Things J.*, vol. 2, no. 2, pp. 133–144, Apr. 2015.

[4] J. Cheng, J. Cheng, M. Zhou, F. Liu, S. Gao, and C. Liu, "Routing in Internet of vehicles: A review," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 5, pp. 2339–2352, Oct. 2015.

[5] B. Cao, X. Chen, Z. Lv, R. Li, and S. Fan, "Optimization of classified municipal waste collection based on the Internet of connected vehicles," *IEEE Trans. Intell. Transp. Syst.*, early access, Apr. 3, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9056517

[6] X. Xu, R. Gu, F. Dai, L. Qi, and S. Wan, "Multi-objective computation offloading for Internet of vehicles in cloud-edge computing," *Wireless Netw.*, vol. 26, no. 3, pp. 1611–1629, Apr. 2020.

[7] X. Xu *et al.*, "An edge computing-enabled computation offloading method with privacy preservation for Internet of connected vehicles," *Future Gener. Comput. Syst.*, vol. 96, pp. 89–100, Jul. 2019.

[8] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.

[9] S. Wan, X. Li, Y. Xue, W. Lin, and X. Xu, "Efficient computation offloading for Internet of vehicles in edge computing-assisted 5G networks," *J. Supercomput.*, vol. 76, no. 4, pp. 2518–2547, Apr. 2020.

[10] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784–1797, Mar. 2018.

[11] F. Zeng, Y. Ren, X. Deng, and W. Li, "Cost-effective edge server placement in wireless metropolitan area networks," *Sensors*, vol. 19, no. 1, p. 32, Dec. 2018.

[12] H. Wang, L. Jiao, and X. Yao, "Two_Arch2: An improved two-archive algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 524–541, Aug. 2015.

[13] B. Cao, W. Dong, Z. Lv, Y. Gu, S. Singh, and P. Kumar, "Hybrid microgrid many-objective sizing optimization with fuzzy decision," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 11, pp. 2702–2710, Nov. 2020.

[14] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Trans. Cloud Comput.*, vol. 5, no. 4, pp. 725–737, Oct. 2017.

[15] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *J. Parallel Distrib. Comput.*, vol. 127, pp. 160–168, May 2019.

[16] T.-L. Chin, Y.-S. Chen, and K.-Y. Lyu, "Queuing model based edge placement for work offloading in mobile cloud networks," *IEEE Access*, vol. 8, pp. 47295–47303, 2020.

[17] A. Santoyo-González and C. Cervelló-Pastor, "Latency-aware cost optimization of the service infrastructure placement in 5G networks," *J. Netw. Comput. Appl.*, vol. 114, pp. 29–37, Jul. 2018.

[18] Y. Li and S. Wang, "An energy-aware edge server placement algorithm in mobile edge computing," in *Proc. IEEE Int. Conf. Edge Comput. (EDGE)*, Jul. 2018, pp. 66–73.

[19] X. Xu *et al.*, "Edge server quantification and placement for offloading social media services in industrial cognitive IoV," *IEEE Trans. Ind. Informat.*, vol. 17, no. 4, pp. 2910–2918, Apr. 2021.

[20] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.

[21] D. Hadka and P. Reed, "Borg: An auto-adaptive many-objective evolutionary computing framework," *Evol. Comput.*, vol. 21, no. 2, p. 231, 2013.

[22] X. Zou, Y. Chen, M. Liu, and L. Kang, "A new evolutionary algorithm for solving many-objective optimization problems," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 38, no. 5, pp. 1402–1412, Oct. 2008.

[23] G. Wang and H. Jiang, "Fuzzy-dominance and its application in evolutionary many objective optimization," in *Proc. Int. Conf. Comput. Intell. Secur. Workshops (CISW)*, Dec. 2007, pp. 195–198.

[24] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Comput.*, vol. 19, no. 1, pp. 45–76, Mar. 2011.

[25] Y. Tian, R. Cheng, X. Zhang, F. Cheng, and Y. Jin, "An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 609–622, Aug. 2018.

[26] E. M. Lopez and C. A. C. Coello, "IGD+−EMOA: A multi-objective evolutionary algorithm based on IGD+," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2016, pp. 999–1006.

[27] M. Asafuddoula, T. Ray, and R. Sarker, "A decomposition-based evolutionary algorithm for many objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 3, pp. 445–460, Jun. 2015.

[28] Y. Zhou, Y. Xiang, Z. Chen, J. He, and J. Wang, "A scalar projection and angle-based evolutionary algorithm for many-objective optimization problems," *IEEE Trans. Cybern.*, vol. 49, no. 6, pp. 2073–2084, Jun. 2019.

[29] M. A. Potter and K. A. D. Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evol. Comput.*, vol. 8, no. 1, pp. 1–29, Mar. 2000, doi: 10.1162/106365600568086.

[30] X. Ma *et al.*, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 275–298, Apr. 2016.

[31] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 97–112, Feb. 2018.

[32] B. Cao, J. Zhao, Z. Lv, Y. Gu, P. Yang, and S. K. Halgamuge, "Multiobjective evolution of fuzzy rough neural network via distributed parallelism for stock prediction," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 5, pp. 939–952, Feb. 2020.

[33] B. Cao, J. Zhao, Z. Lv, and P. Yang, "Diversified personalized recommendation optimization based on mobile data," *IEEE Trans. Intell. Transp. Syst.*, early access, Dec. 15, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9295358, doi: 10.1109/TITS.2020.3040909.

[34] T.-Y. Liang and Y.-J. Li, "A location-aware service deployment algorithm based on K-Means for cloudlets," *Mobile Inf. Syst.*, vol. 2017, pp. 1–10, Jan. 2017, doi: 10.1155/2017/8342859.

[35] L. Chen, J. Wu, G. Zhou, and L. Ma, "QUICK: QoS-guaranteed efficient cloudlet placement in wireless metropolitan area networks," *J. Supercomput.*, vol. 74, no. 8, pp. 4037–4059, Aug. 2018.

[36] Q. Fan and N. Ansari, "Cost aware cloudlet placement for big data processing at the edge," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[37] Q. Fan and N. Ansari, "On cost aware cloudlet placement for mobile edge computing," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 4, pp. 926–937, Jul. 2019.

[38] A. Santoyo-Gonzalez and C. Cervello-Pastor, "Network-aware placement optimization for edge computing infrastructure under 5G," *IEEE Access*, vol. 8, pp. 56015–56028, 2020.

[39] H. Yin *et al.*, "Edge provisioning with flexible server placement," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 4, pp. 1031–1045, Apr. 2017.

[40] N. Mohan, A. Zavodovski, P. Zhou, and J. Kangasharju, "Anveshak: Placing edge servers in the wild," in *Proc. Workshop Mobile Edge Commun. (MECOMM)*, New York, NY, USA, Aug. 2018, pp. 7–12, doi: 10.1145/3229556.3229560.

[41] H. Badri, T. Bahreini, D. Grosu, and K. Yang, "Energy-aware application placement in mobile edge computing: A stochastic optimization approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 4, pp. 909–922, Apr. 2020.
[42] B. Li, P. Hou, K. Wang, Z. Peng, S. Jin, and L. Niu, "Deployment of edge servers in 5G cellular networks," *Trans. Emerg. Telecommun. Technol.*, to be published. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3937, doi: 10.1002/ett.3937.
[43] E. Gelenbe, R. Lent, and M. Douratsos, "Choosing a local or remote cloud," in *Proc. 2nd Symp. Netw. Cloud Comput. Appl.*, Dec. 2012, pp. 25–30.
[44] C. Hu, X. Shu, X. Yan, D. Zeng, W. Gong, and L. Wang, "Inline wireless mobile sensors and fog nodes placement for leakage detection in water distribution systems," *Softw. Pract. Exper.*, vol. 50, no. 7, pp. 1152–1167, Jul. 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.2631
[45] H. Xiang *et al.*, "An adaptive cloudlet placement method for mobile applications over GPS big data," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2016, pp. 1–6.
[46] C. Guerrero, I. Lera, and C. Juiz, "A lightweight decentralized service placement policy for performance optimization in fog computing," *J. Ambient Intell. Humanized Comput.*, vol. 10, no. 6, pp. 2435–2452, Jun. 2019.
[47] C. Guerrero, I. Lera, and C. Juiz, "Evaluation and efficiency comparison of evolutionary algorithms for service placement optimization in fog architectures," *Future Gener. Comput. Syst.*, vol. 97, pp. 131–144, Aug. 2019.
[48] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[49] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.
[50] L. Wang, X. Fu, J. Fang, H. Wang, and M. Fei, "Optimal node placement in industrial wireless sensor networks using adaptive mutation probability binary particle swarm optimization algorithm," in *Proc. 7th Int. Conf. Natural Comput. (ICNC)*, Shanghai, China, Jul. 2011, pp. 2199–2203.
[51] B. Cao, J. Zhao, P. Yang, P. Yang, X. Liu, and Y. Zhang, "3-D deployment optimization for heterogeneous wireless directional sensor networks on smart city," *IEEE Trans. Ind. Informat.*, vol. 15, no. 3, pp. 1798–1808, Mar. 2019.
[52] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Trans. Evol. Comput.*, vol. 10, no. 1, pp. 29–38, Feb. 2006.

Bin Cao (Member, IEEE) received the Ph.D. degree in computer application technology from Jilin University in 2012.

From 2012 to 2014, he was a Postdoctoral Researcher with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He is currently a Professor with the Hebei University of Technology, Tianjin, China. His research interests include intelligent computation with its applications to cyber-physical systems, big data, and graphics and visual media; high performance computing; and cloud computing.

Shanshan Fan received the bachelor's degree in computer science and technology from the Hebei University of Technology, Tianjin, China, in 2018, where she is currently pursuing the master's degree with the School of Artificial Intelligence.

Her research interests include artificial intelligence with its applications to cyber-physical systems, the Internet of Things, edge computing, and cloud computing.

Jianwei Zhao received the master's degree in computer science and technology from the Hebei University of Technology, Tianjin, China, in 2018, where he is currently pursuing the Ph.D. degree.

His research interests include intelligent computation with its applications to cyber-physical systems, pattern recognition, 5G, the Internet of Things, high performance computing, and cloud computing.

Shan Tian received the bachelor's degree in vehicle engineering from the Shandong University of Science and Technology, Qingdao, China, in 2018. He is currently pursuing the master's degree with the School of Artificial Intelligence, Hebei University of Technology, Tianjin, China.

His research interests include artificial intelligence with its applications to cyber-physical systems, edge computing, and cloud computing.

Zihao Zheng received the bachelor's degree in safety engineering from the Hebei University of Technology, Tianjin, China, in 2020, where he is currently pursuing the master's degree with the School of Artificial Intelligence.

His research interests include artificial intelligence with its applications to cyber-physical systems, edge computing, and cloud computing.

Yanlong Yan received the bachelor's degree in automation from Liaoning Shihua University, Fushun, China, in 2020. He is currently pursuing the master's degree with the School of Artificial Intelligence, Hebei University of Technology, Tianjin, China.

His research interests include artificial intelligence with its applications to cyber-physical systems, edge computing, and cloud computing.

Peng Yang received the Ph.D. degree in electric machines and electric apparatus from the Hebei University of Technology, Tianjin, China, in 2001.

Since 2005, he has been with the University of Munich as a Visiting Scholar. He is currently a Professor with the School of Artificial Intelligence, Hebei University of Technology. His research interests include computational intelligence, intelligent control techniques, and prosthetics. He was a recipient of various awards and honors, including the Natural Science Award of Hebei Province, the Science and Technology Progress Award of Hebei Province, and the Natural Science Award of Chongqing.