An Application Deployment Approach for City IoT Applications in Resource-constrained Edge Computing Environments

Jialei Liu

Anyang Normal University <https://orcid.org/0000-0002-9537-9740>

Quanzhen Huang (✉ huang2004_susu@haue.edu.cn)

School of Electrical Information Engineering, Henan Institute of Engineering, China

<https://orcid.org/0000-0001-6425-4665>

Research

Keywords: Edge cloud, city IoT application, energy consumption, resource wastage, resource load imbalance, bandwidth resource consumption, execution delay

Posted Date: October 16th, 2020

DOI: <https://doi.org/10.21203/rs.3.rs-92258/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

An Application Deployment Approach for City IoT Applications in Resource-Constrained Edge Computing Environments

Jialei Liu¹, Quanzhen Huang^{2*}

¹ School of Software Engineering, Anyang Normal University, China

² School of Electrical Information Engineering, Henan Institute of Engineering, China

{jialeiliu@aynu.edu.cn, huang2004_susu@haue.edu.cn}

Abstract: With the development and utilization of more and more city Internet of Things (IoT) applications with high resource requirements, how to reduce the consumption of energy, processor resources and bandwidth resources in resource-constrained edge clouds while ensuring the execution delay of these applications is an urgent problem to be solved. Therefore, an optimal energy-bandwidth tradeoff deployment approach for city IoT application is proposed for resource-constrained edge clouds. In this approach, the city IoT applications are first divided into multiple collaborative tasks and offload to edge clouds. Secondly, a joint optimization model including energy consumption, resource wastage, resource load imbalance and bandwidth resource consumption is established for the task offloading scheme. Thirdly, the city IoT application deployment problem is optimized under the constraints of resource and execution delay. Finally, a comprehensive simulation test is conducted to analyze the deployment approaches from the aspects of performance and effectiveness. The experimental results show that our deployment approach is superior to other related approaches.

Keywords: Edge cloud; city IoT application; energy consumption; resource wastage; resource load imbalance; bandwidth resource consumption; execution delay

1. Introduction

With the rapid development of mobile networks and the wide application of Internet of Things (IoT) in intelligent manufacturing, smart home, smart transportation and other fields, IoT devices such as smartphones, tablets, drones and wearable devices are becoming more and more popular. In 2018, Cisco predicted that the number of global IoT devices will grow from 8.6 billion in 2017 to 12.3 billion in 2022, among which, more than 422 million global IoT devices and connections will adopt 5G [1]. However, these IoT devices are often limited by computing power, storage space, energy capacity, environmental awareness, and other resources, and complex computing tasks such as optical character recognition, face recognition, and augmented reality are locally inefficient [2]. In addition, the diversity of latency-sensitive and resource-intensive city IoT applications often results in a variety of different computing and communication costs [3].

In order to alleviate the limitation of mobile computing power, mobile edge computing has emerged as a new computing paradigm that utilizes resources near IoT devices to provide timely services in conjunction with cloud servers. In mobile edge

computing, resource-intensive and latency-sensitive city IoT application from the edge of the mobile network will usually be broken down into a series of tasks, which can be independent design, development, deployment and operational [4]. These tasks coordinating with each other are offloaded onto the edge clouds or the remote cloud [5], and turn over a cluster consisting of multiple virtual machines or containers for collaborative processing. Furthermore, the IoT devices reduce the energy consumption and speed up the calculation process, and also make the run on IoT devices emerging city IoT applications possible. In the process of computing offloading, multiple tasks split by city IoT applications need to select the best computing node for collaboratively processing by multiple virtual machines or containers. Meanwhile, given the consumption of energy, processor resources, and bandwidth resources by these tasks with the required execution delay, it has become an urgent problem that which deployment approach of city IoT application should be adopted to optimally allocate these tasks to edge clouds.

Compared with the resources in the remote cloud, the resources in the edge cloud are usually: (1) resource-constrained -- limited computing resources, because the IoT devices have small processors and limited power budget; (2) Heterogeneity -- processors with different architectures; (3) Dynamics -- their workload changes and city IoT applications compete for limited resources. These factors cause various resources in the edge cloud (such as CPU, storage, and bandwidth) to be scarcer and more expensive than those in the remote cloud [6]. Therefore, the cost of renting edge servers becomes a major challenge for application operators as they rent lots of edge servers and deploy massive tasks to provide a better user experience. According to RightScale's report [7], 26% of enterprises with more than 1,000 employees spend more than \$6 million annually on public clouds, but 35% of this cloud spending is wasted -- users may always overestimate resource consumption. In addition, application operators always have specific requirements for their applications, which are to maintain some key performance indicators [8], such as average response time. Therefore, it is necessary to study how to find the optimal deployment scheme for a city IoT application while ensuring the actual performance indicators such as execution delay, energy consumption, processor resource consumption and bandwidth resource consumption of the city IoT application.

To solve the above problem, this paper studies the optimal task deployment problem for city IoT applications deployed to edge clouds. The joint optimization of the energy consumption, processor resource consumption and bandwidth resource consumption of edge clouds under the condition of specified execution delay is analyzed and discussed via considering the impact of different task deployment schemes on the execution delay and the system utility of edge clouds. The main contributions are as follows:

- The optimal deployment of city IoT applications is formulated as three sub-problems of energy consumption, processor resource consumption and bandwidth resource consumption, and solved using models such as energy consumption, resource wastage, resource load imbalance and bandwidth resource consumption under the constraints of resource and execution delay.

- On the basis of the above model, a joint optimization model is first established to measure the consumption of energy, processor resources and bandwidth resources of city IoT application deployment scheme. Then, a Differential Evolution [9] based Optimization Deployment of City IoT Application (DEODCA) approach is introduced to minimize the consumption of energy, processor resources and bandwidth resources in the edge clouds under the constraints of resource and execution delay.
- We establish a city mobile edge computing system model to comprehensively evaluate the performance and effectiveness of our proposed deployment approach for city IoT applications.

The rest of this paper is organized as follows. Section 2 introduces the related work. Section 3 introduces the system model. Section 4 proposes the deployment model of city IoT application. Section 5 proposes the design and implementation of city IoT application deployment approach. Section 6 provides the performance evaluation including experimental parameter settings, comparison results, and parameter study. Finally, we conclude this paper with the future work recommendations in Section 7.

2. Related Work

In recent years, many scholars have carried out researches on the IoT application deployment in mobile edge computing, and made many research achievements. For example, Tong et al. [10] designed edge clouds as a tree hierarchy of regionally distributed edge servers to improve cloud resource utilization efficiency, and proposed a workload allocation algorithm to assign mobile applications to some edge servers. Tan et al. [11] proposed a general model for how to distribute and schedule jobs to minimize the weighted response time of all jobs, and then proposed an online distribution and scheduling algorithm with extensibility in the speed enhancement model. Meng et al. [12] jointly considered the management of network bandwidth and computing resources to meet the maximum value of deadlines, and then proposed an online algorithm to greedily schedule newly arrived tasks to meet the new deadline. Wu et al. [13] proposed a heuristic algorithm for service selection in mobile edge computing systems to solve how to allocate composite services of service requests to edge servers and cloud servers to reduce the execution delay. Chen et al. [14] proposed a data-intensive service edge deployment scheme based on genetic algorithm to minimize the response time of data-intensive service deployment under storage constraints and load balancing. Bahreini et al. [15] proposed a mixed integer linear programming formulation to solve the multi-component application placement problem for dynamic user location and network capability, thus minimizing the time cost incurred when running the application. Considering that multiple mobile user devices simultaneously offload their tasks to the mobile edge clouds, Chen et al. [16] effectively solved the multi-user and multi-task computing offloading problem of green mobile edge computing by introducing centralized and distributed greedy maximum scheduling algorithm. Dai et al. [17] jointly optimized user association and computational offload by developing a computational offloading algorithm that considers computational resource allocation and transmission power allocation to

minimize the overall energy consumption. Chen et al. [18] proposed an energy efficient dynamic offloading algorithm to minimize the energy consumption of task offloading while guaranteeing the average queue length. The above researches [10-18] are basically carried out by taking a single indicator as the optimization objective. It is difficult to achieve the purpose of reducing the system utility and network resources of the mobile edge computing environment with ensuring the execution delay.

At present, there are also massive researches on multi-objective optimization deployment of IoT applications. For example, Chen et al. [19] proposed a distributed computing offloading algorithm based on game theory to solve the problem of multi-user computing offloading for multi-channel wireless interference or mobile edge computing in competitive environment by modeling the communication cost and computing cost during offloading. Chen, et al. [20] formulated a data-intensive applications deployment strategies to minimize the delay of the mobile devices and minimize the monetary cost of application service providers under the condition of data transfer among mobile devices, edge servers, and clouds, user movement, and changing load balancing conditions. Deng et al. [7] proposed an approach to generate an appropriate deployment scheme at the minimum cost under the on-demand billing model in the case of the resource constraints of edge servers, the business logic of applications and the average response time of applications. Hu et al. [21] proposed an approximately optimal service allocation strategy that meets the constraints of edge server resources and bandwidth to find the tradeoff between average network delay and load balancing. Wu et al. [22] formalized the mixed task assignment problem of mobile edge computing as a multi-objective optimization problem, and then proposed an efficient offloading framework with intelligent decision-making ability to jointly minimize system utility and bandwidth allocation of each mobile device. Pallewatta et al. [23] proposed an IoT application layout strategy that utilizes micro-service independent deployability and scalability to minimize latency and network utilization. Zhang et al. [24] proposed an adaptive task offloading algorithm to optimize and balance the energy consumption of terminal devices and the overall task execution time. Goudarzi et al. [25] proposed a parallel IoT batch application layout decision approach based on MEME algorithm to minimize the execution time and energy consumption of IoT applications in a computing environment with multiple IoT devices, multiple Fog/Edge servers and cloud servers. Cheng et al. [26] studied the task allocation algorithm in the mobile edge computing system of data sharing, and then proposed three algorithms to reduce the delay and energy consumption needed to process global tasks and separable tasks respectively. Zhang et al. [27] proposed a low-complexity solution scheme based on the improved Newton method and the concept of calculating offloading priority to minimize task execution delay and equipment energy consumption by studying the multi-objective resource allocation problem of multi-user mobile edge computing system. Tao et al. [28] proposed a task offloading algorithm to solve the problem of minimized energy consumption under the constraints of resource capacity and delay, and meet mobile users' demands for low-energy and high-performance tasks. Guo et al. [29] studied the optimal allocation scheme of energy saving resources for multi-user mobile edge computing systems

with inelastic computing tasks and non-negligible task execution time, and proposed a low complexity algorithm to solve the suboptimal solution by combining the optimization problem with the three-stage pipeline scheduling problem and using Johnson algorithm and convex optimization technology. The above studies [7, 19- 29] all solve the corresponding joint optimization problem through the corresponding calculation offloading algorithm. However, their disadvantage lies in that the designed cost does not take into account the bandwidth resource consumption between tasks deployed on different edge servers and the energy consumption of edge clouds.

Based on the above analysis, this paper will study the optimal deployment of city IoT applications in resource-constrained edge clouds. That is, when the multiple collaborative tasks that make up the city IoT applications are offloaded to the resource-constrained edge clouds, how to minimize consumption of energy, processor resources, and bandwidth resources in the edge clouds under the constraints of resource and execution delay.

3. System Model

Mobile edge computing provides the cloud services through pushing the cloud resources (e.g., computing, network and storage) to the edge of the mobile network. The traditional wireless access network has the conditions of intelligence, application localization and close deployment through the integration of wireless network and applications, and provides a high bandwidth and low latency transmission capacity. The mobile edge computing system model is shown in Fig.1.

Fig.1 Mobile edge computing system model.

As shown in Fig.1, the system model is mainly composed of five parts: (1) IoT devices that generate city IoT applications; (2) Edge Clouds (ECs) including a wireless cellular base station and multiple edge servers; (3) Remote Cloud (RC) providing cloud services; (4) Edge server and cloud server accommodating the virtual machines (VMs) or containers (As); (5) Tasks (Ts) assigned to the virtual machines or containers. When an IoT device makes an application offloading request, a virtual machine or container on the edge server or cloud server assists the device in handling the tasks offloaded onto it, and feeds the result back to the IoT device. Considering that the factors such as high-rise buildings in the city have great interference to the wireless signal, all edge clouds are connected by a fiber optic backhaul network based on a full network topology [30]. Furthermore, the propagation delay between edge clouds is load-independent. Each edge cloud is endowed with certain computing and storage capabilities by deploying some heterogeneous edge servers interconnected by

switches, and it can receive, process, and forward the offloading requests from IoT devices via a wireless cellular base station. Since IoT devices appear randomly and generate city IoT applications in an arbitrary order and time, the number of offloading requests for city IoT applications will vary from time to time [31]. Meanwhile, each city IoT application usually has a certain deadline and is modeled as a directed acyclic graph reflecting the task dependence. Therefore, application service providers can rent cellular base stations from communications facility providers to deploy and handle these collaborative tasks.

4. Deployment Model of City IoT Application

This section first introduces the resource wastage model, resource load imbalance model, energy consumption model and bandwidth resource consumption model of city IoT application deployment scheme; and then proposes a joint optimization objective function for the city IoT application deployment.

4.1 Resource wastage model

Since edge cloud has scarce and expensive processing resources compared with remote cloud, how to maximize the resource utilization ratio of edge server is the focus of current attention while deploying city IoT applications to edge clouds. For each edge server, the utilization of certain resources such as CPU, memory, disk, and bandwidth refers to the ratio of the resources used to the total resources [32], as shown in formulation (1). The total resource utilization of the edge cloud can be defined as the average utilization of each resource type of all edge servers. The mean value further reflects the idle resources in the edge clouds, that is, the larger the mean value is, the less the idle resources in the edge clouds will be, and conversely, the more. Therefore, the idle resources (i.e., resource wastage) W in the edge clouds can be calculated by formulation (2).

$$\begin{cases} U_j^{CPU} = \sum_{i=1}^N z_{ij} \cdot R_i^{CPU} / C_j^{CPU} \\ U_j^{RAM} = \sum_{i=1}^N z_{ij} \cdot R_i^{RAM} / C_j^{RAM} \\ U_j^{disk} = \sum_{i=1}^N z_{ij} \cdot R_i^{disk} / C_j^{disk} \\ U_j^{BW} = \sum_{i=1}^N z_{ij} \cdot R_i^{BW} / C_j^{BW} \end{cases} \quad (1)$$

$$W = \frac{4M' - \sum_{j=1}^M (U_{j=1}^{CPU} + U_{j=1}^{RAM} + U_{j=1}^{disk} + U_{j=1}^{BW})}{4M'} \quad (2)$$

where M , M' and N respectively represent the total number of edge servers, startup edge servers and virtual machines or containers; U_j^* represents the utilization of one resource in the j -th edge server; R_i^* represents one resource demand of the i -th virtual machine or container; C_j^* represents the total amount of one resource owned by the j -th edge server; the binary variable z_{ij} indicates whether the i -th virtual machine or container is assigned to the j -th edge server, that is, if the i -th virtual machine or

container is assigned to the j -th edge server, then $z_{ij}=1$, otherwise, $z_{ij}=0$.

4.2 Resource load imbalance model

While improving the utilization rate of various resources of edge servers, the load balancing degree of these edge servers should also be taken into account. The resource load imbalance level IB for all edge clouds can be obtained by averaging the various resource load imbalance levels [32], as shown in the formulation (3).

$$IB = \frac{1}{|\Omega|} \sum_{q \in \Omega} \sqrt{\sum_{j=1}^M (U_j^q - \sum_{j=1}^M U_j^q / M')^2} \quad (3)$$

where Ω represents the set of various resources such as CPU, bandwidth, memory, disk of edge servers.

4.3 Energy consumption model

Relevant studies [33], [34] show that CPU is the most important energy consumption component, and propose that the energy consumption of edge server is linearly related to its CPU utilization, or piecewise linear function. However, existing studies show that the energy consumption of edge servers depends on the comprehensive utilization of CPU, memory, disk and network interface. The energy consumption models of different types of edge servers are different, and the simple linearity is not applicable to many new models [35]. According to the research in literature [36], the fitting error of polynomial model is smaller than that of linear model and piecewise linear model. The energy consumption model of the edge server established by the quadratic polynomial model is more suitable for the actual edge server, as shown in the formulation (4).

$$P = \sum_{j=1}^M (P_j^{idle} + \omega_1 \cdot U_j^{CPU} + \omega_2 \cdot (U_j^{CPU})^2) \quad (4)$$

where ω_1 and ω_2 represent the positive fixed polynomial coefficient; P_j^{idle} represents the power consumed when the j -th edge server is started without any load.

4.4 Bandwidth resource consumption model

Considering that the city IoT applications will be split into multiple collaborative tasks, and offloaded onto the edge servers to hand in a cluster composed of multiple virtual machines or containers for collaborative processing. That is, there will be communication between virtual machines or containers in the same cluster. Therefore, the bandwidth resources and execution time required by the cluster to process a city IoT application are directly related to the location of virtual machines or containers in the edge clouds. This is due to the fact that when virtual machines or containers in the same cluster are on the same edge server, the communication between virtual machines or containers does not consume bandwidth resources and execution time; on the contrary, it consumes a certain amount of bandwidth resources and execution time. Therefore, the bandwidth resources BW consumed by edge servers after processing a batch of city IoT applications can be calculated by formulation (5).

$$BW = \sum_{e=1}^E \sum_{k=1}^{M_e} \sum_{l=1}^{m_k} \sum_{t=1}^{V_l} y_{lt} \cdot bw_{lt} \quad (5)$$

where E denotes the total number of edge clouds in the mobile edge computing environment; M_e denotes the number of started edge servers that the e -th edge cloud has; m_k represents the number of virtual machines or containers on the k -th edge server; V_l represents the number of virtual machines or containers working with the l -th virtual machine or container on the k -th edge server for a city IoT application; bw_{lt} represents the bandwidth value from the l -th virtual machine or container to the t -th virtual machine or container, which is specified as a range of random values; the binary variable y_{lt} indicates whether the l -th virtual machine or container is a data sender, if so, then $y_{lt}=1$ otherwise $y_{lt}=0$.

4.5 Joint optimization formulation

When a batch of city IoT applications are offloaded onto the edge cloud, how to adopt the city IoT application deployment approach to improve the system utility and bandwidth resource utilization of these edge clouds under certain execution delay is the key problem that needs to be solved at present. Since the heterogeneous virtual machines or containers that handle these city IoT applications are assigned to different heterogeneous edge servers [37], different deployment schemes have different effects on the system utility and bandwidth resource consumption of the whole edge computing system. To solve the problem of inconsistent optimization objectives, it is necessary to find an optimal deployment scheme by proposing a city IoT application deployment approach. Furthermore, the resource wastage, resource load imbalance, energy consumption and bandwidth resource consumption of edge clouds are minimized under the constraint of resource regulation and execution delay. The joint optimization objective function of the optimization phase can be expressed by formulation (6).

$$\min : \theta_1 * W + \theta_2 * IB + \theta_3 * P + \theta_4 * BW \quad (6)$$

s.t.

$$\sum_{e=1}^E \sum_{k=1}^{M_e} \sum_{l=1}^{m_k} \sum_{t=1}^{V_l} y_{lt} \cdot \frac{data_l}{bw_{lt}} < L \quad (7)$$

where $\theta_1, \theta_2, \theta_3$, and θ_4 are all the tunable positive weights ($0 < \theta_1, \theta_2, \theta_3, \theta_4 < 1$, and $\theta_1 + \theta_2 + \theta_3 + \theta_4 = 1$); the formulation (7) represents the constraint condition of execution delay, that is, the total execution time required by the edge servers to process a batch of city IoT applications should be less than the preset threshold value L ; $data_l$

represents the amount of data that the l -th virtual machine or container sends to the t -th virtual machine or container.

5. Design of City IoT Application Deployment Approach

This section will introduce the implementation details of DEODCA approach. Firstly, the DE algorithm is briefly introduced, then the improvement measures of DE algorithm are proposed, and finally the detailed implementation scheme of DEODCA approach is given.

4.1 DE algorithm

DE algorithm is a heuristic random search algorithm based on population difference proposed by Storn and Price in 1995 to solve Chebyshev polynomials [9]. Unlike other common evolutionary algorithms, it adopts an arithmetic operator to modify the internal representation of an individual to make a difference. Next, through the evaluation of the generated difference vector, if its fitness value is better than the fitness of the predetermined current individual, the newly generated vector will be used to replace the current individual. At present, DE algorithm has developed many different optimization strategies based on the number of disturbed individuals and weighted difference vectors [38]. In order to maintain the population diversity, the DE/rand/1/bin strategy is adopted to randomly select the disturbed vectors, thus three operations of variation, crossover, and selection can be obtained as described below [9]. For the G -th iteration, a population is composed of NP D -dimensional parameter vectors $x_{n,G} = (v_{1n,G}, v_{2n,G}, \dots, v_{Dn,G})$, $n=1, 2, \dots, NP$; where a D -dimension parameter vector represents an individual containing D optimization parameters.

1) Variation operation

For each target vector $x_{n,G}$, a variation vector $v_{n,G+1} = (v_{1n,G+1}, v_{2n,G+1}, \dots, v_{Dn,G+1})$ is generated according to formulation (8).

$$v_{n,G+1} = x_{r1,G} + F \cdot (x_{r2,G} - x_{r3,G}) \quad (8)$$

where $r1 \neq r2 \neq r3 \neq n$ represents four randomly selected individuals that are different from each other, and the number of parameter vectors in the cluster must be at least four; $F \in [0,1)$ represents a pre-set scaling factor used to control the scaling of the difference vector, so that the search step size can be controlled.

2) Crossover operation

This operation will produce a trial vector $u_{n,G+1} = (u_{1n,G+1}, u_{2n,G+1}, \dots, u_{Dn,G+1})$, which is obtained by discrete crossover of the target vector and the variation vector via the formulation (9).

$$u_{gn,G+1} = \begin{cases} v_{dn,G+1} & \text{if } (r \leq CR) \text{ or } d = d_{rand} \\ x_{dn,G} & \text{if } (r > CR) \text{ or } d \neq d_{rand} \end{cases} \quad (9)$$

where r is the random number of the range $[0,1]$; $CR \in [0,1]$ is the pre-set crossover probability, and $CR=0$ represents that the corresponding vector element has no crossover; $d=1,2,\dots,D$, d_{rand} is a randomly selected index from the range $[1,D]$, it ensures that $u_{n,G+1}$ can obtain at least a one-dimensional element from the variation vector $v_{n,G+1}$, otherwise the trial vector will be the same as $x_{dn,G}$, and the population will no change.

3) Selection operation

Based on the first two operations, a greedy selection operation is needed to compare the fitness values $f(*)$ of the children and the corresponding parent, and the better ones are saved into the $G+1$ generation population, as shown in formulation (10).

$$x_{n,G+1} = \begin{cases} u_{n,G+1} & \text{if } (f(u_{n,G+1}) \leq f(x_{n,G})) \\ x_{n,G} & \text{otherwise} \end{cases} \quad (10)$$

4.2 Improvement of DE algorithm

The essence of city IoT application deployment is to establish a reasonable mapping relationship between application task set and edge server resources in edge clouds based on resource and execution delay constraint rules, so as to obtain the smallest value of joint optimization objective function. To solve the above problems, it is necessary to improve the variation operation, crossover operation, selection operation and relevant control parameters of DE algorithm on the basis of formulating a chromosome coding scheme, and then put forward an effective improved DE algorithm.

1) Chromosome coding scheme

According to the characteristics of the city IoT application deployment, this paper adopts a more concise and understandable real coding approach to encode chromosomes. In the encoding scheme, chromosome length is expressed as the total number of tasks in a batch of city IoT applications. Each gene fragment represents a task number, its bit value represents the edge server number assigned to an edge cloud, and the final chromosome encoding scheme is shown in Fig.2.

Fig.2 Chromosome encoding scheme.

As shown in Fig.2, each edge cloud has M_e heterogeneous edge servers; each city IoT application (APP) can consist of multiple tasks with different quantity and computing resource requirements, which are represented by a directed acyclic graph; when Q city IoT applications are deployed to a mobile edge computing environment consisting of E heterogeneous edge clouds, the edge server in each edge cloud and the task in each city IoT application are both numbered from 1. Please note that the total number of tasks is much larger than the number of resources in the edge clouds and is adopted to represent the length of chromosomes. Then the corresponding decoding can be obtained according to the encoding scheme in Fig.2, as shown in the formulation (11).

$$EC_1(1): \{APP_1:T_1, APP_2:T_1\} \quad EC_1(2): \{APP_Q:T_1\} \quad \text{L} \quad EC_E(M_e): \{APP_Q:T_Q\} \quad (11)$$

2) Improved algorithm

In order to solve the optimizing deployment problem of city IoT application, the standard DE algorithm is required to be improved via the chromosome coding scheme in Fig.2. In the process of improvement, three control parameters (i.e., the population size NP , crossover probability CR and scaling factor F) need to be set. Considering the two control parameters of F and CR include constant, random and adaptive strategies, different setting strategies can have a large effect on the convergence speed, diversity and search space of the algorithm [39]. Therefore, to avoid the problem of falling in local optimal and low convergence speed, this paper exploits a variety of optimization strategies to improve the diversity and convergence speed of the algorithm, so as to better solve the optimal deployment problem of city IoT application.

In term of the adaptive strategies for controlling parameter CR and F , the size of F is a positive relationship with the variation search scope of DE algorithm. F will be reduced with the operation of the algorithm, which ensures the initial population diversity and the optimal solution protection of the later algorithm. F_n of the

individual x_n adaptively changes in the iterative process, as shown in formulation

(12). Given that the larger cross probability CR increases the probability that individuals with low fitness are entering the next generation, the smaller cross probability CR is very good for global search ability and population diversity. CR_n of

the individual x_n adaptively changes in the iterative process, as shown in formulation

(13).

$$F_n = F_0 \cdot 5^{\lambda} \quad \text{且} \quad \lambda = e^{\frac{1 - \frac{f_{\max} - f_{\min}}{f_n - f_{\min}}}{f_n - f_{\min}}} \quad (12)$$

$$CR_n = \begin{cases} CR_{\min} + (CR_{\max} - CR_{\min}) \cdot \frac{f_n - f_{\min}}{f_{\max} - f_{\min}}, & \text{if } f_n < \bar{f} \\ CR_{\min}, & \text{if } f_n \geq \bar{f} \end{cases} \quad (13)$$

where $f_{\max}$ and $f_{\min}$ respectively represent the fitness of the worst individuals and the optimal individuals in the current iteration population; $\bar{f}$ represents the average fitness value of the current population; f_n represents the fitness of the individual x_n ; $CR_{\max}$ represents the maximum crossover probability; $CR_{\min}$ represents the minimum crossover probability; F_0 represents the initial scaling factor.

During the runtime of the algorithm, except for that the variation operation produces the variation vector $v_{n,G+1} = (v_{1n,G+1}, v_{2n,G+1}, \dots, v_{Dn,G+1})$ and the crossover operation produces the trial vector $u_{n,G+1} = (u_{1n,G+1}, u_{2n,G+1}, \dots, u_{Dn,G+1})$, the crossover operation can also produce another trial vector $w_{n,G+1} = (w_{1n,G+1}, w_{2n,G+1}, \dots, w_{Dn,G+1})$ to keep genetic information, as shown in the formulation (14). The variation vector $v_{n,G+1}$ and the trial vector $w_{n,G+1}$, $u_{n,G+1}$ can be used to the selection operation to choose the smallest individual of the three vectors, as shown in the formulation (15), which can further keep the genetic information of each iteration.

$$w_{gn,G+1} = \begin{cases} x_{dn,G+1} & \text{if } (r \leq CR) \text{ or } d = d_{rand} \\ v_{dn,G} & \text{if } (r > CR) \text{ or } d \neq d_{rand} \end{cases} \quad (14)$$

$$x_{n,G+1} = \begin{cases} u_{n,G+1} & \text{if } (f(u_{n,G+1}) \leq f(x_{n,G})) \\ v_{n,G+1} & \text{else if } (f(v_{n,G+1}) \leq f(x_{n,G})) \\ w_{n,G+1} & \text{else if } (f(w_{n,G+1}) \leq f(x_{n,G})) \\ x_{n,G} & \text{otherwise} \end{cases} \quad (15)$$

Based on the above encoding scheme, the DEODCA approach can be designed by using these improved measures to improve the standard DE algorithm, and solve well the optimization deployment problem of city IoT applications. The implementation scheme of the DEODCA approach is shown in Algorithm 1.

As described in Algorithm 1, line 1~2 initialize the system parameters such as the population size NP , chromosome length D , maximum iteration times $\max Gen$, maximum crossover probability $CR_{\max}$, minimum crossover probability $CR_{\min}$, initial scaling factor F_0 , edge cloud number E and its edge server number $M_e, e=1,2,\dots,E$, and also randomly initial population. Line 4~11 execute the variation operation, which first selects three different individuals from the population and sorts them from small to large according to their fitness, then calculates scaling factor of the current

individual, and finally exploits the formulation (8) to obtain the variation vector $v_{n,G+1}$ for the variation of the D gene fragment in the former individual. Please note that if the assignment R_i of one mutated gene fragment exceeds the specified assignment range, the formulation $\lfloor |R_i| \rfloor / \sum_{e=1}^E M_e$ is required to map the variation value to the specified range of assignment. Line 12~17 execute the crossover operation, which exploits the formulation (13) to compute the crossover probability CR_n of individual, and adopts the formulation (9) and (14) to produce two trial vectors $u_{n,G+1}$ and $w_{n,G+1}$. The loop in line 18~20 execute the selection operation, which obtains the optimal individual as the next iteration from the current individual vector $x_{n,G}$, the variation vector $v_{n,G+1}$, the trial vector $u_{n,G+1}$ and $w_{n,G+1}$.

Algorithm 1: Differential Evolution based Optimization Deployment of City IoT Application (DEODCA)

Input: the system parameters

Output: $\arg \min_p f(x_p)$ individual

- 1 initialize the system parameters
- 2 Randomly generate NP D -dimensional parameter vectors
$$x_{n,G} = (v_{1n,G}, v_{2n,G}, \dots, v_{Dn,G}), n = 1, 2, \dots, NP$$
- 3 **for** $G=1$ to $\max Gen$ **do**
- 4 **for** $n=1$ to NP **do** //variation operation
- 5 Calculate the fitness $f_{\max}$ and $f_{\min}$ of the worst individual and the optimal individual, the average fitness $\bar{f}$ and the fitness f_n of individual x_n
- 6 Select three different individuals $x_{r1,G}$, $x_{r2,G}$, and $x_{r3,G}$ from $x_{n,G}$
- 7 Compute the scaling factor $F_n = F_0 \cdot 5^\lambda$ & $\lambda = e^{\frac{f_{\max} - f_{\min}}{f_n - f_{\min}}}$ for the current individual
- 8 **for** $d=1$ to D **do**
- 9 Generate variation vector $v_{n,G+1} = (v_{1n,G+1}, v_{2n,G+1}, \dots, v_{Dn,G+1})$ via the formulation (8)
- 10 **end for**
- 11 **end for**
- 12 **for** $n=1$ to NP **do** // crossover operation
- 13 Compute crossover probability CR_n of current individual via the formulation (13)

```

14      for d=1 to D do
15          Generate trial vectors  $u_{n,G+1}$  and  $w_{n,G+1}$  via the formulation (9) and (14)
16      end for
17  end for
18  for n=1 to NP do      // selection operation
19      
$$x_{n,G+1} = \begin{cases} u_{n,G+1} & \text{if } (f(u_{n,G+1}) \leq f(x_{n,G})) \\ v_{n,G+1} & \text{else if } (f(v_{n,G+1}) \leq f(x_{n,G})) \\ w_{n,G+1} & \text{else if } (f(w_{n,G+1}) \leq f(x_{n,G})) \\ x_{n,G} & \text{otherwise} \end{cases}$$

20  end for
21 end for
22 return  $\arg \min_p f(x_p)$  individual

```

3) Complexity Analysis

In this section, we analyze the complexity of the Algorithm 1, which mainly includes three parts, i.e., mutation operation, crossover operation and selection operation. As we have seen before, the population size and the individual vector parameter dimension are respectively set to NP and D . The complexity of the variation operation is determined by lines 4-11; it has the complexity $O(NP * D)$ to traverse all individuals and parameter vectors in the population. The complexity of the crossover operation is determined by lines 12-17; it has the complexity $O(NP * D)$ to achieve the crossover between individuals in a population. The complexity of the selection operation is determined by lines 18-20; it has complexity $O(NP)$ to select the superior individual. Therefore, the maximum time complexity required for the algorithm to iteration once is $O(NP * D)$. The time complexity of algorithm iteration

$\max Gen$ is polynomial and can be expressed as $O(\max Gen * NP * D)$.

5. Performance Evaluation

This section first introduces the setting of the simulation experiment environment, then compares the DEODCA approach with some related approaches, and finally evaluates and analyzes these approaches from two aspects of performance and effectiveness.

5.1 Experimental setup

We can set up a city mobile edge computing experiment environment with 25 edge clouds and 20 IoT devices by extending the CloudSim simulator [40], [41]. In this experimental environment, each edge cloud includes a base station interconnecting

with other base stations via the full network topology- based optical fiber backhaul network, a certain number of heterogeneous edge servers, and multiple IoT devices interconnecting via wireless access. The configuration parameters of each edge server are randomly selected from the set {HP ProLiant G4, HP ProLiant G5} [42], and deployed to multiple edge clouds. When 20 IoT devices appear at some point and produce a batch of city IoT applications, each city IoT application is split into three collaborative tasks and assigned to 60 heterogeneous virtual machines based on the task size.

The bandwidth requirement of each virtual machine is selected randomly from the set [10, 50] Mbps; the data amount sent of each virtual machine is randomly selected from the set [1, 2] Mb; the CPU and memory requirement of each virtual machine is randomly selected from the set {2000MIPS and 3.75GB, 500MIPS and 0.6GB, 1000MIPS and 1.7GB, 2500MIPS and 0.85GB} [42]; the disk requirement of each virtual machine is set at 1GB. When these edge servers are started without any load, the power of edge servers HP ProLiant G4 and HP ProLiant G is respectively set to 86w and 93.7w [42]. The value of ω_1 and ω_2 is set to 1.30447 and 0.02867, respectively [36]. The initial population size NP is set to 100; the chromosome length D is set to 60; and the maximum iteration number is set to 2000. $CR_{\max}$, $CR_{\min}$, and F_0 are respectively set to 0.90, 0.1 and 0.2, respectively. The execution delay threshold L is set to 2.36s. The value of the tunable positive weights $\theta_1, \theta_2, \theta_3$, and θ_4 is all set to 0.5.

To evaluate the performance and effectiveness of the DEODCA approach, we compare it with the following benchmark approaches.

- **Random Deployment (RD):** When multiple edge server candidates meet resource constraints, randomly select the edge server to run each virtual machine.
- **First Fit Deployment (FFD):** When there are several candidate edge servers that meet the constraints, select the edge server that meets the resource requirements for the first time to run each virtual machine.
- **Particle Swarm Optimization (PSO):** When a number of edge server candidates meet the resource constraints, the edge server that meets the resource requirements is selected to run each virtual machine via the particle swarm optimization algorithm [43].

Fig.3 Comparison of resource wastage level Fig.4 Comparison of resource load imbalance level

Fig.5 Comparison of energy consumption Fig.6 Comparison of bandwidth resource consumption

- **Multi-objective Grouping Genetic Algorithm (MGGA):** When a number of edge server candidates meet the resource constraints, the edge server that meets the resource requirements is selected via the multi-objective grouping genetic algorithm to run each virtual machine [44].

5.2 Experimental results and evaluation

We first compared the resource wastage level, resource load imbalance level, energy consumption and bandwidth resource consumption of the edge clouds while dealing with a batch of city IoT applications under the condition of specified execution delay; and then analyzed and studied the relevant experimental parameters including the number of base stations and the number of city IoT applications.

1) Performance analysis

Fig.7 Comparison of execution time

As shown in Figs. 3 to 7, since RD approach randomly deploys the virtual machines that the tasks reside, its resource wastage level and resource load imbalance level are both the highest of four approaches. FFD approach exploits the first adaption to deploy these virtual machines, that is, the next edge server will be selected to host residual virtual machines, only when the first appropriate edge server cannot accommodate them. Although the resources of the edge server can be fully utilized,

Fig.8 Effect of the number of base stations

the resource wastage level and the resource load imbalance level are still very high. MGGA and PSO both exploit intelligent optimization algorithms to deploy the virtual machines that these tasks are located, but they still have higher resource wastage level and resource load imbalance level than the DEODCA approach. The above reasons also result in the decreasing energy consumption of the RD, FFD, MGGA, PSO, and DEODCA. Given that the tasks deployed on the virtual machine need to communicate with other tasks in the same cluster, the bandwidth resources consumed by different deployment solutions for these five approaches and the execution time processing a batch of city IoT applications are also diminishing. Since the execution time of the RD approach exceeds the execution latency threshold, it is not recommended for the deployment of the city IoT applications.

2) Parameter study

In view of the above approaches, we further analyzed the impact of the number of base stations and the number of city IoT applications on the resource wastage level, resource load imbalance level, energy consumption and bandwidth resource consumption of the edge clouds, as shown in Fig.8 and Fig.9.

(1) Effect of the number of base stations

Fig.8 shows the impact of the number of base stations on all approaches. In order to make this effect more clearly, the number of virtual machines and the city IoT applications was respectively set up to 20 and 60; the number of base stations varied from 15 to 30 by step size 5, and the number of edge servers changed accordingly to 76, 101, 127, and 152. The analysis of these figures can be obtained that although the resource wastage level, the resource load imbalance level, energy consumption and bandwidth resource consumption fluctuate with the increase of the number of base stations, it can be thought that there is no such effect. In these five approaches, the DEODCA approach always has the lowest resource wastage level, the resource load imbalance level, energy consumption and bandwidth resource consumption.

Fig.9 Effect of the number of city IoT applications

(2) Effect of the number of city IoT applications

Fig.9 shows the impact of the number of city IoT applications on all approaches. In order to make this effect more clearly, the number of base stations and the edge servers was respectively set up to 25 and 127; the number of city IoT applications varied from 5 to 20 by step size 5, and the number of virtual machines changed accordingly to 15, 30, 45, and 60. The analysis of these figures can be found that although the resource wastage level fluctuates with the increase of the number of city IoT applications, it can be thought that there is no such effect. The resource load imbalance level, energy consumption and bandwidth resource consumption of all approaches increase as the number of city IoT applications increases; this is due to that the increase in the number of city IoT applications requires more virtual machines to deal with their tasks. Furthermore, in these five approaches, the DEODCA approach always has the lowest resource wastage level, resource load imbalance level, energy consumption and bandwidth resource consumption.

6. Conclusion and Future Work

With the advent of the 5G era, an increasing number of IoT devices have adopted 5G communication technology, and produce a large number of city IoT applications with high resource demand and delay sensitivity. Considering that IoT devices with limited resources cannot meet the resource demands of these city IoT applications, these IoT devices offload some tasks of city IoT applications onto the edge clouds. In order to improve the system utility and processing efficiency of edge clouds, this paper presented an optimal deployment approach for city IoT applications of resource-constrained edge clouds to allocate their collaborative tasks. Through the comparative analysis of experimental results, our proposed approach is superior to other related approaches in terms of effectiveness and efficiency.

In future work, we will need to further reduce the time complexity of the our proposed approach. Meanwhile, we will optimize the consumption of energy, processor resources and bandwidth resources for mobile edge computing environment under the circumstance that city IoT applications are offloaded to edge cloud and remote cloud.

List of Abbreviations

- (1) IoT: Internet of Things
- (2) EC: Edge Cloud
- (3) RC: Remote Cloud
- (4) A: container
- (5) T: Task
- (6) VM: Virtual Machine
- (7) DE: Differential Evolution
- (8) APP: city IoT application
- (9) BW: bandwidth resources
- (10) RD: Random Deployment
- (11) FFD: First Fit Deployment
- (12) PSO: Particle Swarm Optimization
- (13) MGGA: Multi-objective Grouping Genetic Algorithm

Competing interests

We declare that there is no conflict of interest regarding this submission.

Funding

Partial work of this paper is supported by the National Natural Science Joint Fund project (No. U1804162), Key Science and Technology Program of Henan Province (No. 202102210152, 202102210109), and the Research and Cultivation Fund Project of Anyang Normal University (No. AYNUPY-2019-24).

Authors' contributions

Jialei Liu designed the algorithm and finished the English writing. Quanzhen Huang introduced the idea of this work and completed the experiments.

Acknowledgements

Not applicable.

References

- [1] Cisco V. Cisco Visual Networking Index: Forecast and Trends, 2017 – 2022. White Paper, 2018.
- [2] X. Sun and N. Ansari, "EdgeIoT: Mobile Edge Computing for the Internet of Things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22–29, 2016.
- [3] L. Qi, Q. He, F. Chen, X. Zhang, W. Dou, and Q. Ni, "Data-Driven Web APIs Recommendation for Building Web Applications," *IEEE Transactions on Big Data*, 2020, online publishing.
- [4] L. Wang, X. Zhang, R. Wang, C. Yan, and L. Qi, "Diversified service recommendation with high accuracy and efficiency," *Knowledge-Based Systems*, vol. 204, p. 106196, 2020.
- [5] K. Zhang, X. Gui, D. Ren, J. Li, J. Wu, and D. Ren, "Survey on Computation Offloading and Content Caching

- in Mobile Edge Networks," *Journal of Software*, vol. 30, no. 8, pp. 2491-2516, 2019.
- [6] C. Hong and B. Varghese, "Resource Management in FogEdge Computing A Survey on Architectures, Infrastructure, and Algorithms," *ACM Computing Surveys*, vol. 52, no. 5, pp. 97:1-97:37, 2019.
- [7] S. Deng, Z. Xiang, J. Taheri, K. Mohammad, J. Yin, A. Zomaya, and S. Dustdar., "Optimal Application Deployment in Resource Constrained Distributed Edges," *IEEE Transactions on Mobile Computing*, online publishing.
- [8] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng, J. Chen, Z. Wang, and H. Qiao, "Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications," in *Proceedings of the web conference*, 2018, pp. 187-196.
- [9] R. Storn and K. Price, "Differential Evolution - A Simple and Efficient Heuristic for global Optimization over Continuous Spaces," *Journal of Global Optimization*, vol. 11, pp. 341-359, 1997.
- [10] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proceedings of the 35th Annual IEEE International Conference on Computer Communications*, IEEE, 2016, pp. 1-9.
- [11] H. Tan, Z. Han, X.-Y. Li, and F. C. Lau, "Online job dispatching and scheduling in edge-clouds," in *Proceedings of the IEEE Conference on Computer Communications*, IEEE, 2017, pp. 1-9.
- [12] J. Meng, H. Tan, C. Xu, W. Cao, L. Liu, and B. Li, "Dedas: Online task dispatching and scheduling with bandwidth constraint in edge computing," *Proceedings of IEEE Conference on Computer Communications*, IEEE, 2019, pp. 2287-2295.
- [13] H. Wu, S. Deng, W. Li, M. Fu, J. Yin, and A. Y. Zomaya, "Service selection for composition in mobile edge computing systems," in *Proceedings of the IEEE International Conference on Web Services*, IEEE, 2018, pp. 355-358.
- [14] Y. Chen, S. Deng, H. Ma, and J. Yin, "Deploying Data-intensive Applications with Multiple Services Components on Edge," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 426-441, 2020.
- [15] T. Bahreini and D. Grosu, "Efficient placement of multi-component applications in edge computing systems," in *Proceedings of the 2th ACM/IEEE Symposium on Edge Computing*, IEEE/ACM, 2017, pp. 5.
- [16] W. Chen, D. Wang, and K. J. Li, "Multi-user Multi-task Computation Offloading in Green Mobile Edge Cloud Computing," *IEEE Transactions on Services Computing*. vol. 12, no. 5, pp. 726-738, 2019.
- [17] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint Computation Offloading and User Association in Multi-task Mobile Edge Computing," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 12, pp. 12313-12325, 2018.
- [18] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. S. Shen, "Energy Efficient Dynamic Offloading in Mobile Edge Computing for Internet of Things," *IEEE Transactions on Cloud Computing*, 2019, online publishing.
- [19] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795-2808, 2016.
- [20] Y. Chen, S. Deng, H. Zhao, Q. He, Y. Li, and H. Gao, "Data-intensive application deployment at edge: A deep reinforcement learning approach," in *Proceedings of the IEEE International Conference on Web Services*, IEEE, 2019, pp. 355-359.
- [21] B. Hu, J. Chen, and F. Li, "A dynamic service allocation algorithm in mobile edge computing," in *Proceedings of the IEEE International Conference on Information and Communication Technology Convergence*, IEEE, 2017, pp. 104-109.
- [22] H. Wu, Z. Zhang, C. Guan, K. Wolter, and M. Xu, "Collaborate Edge and Cloud Computing with Distributed Deep Learning for Smart City Internet of Things," *IEEE Internet of Things Journal*, 2020, online publishing.

- [23] S. Pallevatta, V. Kostakos, and R. Buyya, "Microservices-based IoT Application Placement within Heterogeneous and Resource Constrained Fog Computing Environments," in Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing, IEEE/ACM, 2019, pp. 71-81.
- [24] X. Zhang and S. Debroy, "Adaptive task offloading over wireless in mobile edge computing," in Proceedings of the 4th ACM/IEEE Symposium on Edge Computing, IEEE/ACM, 2019, pp. 323-325.
- [25] M. Goudarzi, H. Wu, M. S. Palaniswami, and R. Buyya, "An Application Placement Technique for Concurrent IoT Applications in Edge and Fog Computing Environments," IEEE Transactions on Mobile Computing, 2020, online publishing.
- [26] S. Cheng, Z. Chen, J. Li, and H. Gao, "Task Assignment Algorithms in Data Shared Mobile Edge Computing Systems," in Proceedings of the IEEE 39th International Conference on Distributed Computing Systems, IEEE, 2019, pp. 997-1006.
- [27] X. Zhang, Y. Mao, J. Zhang, and K. B. Letaief, "Multi-objective resource allocation for mobile edge computing systems," in Proceedings of the IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications, IEEE, 2017, pp. 1-5.
- [28] X. Tao, K. Ota, M. Dong, H. Qi, and K. Li, "Performance guaranteed computation offloading for mobile-edge cloud computing," IEEE Wireless Communications Letters, vol. 6, no. 6, pp. 774-777, 2017.
- [29] J. Guo, Z. Song, Y. Cui, Z. Liu, and Y. Ji, "Energy-efficient resource allocation for multi-user mobile edge computing," in Proceedings of the IEEE Global Communications Conference, IEEE, 2017, pp. 1-7.
- [30] J. Oueis, E. Calvanese-Strinati, A. De Domenico, and S. Barbarossa, "On the impact of backhaul network on distributed cloud computing," Proceedings of IEEE Wireless Communications and Networking Conference Workshops, IEEE, 2014, pp. 12-17.
- [31] L. Qi, C. Hu, X. Zhang, M. Khosravi, S. Sharma, S. Pang, T. Wang, "Privacy-aware Data Fusion and Prediction with Spatial-Temporal Context for Smart City Industrial Environment," IEEE Transactions on Industrial Informatics, 2020, online publishing.
- [32] X. Liu, B. Cheng, and S. Wang, "Availability-aware and Energy-efficient Virtual Cluster Allocation Based on Multi-objective Optimization in Cloud Datacenters," IEEE Transactions on Network and Service Management, 2020, online publishing.
- [33] C.-H. Lien, Y.-W. Bai, and M.-B. Lin, "Estimation by software for the power consumption of streaming-media servers," IEEE transactions on instrumentation and measurement, vol. 56, no. 5, pp. 1859-1870, 2007.
- [34] V. De Maio, G. Kecskemeti, and R. Prodan, "An improved model for live migration in data centre simulators," in Proceedings of the IEEE/ACM International conference utility and cloud computing, IEEE/ACM, 2016, pp. 108-117.
- [35] D. Tsirogiannis, S. Harizopoulos, and M. A. Shah, "Analyzing the energy efficiency of a database server," in Proceedings of the ACM International Conference on Management of Data, ACM, 2010, pp. 231-242.
- [36] H. Zhao, J. Wang, F. Liu, Q. Wang, W. Zhang, and Q. Zheng, "Power-Aware and Performance-Guaranteed Virtual Machine Placement in the Cloud," IEEE Transactions on Parallel and Distributed Systems, vol. 29, no. 6, pp. 1385-1400, 2018.
- [37] W. Zhong, X. Yin, X. Zhang, S. Li, W. Dou, R. Wang, L. Qi, "Multi-Dimensional Quality-Driven Service Recommendation with Privacy-Preservation in Mobile Edge Environment. Computer Communications," 2020, online publishing.
- [38] R. Storn, "On the usage of differential evolution for function optimization," in Proceedings of the North American Fuzzy Information Processing Society, IEEE, 1996, pp. 519-523.
- [39] L. Tang, Y. Dong, and J. Liu, "Differential Evolution With an Individual-Dependent Mechanism," IEEE Transactions on Evolutionary Computation, vol. 19, no. 4, pp. 560-574, 2015.

- [40] J. Liu, S. Wang, A. Zhou, S. Kumar, F. Yang, and R. Buyya, "Using Proactive Fault-tolerance Approach to Enhance Cloud Service Reliability," *IEEE Transactions on Cloud Computing*, vol. 6, no. 4, pp. 1191-1202, 2018.
- [41] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, "ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1275-1296, 2017.
- [42] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397-1420, 2012.
- [43] S. Wang, Z. Liu, Z. Zheng, Q. Sun, and F. Yang, "Particle swarm optimization for energy-aware virtual machine placement optimization in virtualized data centers," *Proceeding of IEEE International Conference on Parallel and Distributed Systems*, IEEE, 2013, pp. 102-109.
- [44] J. Xu and J. A. Fortes, "Multi-objective virtual machine placement in virtualized data center environments," in *Proceeding of the IEEE/ACM International Conference on Green Computing and Communications*, IEEE/ACM, 2010, pp. 179-188.

Figures

Figure 1

Mobile edge computing system model.

Figure 2

Chromosome encoding scheme.

Figure 3

Comparison of resource wastage level

Figure 4

Comparison of resource load imbalance level

Figure 5

Comparison of energy consumption

Figure 6

Comparison of bandwidth resource consumption

Figure 7

Comparison of execution time

Figure 8

Effect of the number of base stations

Figure 9

Effect of the number of city IoT applications