Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8936576, 17 pages
https://doi.org/10.1155/2022/8936576

Research Article

Cost-Aware Placement Optimization of Edge Servers for IoT Services in Wireless Metropolitan Area Networks

Yanling Shao, Zhen Shen, Siliang Gong, and Hanyao Huang

School of Computer and software, Nanyang Institute of Technology, Nanyang 473000, China

Correspondence should be addressed to Yanling Shao; shaoyl@nyist.edu.cn

Received 16 May 2022; Accepted 16 July 2022; Published 27 July 2022

Academic Editor: Yanxiang Jiang

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

Edge computing migrates cloud computing capacity to the edge of the network to reduce latency caused by congestion and long propagation distance of the core network. And the Internet of things (IoT) service requests with large data traffic submitted by users need to be processed quickly by corresponding edge servers. The closer the edge computing resources are to the user network access point, the better the user experience can be improved. On the other hand, the closer the edge server is to users, the fewer users will access simultaneously, and the utilization efficiency of nodes will be reduced. The capital investment cost is limited for edge resource providers, so the deployment of edge servers needs to consider the trade-off between user experience and capital investment cost. In our study, for edge server deployment problems, we summarize three critical issues: edge location, user association, and capacity at edge locations through the research and analysis of edge resource allocation in a real edge computing environment. For these issues, this study considers the user distribution density (load density), determines a reasonable deployment location of edge servers, and deploys an appropriate number of edge computing nodes in this location to improve resource utilization and minimize the deployment cost of edge servers. Based on the objective minimization function of construction cost and total access delay cost, we formulate the edge server placement as a mixed-integer nonlinear programming problem (MINP) and then propose an edge server deployment optimization algorithm to seek the optimal solution (named Benders_SD). Extensive simulations and comparisons with the other three existing deployment methods show that our proposed method achieved an intended performance. It not only meets the low latency requirements of users but also reduces the deployment cost.

1. Introduction

Due to long-distance network communication, data transmission has a long round-trip time. And remote cloud computing services for delay-sensitive IoT applications [1–3] (i.e., Internet of Vehicles, intelligent industrial control, virtual reality/augmented reality, and online games) may lead to poor user satisfaction. To enable users to access service nodes in a timely manner and quickly meet user requirements, service content is distributed to appropriate access sites. In general, the access site close to the user is preferred to be processed in time by reducing transmission distance, and tasks are going to be executed in multiaccess edge computing (MEC) servers [4]. MEC refers to processing, analyzing, and storing data closer to where it is generated to enable rapid, near real-time analysis and response, which can alleviate the backhaul link pressure with the traditional network architecture. And it can offload a large number of complex computations to edge servers, reducing the cost of remote network communication and effectively meeting the requirements of low latency and bandwidth efficiency. Therefore, the operating costs of edge servers are significantly reduced [5].

A wireless metropolitan area network (WMAN) is a computer network, usually as a public utility, that provides wireless Internet coverage to mobile users in metropolitan areas. The core idea of mobile edge networks is to move network functions, contents, and resources closer to end users.

The network resources mainly include computing, storage or caching, and communication resources. A mobile edge network scenario mainly includes the following four parts: MEC server, base station, terminal equipment, and core network equipment. When the edge server is deployed in the metropolitan area network, due to the expansion of the service area and the increase of deployed resources, the scale of the problems to be solved becomes larger and the computational complexity increases. With a large network, service providers can take advantage of economies of scale when providing edge services [6, 7]. So in this work, we focus on the edge server placement in collaborative edge computing environments that provides wireless internet coverage for mobile users in a large-scale metropolitan area. First, a large number of mobile users access edge servers in edge computing environments because the metropolitan area that it covers has a high population density. Second, because of the size of the network, service providers can take advantage of economies of scale when offering edge server services by making edge server services more affordable to the general public. Therefore, how to place edge server becomes a critical and meaningful research topic.

The deployment location of the edge server and the number of servers in each edge micro data center profoundly impact on the costs and the performance of edge services or 6G networks [8], such as the end-to-end delay and resource utilization. On the one hand, if the edge server is deployed far away from the user, the user can only access the nearest edge site through multiple forwarding. The deployment of the servers affects transmission latency in the scenario where data needs to be analyzed in real-time and used for precise control. In addition, the rental cost of the server deployment location varies with geographic location, which significantly impacts on deployment costs. Therefore, edge service providers should set out server sites to achieve a high quality of service for low-latency applications. After the deployment location is determined, each server group serving the surrounding users has limited transmission resources within the region. The resources available within the deployment group, such as the number of servers, should be appropriately adjusted according to the density of users in the region and the service requirements required by users. Deploying a relatively large number of servers in areas with low user density or a relatively small number of servers in areas with high user density is not a reasonable deployment strategy. Unreasonable deployment causes overload or underload of cloud servers and brings the same problems to the transmission process. So we propose Benders_SD algorithm to optimize the deployment of edge servers. This study owns threefold specific contributions as follows.

(1) Considering user distribution density, deployment cost, and network access delay in each service area by the candidate locations, the edge server placement problem in the WMAN area is transformed into a MINP problem

(2) The Benders_SD optimization algorithm for sparse edge server deployment is proposed to minimize the total capital investment cost while meeting the low latency requirements of users in collaborative edge computing environments

(3) The simulation results show that our presented Benders_SD optimization algorithm can successfully solve the above problems, reducing the user delay requirements and the deployment cost to the greatest extent

The remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 describes the analysis and modeling of edge server deployment problems, and Section 4 presents Benders decomposition of edge server deployment problems. Section 5 gives implementation of edge server deployment algorithm based on Benders decomposition. Section 6 shows an example of edge server deployment. Section 7 evaluates these algorithms by extensive simulations. Section 8 gives the conclusion.

2. Related Work

The locations of edge servers have an important influence on user access delay and resource utilization of edge server. Therefore, the strategic placement of edge servers will significantly improve the performance of edge computing systems. To perfect MEC standardization, European Telecommunication Standards Association Working Group on Mobile Edge Computing and Heavy Reading gathered typical use cases and deployment scenarios [9, 10]. However, compared with the research on edge computing resource scheduling, there are relatively few works focusing on edge server placement [6]. In some of these studies, edge server placement is modeled as optimization problems, such as multiobjective constrained optimization problems [6, 11–15], integer linear programming (ILP) problems [16–18], and MINP problems [19, 20]. The most commonly used methods are k-means clustering [12, 21–23], heuristic algorithm [7, 11, 15–19], branch and bound method [16, 20], and so on.

Edge server placement was modeled as a multiobjective optimization problem in some work. Wang et al. [6] adopted mixed-integer programming (MIP) to find the optimal edge server placement with workload balancing among edge servers and minimizing the edge server access delay. Kasi et al. [11] used genetic algorithms and local search algorithms to find an edge server allocation strategy. Guo et al. [12] proposed an approximate approach that adopted the k-means and mixed-integer quadratic programming to balance the workload between edge clouds and minimize the service communication delay of mobile users. Li et al. [13] studied the deployment of edge servers in a smart city mobile edge computing environment. And the optimal solution was found by using mixed-integer programming to balance the workload of edge servers and minimize the access delay between mobile users and edge servers. Li et al. [13] proposed the optimal deployment and allocation strategy of edge servers, which could optimize the number and location of edge servers and the allocation of mobile users in a given ultradense networking environment. It proposed a

strategy based on queuing model and vector quantization to solve it. Considering transmission delay, workload balancing, energy consumption, deployment costs, network reliability, and edge server quantity, Cao et al. [14] studied the placement problem of edge servers in the Internet of Vehicle (IoV). Considering the density of mobile users and the location of cloudlets in the mobile edge computing environment, Fan and Ansari [15] studied the optimal deployment strategy of cloudlets that balanced the deployment cost and end-to-end delay cost and proposed to use the mixed-integer programming (MIP) tool CPLEX solver to find the suboptimal solution.

The integer linear programming model (ILP) was used to model the edge server positioning under constraints. Considering load balancing between edge servers, Li et al. [16] proposed the greedy algorithm is and combined with the GA to solve the edge server placement problem. To minimize the access delay between mobile equipment and cloudlet, Xu et al. [17] proposed a heuristic greedy algorithm to solve it with an exact solution. To extend the poor scalability of the ILP, efficient approximation algorithms with identical and different cloudlet capacities were proposed. To optimize edge facilities' overall performance and cost, Yin et al. [18] proposed Tentacle decision support framework and flexible edge server deployment method. Considering the proximity between users and edge servers, cost budget, the capacity of edge sites, the fault tolerance of edge sites, and other factors in deploying edge servers, the heuristic algorithm was used to select the ideal location of edge server deployment and find the exact deployment location of the server closest to the ideal location. Ahat et al. [19] proposed the MILP model to optimize the multilevel computing of the design infrastructure to maximize the expected revenue of operators. The proposed model considered operators' limited budgets and service requirements and introduced a heuristic approach based on Lagrange relaxation to solve complex and scalability problems and huge instances. Finally, a greedy heuristic solution was proposed to solve the computational time complexity problem.

To minimize the average delay time of job hunting, Jia et al. [21] designed Heaviest-AP First Placement (HAF) strategy and the K-median algorithm. HAF placed cloudlets at the BSs with the heaviest workloads and the K-median algorithm selected some strategic positions. Xiang et al. [22] proposed an adaptive cloudlet placement method for mobile applications to maximize the number of mobile devices covered in cloudlet, and the gathering areas of the mobile devices were identified based on k-means algorithm. Lähderanta et al. [23] proposed the PACK algorithm, which placed a fixed number of servers, minimized the delay between users and edge servers, balanced the system workload, and met the lower and upper limits of the server capacity. And PACK was considered as a variant of the k-means clustering with capacity constraints, and the integer programming step block coordinate descent algorithm was used to solve it.

The above-mentioned researches on the cloudlet/edge server deployment problem are valid; however, these researches [11–13, 16, 17, 21] focused on the access delay or balancing workload. Inspired by this, economic cost and delay cost are considered comprehensively from the perspective of edge service provider and user requests in our solution, as well as collaborative edge computing environments in metropolitan area networks. The edge server deployment problem in collaborative edge computing environments is modeled as MINP problem, and the Benders algorithm is adopted to solve it, which can efficiently find the optimal solution for edge server deployment economic cost and low delay balance. Furthermore, based on our previous work [20], this paper further deepens Benders decomposition theory of edge server deployment problems and extends the evaluation test of edge MDC deployment algorithm under different candidate edge locations. In addition, we show an example of edge server deployment based on Benders_SD algorithm to illustrate the effectiveness of our work.

3. Analysis and Modeling of Edge Server Deployment Problems

3.1. Analysis of Edge Server Deployment Problems. In densely populated metropolitan area network coverage areas, edge computing servers are deployed to provide edge services for many users for improving the benefits of edge services by making full use of edge computing resources [17–19]. In addition, edge computing infrastructure providers can use economies of scale to enable edge services to benefit more users. Therefore, the network environment for edge computing server deployment selected in this paper is a metropolitan area network. The edge computing server is closer to the user network access point, and the better the user experience. But the closer is to the user, the fewer users, and the efficiency of the edge server will decrease. For edge resource providers, deployment costs are limited, so the deployment of edge servers needs to consider the balance between user experience and server efficiency. At present, edge computing is usually deployed in small and medium-sized edge data centers at the convergence of metropolitan areas or lower [9]. According to the specific network environment and business requirements, the server is often deployed to be close to the edge communication equipment of the user end, such as the base station. Deploy server in cost-effective IP convergence points to reduce network switching due to user movements, such as the location of routers or switches. Edge services are deployed in computer clusters within schools or enterprises. One or more edge servers are placed on each location to form a small edge data center. Figure 1 is an example of edge server deployment based on the wireless metropolitan area network architecture.

Considering the deployment cost of edge computing nodes and the sharing of edge computing resources, the deployment region does not need to cover all network access points and only requires sparse deployment. Further, the number of edge servers deployed increases with high user density in these areas. Conversely, the number of edge servers in regions with low user distribution density will decrease accordingly. Therefore, according to the distribution density of service users and the deployment cost in different

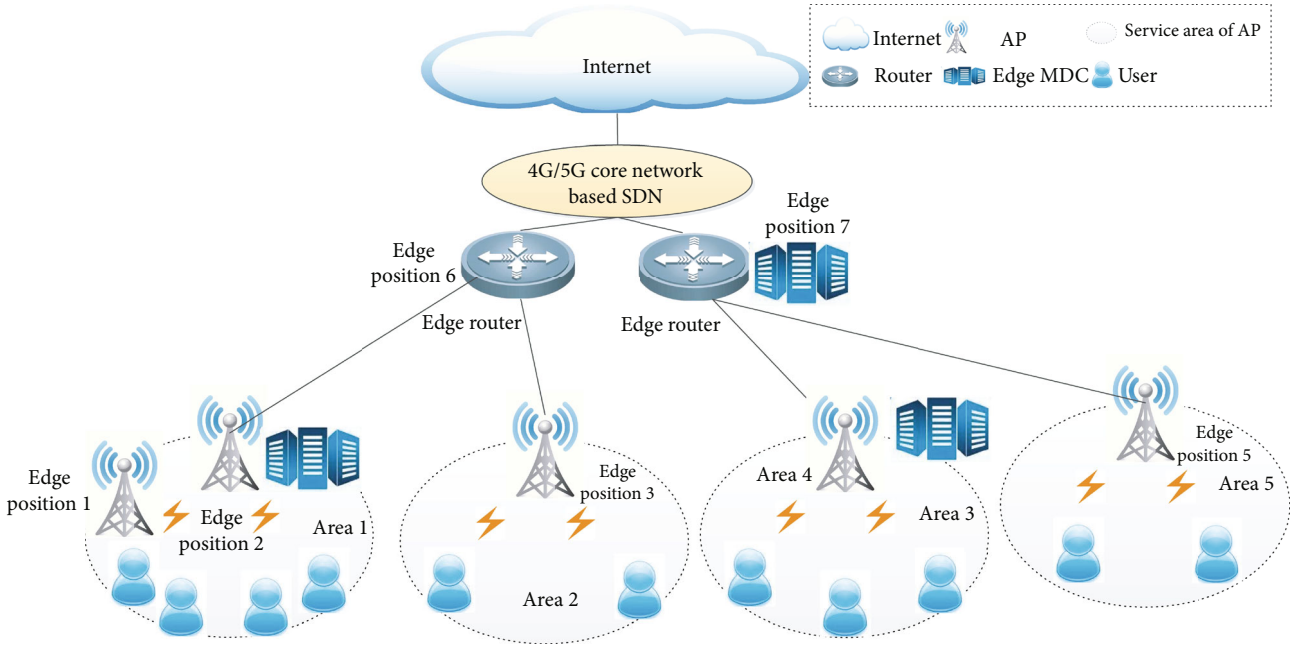

Figure 1: Deployment example of edge server based on WMAN architecture.

geographic locations, it is vital to choose an appropriate location to reasonably deploy edge computing resources. It should satisfy the users' low-latency application requirements and minimize the deployment cost of service providers.

Neither implementation of cost nor low-latency optimization alone can meet the requirements of edge computing infrastructure providers. Therefore, this paper considers the user distribution density (load density) based on edge service's actual diversified scenario requirements under the constraints of satisfying user low-latency applications. This approach determines a suitable edge server deployment location and deploys an appropriate number of edge computing nodes in this location to achieve high resource utilization and minimize edge server deployment costs. This work mainly explores the locations of LTE macro base stations and multistandard base station convergence point routers within the scope of metropolitan area networks. The balance between deployment cost and network access delay is optimized according to the user distribution density in each service area. This work needs to solve three key issues: (1) the edge location problem: select the ideal edge location from the set of candidate locations; (2) the user association problem: edge server provides the service for the user; and (3) the problem of edge location capacity: according to the user distribution density (load density), determine the appropriate number of servers in each edge location. These factors are usually tightly combined, resulting in a huge search space. This work comprehensively weighs these factors and searches for the best edge server deployment strategy under multiple constraints. The user refers to the user terminal that submits a task request to the local edge server. The candidate edge location refers to a wireless or wired network access point, which can be a base station, router, or gateway.

3.2. Problem Description. The deployment scope studied in our work is the WMAN. The base station located close to the user equipment and the router device locations of the data convergence point are selected as candidate deployment locations. And the edge server deployment issues are described as follows [20]:

Within WMAN coverage, given the deployment location set of potential edge servers and the service coverage area set, the coverage area is the service range of the base station or within one hop distance of the router. The user connects to the edge micro data center through the base station. The edge micro data center can process requests and data offloaded by user terminals. Due to different user distribution densities and loads in each coverage area, the cost of renting each potential location and the number of edge servers needing deployment are not the same. The target of this work is to select suitable places to deploy edge servers from these potential edge locations to meet the low-latency requirements of the application and determine the number of nodes in each edge micro data center based on the user distribution density. So that low-latency applications can be satisfied. Under the premise of restrictions, the overall cost is the smallest.

3.3. Model of Edge Server Deployment

Definition 1 (PNN [18]). The distance between the user and a certain location is calculated. The greater the PNN, the greater distance between user and location, and vice versa, the closer the distance between the two set s_i as the position of i; PNN_{li} indicates the proximity between user l and edge position i ($i \in I$); I is the network access point, defined as

$$PNN_{li} = \|s_l - s_i\|. \quad (1)$$

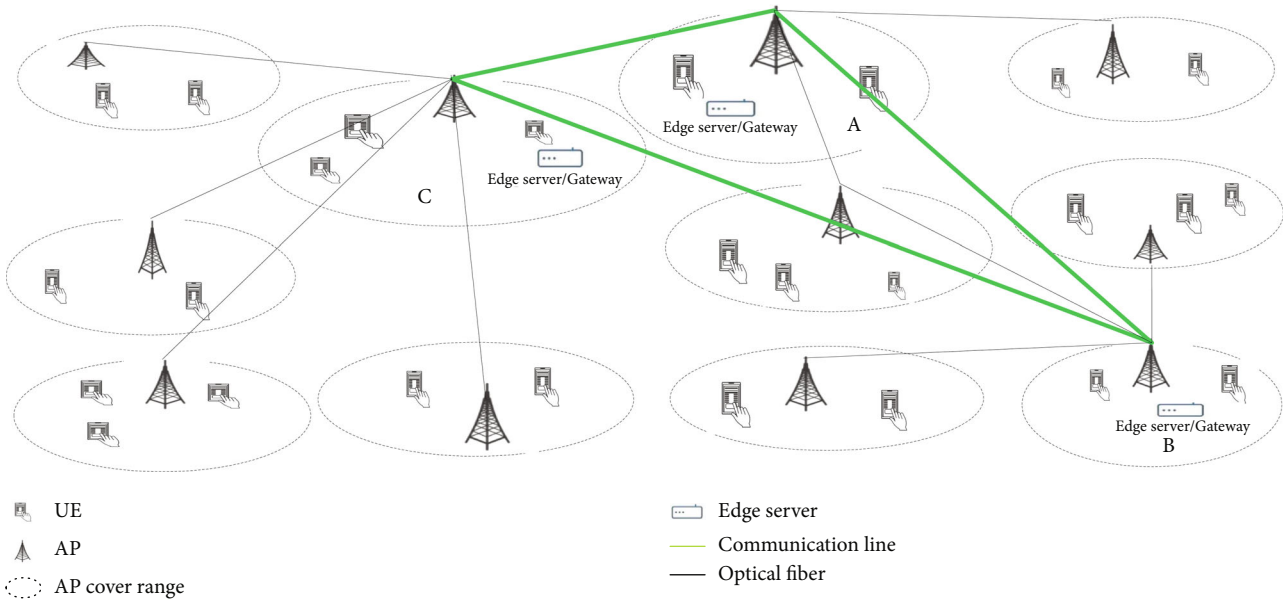

Figure 2: A deployment diagram of edge servers in a WMAN.

Note that it is challenging and costly to directly measure the network distance (delay) between the user and the selected edge location. Therefore, it is a more critical issue in the deployment of edge servers to evaluate the network distance (delay) between the user and the edge location. Geographical coordinate (GC) [18] provides a lightweight network delay evaluation scheme. This work uses the delay level provided by GC to search for the ideal edge position. Like most coordinate systems, this work's distance prediction between geographic coordinate positions is also based on the Euclidean distance calculation model.

Definition 2 (access point coverage area [15]). The coverage area of the base station is the range within which users can generally receive the transmitted signal. The design coverage distance of the base station in the urban area is about 100-200 meters; in the suburbs, it can generally cover a radius of about 3 kilometers; the coverage area of a router or gateway is defined as within a hop. This study defines the maximum distance that users and edge locations can tolerate in the coverage area as $D_{\max}$.

3.3.1. System Model. In the wireless metropolitan area network WMAN $= (V, E)$, $V = I \cup S$, S is the edge server deployment location, and E denotes a link set of access points and edge server potential location. $U = \{u_1, u_2, \cdots u_n\}$ represents the set of all user equipment. Users reasonably use edge computing resources through appropriate access points according to their own needs and geographical location set u_l as lth user, $l \in \{1, 2, \cdots, n\}$.

It is assumed that the edge server and base station or network aggregation equipment (router and switch) and other edge locations are collocated. J is the set of service coverage areas of different network access points, and server clusters are deployed at selected edge locations according to the user distribution density $\{s_1, s_2, \cdots, s_k\}$, $k \geq 1$. $j \in J$ is jth area. Figure 2 describes a server deployment example, including the location information of base stations and user edge servers. As shown in the figure, 11 base stations are used as candidate locations and the dotted line indicates the service area of a server. The user sends a request, and then, the local manager distributes the load and task according to the available resources and user requirements in the edge cluster. The rental price of edge servers deployed in remote areas compared with the central location is relatively low, but the distribution density of users within one hop in the service area is high.

Locations A, B, and C have high user distribution density within one hop of the server, and multiple edge servers are deployed to meet user needs. However, when the scale of access point is large, there are many candidate locations. Choosing the optimal feasible solution is a more complex problem under various constraints such as delay and lease cost.

Note that y_i and x_{ij} are used for illustrating the deployment of edge servers. When the edge server is deployed at the i-th candidate edge position, $y_i = 1$; otherwise, if the edge server is deployed at other edge positions, $y_i \neq 1$. Users' requests in one area may be distributed to different edge MDCs for processing, a continuous variable x_{ij} $(0 \leq x_{ij} \leq 1)$ denotes the load ratio allocated from space j to edge MDC. The nonnegative integer variable χ_i is the number of edge servers located at edge position i. Aiming at low delay and minimum deployment cost, according to users' service requests in different server areas, this method selects K edge locations from candidate set I as the deployment server locations.

3.3.2. Cost of Edge Server Deployment. The overall cost of edge server deployment is determined by two parts: the total

Table 1: Symbol used in this work.

Symbol	Meaning
$WMAN(V, E)$	A metropolitan area network
I	Collection of candidate locations for edge server deployment
J	Base station or router service coverage area
y_i	Decision variable, whether to place an edge server in i
d_{ij}	The delay caused by the access point j and edge position i in the area caused by the access object through the link
f_i	Rental price of edge location i
χ_i	Decision variables, number of servers deployed in edge location i
x_{ij}	Decision variable, load ratio allocated to edge server cluster i in area j
λ_l	User l's request arrival rate
p_{lj}	User request l percentage of staying time in the area
ω_j	Average user request load in area j
k	Total number of selected edge micro data center deployment locations

Input: I:the AP set of base station; U:user set; J: the area set
Output: Y: the deploy site set; χ_i:the server number of the deploy site i
Begin
1: initialize g_i, the server price; f_i, the price of an edge position ; s,the maximum load of a server; c, the largest server number in a single edge position which accommodates
2: initialize $UB = +\infty$, $LB = -\infty$, $k = 0$;
3: do{
4: Select the initial server deployment scenario

$$\overline{y_i} = \begin{cases} 1, & \text{if } d_{ij} \le D, \forall i, j \\ 0, & \text{else} \end{cases}$$

5: In the first step, all nodes in the region j that satisfy the delay condition are selected for initial deployment
6: Initialize the main problem model MP(16)-(17)
7: Compute $C_k = \sum_j \beta_j + \sum_i c\bar{y}_i \gamma_i$ by(14) and (15)
8: if $(C_k < UB) UB = C_k$
9: Solving MP to get the lower bound Lk by Benders cut constraints $\varsigma \le \sum_j \beta_j^* + \sum_i c\bar{y}_i \gamma_i^*$
10: if($lk > LB$) $LB = lk$;
11: if the MP problem has no solution, the original problem has no solution and the algorithm ends.
12: update $\bar{y}_i$ and χ_i by MP's solution
13: $k = k+1$
14: while $((UB - LB)/UB > 0.001 \| k < 100)$
End

Algorithm 1: Edge MDC deployment algorithm based on Benders decomposition (Benders_SD).

user access delay and the edge MDC construction cost. And the construction cost is the cost of resource investment.

(1) Total User Access Delay. When a user access request is distributed to an edge micro data center for processing, it requests to connect to the base station where the edge server is located after passing through the network access point it serves. Therefore, the end-to-end delay between the user and the edge server consists of two parts: the access delay between the user and the access point and the network delay between the access point and the base station where the edge server is located. Since the network delay between the user and the access point is not affected by the edge server deployment location, in this work, we only consider the delay between the access point and the edge location where the edge server is located [16]. Based on advanced SDN network technology, the network controller can monitor the delay d_{ij} [24, 25] between the access point in area j and the location i where the edge server is located.

Due to the mobility of users, the load in different areas changes with time and time. The total delay of user access is affected by the distribution of users, the rate of user

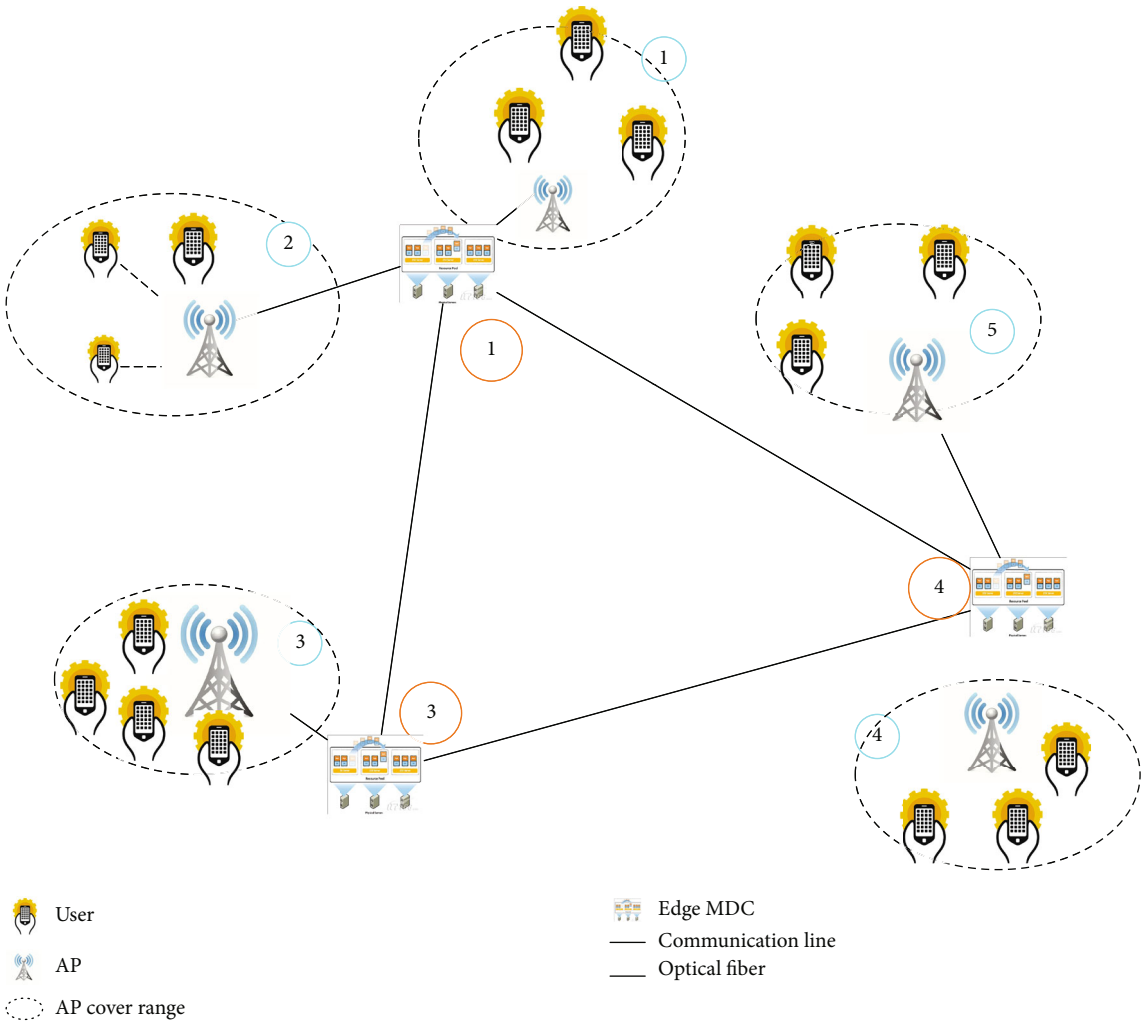

Figure 3: Schematic diagram of edge server deployment example.

Table 2: Rental cost of candidate locations and user request load in the coverage area.

	Location 1	Location 2	Location 3	Location 4	Location 5
Rental price (unit: $)	20000	50000	40000	60000	70000
User load in the coverage area (unit of request quantity: number)	750	1350	1500	3000	3900

requests, and the time the users stay in the area. User distribution characteristics are described by user density. Collect the number of user terminals at time slot τ in area j. The average user density in this area is as in equation (2); ψ_t is the number of user terminals in the coverage area of the first time slot.

$$\overline{\psi_j} = \frac{\sum_{\tau=1}^{\delta} \psi_\tau}{\delta}. \tag{2}$$

According to the average user density in the area ψ_j, the request arrival ratio λ_l of user l and percentage p_{lj} of users stay in area j. The average number of user requests (load) in area j is calculated as

$$\omega_j = \sum_{l \in \{1,2,\cdots,n\}} p_{lj} \lambda_l \overline{\psi_j}. \tag{3}$$

The requests of different users in the area j are distributed to the edge clusters in different locations. d_{ij} is the unit delay caused by accessing the object through the logical link from the access point in the area to the edge location i where the edge server is deployed. The continuous variable $0 \le x_{ij} \le 1$ represents the ratio of the request load

Table 3: The unit access delay of users in the coverage area to the corresponding edge site (unit: ms).

	Location 1	Location 2	Location 3	Location 4	Location 5
Area 1	5	10	15	20	25
Area 2	10	5	10	15	20
Area 3	15	10	5	10	15
Area 4	20	15	10	5	10
Area 5	25	20	15	10	5

Table 4: *UB* and *LB* iteration results.

k	UB	LB	k	UB	LB
1	216584.3	201640	10	204187.8	201737.5
2	212941.1	201655.2	11	203476.6	201747
3	210296. 6	201674.4	12	203387.6	201760.5
4	210087.1	201683.6	13	202988.9	201771
5	209884.3	201698.8	14	202596.9	201786.5
6	209027.1	201711.1	15	202406.2	201799
7	208606.3	201722.3	16	202240.8	201816.1
8	208395.6	201726.9	17	202123.9	201817.6
9	206474.4	201725.5	18	202125	201999

allocated to the edge position i in the area j. The total user access delay is denoted in

$$T_j = \sum_{i \in I} \omega_j x_{ij} d_{ij}. \tag{4}$$

(2) The Edge MDC Construction Cost. Deployment of an edge server needs to select the appropriate location and equip it with infrastructure. The edge MDC construction cost includes the cost of location leasing and the cost of equipment required for deployment. f_i is the set construction cost of edge location i (including leasing structure and other primary resource allocation), g_i is server unit price, and χ_i is the number of server nodes. The edge MDC construction cost is calculated as

$$\text{Cost}_1 = \sum_{i \in I} (f_i + g_i \chi_i) y_i. \tag{5}$$

3.3.3. Edge Server Deployment Modeling. To minimize end-to-end delay, the nearest edge computing resources should be provided to the user after receiving the user's request. As the number of edge micro data centers and their servers deployed increases, the end-to-end latency of user request processing will decrease correspondingly, resulting in an increase in the capital investment cost of edge server deployment. It is crucial for the edge facility provider to balance the capital investment cost and end-to-end network delay. Therefore, this work proposes a strategy [20] to use the lowest capital investment cost and minimize end-to-end network delay for satisfying users. Edge server deployment cost includes the sum cost of edge MDC construction cost and total access delay, which can be defined as shown in

$$\Gamma = \sum_{i \in I} (f_i + g_i \chi_i) y_i + \varsigma \sum_{i \in I} \sum_{j \in J} \omega_j x_{ij} d_{ij}. \tag{6}$$

Note ς is the adjustment constant used to adjust the total access delay cost and the proportion of edge server investment cost. The definition ς is shown in

$$\varsigma = \frac{\left\lceil \theta_2 \sum_{i=1}^{k} (f_i^{\max} + g_i c) \right\rceil}{\left\lceil \theta_1 \sum_j \omega_j d_j^{\max} \right\rceil}. \tag{7}$$

$\varsigma > 0$, $d_j^{\max}$ is the maximum delay of the farthest edge server in area j, and c is the maximum number of servers in the edge location. $\sum_{i=1}^{k} f_i^{\max} + g_i c$ indicates the highest investment cost. $\sum_j \omega_j d_j^{\max}$ represents the maximum total delay of all user requests in area j, θ_1 and θ_2 denote balance parameter, $\theta_1 + \theta_2 = 1$, and $\theta_1, \theta_2 \in [0, 1]$. Therefore, the comprehensive cost minimization model of edge server deployment [25] can be described as

$$\mathbf{P1}: \min \left(\sum_{i \in I} (f_i + g_i \chi_i) y_i + \varsigma \sum_{i \in I} \sum_{j \in J} \omega_j x_{ij} d_{ij} \right) \tag{8}$$

$$\begin{aligned} &\text{s.t.} \\ &\text{C1}: \sum_{j \in J} \omega_j x_{ij} \le s \chi_i, && \forall i \in I \\ &\text{C2}: \sum_{i \in I} x_{ij} = 1, && \forall j \in J \\ &C3: \chi_i \le c y_i, && \forall i \in I \\ &\text{C4}: \sum_i y_i \le k, && \forall i \in I \\ &\text{C5}: x_{ij} \in [0, 1] && \forall i \in I, \forall j \in J \\ &\text{C6}: y_i \in \{0, 1\}, && \forall i \in I \\ &\text{C7}: \chi_i \in Z_0^+, && \forall i \in I. \end{aligned} \tag{9}$$

Among them, s is the available capacity of a single edge server related to user request load, and c is the maximum value of the server at the edge location. It is obvious that the objective function equation (8) works for minimizing these two kinds of cost within the metropolitan area network. Constraint (C1) restricts the assigned tasks from exceeding the maximum load of the edge server cluster at that location. Constraint (C2)

Table 5: Parameter setting.

Parameter	Value
Number of user requests at each access point	[50,200]
Time slot length (minutes)	10
Latency of edge location and client access point (ms)	[5,50]
Number of candidate edge positions	{200,400,600,800,1000}
The unit price of a server ($)	1000
The maximum number of servers in an edge micro data center	10
Maximum load of a server (number of requests processed)	50
Edge location rental cost ($)	[10000,80000]
Equalization parameters θ_1 and θ_2	{0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9}

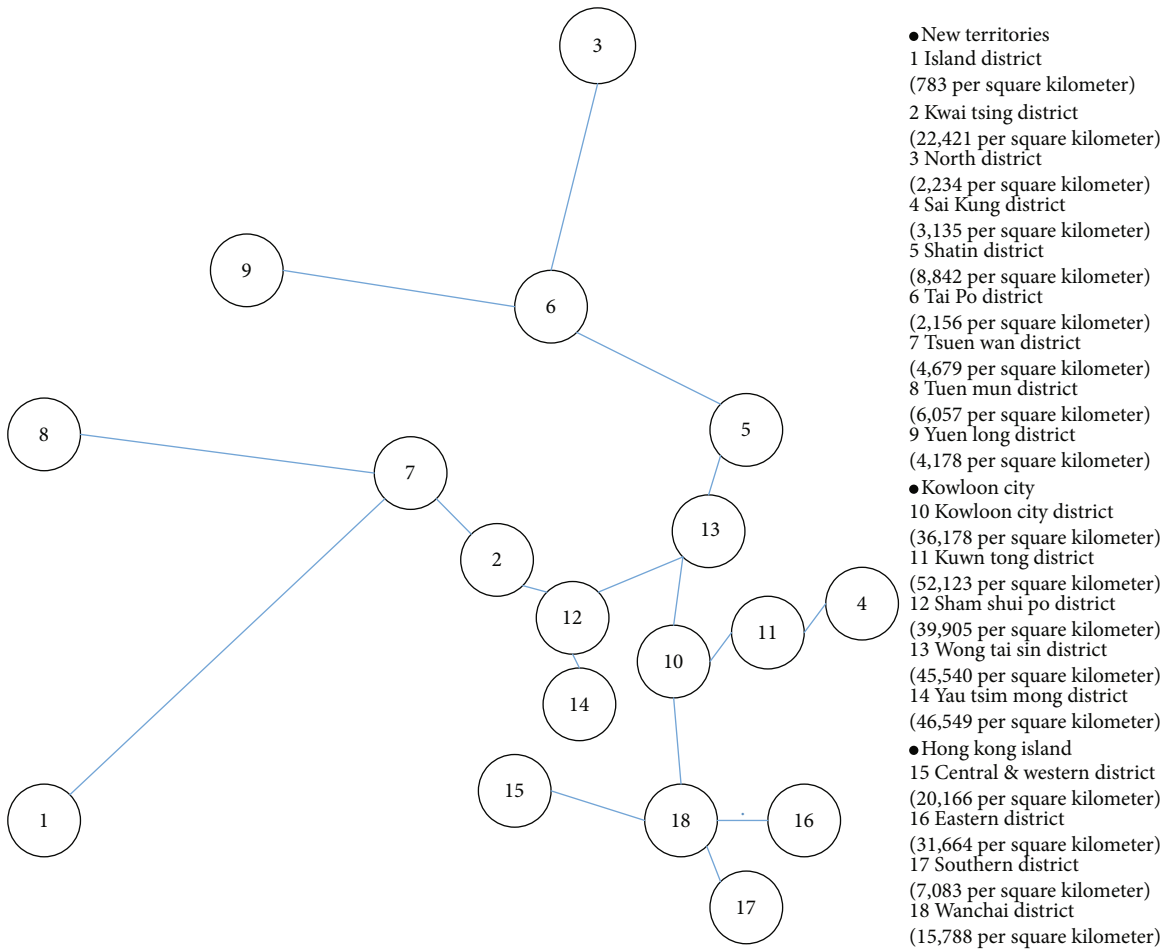

Figure 4: Hong Kong subway map.

ensures that all regional loads are distributed to different edge server clusters. The load ratio of edge server m in area j is a continuous variable between 0 and 1. Constraints (C3) and (C4) ensure that the total number of locations deployed in the metropolitan area network do not exceed the maximum limit. Constraints (C5-C7) define the value range of variables. y_i is an integer binary decision variable, χ_i is an integer decision variable, x_{ij} as a continuous decision variable. Since the objective function equation (8) contains product two decision variables χ_i and y_i, so the model is nonlinear. Here, the discrete 0-1 integer variable y_i increases the difficulty of the solution and is regarded as a "complex variable." Through the model analysis of the edge server deployment cost minimization problem, it can be seen that the variable of the number of servers in a certain location is an integer

Table 6: The characteristics of the data set used in existing research.

Research	Data set	Characteristic
[6]	Data set collected by Shanghai Telecom base station	The data set of Shanghai Telecom's base station, which contains Internet information of mobile users accessing 3,233 base stations, has 3000 practical base stations. The data set includes the exact start and end times of each mobile user's base station visit.
[15]	Specify the number of geographic areas and edge locations	For a network with a coverage area of 80 square kilometers, each base station covers an area of 4 square kilometers; the number of edge locations is 20; the number of users is 1,000; the average user request arrival rate is 2, and the variance is a normal distribution of 0.5.
[17]	Hong Kong Metro wireless network topology, synthetic data set	The deployment scope is a wireless metropolitan area network; the edge position of the real data set is fixed (18); the edge position of the GT-ITM synthetic data set varies from (200 to 1000), and the edge-to-edge connection probability is 0.02. Assume that the number of edge MDCs is 10% of the network scale; the random value of the number of user requests for each AP access point is [50,500].
[18]	Computer clusters distributed around the world, PlanetLab project and measurement nodes deployed in mainland China	The deployment scope of the edge location is the worldwide network topology and the mainland China-wide network topology; the user demand delay is divided into 50-90 ms and 20-40 ms based on the geographical range; the number of users is 1,116,000 and 20,000, respectively, based on the geographical range.
[21]	Hong Kong Metro wireless network topology, synthetic data set	The scope of deployment is wireless metropolitan area network (WMAN); the edge position of the real data set is fixed (18); the network topology in the synthetic data set is a randomly generated nonscale network; the number of users is 150 per AP.
[26]	Synthetic network topology	The range of wireless metropolitan area network; the variation range of candidate edge position is 200 to 1000, the probability of connection between each AP is 0.02; the edge delay is randomly distributed in [5,50]ms; the number of user requests for each AP is randomly [50,500]; the amount of resources requested by each user is [50,200] MHz.

variable. The load distribution variable of a certain edge location in the area is continuous, and the edge server deployment model is a MINP problem.

According to the above analysis, edge server deployment mainly includes two parts: edge positioning based on the proximity of users and edge locations and determination of the number of edge server nodes based on user distribution and deployment costs. In this paper, the multiple edge server deployment issues studied include not only the user's attribution but also the spatial positioning of the edge server within the metropolitan area network. Several locations are selected among the candidate network access points, and the user is assigned to this location, which can be summarized as a capacity-limited multifacility positioning problem in discrete space (capacitated facility location problem (CFLP)). Since this problem involves thousands of network access points in the metropolitan area network, the problem scale is relatively large and an NP-complete combinatorial optimization problem. Therefore, selecting an appropriate method for solving integer planning is a vital issue to ensure the accuracy and efficiency of the optimal deployment of edge servers. The symbols used in this work and their meanings are shown in Table 1.

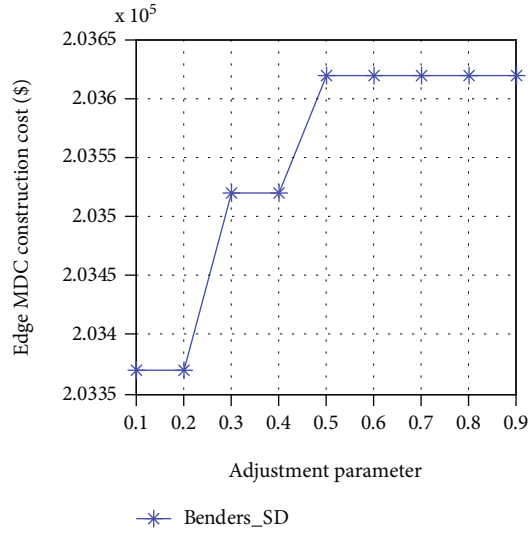

Figure 5: The comparison of edge MDC construction cost as the parameter value varies.

3.4. Linearization Process of Original Problem. For integer nonlinear programming problem P1, the original problem is transformed into the mixed-integer linear programming problem (MIP) P2 by linear transformation $\varphi_i = y_i\chi_i$.

$$\mathrm{P2}: \min\left(\sum_{i\in I} f_i y_i + \sum_{i\in I} g_i\phi_i + \varsigma\sum_{j\in J}\sum_{i\in I}\omega_j x_{ij} d_{ij}\right) \tag{10}$$

$$\begin{aligned}
&\text{s.t.}\\
&\mathrm{C1}: \sum_{j\in J}\omega_j x_{ij} \le s\chi_i, && \forall i \in I\\
&\mathrm{C2}: \sum_{i\in I} x_{ij} = 1, && \forall j \in J\\
&\mathrm{C3}: \chi_i \le c y_i, && \forall i \in I\\
&\mathrm{C4}: \sum_i y_i \le k, && \forall i \in I\\
&\mathrm{C5}: \phi_i\text{-}\chi_i \le 0, && \forall i \in I\\
&\mathrm{C6}: \phi_i \le c y_i, && \forall i \in I\\
&\mathrm{C7}: x_{ij} \ge 0 && \forall i \in I, \forall j \in J\\
&\mathrm{C8}: y_i \in \{0,1\}, && \forall i \in I\\
&\mathrm{C9}: \phi_i \in Z_0^+, && \forall i \in I\\
&\mathrm{C10}: \chi_i \in Z_0^+, && \forall i \in I.
\end{aligned} \tag{11}$$

Generally, the solution to the NP-complete problem can use accurate and approximate algorithms. Commonly used accurate algorithms for solving MIP include the branch and bound method and Benders decomposition method. Approximate algorithms include heuristic algorithms and intelligent optimization algorithms. The branch and bound method is a deterministic algorithm based on search and iteration with a large calculation. The well-known commercial software standard mathematical programming optimizer CPLEX, based on the branch and bound method, combined with cutting planes, heuristics, and other technologies, can quickly solve mixed-integer linear programming problems. Currently, it has been applied in solving the facility location problem, but CPLEX can obtain the optimal solution for small and medium-scale mixed-integer programming problems. The scale of edge server deployment problems in the metropolitan area network environment is relatively large, and CPLEX takes too much time. Get the optimal solution, and it may even be impossible to get a feasible solution. The Benders decomposition algorithm shows performance better to solve the MIP problem [26–29]. Thus, this work uses the Benders decomposition algorithm to solve this problem.

4. Benders Decomposition of Edge Server Deployment Problems

The Benders decomposition [26–29] algorithm is suitable for solving mixed-integer programming problems; it decomposes the original problem into the main problem containing complex integer decision variables and the subproblems containing only continuous variables according to the different types of variables. So it is suitable to apply this algorithm to solve the edge server deployment problem. In solving the main problem and the subproblems iteratively, the main problem provides a lower bound for the original problem. The obtained integer solution is passed to the subproblem, and the subproblem provides an upper bound for the original problem and returns to Benders cut to the main problem. The algorithm stops when the main and subproblems alternate solve until the upper and lower bounds are equal. At this time the optimal solution to the original problem is obtained.

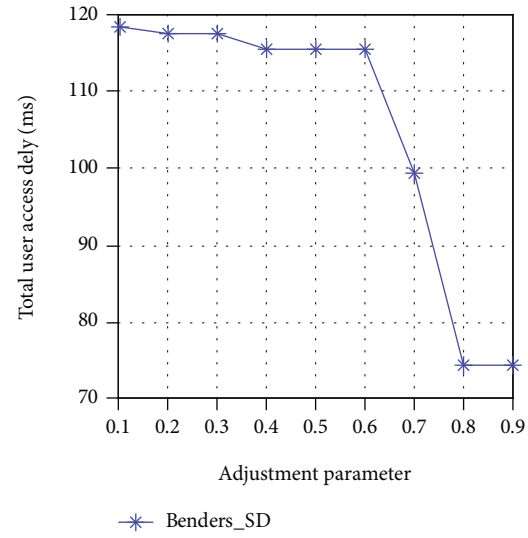

Figure 6: The comparison of total end-to-end delay as the parameter value varies.

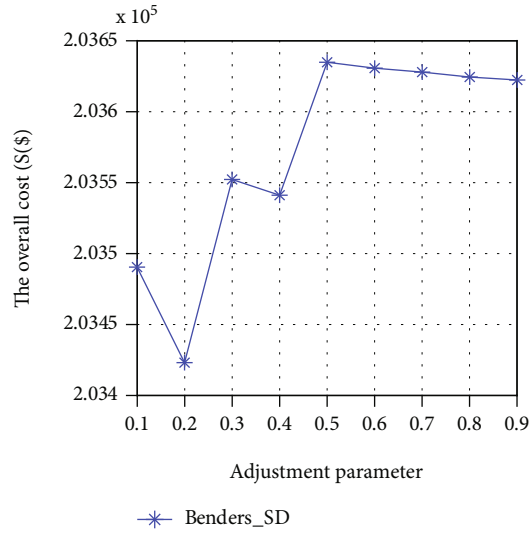

Figure 7: The comparison of the overall cost as the parameter value varies.

4.1. Subproblems of Benders Decomposition Algorithm. The fixed 0-1 integer problem variable y_i decomposes $\overline{y_i}$ the subproblem P3:

$$\mathrm{P3}: \min_{x,\phi}\left(\sum_i g_i\phi_i + \varsigma\sum_{j\in J}\sum_{i\in I}\omega_j x_{ij} d_{ij}\right) \tag{12}$$

$$\begin{aligned} &\text{s.t.}\\ &\mathrm{C1}: \sum_{j\in J}\omega_j x_{ij} \le s\phi_i, \qquad \forall i \in I\\ &\mathrm{C2}: \sum_{i\in I} x_{ij} = 1, \qquad \forall j \in J\\ &\mathrm{C3}: \phi_i \le c\overline{y_i}, \qquad \forall i \in I\\ &\mathrm{C4}: x_{ij} \ge 0, \qquad \forall i \in I, \forall j \in J. \end{aligned} \tag{13}$$

Define the dual variable of the constraint (C1) as $\alpha = \{\alpha_i \ge 0 | i \in I\}$, the dual variable of the constraint (C2) as $\beta = \{\beta_j | j \in J\}$, and the dual variable of the constraint (C3) as $\gamma = \{\gamma_{\mathrm{i}} \ge 0 | i \in I\}$. Substitute P3; then, $\max - (\sum_{\mathrm{i}} g_i\phi_i + \sum_{j\in J} \sum_{i\in I}\omega_j x_{ij} t_{ij}) + \alpha_i(\sum_{j\in J}\omega_j x_{ij} - s\phi_i) + \beta_j(\sum_{i\in I} x_{ij} - 1) + \gamma_{\mathrm{i}}(\phi_{\mathrm{i}} - c\overline{y_i})$. Then, the dual problem P4 of P3 is

$$\mathbf{P4}: \max \sum_j \beta_j + \sum_i c\overline{y_i}\gamma_i \tag{14}$$

$$\begin{aligned} &\text{s.t.}\\ &\mathrm{C1}: \alpha_i\omega_j + \beta_j - \omega_j t_{ij} \ge 0, \quad i \in I, j \in J\\ &\mathrm{C2}: \gamma_i - g_i - sa_i \ge 0, \qquad i \in I\\ &\mathrm{C3}: \alpha_i \le 0\\ &\mathrm{C4}: \gamma_{\mathrm{i}} \le 0. \end{aligned} \tag{15}$$

Suppose P3 has a feasible solution, according to the duality principle. In that case, dual problem P4 has a bounded solution, and the bounded solution is the pole of a polyhedron composed of constraints (C1) and (C2), and then, an optimal Benders cut can be obtained. If P3 is not feasible, then the dual problem P4 is unbounded. Then, a feasible Benders cut can be obtained for a polar ray. From this, the optimal Benders cut of P3 is $\varsigma \le \sum_j \beta_j^* + \sum_{\mathrm{i}} c\overline{y_i}\gamma_i^*$. The feasible Benders cut is $\sum_j \beta_j^* + \sum_i c\overline{y_i}\gamma_i^* \ge 0$, $(\beta_j^*, \alpha_i^*, \gamma_i^*) \in P_\Omega$ is a pole of the polyhedron Ω, and $(\beta_j', \alpha_i', \gamma_i') \in Q_\Omega$ is a polar ray of a polyhedron Ω. Suppose ς is an auxiliary decision variable for Benders' main problem, the optimal Benders cut can raise the lower bound of the Benders main problem, and the feasible Benders cut will get the effective lower bound of the original problem. Since generating the optimal Benders cut will speed up the convergence speed of the Benders decomposition algorithm, having more optimal Benders cuts and limiting the feasible Benders cuts is an effective way to accelerate the decomposition algorithm.

4.2. Main Problem of Benders Decomposition. Based on the optimal and feasible Benders cut, the main problem MP is

$$\mathbf{MP}: \min\left(\sum_i f_i y_i + \varsigma\right) \tag{16}$$

$$\begin{aligned} &\text{s.t.}\\ &\mathrm{C1}: \sum_i y_i \le k,\\ &\mathrm{C2}: \varsigma \le \sum_j \beta_j^* + \sum_i c\overline{y_i}\gamma_i^*\\ &\mathrm{C3}: \sum_j \beta_j^* + \sum_i c\overline{y_i}\gamma_i^* \ge 0\\ &\mathrm{C4}: \varsigma \ge 0\\ &C5: y_i \in \{0, 1\}, \forall i \in I. \end{aligned} \tag{17}$$

Although equation (17) considers many linear constraints in theory, only a small part of these constraints are active constraints at the optimal solution. Therefore, the direction constructs a relatively simple form of expression by utilizing the poles and extremes corresponding to these constraints.

5. Implementation of Edge Server Deployment Algorithm Based on Benders Decomposition

5.1. Algorithm Implementation. The edge MDC deployment algorithm based on Benders decomposition proposed in our work is shown in Algorithm 1 [25]. It can be seen from Algorithm 1 that in the second row, the maximum upper limit UB and the minimum lower limit LB are initialized. A feasible initial position is selected. In the iterative process of Algorithm 1, the dual problem C_k provides an upper bound for the original problem in line 7 and returns the Benders cut to the MP problem to constrain the main problem and update UB to form a new main problem for solving edge server configuration. In line 4, the optimal solution of the main problem MP is to provide a lower limit for the original problem. Since UB does not necessarily decrease at each iteration, in line 5, the upper limit is selected as $UB = \min(C_k, UB)$; then, update LB. In addition, in order to avoid the MP main problem being unlimited in the first few iterations, many cuts were generated in the feasible solution initially added to MP.

5.2. Algorithm Correctness Analysis. The edge server deployment model is a mixed-integer nonlinear programming problem (MINP). The Benders decomposition (including the generalized Benders decomposition) algorithm is a method to solve the problem by decomposing the MINP problem according to the duality theory. According to the different data types of variables, the decomposition algorithm first linearizes the nonlinear programming problem of edge server deployment and then decomposes the mixed-integer programming problem into main and subproblems and iteratively solves them. The main problem MP is used to solve the location of server deployment, the number of servers in each deployment location of the subproblem SP, and the ratio of server resource allocation. During the iterative solution process, the lower limit LB of the main problem MP and the upper limit UB of the subproblem SP are constantly updated. According to the difference between the upper limit and the lower limit of the main

problem and the subproblem and the result of the subproblem, different Benders cut constraints are formed and added to the MP, and they are optimized and corrected until the conditions are met, and the optimal solution is obtained. It has been proved by literatures [26–28] that the Benders decomposition algorithm can achieve convergence in a limited number of steps. Obviously, choosing the Benders decomposition algorithm can efficiently solve the problem of optimal deployment of edge servers.

6. Example of Edge Server Deployment

Suppose there are five candidate locations for edge server deployment (such as base station or router locations), and each location corresponds to a coverage area as shown in Figure 3. In 10 years, the rental cost ($) of each location and the number of user requests within 5 seconds of the corresponding coverage area is shown in Table 2. The size of each request content is 100 M. The number of edge servers deployed at each edge location is no more than 30, and the unit price of each server is $2000. The maximum processing capacity of each edge server is 300 requests/time. Table 3 shows the average unit access delay corresponding to each edge site of the user set in each area. There are three decision variables y_i, $x_{i,j}$, and χ_i; y_i is a binary decision variable.

Using Benders to solve the edge server deployment, after 18 iterations, $(UB - LB)/UB = 0.00062 < 0.001$. The upper bound value of the objective function is 202125, and the lower bound of the objective function is 201999. The changes of UB and LB during the specific execution are shown in Table 4. During the entire edge server deployment period, the 10 servers are deployed in the first location, and the load distribution of area 1 and area 2 to the edge MDC_1 is 100%. Four servers are deployed in the third location, and the load distribution ratio of area 3 to the edge MDC_2 is 100%. 22 servers are deployed in the fourth location, and the load distribution ratio of areas 4 and 5 to the edge MDC_2 is 100%.

7. Performance Evaluation

This simulation [20] is carried out on a personal laptop equipped with Inter (R) Core (TM) i7-3770 CPU@3.40 GHz processor, RAM 12.0 GB memory, and 1 T hard disk space. The algorithm is programmed in C++ language. The simulation results were independently performed 25 times under the same conditions, and then, the average value was taken.

7.1. The Setup of Simulation. This section evaluates the performance of the proposed algorithm based on real and synthetic network topology data sets [17, 20, 21]. The amount of resources requested by each user is a random value in the range of [50,200] MHz. Each server can handle up to 50 requests. And the edge delay is randomly generated between 5 ms and 50 ms. Assume that mobile users usually stay in several places most of the day, such as home and work. Therefore, this work assumes that the location of each user in a specific area covered by five BSs changes randomly. Assume that the maximum number of edge micro data centers is 10% of the number of network access points. These main parameter settings are shown in Table 5.

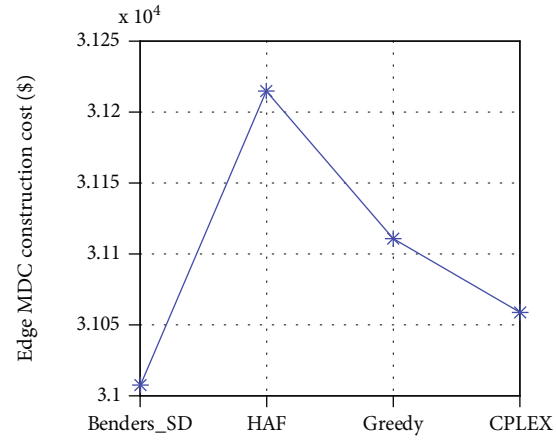

Figure 8: Edge MDC construction cost of four algorithms.

7.1.1. Data Set. This simulation refers to the experimental settings of the Australian National University [17], the Hong Kong Polytechnic University Cao JN team [21], and the literature [18]. The real data set comes from the network topology of the Hong Kong Metro (HKMTR), including 18 in Hong Kong. The region corresponds to 18 potential edge locations. The number of requests in each area is directly proportional to the number of people in the AP coverage area. Figure 4 shows the Hong Kong subway map used as a WMAN template. Although the network topology of the Hong Kong area is not public, the Hong Kong subway map is used to infer the wired connections between hubs in each area to represent the wired hub in WMAN to the edge of the hub. This paper conducts a comparative analysis of related data sets to test the algorithm's adaptability, as shown in Table 6.

7.1.2. Comparison Algorithm. To evaluate the performance of the proposed algorithm Benders_SD, this paper selects the heaviest load priority placement (HAF [21]), greedy algorithm [17], and CPLEX algorithm for comparative analysis.

(1) Heaviest Load Priority Placement (HAF). HAF deploys the edge micro data center with the heaviest user load at the network access point. First, sort the locations of base stations or edge routers from large to small according to the accumulated request reach rate of users, and deploy edge computing resources at the edge positions of the first k locations. However, the HAF algorithm has two main disadvantages: first, the access point with the heaviest workload is not always the closest to the user; second, assigning users to the nearest edge MDC will cause uneven user distribution, which will lead to load imbalance in some edge MDCs.

(2) Greedy Algorithm (Greedy). The deployment strategy of the greedy algorithm is to select edge sites one by one from the candidate edge locations. It selects a site that achieves

the minimum and maximum delay time of user server round-trip in the first round. The rest selects k-1 edge sites in the k-1 round. According to this strategy, when the selected edge site meets the bandwidth requirements of all users, the selection process ends.

(3) CPLEX Mixed-Integer Programming Optimization Algorithm. IBM's WebSphere ILOG CPLEX algorithm can realize the basic algorithm with the fastest speed and the most reliability. CPLEX provides a flexible high-performance optimization program to solve problems such as mixed-integer planning.

7.1.3. Performance Parameters. Performance evaluation indicators include edge MDC construction cost ($), total user access delay (s), and overall cost. The overall cost is derived from equation (6). The unit of comprehensive cost is delay (s) and creation cost ($) joint decision, represented by S ($) in this work.

7.2. Results and Analysis

7.2.1. Sensitivity Test of Parameter θ_1. The values θ_1 of the delay sensitivity parameters are, respectively, {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}, and θ_1 gradually increases, indicating that the more sensitive to delay, this simulation sets 200 candidate edge positions.

It can be seen from Figure 5 that with the gradual increase of the adjustment parameter θ_1, the construction cost of the edge MDC gradually increases. Figure 6 describes the process in which the total user access delay gradually decreases with the parameter increase θ_1. The larger one makes the edge server deployment more sensitive to the end-to-end delay. When θ_1 increasing, the algorithm proposed in this work pays more attention to the delay cost; thus, more servers are needed to reduce the delay and the corresponding edge MDC construction cost increases. It can be seen that adjusting the parameter has a more significant impact on the algorithm results. Figure 7 depicts the overall cost change with the adjustment parameter θ_1. The overall cost is the smallest when $\theta_1 = 0.2$.

Therefore, this work comprehensively considers the comprehensive benefits of edge computing service providers and users to minimize the comprehensive cost and set the system adjustment parameter θ_1 to 2.

7.2.2. Performance Evaluation of MDC Deployment Algorithm at the Lower Edge of the Small Network Service Area. This group of simulations uses the real Hong Kong subway network HKMTR data set to evaluate the edge server deployment algorithm proposed in this work. There are eighteen AP access points, and three of them are selected as deployment locations. This group of simulations relies on the results of parameter sensitivity simulations and sets system adjustment parameter $\theta_1 = 0.2$.

Figure 8 shows the creation cost of edge MDC under the four algorithms of Benders_SD, HAF, Greedy, and CPLEX. However, compared with HAF Greedy and CPLEX, Benders_SD reduces the MDC construction cost by an average of 200$, 100$, and 50$, respectively. It can be seen that

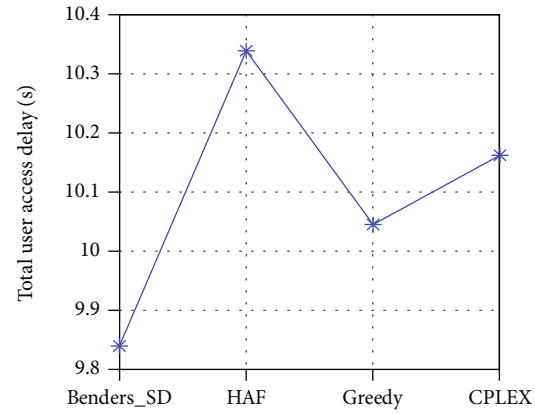

Figure 9: The total end-to-end delay cost of the four.

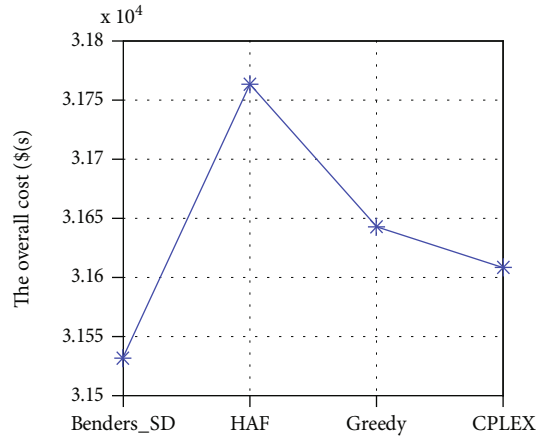

Figure 10: The overall cost.

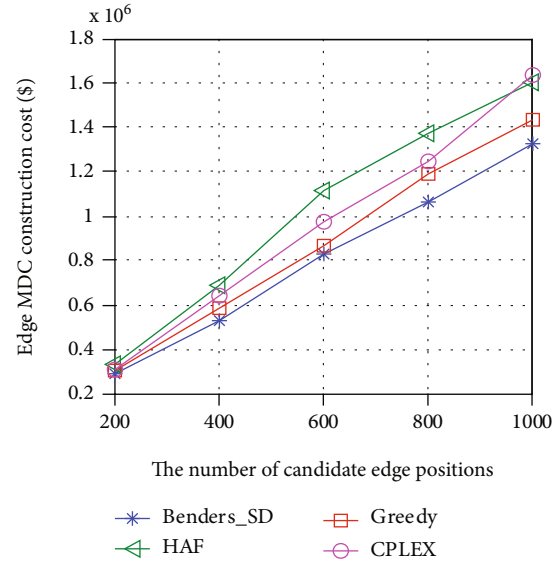

Figure 11: The comparison of Edge MDC creation cost as the number of candidate edge positions varies.

the Benders_SD algorithm proposed in this work has the lowest construction cost, which is better than the other three comparison algorithms. The HAF algorithm has the highest server deployment cost.

Figure 9 depicts the total end-to-end access latency cost under the four algorithms. Compared with HAF Greedy and CPLEX, Benders_SD reduces the total user access delay by an average of 0.51 s, 0.21 s, and 0.33 s, respectively. At the same time, Benders_SD reduces the total cost by an average of 227.26, 111.23, and 67.64, respectively. The total cost of the Benders_SD algorithm is lower for HAF, Greedy, and CPLEX algorithms in Figure 10. It is shown that the Benders_SD proposed in this work outperforms than the others.

Based on the above comparison results, from the evaluation results of the four algorithms on the HKMTR data set, the Benders_SD proposed in this work has the best performance in three aspects: edge MDC creation cost, end-to-end delay, and total cost. It can minimize the cost of edge computing infrastructure providers and the end-to-end delay of user access.

7.2.3. Evaluation of Edge MDC Deployment Algorithm under Different Number of Candidate Edge Positions. This group of simulations uses a synthetic network data set, the network scale becomes larger, and the number of candidate edge positions in the network changes from 200 to 1000. The range of change in the number of user requests for each candidate edge location (AP access point) is [50,200].

It can be seen from Figure 11 that when the number of candidate edge positions changes from 200 to 1000, the creation cost of edge MDC gradually increases. The creation cost of the Benders_SD edge server deployment algorithm proposed in this work is significantly lower than the cost of HAF and CPLEX, which is slightly close but still better than the Greedy algorithm. When the number of candidate edge positions is equal to 800, the creation cost of the Benders_SD algorithm is 125,819$ less than Greedy and 307,839$ than HAF; when the number of candidate edge positions is 1000, it is 309,072$ less than CPLEX. This shows that as the number of candidate edge positions increases, the Benders_SD algorithm has more advantages.

Figure 12 describes that as the number of candidate edge positions increases, the total end-to-end delay also increases, and Benders_SD outperforms other algorithms. As the problem scale becomes larger, the CPLEX algorithm's performance worsens. Figure 13 describes the increase in the number of candidate edge positions as the network size increases. The performance of the four algorithms on the overall cost is consistent with the trend of edge MDC creation and the overall end-to-end latency cost. Compared with the three comparison algorithms, the Benders_SD algorithm performs best.

8. Conclusion

The emergence of edge computing plays a crucial role in low latency IoT applications. A MAN contains a large number of base stations that serve as candidate deployment locations for edge servers. Selecting a location for the edge server and determining the number of servers in the location for low latency and high node utilization is an urgent problem to be solved. This work proposes a cost-aware edge server optimization deployment method based on the Benders decomposition algorithm. An objective function is established based on edge server deployment and access cost minimization by using the resource allocation ratio, regional average load, access delay between users, and edge node serving location. Compared with the traditional server deployment strategy, our optimal strategy can more accurately decide the edge MDC's location and the number of each edge server to ensure low latency and low deployment costs.

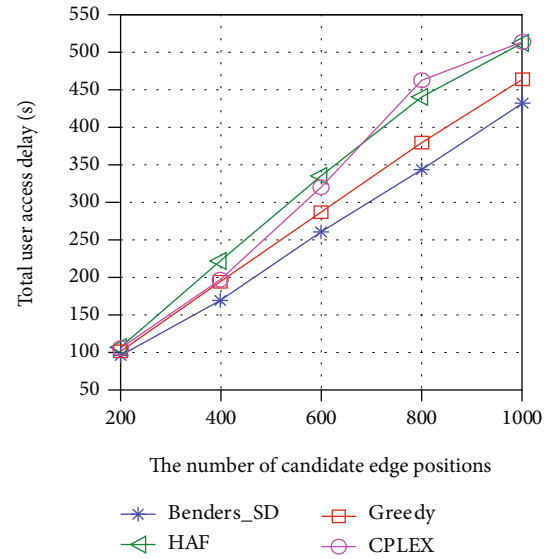

Figure 12: The comparison of total end-to-end delay as candidate edge position varies.

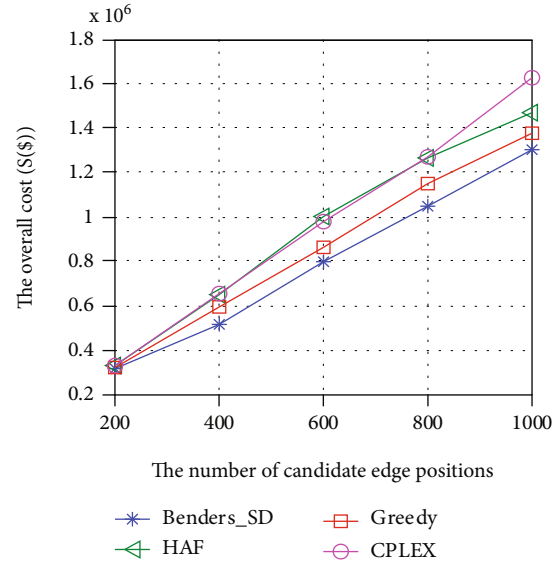

Figure 13: The comparison of total cost as candidate edge position varies.

In further research, the optimal allocation and deployment of edge computing resources for complex and diverse Internet of things services will be studied from multiple perspectives around computing offloading, resource allocation, and cache content placement to improve system performance, edge service quality, and user experience.

Data Availability

The data that support the findings of this study are available from the corresponding author Shao, upon reasonable request.

Disclosure

Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the above agencies.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work was supported by the National Natural Science Foundation of China (NSFC) under grants (No. 62102200) and Henan Science and Technology Research Project (No. 222102210134), cross science research project of Nanyang Institute of Technology.

References

[1] C. Chen, Y. Zhang, Z. Wang, S. Wan, and Q. Pei, "Distributed computation offloading method based on deep reinforcement learning in ICV," *Applied Soft Computing*, vol. 103, pp. 107108–107111, 2021.

[2] A. D. Boursianis, M. S. Papadopoulou, A. Gotsis et al., "Smart irrigation system for precision agriculture-the ARETHOU5A IoT platform," *IEEE Sensors Journal*, vol. 21, no. 16, pp. 17539–17547, 2021.

[3] J. Shuja, M. A. Humayun, W. Alasmary, H. Sinky, E. Alanazi, and M. K. Khan, "Resource efficient geo-textual hierarchical clustering framework for social IoT applications," *IEEE Sensors Journal*, vol. 21, no. 22, pp. 25114–25122, 2021.

[4] W. Wei, R. Yang, H. Gu, W. Zhao, C. Chen, and S. Wan, "Multi-objective optimization for resource allocation in vehicular cloud computing networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 2021, pp. 1–10, 2021.

[5] S. Zhou, W. Jadoon, and J. Shuja, "Machine learning-based offloading strategy for lightweight user mobile edge computing tasks," *Complexity*, vol. 2021, Article ID 6455617, 11 pages, 2021.

[6] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C. Hsu, "Edge server placement in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 160–168, 2019.

[7] Y. Qu, L. Wang, H. Dai et al., "Server placement for edge computing: a robust submodular maximization approach," *IEEE Transactions on Mobile Computing*, vol. 2021, pp. 1–16, 2021.

[8] L. Zhao, C. Wang, K. Zhao, D. Tarchi, S. Wan, and N. Kumar, "INTERLINK: a digital twin-assisted storage strategy for satellite-terrestrial networks," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 2022, pp. 1–14, 2022.

[9] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "ETSI white paper no. 11: mobile edge computing: a key technology towards 5G," http://www.etsi.org/images/files/ETSIWhitepapers/etsiwp11mecakeytechnologytowards5g.pdf.

[10] G. Brown, "Mobile edge computing use cases& deployment options," https://www.juniper.net/assets/uk/en/local/pdf/whitepapers/2000642-en.pdf.

[11] S. K. Kasi, M. K. Kasi, K. Ali et al., "Heuristic edge server placement in industrial internet of things and cellular networks," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10308–10317, 2021.

[12] Y. Guo, S. Wang, A. Zhou, J. Xu, J. Yuan, and C. H. Hsu, "User allocation-aware edge cloud placement in mobile edge computing," *Software: Practice and Experience*, vol. 50, no. 5, pp. 489–502, 2020.

[13] B. Li, P. Hou, H. Wu, and F. Hou, "Optimal edge server deployment and allocation strategy in 5G ultra-dense networking environments," *Pervasive and Mobile Computing*, vol. 72, p. 101312, 2021.

[14] B. Cao, S. Fan, J. Zhao et al., "Large-scale many-objective deployment optimization of edge servers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3841–3849, 2021.

[15] Q. Fan and N. Ansari, "Cost aware cloudlet placement for big data processing at the edge," in *IEEE International Conference on Communications*, pp. 1–6, 2017.

[16] X. Li, F. Zeng, G. Fang, Y. Huang, and X. Tao, "Load balancing edge server placement method with QoS requirements in wireless metropolitan area networks," *IET Communications*, vol. 14, no. 21, pp. 3907–3916, 2020.

[17] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Efficient algorithms for capacitated cloudlet placements," *Transactions on Parallel & Distributed Systems*, vol. 27, no. 10, pp. 2866–2880, 2016.

[18] H. Yin, X. Zhang, H. Liu et al., "Edge provisioning with flexible server placement," *IEEE Transactions on Parallel & Distributed Systems*, vol. 28, no. 4, pp. 1031–1045, 2017.

[19] B. Ahat, A. C. Baktır, N. Aras, İ. K. Altınel, A. Özgövde, and C. Ersoy, "Optimal server and service deployment for multi-tier edge cloud computing," *Computer Networks*, vol. 199, p. 108393, 2021.

[20] Y. Shao and W. Dong, "Considering user distribution and cost awareness to optimize server deployment," in *CCF Conference on Big Data*, vol. 1120, pp. 135–147, Singapore, 2019.

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

[22] H. Xiang, X. Xu, H. Zheng et al., "An adaptive cloudlet placement method for mobile applications over GPS big data," in *Global Communications Conference*, pp. 1–6, 2017.

[23] T. Lähderanta, T. Leppänen, L. Ruha et al., "Edge computing server placement with capacitated location allocation," *Journal of Parallel and Distributed Computing*, vol. 153, pp. 130–149, 2021.

[24] C. Yu, C. Lumezanu, A. Sharma, Q. Xu, G. Jiang, and H. V. Madhyastha, "Software-defined latency monitoring in data center networks," in *International Conference on Passive and Active Network Measurement, Cham: Springer Verlag*, vol. 8995, pp. 360–372, 2015.

[25] P. Francis, S. Jamin, C. Jin, and Y. Jin, "Idmaps: a global internet host distance estimation service," *IEEE/ACM Transactions on Networking*, vol. 9, no. 5, pp. 525–540, 2001.

[26] A. M. A. Costa, *Survey on Benders Decomposition Applied to Fixed-Charge Network Design Problems*, vol. 32, no. 6, 2005Elsevier Science Ltd., 2005.

[27] J. N. Hooker, "Planning and scheduling by logic-based benders decomposition," *Operations Research*, vol. 55, no. 3, pp. 588–602, 2007.

[28] L. Ma, J. Wu, and L. Chen, "DOTA: delay bounded optimal cloudlet deployment and user association in WMANs," in *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pp. 196–203, 2017.

[29] J. Shuja, K. Bilal, S. A. Madani, and S. U. Khan, "Data center energy efficient resource scheduling," *Cluster Computing*, vol. 17, no. 4, pp. 1265–1277, 2014.