Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1237619, 17 pages
https://doi.org/10.1155/2022/1237619

Research Article

Dynamic Rendering-Aware VR Service Module Placement Strategy in MEC Networks

Chunyu Liu, Heli Zhang, Xi Li, and Hong Ji

Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China

Correspondence should be addressed to Heli Zhang; zhangheli@bupt.edu.cn

Received 10 April 2022; Revised 25 July 2022; Accepted 1 August 2022; Published 18 August 2022

Academic Editor: A.H. Alamoodi

Combining multiaccess edge computing (MEC) technology and wireless virtual reality (VR) game is a promising computing paradigm. Offloading the rendering tasks to the edge node can make up for the lack of computing resources of mobile devices. However, the current offloading works ignored that when rendering is enabled at the MEC server, the rendering operation depends heavily on the environment deployed on this MEC serve. In this paper, we propose a dynamically rendering-aware service module placement scheme for wireless VR games over the MEC networks. In this scheme, the rendering tasks of VR games are offloaded to the MEC server and closely coupled with service module placement. At the same time, to further optimize the end-to-end latency of VR video delivery, the routing delay of the rendered VR video stream and the costs of the service module migration are jointly considered with the proposed placement scheme. The goal of this scheme is to minimize the sum of the network costs over a long time under satisfying the delay constraint of each player. We model our strategy as a high-order, nonconvex, and time-varying function. To solve this problem, we transform the placement problem into the min-cut problem by constructing a series of auxiliary graphs. Then, we propose a two-stage iterative algorithm based on convex optimization and graphs theory to solve our object function. Finally, extensive simulation results show that our proposed algorithm can ensure low end-to-end latency for players and low network costs over the other baseline algorithms.

1. Introduction

Wireless virtual reality (VR) games are becoming more and more popular, and it is reported that the global VR gaming market size is projected to reach 45 billion dollars by 2025. A wireless VR game application is generally composed of two parts: a collection module and a service module. The collection module is used to collect the geographic location and actions of players and then delivers the collected information to the service module. The service module encapsulates all the necessary environments to perform logical calculations, render the scene, and synchronize the game information among players [1]. Players of different VR games need different service modules to perform their respective rendering tasks. Players of the same VR game use the same service modules and need to synchronize the information of this VR game with each other (such as character position and score). However, offering low-latency and high-quality VR gaming services to mass wireless players at any time and anywhere is always a major challenge [2–8].

Recently, introducing multiaccess edge computing (MEC) technology to wireless VR games has been a promising computing paradigm to address the above challenges [9–14]. By offloading the rendering tasks from the mobile devices (e.g., VR headsets) to the proximal MEC servers, the players' requirements for ultrahigh computational capacity and strict response latency would be satisfied. Rendering refers to the process of generating images from a model, which is a representation of a 3D object or virtual environment defined by a programming language or data structure. Specifically, since the MEC server has higher computing power than the mobile device, the delay of rendering VR tasks on the MEC server is less than the delay of rendering the same VR tasks on the mobile device [15–19].

However, edge rendering inevitably introduces the edge computing delay and the transmission delay caused by the rendered VR game video stream back to the mobile terminal. Especially, since the data volume of VR video streams is generally huge, the increase in delay will be even more pronounced. Therefore, it is particularly important to optimize the routing of rendered VR game video streams and reasonably allocate the edge resource including wireless spectrum and computation. In addition, it should be noted that deploying service modules on MEC servers increases placement costs, and limited by the storage capacity, service modules of all kinds of VR games cannot deploy on each MEC at the same time [20–23]. But the premise of performing the rendering task of the player on the MEC server is that the service module of the VR game that this user participates in has been deployed on this MEC server [24–26]. Based on the above discussion, the service module placement optimization and the computation resource allocation should be closely coupled to jointly optimize the wireless VR game delivery performance [27–29].

Moreover, in a MEC network scenario of concurrent multiple kinds of wireless VR games, the geographical position of players may change with time, and their access base stations (BSs) may change as they move. To ensure the low routing cost of the rendered VR video streams of one group, the corresponding VR service module serving this group may need to migrate to a new base station. The above situation would increase migration costs [30–34] including hardware wear-and-tear costs and data migration delay costs. Dynamically optimizing the trade-off between the routing cost and migration cost is necessary.

In this paper, we propose a dynamically rendering-aware service module placement scheme. In this scheme, the rendering tasks of VR games are offloaded to the MEC server and closely coupled with service module placement. At the same time, to further optimize the end-to-end latency of VR video delivery, the rendered VR video stream routing delay and service module migration costs are considered with the proposed placement scheme. Specifically, the strategies jointly consider the bandwidth, computing, and storage resource allocation scheme within each time slot and the service module migration cost optimization between different base stations in the adjacent time slot. The goal of this scheme is to minimize the sum of the network costs over a long time under satisfying the delay constraint of each player.

(i) In this paper, we propose a dynamically rendering-aware service module placement scheme, which jointly optimizes service module placement and the associated rendering computation allocation. The goal of this scheme is to minimize the whole network cost based on satisfying the players' low end-to-end delay and high-computing requirements

(ii) We study the problem of how to dynamically place the VR service module to achieve a good balance between the routing delay cost of the rendered VR video stream and the migration cost of the corresponding service module

(iii) We transform our placement problem into the minimal cut problem by developing algebraic conversions and constructing a series of auxiliary graphs. Then, we propose a two-stage iterative algorithm based on convex optimization and graphs theory to solve our objective function within polynomial time

The rest of this paper is organized as follows. Section 2 introduces the system model. Section 3 presents the problem formulation. The proposed solution is presented in Section 4. In Section 5, simulation results are presented and discussed. Finally, the conclusion is given in Section 6.

1.1. Related Work. At present, most of the research on placement strategy focuses on reducing network delay and network overhead for the user by reasonably deploying the services, data, or virtual machines in a suitable location with limited network resources. But the current works ignore considering the dependency relationships between computing and storage. Paper [24] proposes a two-time scale framework that jointly optimizes service placement and request scheduling considering system stability and operation cost. Paper [1] provides a mix of cost models to optimize the deployment of collaborative edge applications to achieve the best overall system performance. Paper [25] proposes a distributed algorithm based on games theory to optimize virtual machine placement in mobile cloud gaming through resource competition to meet the overall requirements of players in a cost-effective manner. Paper [35] proposes a novel offline community discovery and online community adjustment schemes to reduce the internode traffic and the system overhead, which solve the replica placement problem in a scalable and adaptive way. Paper [36] has some similarities with our work, which studies the joint optimization of service placement and request routing in the MEC networks with multidimensional (storage-computation-communication) constraints. In paper [5], the author proposes a MEC-based dynamic cache strategy and an optimized unload strategy to minimize system delay and energy. Paper [27] proposes a rendering-aware tile caching scheme to optimize the end-to-end latency for VR video delivery over multicell MEC networks. Paper [28] designs a view synthesis-based 360 VR caching system to meet the requirements of wireless VR applications and enhance the quality of the VR user experience, which supports MEC and hierarchical caching.

The goal of the recent research on wireless VR mainly focuses on improving the quality of service (QoS), reducing network overhead, or both by proper resource allocation, transcoding technology, introducing edge networks, and etc. Insufficient consideration is given to players' mobility and the network scenario of concurrent multiple kinds of wireless VR games. Paper [4] proposes a blockchain-supported task offloading scheme to resist malicious attacks, which reduces the computing load of virtual machines and

satisfy the high QoE of VR users. Paper [10] proposes a wireless VR network that supports MEC. The network uses a recurrent neural network (RNN) to predict the field of view of each VR user in real-time and transfers the rendering task of VR from the VR device to the MEC server through the rendering model migration function. Paper [16] proposes an adaptive MEC-assisted virtual reality framework, which can adaptively assign real-time virtual reality rendering tasks to MEC servers. Meanwhile, the caching capability of MEC servers can further improve network performance. Paper [37] proposes a task offloading, and resource management scheme based on wireless virtual reality is proposed. The scheme comprehensively considers the factors of cache, computing, and spectrum allocation and minimizes the content delivery delay while guaranteeing quality. Paper [38] studies a multilayer wireless VR video service scenario based on a MEC network. Its main goal is to minimize system energy consumption and delay and to find a balance between these two indicators. Paper [11] proposes to minimize the long-term energy consumption of MEC systems based on THz wireless access by jointly optimizing viewport rendering offloading and downlink transmission power control to support high-quality immersive VR video services. Paper [39] proposes a novel transcoding-enabled VR video caching and delivery framework for edge-enhanced next-generation wireless networks. Paper [40] investigates the optimal wireless streaming of a multi-quality-tiled VR video from a server to multiple users by effectively utilizing characteristics of multi-quality-tiled VR videos and computation resources at the users' side.

2. System Model

The MEC server is a microdata center that is typically deployed with a cellular base station or WiFi access point. Some lightweight virtualization technologies are used to virtualize the hardware resources in the MEC server to realize the flexible sharing of resources.

In this section, as illustrated in Figure 1, we consider a scenario of concurrent multiple kinds of VR games under the cellular network equipped with MEC servers. In this network scenario, there are U players and M base stations (BSs), where each BS is deployed with a MEC server. We represent the set of BSs as $\mathcal{U} = \{1, 2, 3, \cdots, u, \cdots, U\}$ and represent the set of users as $\mathcal{M} = \{1, 2, 3, \cdots, m, \cdots, M\}$. The base stations are connected to each other in a wired way. We assume that there are H kinds of VR games in this scenario, denoted by the set $\mathcal{H} = \{1, 2, 3, \cdots, h, \cdots, H\}$. Therefore, H different service modules are required to support these VR games. In addition, to make dynamic decisions, we model our problem as a time-slotted system, where we use $\mathcal{T} = \{1, 2, 3, \cdots, t, \cdots, T\}$ to denote the set of consecutive time slots under consideration. We assume that each time slot is much larger than the delay caused by transmission and processing.

In the remaining subsections, the mathematical models for communication, dynamic placement, rendering computation, and whole network cost are discussed. Some important notations are summarized in Table 1.

2.1. Placement Cost. In this section, we investigate the dynamic placement scheme of all VR service modules in the system.

We assume that the set of service module placement strategies can be denoted as $\Delta = \{\delta_{mh}^t | m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}\}$, where $\delta_{mh}^t = 1$ represents that the VR module service h is storaged in the BS m; otherwise at the time t, $\delta_{mh}^t = 0$.

The cost for using the storage resources when placing service module h on edge node m is characterized by λ_{mh}. The cost of the placement VR service module can be expressed by the following formula:

$$\text{Cost}_P^t = \sum_{m=1}^{M} \sum_{h=1}^{H} \lambda_{mh} \delta_{mh}^t. \tag{1}$$

We assume the storage capacity of BS m is Π_m, and the size of VR service module h is w_h. Due to the total size of the VR service modules deployed in BS m should not exceed the maximum storage capacity of BS m, the constraint should be expressed as

$$\sum_{h=1}^{H} \delta_{mh}^t w_h \le \Pi_m, \forall m \in \mathcal{M}. \tag{2}$$

2.2. Migration Cost. When the players move, due to the changes in the geographical location, the BS that transmits the rendered data to the players may change. At the same time, the BS that originally provided the rendering service for the game group may no longer be the best choice to provide service. The group may need to select a suitable new BS to perform rendering and even may need to deploy the corresponding VR service module on the new selected BS. That is to say, the data information of the service module may need to be migrated from the old MEC server to the new MEC server and built the environment on the new MEC. However, the migration of the VR service module will cause hardware wear-and-tear costs and impose data migration latency costs. The migration delay of each player belonging to the same group is equal and can be expressed as

$$D_{u,t}^{\text{mig}} = \sum_{m=1}^{M} \sum_{h=1}^{H} p_u^h g\left(\delta_{mh}^t, \delta_{mh}^{t-1}\right). \tag{3}$$

In addition, the all migration costs can be expressed as

$$\text{Cost}_M^t = \sum_{m=1}^{M} \sum_{h=1}^{H} \left[f\left(\delta_{mh}^t, \delta_{mh}^{t-1}\right) + g\left(\delta_{mh}^t, \delta_{mh}^{t-1}\right) \right], \tag{4}$$

where $f(\delta_{mh}^t, \delta_{mh}^{t-1})$ and $g(\delta_{mh}^t, \delta_{mh}^{t-1})$ can be, respectively, defined as

$$f\left(\delta_{m,h}^t, \delta_{m,h}^{t-1}\right) = \begin{cases} f_h, & \delta_{mh}^t > \delta_{mh}^{t-1}, \\ 0, & \delta_{mh}^t \le \delta_{mh}^{t-1}, \end{cases} \tag{5}$$

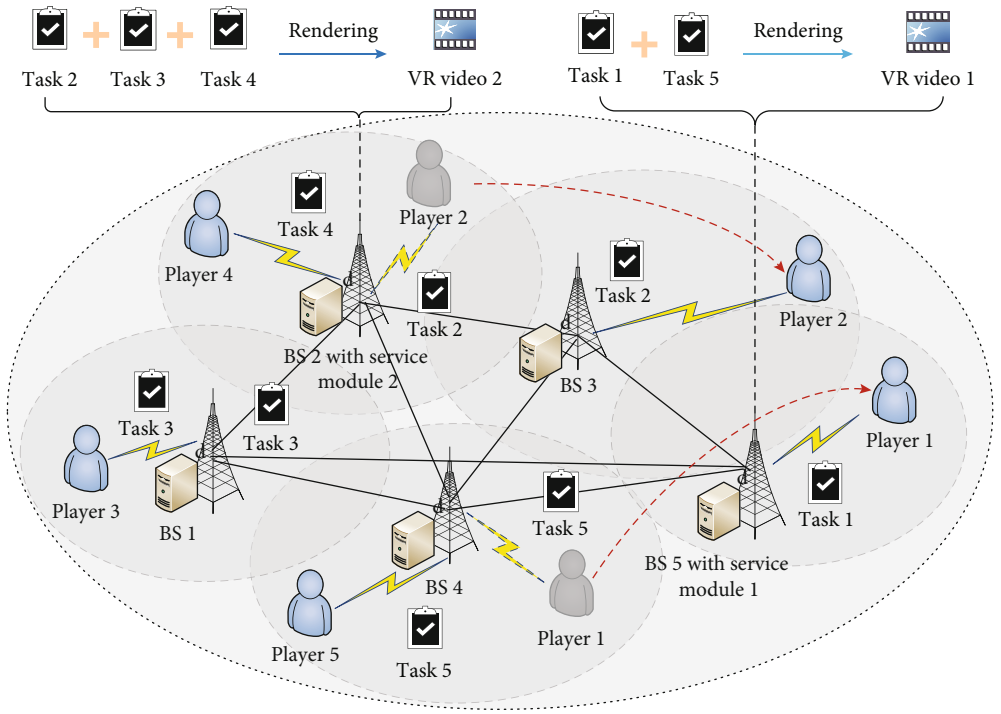


Figure 1: System model. Players 1 and 5 belong to the same VR game and need VR service module 1 to perform rendering. Players 2–4 belong to the same VR game and need VR service module 2 to perform rendering. Among them, the player 1 migrates from the coverage of BS 4 to the coverage of BS 5, and the player 2 migrates from the coverage of BS 2 to the coverage of BS 3. Player 3 access to BS 1 and offload the task 3 from BS 1 to BS 2.

Table 1: List of key notations.

Notation	Definition
$\mathcal{M}$	Set of BSs
$\mathcal{U}$	Set of users
$\mathcal{H}$	Set of VR games
$\mathcal{T}$	Set of consecutive time slots
δ_{mh}^t	Placement indicator
Π_m	The maximum storage capability of BS m
σ^2	The variance of additive white Gaussian noise
B^t	The maximum bandwidth of BS at time t
K_m	The maximum computing capability of BS m
a_{mu}^t	The access indicator at the time t
s_{mh}^t	The BS selection indicator at the time t
$d_{m,m'}$	The delay of routing one bit of data from BS m to BS m'
p_u^h	The indicator of player u whether joining in group$h$

$$g\left(\delta_{m,h}^t, \delta_{m,h}^{t-1}\right) = \begin{cases} \upsilon g_h, & \delta_{mh}^t > \delta_{mh}^{t-1}, \\ 0, & \delta_{mh}^t \le \delta_{mh}^{t-1}, \end{cases} \tag{6}$$

where g_h represents the migration delay of the VR service module h and f_h represents the cost of reconfiguring the VR service module h. To be reasonable, we make the values of $f(\delta_{mh}^t, \delta_{mh}^{t-1})$ and $g(\delta_{mh}^t, \delta_{mh}^{t-1})$ to the same order of magnitude by adjusting the parameter υ.

2.3. Rendering Cost. Players in the same group may have overlapping computational tasks; in this section, we assume that the MEC server computes centrally after collecting all the information of the players in the group. Therefore, we allocate the computing resources on each server by the group.

In the MEC network, when the MEC server is serving only one group, that group can certainly get more computing resources to perform rendering, resulting in a low processing latency experience. However, in general, each MEC server needs to serve multiple groups at the same time, which can lead to competition for computation resources. In particular, if too many groups render on the same MEC server, the delays for all groups connected to this server will increase dramatically.

$p_u^h \in \{0, 1\}$ is the indicator, to represents whether the players u join in the VR game h. Due to one player can only join in one kind of game, so the corresponding constraints can be, respectively, formulated as

$$\sum_{h=1}^{H} p_u^h = 1, \forall u \in \mathcal{U}. \tag{7}$$

We use $\mathcal{S} = \{s_{mh}^t | m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}\}$ to denote the set of the rendering base station selection strategies. When the

group h selects the MEC server m to perform the rendering task, $s_{mh}^t = 1$ at the time t; otherwise, $s_{mh}^t = 0$.

In order to ensure the information synchronization between users in the same group, we assume that a group can only select one MEC server to process tasks at a time slot, so the corresponding constraints can be formulated as

$$\sum_{m=1}^{M} s_{mh}^t = 1, \forall h \in \mathcal{H}, t \in \mathcal{T}. \tag{8}$$

Since the cost of putting the VR service module on the server is high, we put the VR service module on the BS, which has been selected to process the groups' tasks. So, we can get the following formula:

$$\delta_{mh}^t = s_{mh}^t, \forall t \in \mathcal{T}, h \in \mathcal{H}, m \in \mathcal{M}. \tag{9}$$

We assume that the maximum computing capability of the MEC server m is K_m (Hz) and the computing resource of the BS m allocated to group h at time t is k_{mh}^t. We use $\mathcal{K} = \{k_{mh}^t | m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}\}$ to represent the computing resource allocation scheme. C_h^t represents the computing resource needed for group h at time t. The rendering delay of players belonging to the same group is equal. So, the rendering delay of player u at time slot t can be expressed as

$$D_{u,t}^{\text{rend}} = \sum_{m=1}^{M} \sum_{h=1}^{H} s_{mh}^t p_u^h \frac{C_h^t}{k_{mh}^t}, t \in \mathcal{T}. \tag{10}$$

So, the rendering cost can be denoted by the sum of the rendering latency of all groups, which can be expressed by

$$\text{Cost}_R^t = \sum_{m=1}^{M} \sum_{h=1}^{H} s_{mh}^t \frac{C_h^t}{k_{mh}^t}, t \in \mathcal{T}. \tag{11}$$

At the same time, a MEC server cannot allocate more computing resources to the groups; it serves than its maximum computing resources. Therefore, the corresponding computing resources constraints can be formulated as

$$\sum_{h=1}^{H} k_{mh}^t \le K_m, \forall m \in \mathcal{M}, t \in \mathcal{T}. \tag{12}$$

2.4. Communication Cost. In this section, we present the communication model in the mobile edge computing networks based on mmWave, which concentrates on the downlink transmission. At the same time, we introduce the routing transmission delay.

2.4.1. Downlink Delay. We use $\mathcal{A} = \{a_{mu}^t | m \in \mathcal{M}, u \in \mathcal{U}, t \in \mathcal{T}\}$ as the access scheme, where the $a_{mu}^t = 1$ means that players u is associated with BS m at the time t to obtain the rendered game video stream, while $a_{mu}^t = 0$ denotes that players u is not served by BS m at the time t.

Moreover, players cannot connect to multiple base stations at the same time, and we need to ensure that each player can connect to a suitable one. So we get the following constraint formula:

$$\sum_{m=1}^{M} a_{mu}^t = 1, \forall u \in \mathcal{U}. \tag{13}$$

We adopt the orthogonal spectrum reuse scheme in this system; i.e., all BS share the total frequency bandwidth, and there is no interference between the users served by the same BS. The data amount of the uplink transmission is small, only including some players' information, such as commands and actions. So, the delay and cost of this process are ignored in this paper.

The downlink transmission is used to transmit the rendered VR video stream, in which the amount of data is larger. Therefore, millimeter Wave technology with large bandwidth is adopted for downlink transmission. Assume that all channels are subject to independent identically distributed quasistatic Rayleigh block fading. The path loss can be expressed as follow:

$$L_{mu}^t = \eta^t \left(\left| d_{mu}^t \right|^{-\varsigma^t} \right), \tag{14}$$

where η^t is the downlink constant related to frequency, ς^t is the downlink path loss exponent at time t, and $|d_{mu}^t|$ is the distance between the players u and BS m at time t.

Millimeter wave has the characteristics of short wavelength, small power, and directional antenna. The interference between the same frequency beam can be reduced well by millimeter wave interference cancelation technology. As the interference cancelation technology is not the focus of this paper and the millimeter transmission tends to be noise-limited and weak-interference, the interference in the transmission process of millimeter waves is ignored in this paper by referring to papers [16, 41, 42]. So, the signal-to-interference-plus-noise ratio received by the players u from the BS u is expressed as follows:

$$\text{SINR}_{mu}^t = \frac{p_{mu} g_{mu}^t L_{mu}^t}{\sigma^2}, \tag{15}$$

where g_{mu}^t is the downlink antenna gain using direction beamforming between players u and BS m at the time t, p_{mu} is the transmission power between players u and BS m, and σ^2 is the variance of additive white Gaussian noise (AWGN).

We assume that the spectrum bandwidth allocated to players u from BS m at time t is B_{mu}^t and use $\mathcal{B} = \{B_{mu}^t | m \in \mathcal{M}, u \in \mathcal{U}, t \in \mathcal{T}\}$ as the bandwidth allocation scheme. Since the total bandwidths that the BS m allocates to its access players do not exceed the whole bandwidths in the wireless access network at time t, which is B^t, corresponding bandwidth constraints can be formulated as

$$\sum_{u=1}^{U} B_{mu}^t \le B^t, \forall m \in \mathcal{M}, t \in \mathcal{T}. \tag{16}$$

Then, the uplink transmission rate between the players u and the BS m at time t is

$$r_{mu}^{t} = B_{mu}^{t}\log_2\left(1 + \mathrm{SINR}_{mu}^{t}\right). \tag{17}$$

We assume that the size of the video images needed to transmit to the players u at time t is o_u^t, so the delay of downlink transmission for players u at time t is

$$D_{u,t}^{\mathrm{down}} = \sum_{m=1}^{M} a_{mu}^{t}\frac{o_u^t}{r_{mu}^{t}}. \tag{18}$$

The delay of downlink transmission for all players at time t, i.e., the downlink communication cost of the network, is

$$E_1^t = \sum_{u=1}^{U} D_{u,t}^{\mathrm{down}}. \tag{19}$$

2.4.2. Routing Delay. In this section, we divided the players into H groups based on the differences in VR games they participate in. Different groups need different service modules to perform rendering. We need to select an appropriate MEC server to perform rendering for group h and route the rendered video stream quickly to the access base station of the user belonging to the group h. The selected MEC server needs to have deployed the corresponding VR service modules and has sufficient computing resources to perform rendering tasks.

According to the above assumption, at the time slot t, the delay of routing the rendered VR content requested by user u from the working (rendering) BS m to this user's access BS m' can be expressed as

$$D_{u,t}^{\mathrm{rout}} = \sum_{m=1}^{M}\sum_{h=1}^{H}\sum_{m'=1}^{M} p_u^h a_{m'u}^{t} s_{mh}^{t} d\left(m, m'\right) o_u^t, \tag{20}$$

where $d(m, m')$ is the delay of routing one bit of data from BS m to BS m', when $m = m'$, $d(m, m') = 0$.

The routing delay of all players at time t, i.e., the routing cost of the network, is

$$E_2^t = \sum_{u=1}^{U} D_{u,t}^{\mathrm{rout}}. \tag{21}$$

So, the communication cost at time t can be expressed as the sum of downlink transmission delay and routing delay.

$$\mathrm{Cost}_C^t = E_1^t + E_2^t, t \in \mathcal{T}. \tag{22}$$

3. Problem Formulation

Our goal is to develop dynamical service module placement strategies based on rendering-aware. The goal of those strategies is to minimize the sum of the whole network costs over a long time under satisfying the delay constraint of each player. The strategies jointly consider the resource allocation scheme within each time slot and the service module migration scheme between different base stations in the adjacent time slot.

We assume that the maximum tolerance delay of the group u is $\mathscr{D}_u$. According to the above formula, the actual end-to-end delay of player u at time slot t can be expressed by the following:

$$\mathscr{D}_{u,t}{}' = D_{u,t}^{\mathrm{down}} + D_{u,t}^{\mathrm{rout}} + D_{u,t}^{\mathrm{rend}} + D_{u,t}^{\mathrm{mig}}. \tag{23}$$

We define $\varepsilon_1 - \varepsilon_4$ as the weight coefficients, which represent the proportion of communication cost, rendering cost, placement cost, and migration cost in the objective function, respectively. So, the optimization problem can be formulated as follows:

$$\begin{aligned}
\Gamma_1 : \quad & \min_{\mathscr{A},\mathscr{S},\mathscr{B},\mathscr{K},\Delta} \sum_{t=1}^{T} \varepsilon_1 \mathrm{Cost}_C^t + \varepsilon_2 \mathrm{Cost}_R^t + \varepsilon_3 \mathrm{Cost}_P^t + \varepsilon_4 \mathrm{Cost}_M^t \\
\text{s.t.} \quad & C1 : \sum_{m=1}^{M} a_{mu}^{t} = 1, \forall u \in \mathscr{U}, t \in \mathcal{T} \\
& C2 : \sum_{m=1}^{M} s_{mh}^{t} = 1, \forall h \in \mathscr{H}, t \in \mathcal{T} \\
& C3 : \sum_{u=1}^{U} B_{mu}^{t} \le B^{t}, \forall m \in \mathscr{M}, t \in \mathcal{T} \\
& C4 : \sum_{h=1}^{H} k_{mh}^{t} \le K_m, \forall m \in \mathscr{M}, t \in \mathcal{T} \\
& C5 : \sum_{h=1}^{H} \delta_{mh}^{t} w_h \le \Pi_m, \forall m \in \mathscr{M} \\
& C6 : \mathscr{D}'_{u,t} \le \mathscr{D}_u \\
& a_{mu}^{t}, s_{mh}^{t}, \delta_{mh}^{t} \in \{0, 1\}.
\end{aligned} \tag{24}$$

Constraint C_1 ensures that a player cannot connect to multiple base stations at the same time; meanwhile, each user can connect to a BS. Constraint C_2 ensures that a group can only select one MEC server to perform rendering tasks at a time slot. Constraint C_3 ensures that the total bandwidths that the BS m allocates to its access players do not exceed the whole bandwidths in the wireless access network at time t. Constraint C_4 ensures that a MEC server cannot allocate more computing resources to the groups; it serves than its maximum computing resources. Constraint C_5 ensures that the total size of the VR service modules storage in BS m should not exceed the maximum storage capacity of BS m. Constraint C_6 ensures the total delay of each group cannot exceed its maximum tolerance delay.

4. Solution

In this section, in order to solve the original problem efficiently, we decompose the original problem into two subproblems including dynamic access and service module placement scheme and the quasistatic resource allocation. Then, we use minimum cut theory and convex optimization to solve the above subproblems, respectively.

4.1. Problem Reformulation. Firstly, to get rid of constraint 1 and constraint 2, we redefine sets $\mathscr{A} = \{a_{mu}^t | m \in \mathscr{M}, u \in \mathscr{U}, t \in \mathscr{T}\}$ and $\mathscr{S} = \{s_{mh}^t | m \in \mathscr{M}, h \in \mathscr{H}, t \in \mathscr{T}\}$ as $\mathscr{A}_* = \{a_{u*}^t, u \in \mathscr{U}, t \in \mathscr{T}\}$ and $\mathscr{S}_* = \{s_{h*}^t, h \in \mathscr{H}, t \in \mathscr{T}\}$, respectively, where $\mathscr{A}_*^t = \{a_{u*}^t, u \in \mathscr{U}\}$ is the set of access decisions at time t and $a_{u*}^t \in \mathscr{M}$ represents the BS accessed by the players u, and there is a one-to-one mapping relationship between it and the set $\mathscr{A}_u^t = \{a_{mu}^t, m \in \mathscr{M}\}$. That is, $a_{mu}^t = 1$ and $\{a_{iu}^t = 0 | i \in \mathscr{M}, i \neq m\}$ when $a_{u*}^t = m$. This way of coding can satisfy the constraint $C1$ that a player can only access one base station at the same time.

In the same way, $\mathscr{S}_*^t = \{s_{h*}^t, h \in \mathscr{H}\}$ is the set of BS selection scheme at time t. $s_{h*}^t \in \mathscr{M}$ represents BS serving group h at time t, and there is a one-to-one mapping relationship between it and the set $\mathscr{S}_h^t = \{s_{mh}^t, m \in \mathscr{M}\}$. This way of coding can satisfy the constraint $C2$ that a group can only select one MEC server to perform editing tasks at a time slot.

So, the δ_{mh}^t can be redefined as

$$\delta_{mh}^t = \begin{cases} 1, & s_{h*}^t = m, \forall h \in \mathscr{H}, \\ 0, & \text{otherwise.} \end{cases} \tag{25}$$

Moreover, $\mathscr{B}_*^t = \{B_{u*}^t, u \in \mathscr{U}\}$ is the set of bandwidth allocation scheme at time t. $B_{u*}^t = B_{a_{u*}^t u}^t \in [0, B^t]$ is the bandwidth that BS a_{u*}^t allocate to the players u at time t. $\mathscr{K}_*^t = \{k_{h*}^t, h \in \mathscr{H}\}$ is the set of computing resources allocation scheme at time t. $k_{h*}^t = k_{s_{h*}^t h}^t \in [0, K_m]$ is the computing resources that BS s_{h*}^t allocate to the group h at time t.

Thus, we transform the original problem into the following problem:

$$\begin{aligned} \Gamma_2 : \min_{\mathscr{A}_*, \mathscr{S}_*, \mathscr{B}_*, \mathscr{K}_*} & \sum_{t=1}^{T} \left(\varepsilon_1 \sum_{u=1}^{U} \frac{o_u^t}{B_{a_{u*}^t u}^t log_2\left(1 + SINR_{u*}^t\right)} \right. \\ & + \varepsilon_1 \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h d\left(s_{h*}^t, a_{u*}^t\right) + \varepsilon_2 \sum_{h=1}^{H} \frac{C_h^t}{k_{s_{h*}^t h}^t} \\ & \left. + \varepsilon_3 \sum_{m=1}^{M} \sum_{h=1}^{H} \lambda_{mh} 1\left(s_{h*}^t = m\right) + \varepsilon_4 \sum_{h=1}^{H} \left[f\left(s_{h*}^t, s_{h*}^{t-1}\right) + g\left(s_{h*}^t, s_{h*}^{t-1}\right) \right] \right) \\ C4' : & \sum_{h \in H_m^t} k_{h*}^t \leq K_m, \forall m \in \mathscr{M}, t \in \mathscr{T} \\ C5' : & \sum_{h=1}^{H} \delta_{mh}^t w_h \leq \Pi_m, \forall m \in \mathscr{M} \\ C6' : & \mathscr{D}'_{ut} \leq \mathscr{D}_u \\ & a_{u*}^t, s_{h*}^t, \in \mathscr{M}, \end{aligned} \tag{26}$$

where H_m^t represents the set of all the groups that render on the BS m and U_m^t represents the set of all the players that access the BS m at time t. Constraint 5 can be satisfied by the k-size minimum cut algorithm. $1(.)$ is a binary function that equals 1 if the specified condition holds and 0 otherwise, where A is the penalty function, which can be expressed as $\mathscr{D}'_{ut}$:

$$\begin{aligned} \mathscr{D}'_{ut} = & \sum_{h=1}^{H} p_u^h \left[g\left(s_{h*}^t, s_{h*}^{t-1}\right)\right) + \frac{C_h^t}{k_{s_{h*}^t h}^t} + d\left(s_{h*}^t, a_{u*}^t\right) \Bigg] \\ & + \frac{o_u^t}{B_{a_{u*}^t u}^t log_2\left(1 + SINR_{u*}^t\right)}. \end{aligned} \tag{27}$$

Due to our objective function containing dynamic optimization and quasistatic optimization, we divide the target function into two parts.

For the part one,

$$\begin{aligned} \text{Cost}_I = \sum_{t=1}^{T} \Bigg(& \varepsilon_1 \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h d\left(s_{h*}^t, a_{u*}^t\right) + \varepsilon_3 \sum_{m=1}^{M} \sum_{h=1}^{H} \lambda_{mh} 1\left(s_{h*}^t = m\right) \\ & + \varepsilon_4 \sum_{h=1}^{H} \left[f\left(s_{h*}^t, s_{h*}^{t-1}\right) + g\left(s_{h*}^t, s_{h*}^{t-1}\right) \right] \Bigg). \end{aligned} \tag{28}$$

We design an iterative algorithm to update the access decisions of players and the placement schemes of the VR service module in each round by performing an operation called α expansion. Furthermore, we optimize the expansion by minimizing graph cuts.

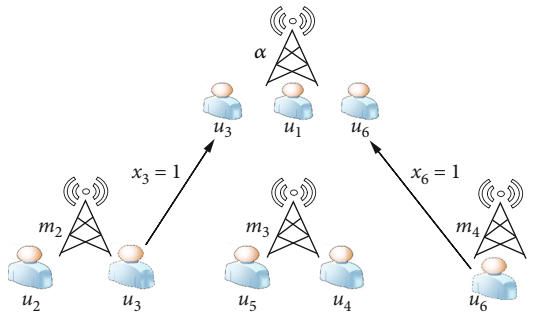
For the part two,

$$\text{Cost}_{II} = \sum_{t=1}^{T} \left(\varepsilon_1 \sum_{u=1}^{U} \frac{o_u^t}{B_{u*}^t \log_2\left(1 + \text{SINR}_{u*}^t\right)} + \varepsilon_2 \sum_{h=1}^{H} \frac{C_h^t}{k_{h*}^t} \right). \tag{29}$$

We use convex optimization to solve the resource allocation problem at each time slot.

4.2. Optimizing Dynamic Access and Placement Strategies by Graph Cuts. In this section, we introduce the α expansion algorithm and how to construct a helper graph and encode the costs of part I into weights on the graph edges. Then, we demonstrate that the min-cut of the graph corresponds to the optimal decisions for the α expansion.

4.2.1. α Expansion. An α expansion can be defined as a binary optimization and reflects the trend of moving the module served for group h from the current base station to the base station α and the trend of users accessing base station α from the current base station. As shown in Figure 2, when we selected BS α as the expansion, $a_{u*}^{t\alpha}$ has a binary choice to stay as $a_{u*}^{t\alpha} = a_{u*}^t$ or change to $a_{u*}^{t\alpha} = \alpha$. In the same way, $s_{h*}^{t\alpha}$ has a binary choice to stay as $s_{h*}^{t\alpha} = s_{h*}^t$ or change to $s_{h*}^{t\alpha} = \alpha$.



	u_1	u_2	u_3	u_4	u_5	u_6
Before	α	m_2	m_2	m_3	m_3	m_4
After	α	m_2	α	m_3	m_3	α
x	1	0	1	0	0	1

Figure 2: α expansion. The player u_3 changes the access base station from m_2 to α, and u_6 changes the access base station from m_4 to α, respectively.

For the sake of calculation, the resultant after expansion also can be expressed by two indicator vectors with binary decision variables. (1) $x't = \{x_1't, \cdots, x_u't\}$, where for all $u \in U$, we define $x_u't = 1$ if $a_{u*}^{t\alpha} = \alpha$; otherwise, $x_u't = 0$. (2) $x = \{x_1^t, \cdots, x_h^t\}$, where for all $h \in H$, we define $x_h^t = 1$ if $s_{h*}^{t\alpha} = \alpha$; otherwise, $x_h = 0$. Note that, if the module served for group h is already on BS α, $x_h^t = 1$, if the players u is already access BS α, $x_u't = 1$.

4.2.2. Transforming the Cost$_I$. After performing an "α expansion," we reconstruct the Cost$_I$ as Cost$_I^\alpha$ using binary variables $x_u^{t'}$ and x_h^t; at the same time, we define $\bar{x}_u't = 1 - x_u't$ and $\bar{x}_h^t = 1 - x_h^t$. And we can get

$$\begin{aligned}
&\varepsilon_1 \sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{u=1}^{U} p_u^h o_u^t d\left(s_{h*}^t, a_{u*}^t\right)^\alpha \\
&\quad = \varepsilon_1 \sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{u=1}^{U} p_u^h o_u^t \Big[d\left(s_{h*}^t, a_{u*}^t\right)\bar{x}_h^t \bar{x}_u' t \\
&\qquad + d\left(\alpha, a_{u*}^t\right) x_h^t \bar{x}_u' t + d\left(s_{h*}^t, \alpha\right)\bar{x}_h^t x_u^{t'} \Big],
\end{aligned} \tag{30}$$

$$\begin{aligned}
\varepsilon_4 \sum_{t=1}^{T}\sum_{h=1}^{H} f\left(s_{h*}^t, s_{h*}^{t-1}\right)^\alpha &= \varepsilon_4 \sum_{t=1}^{T}\sum_{h=1}^{H} \Big[f\left(s_{h*}^t, s_{h*}^{t-1}\right)\bar{x}_h^t \bar{x}_h^{t-1} \\
&\quad + f\left(\alpha, s_{h*}^{t-1}\right) x_h^t \bar{x}_h^{t-1} + f\left(s_{h*}^t, \alpha\right)\bar{x}_h^t x_h^{t-1} \Big],
\end{aligned} \tag{31}$$

$$\begin{aligned}
\varepsilon_4 \sum_{t=1}^{T}\sum_{h=1}^{H} g\left(s_{h*}^t, s_{h*}^{t-1}\right)^\alpha &= \varepsilon_4 \sum_{t=1}^{T}\sum_{h=1}^{H} \Big[g\left(s_{h*}^t, s_{h*}^{t-1}\right)\bar{x}_h^t \bar{x}_h^{t-1} \\
&\quad + g\left(\alpha, s_{h*}^{t-1}\right) x_h^t \bar{x}_h^{t-1} + g\left(s_{h*}^t, \alpha\right)\bar{x}_h^t x_h^{t-1} \Big].
\end{aligned} \tag{32}$$

Then, based on the definition of δ_{mh}^t, we can rewrite it as

$$\varepsilon_3 \sum_{m=1}^{M}\sum_{h=1}^{H} \lambda_{mh} 1\left(s_{h*}^t = m\right)^\alpha = \sum_{m=1}^{M}\sum_{h=1}^{H} \left[\lambda_{\alpha h} x_h^t + \lambda_{mh}\bar{x}_h^t\right]. \tag{33}$$

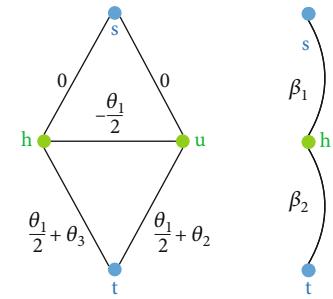


Figure 3: Graph construction. The first figure correspond to $\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h$; the last figure corresponds to $\beta_1 s_h + \beta_2\bar{s}_h$.

4.2.3. A Simple Example of Graph Cut. Based on the derivation above, we find that $\sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{u=1}^{U} p_u^h o_u^t d(s_{h*}^t, a_{u*}^t)^\alpha$ and $\sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{u=1}^{U} f(s_{h*}^t, s_{u*}^{t-1})^\alpha$ correspond to the sum of the products of pairs of binary variables; $\sum_{m=1}^{M}\sum_{h=1}^{H}\lambda_{mh}1(s_{h*}^t = m)^\alpha$ corresponds to the sum of binary variables.

Taking $\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h$ and $\beta_1 s_h + \beta_2\bar{s}_h$ as simple examples, we next will introduce how to minimize them, respectively, by constructing a graph. The basic idea is to construct a helper graph to make the sum of the weights of the min-cut of the graph equal the optimal value of the objective function. The above cut edges divide the nodes in the graph into two parts: one part of the nodes is on the side of node s, and the corresponding value is 0. The other part of the nodes is on the side of node t, and the corresponding value is 1. In addition, the minimum cut can be computed in polynomial time only if all the edge weights are nonnegative. Next, we will introduce how to build a diagram for our example.

For $\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h$, we reformulate the expression to construct each edge in a subgraph.

$$\begin{aligned}
&\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h \\
&\quad = \frac{\theta_1}{2}\bar{x}_u\bar{s}_h + \frac{\theta_1}{2}\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h \\
&\quad = -\frac{\theta_1}{2}x_u\bar{s}_h - \frac{\theta_1}{2}\bar{x}_u s_h + \left(\frac{\theta_1}{2} + \theta_2\right)\bar{x}_u + \left(\frac{\theta_1}{2} + \theta_3\right)\bar{s}_h.
\end{aligned} \tag{34}$$

As illustrated in the first figure in Figure 3, the weight of edge between node u and node h is $-\theta_1/2$, the weight of edge between node u and node t is $\theta_1/2 + \theta_2$, and the weight of edge between node h and node t is $\theta_1/2 + \theta_3$, where $-\theta_1/2 \geq 0$. For example, when we divide the first graph's nodes in Figure 3 into two parts by cutting the edge between nodes s and h, the edge between nodes h and u, and the edge between nodes u and t, node u and node s are in the same part, and node h and node t are in the same part (i.e., $x_u = 0$, $s_h = 1$, and $\bar{x}_u = 1$, $\bar{s}_h = 0$). The value of the first graph function is $\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h = \theta_2$, which is equal to the sum of the weights of the cut edges. In the last figure in Figure 3, the weight of edge between node h and node s is β_1, and the weight of edge between node h and node t is β_2.

4.2.4. Constructing a Graph to Solve the Subproblem. In this section, we construct a graph $\mathcal{G} \ll (\mathcal{V}, \mathcal{E})$ to make the sum of the edges' weights in the minimal cut set equals the optimal value of our objective function. In this graph, there are $T * U$ vertices corresponding to the players, and $T * H$ vertices corresponding to the groups. Moreover, a source vertex s and a terminal vertex t are also in the vertex set. As a result, the set of vertices in $\mathcal{G}$ is given by $\{x_u' t | u \in \mathcal{U}, t \in \mathcal{T}\} \cup \{x_h^t | h \in \mathcal{H}, t \in \mathcal{T}\} \cup \{s, t\}$.

In the next section, we add edges to the graph and give each edge an appropriate weight. Firstly, based on the example of the last figure in Figure 3. The weights of the edges between node x_h^t and node s can be represented as $\lambda_{\alpha h}$, and the weights of the edges between node x_h^t and node t can be represented as λ_{mh}.

Next, we rewrite formulas (30) and (31) to formulas (40) and (41) based on the example of the first figure in Figure 3.

Therefore, the weight of the edge between the vertex $x_u' t$ and vertex x_h^t is

$$p_u^h o_u^t \frac{d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - d(s_{h*}^t, a_{u*}^t)}{2}, \tag{35}$$

where $d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - d(s_{h*}^t, a_{u*}^t)$ is always satisfied, which can be proved by the triangle inequality.

In the same way, the weight of the edge between the vertex x_h^{t-1} and vertex x_h^t is

$$\frac{f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})}{2}, \tag{36}$$

where $f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})$ is always satisfied, which can be proved by the triangle inequality.

In addition, based on the above derivation, we can also get that the partial of weight of the edge between vertex x_h^t and vertex t is

$$\frac{d(s_{h*}^t, a_{u*}^t) - d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha)}{2}. \tag{37}$$

The partial of weight of the edge between vertex $x_u' t$ and vertex t is

$$\frac{d(s_{h*}^t, a_{u*}^t) + d(\alpha, a_{u*}^t) - d(s_{h*}^t, \alpha)}{2}. \tag{38}$$

Moreover, the partial of weight of the edge between vertex x_h^t and vertex t is

$$\frac{-f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})}{2} + \frac{f(\alpha, s_{h*}^{t-1}) - f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})}{2}. \tag{39}$$

Therefore, we can perform the following transformation of the objective function based on the above analysis:

$$\begin{aligned}
&\varepsilon_1 \sum_{t=1}^{T} \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h o_u^t d(s_{h*}^t, a_{u*}^t)^{\alpha} \\
&= \varepsilon_1 \sum_{t=1}^{T} \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h o_u^t \Big[d(s_{h*}^t, a_{u*}^t) \bar{x}_h^t \bar{x}_u' t \\
&\quad + d(\alpha, a_{u*}^t)(1 - \bar{x}_h^t) \bar{x}_u' t + d(s_{h*}^t, \alpha) \bar{x}_h^t \left(1 - \bar{x}_u^{t'}\right) \Big] \\
&= \varepsilon_1 \sum_{t=1}^{T} \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h o_u^t \Big[d(\alpha, a_{u*}^t) \bar{x}_u' t + d(s_{h*}^t, \alpha) \bar{x}_h^t \\
&\quad + \left(d(s_{h*}^t, a_{u*}^t) - d(\alpha, a_{u*}^t) - d(s_{h*}^t, \alpha) \right) \bar{x}_h^t \bar{x}_u' t \Big] \\
&= \varepsilon_1 \sum_{t=1}^{T} \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h o_u^t \Bigg[\frac{d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - (d(s_{h*}^t, a_{u*}^t)}{2} \bar{x}_h^t x_u' t \\
&\quad + \frac{d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - d(s_{h*}^t, a_{u*}^t)}{2} x_h^t \bar{x}_u' t \\
&\quad + \frac{d(s_{h*}^t, a_{u*}^t) + d(\alpha, a_{u*}^t) - d(s_{h*}^t, \alpha)}{2} \bar{x}_u' t \\
&\quad + \frac{d(s_{h*}^t, a_{u*}^t) - d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha)}{2} \bar{x}_h^t \Bigg],
\end{aligned} \tag{40}$$

$$\begin{aligned}
&\varepsilon_4 \sum_{t=1}^{T} \sum_{h=1}^{H} f(s_{h*}^t, s_{h*}^{t-1})^{\alpha} \\
&= \varepsilon_4 \sum_{t=1}^{T} \sum_{h=1}^{H} \Big[f(s_{h*}^t, s_{h*}^{t-1}) \bar{x}_h^t \bar{x}_h^{t-1} \\
&\quad + f(\alpha, s_{h*}^{t-1})(1 - \bar{x}_h^t) \bar{x}_h^{t-1} + f(s_{h*}^t, \alpha) \bar{x}_h^t (1 - \bar{x}_h^{t-1}) \Big] \\
&= \varepsilon_4 \sum_{t=1}^{T} \sum_{h=1}^{H} \Big[f(\alpha, s_{h*}^{t-1}) \bar{x}_h^{t-1} + f(s_{h*}^t, \alpha) \bar{x}_h^t + \left(f(s_{h*}^t, s_{h*}^{t-1}) \right) \\
&\quad - f(\alpha, s_{h*}^{t-1}) - f(s_{h*}^t, \alpha)) \bar{x}_h^t \bar{x}_h^{t-1} \Big] \\
&= \varepsilon_4 \sum_{t=1}^{T} \sum_{h=1}^{H} \Bigg[\frac{f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})}{2} x_h^t \bar{x}_h^{t-1} \\
&\quad + \frac{f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})}{2} \bar{x}_h^t x_h^{t-1} \\
&\quad + \frac{-f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})}{2} \bar{x}_h^t \\
&\quad + \frac{f(\alpha, s_{h*}^{t-1}) - f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})}{2} \bar{x}_h^{t-1} \Bigg].
\end{aligned} \tag{41}$$

The detail process of the auxiliary diagram construction is concluded in Algorithm 1.

4.3. Resource Allocation Scheme Based on Convex Optimization. In this section, we mainly focus on the optimization of Cost_{II}, that is, minimizing the total transmission and editing delay in each time interval through the reasonable allocation of computing and spectrum resources. When $\mathcal{A} *$ and $\mathcal{S} *$ are determined, the original optimization problem can be expressed in the following form:

Input: The network delay between BS m and BS m' $d(m, m')$; The switching cost of group h at time t $f(s_{h*}^t, s_{h*}^{t-1})$; The migration delay of group h at time t $g(s_{h*}^t, s_{h*}^{t-1})$;
Output: The value of binary variables $x_u't$ and x_h^t; The auxiliary graph $\mathcal{G} = (\mathcal{E}, \mathcal{V})$; the variables $s_{h*}^{t\alpha}$ and $a_{u*}^{t\alpha}$

Initialization $\mathcal{V} = \{x_u't | u \in \mathcal{U}, t \in \mathcal{T}\} \cup \{x_h^t | h \in \mathcal{H}, t \in \mathcal{T}\} \cup \{source, terminal\}$; $\mathcal{E} = 0$;
for $t = 1 : T$ **do**
  for $h = 1 : H$ **do**
    $u = 1 : U$ **do**
      $e(x_h^t, x_u't) = p_u^h o_u^t (d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - d(s_{h*}^t, a_{u*}^t)/2)$;
      $e(terminal, x_u't) = d(s_{h*}^t, a_{u*}^t) + d(\alpha, a_{u*}^t) - d(s_{h*}^t, \alpha)/2$;
    end for
    for Algorithm 1 $m = 1 : M$ **do**
      $e(x_h^t, x_h^{t-1}) = f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})/2$;
      $e(terminal, x_h^t) = (-f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})/2) + (f(\alpha, s_{h*}^{t-1}) - f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})/2) + (d(s_{h*}^t, a_{u*}^t) - d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha)/2) + \lambda_{mh}$;
      $e(source, x_h^t) = \lambda_{\alpha h}$
    end for
  end for
end for
Solve the k-size s-t min cut [43] of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$;

Algorithm 1: Auxiliary graph construction and solving algorithm.

$$
\begin{aligned}
&\Gamma_2' \min_{\mathcal{K},\mathcal{B}} \sum_{t=1}^{T} \sum_{m=1}^{M} \left(\varepsilon_1 \sum_{u=1}^{U} \frac{a_{mu}^t o_u^t}{B_{mu}^t \log_2\left(1 + \mathrm{SINR}_{mu}^t\right)} \varepsilon_2 \sum_{h=1}^{H} \frac{s_{mh}^t C_h^t}{k_{mh}^t} \right) \\
&\quad + \mathrm{Cost}_I + \sum_{t=1}^{T} \sum_{u=1}^{U} \Lambda_u^t \\
&C3' : \sum_{u \in U} B_{mu}^t \le B^t, \forall m \in \mathcal{M}, t \in \mathcal{T} \\
&C4' : \sum_{h \in H} k_{mh}^t \le K_m, \forall m \in \mathcal{M}, t \in \mathcal{T},
\end{aligned}
\tag{42}
$$

where Λ is penalty function, which can be expressed as

$$
\begin{aligned}
Z * \max \Bigg(& \sum_{h=1}^{H} p_u^h \Bigg[\left(\frac{a_{mu}^t o_u^t}{B_{u*}^t \log_2\left(1 + \mathrm{SINR}_{u*}^t\right)} + d\left(s_{h*}^t, a_{u*}^t\right) \right) \\
& + g\left(s_{h*}^t, s_{h*}^{t-1}\right) \Bigg) + \frac{s_{mh}^t C_h^t}{k_{h*}^t} \Bigg] - \mathcal{D}_u, 0 \Bigg),
\end{aligned}
\tag{43}
$$

where Z goes to infinity. $d(s_{h*}^t, a_{u*}^t)$ and $g(s_{h*}^t, s_{h*}^{t-1})$ are constants, when $\mathcal{A} *$ and $\mathcal{S} *$ are fixed.

Since the structure like $1/B_{mu}^t$ is a well-known convex function, the optimization problem can be proved to be a convex problem.

Since the variable k_{mh}^t can affect multiple spectrum allocation variables, we denote those as global variables. Next, the local copy of the global variables would be introduced. Each base station can obtain a distributed feasible solution by decoupling the above problem.

For BS m, we introduce the new variables $\widehat{\mathbf{k}}_m = \{\widehat{k}_{mh}^{et} | e \in \mathcal{M}, m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}\}$ as the local information.

$$
\widehat{k}_{mh}^{et} = k_{eh}^t, \forall e \in \mathcal{M}, m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}. \tag{44}
$$

$\widehat{\mathbf{B}}_m = \{\widehat{B}_{mu}^t | m \in \mathcal{M}, u \in \mathcal{U}\}$ is the local variation and represents the bandwidth resource allocation scheme of the BS m. Thus, the feasible local variables of the BS m can be denoted as $\Phi_m = (\widehat{\mathbf{k}}_m, \widehat{\mathbf{B}}_m)$ and the constraint set of the objective function can be denoted as Ω.

Let $\Psi(\Phi_m)$ be the penalty function, when the Φ_m belongs to the constraint set Ω, i.e.,$\Phi_m \in \Omega$, we can get $\Psi(\Phi_m) = 0$. Otherwise, $\Psi(\Phi_m) = +\infty$. So, the objective functions equivalent to

$$
\begin{aligned}
&\min_{\Phi_m}, \quad \sum_{m=1}^{M} \Xi_m(\Phi_m) + \Psi(\Phi_m) + \mathrm{Cost}_I \\
&\text{s.t.} \quad \widehat{k}_{mh}^{et} - k_{eh}^t = 0, \forall e \in \mathcal{M}, m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T},
\end{aligned}
\tag{45}
$$

where $\Xi_m(\Phi_m) = \sum_{t=1}^{T} (\varepsilon_1 \sum_{u=1}^{U} (o_u^t / B_{mu}^t \log_2(1 + \mathrm{SINR}_{mu}^t)) + \varepsilon_2 \sum_{h=1}^{H} (C_h^t / k_{mh}^t))$, and in the above objective function, we can view Cost_I as a constant.

We separate the objective function into multiple local function of the corresponding BS. Each local function can determine its local variable by using local information. The Lagrange formula of the augmented problem is

$$\begin{aligned}\mathbb{L}\left(\{\Phi_m\}_{m\in\mathcal{M}}, \mathbf{k}, \{\xi_m\}_{m\in\mathcal{M}}\right) &= \sum_{m=1}^{M} \Xi_m(\Phi_m) + \Psi(\Phi_m) + Cost_I \\ &+ \sum_{m=1}^{M}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T} \xi_{mh}^{et}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t}\right) \\ &+ \frac{\zeta}{2}\sum_{m=1}^{M}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t}\right)^2,\end{aligned} \tag{46}$$

where $\xi_m = \{\xi_{mh}^{et}\}$ are the vectors of the Lagrange multipliers, and the penalty parameter is $\zeta/2 \in \mathbb{R} + +$.

In order to solve the above problems (46), the iterative process is as follows.

4.3.1. Local Variables.

$$\begin{aligned}\Phi_m^{[\iota+1]} = \arg\min_{\Phi_m} &\Xi_m(\Phi_m) + \Psi(\Phi_m) + Cost_I \\ &+ \sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T} \xi_{mu}^{et[\iota]}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t[\iota]}\right) \\ &+ \frac{\zeta}{2}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t[\iota]}\right)^2,\end{aligned} \tag{47}$$

where ι denotes the iteration times.

Since the updating process of Φ_m of each BS is independent, we can decouple the problem into M independent subproblems. We can update the local variables by solving the problem as follow:

$$\begin{aligned}\min_{\Phi_m} \quad & (\Phi_m) + \Psi(\Phi_m) + \mathrm{Cost}_I + \sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T} \xi_{mh}^{et[\iota]}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t[\iota]}\right) + \frac{\zeta}{2}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t[\iota]}\right)^2 \\ \text{s.t.} \quad & \Phi_m \in \Omega.\end{aligned} \tag{48}$$

We solve the above problem by CVX, due to it being convex, and then, broadcast the decision of each BS to other BSs.

4.3.2. Global Variables.

$$\begin{aligned}\mathbf{k}^{[\iota+1]} = \arg\min_{k_{eh}} &\sum_{m=1}^{M}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{T=1}^{T} \xi_{mh}^{et[\iota]}\left(\widehat{k}_{mh}^{et[\iota+1]} - k_{eh}^{t}\right) \\ &+ \frac{\zeta}{2}\sum_{m=1}^{M}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T}\left(\widehat{k}_{mh}^{et[\iota+1]} - k_{eh}^{t}\right)^2.\end{aligned} \tag{49}$$

The above problems are strictly convex and unconstrained quadratic problems, because we add the quadratic regular term to the augmented Lagrangian. Let the gradient of $\mathbf{k}$ be zero. We can get the following results:

$$\sum_{m=1}^{M} \xi_{mh}^{et[\iota]} + \zeta\sum_{m=1}^{M}\left(\widehat{k}_{mh}^{et[\iota+1]} - k_{eh}^{t}\right) = 0, \forall e, h, t. \tag{50}$$

And then, we can derive

$$k_{eh}^{t[\iota+1]} = \frac{1}{M\zeta}\sum_{m=1}^{M} \xi_{mh}^{et[\iota]} + \frac{1}{M}\sum_{m=1}^{M} \widehat{k}_{mh}^{et[\iota+1]}, \forall e, u, t. \tag{51}$$

By using $\sum_{m=1}^{M}\xi_{mh}^{et[\iota]} = 0$, we can derive

$$k_{eh}^{t[\iota+1]} = \frac{1}{M}\sum_{m=1}^{M} \widehat{k}_{mh}^{et[\iota+1]}, \forall e, u, t. \tag{52}$$

In other words, we can obtain global variables by averaging the corresponding updated local variables in each iteration.

4.3.3. Lagrange Multipliers.

$$\xi_m^{[\iota+1]} = \xi_m^{[\iota]} + \zeta\left(\widehat{\mathbf{k}}_m^{[\iota+1]} - \mathbf{k}^{[\iota+1]}\right). \tag{53}$$

At each iteration, we can calculate the Lagrange multipliers directly by using the updated local variables $\{\Phi_m\}$ and global variables $\{\mathbf{k}\}$. The formulation can be represented as follows:

$$\xi_{mh}^{et[\iota+1]} = \xi_{mh}^{et[\iota]} + \zeta\left(\widehat{k}_{mh}^{et[\iota+1]} - x_{eh}^{t[\iota+1]}\right). \tag{54}$$

4.3.4. Stopping Criterion and Convergence. The above problem is a convex problem with strong duality. When the number of iterations approaches infinity, the algorithm satisfies convergence. Therefore, the reasonable stopping criteria are given as follows:

$$\left\|\widehat{\mathbf{k}}_m^{[\iota+1]} - \mathbf{k}^{[\iota+1]}\right\|_2 \le \kappa_{\mathrm{pri}}, \forall m \in M, \tag{55}$$

$$\left\|\mathbf{k}^{[\iota+1]} - \mathbf{k}^{[\iota]}\right\|_2 \le \kappa_{\mathrm{dual}}, \forall m \in M, \tag{56}$$

where $\xi_{\mathrm{pri}} > 0$ and $\xi_{\mathrm{dual}} > 0$ indicate the primal feasibility and dual feasibility conditions, respectively, which are the small positive constant scalars.

The above iteration process based on convex optimization is concluded in Algorithm 2.

4.3.5. Two-Stage Iterative Algorithm Based on $\boldsymbol{\alpha}$ Expansion. Because there are many optimization variables in the original problem, the complexity of the algorithm is high. In order to reduce the algorithm complexity and obtain the

1: Initialization the number of iterations $\iota = 0$, global variables $\mathbf{k}^{[0]}$
and Lagrange multipliers $\xi^{[0]}$;
2: Set the maximum number of iterations $\iota_{\max}$ and the stopping criterion threshold ξ_{dual};
3: **while** $\iota < \iota_{\max}$, $\|\widehat{\mathbf{k}}_m^{[\iota+1]} - \mathbf{k}^{[\iota+1]}\|_2 > \kappa_{pri}$ and $\|\mathbf{k}^{[\iota+1]} - \mathbf{k}^{[\iota]}\|_2 > \kappa_{dual}$
4: Each BS m update Φ_m by solving problem (48), and share the local solution to other BSs;
5: Update the global variables $\mathbf{k}$ according to the formula (52);
6: Update the Lagrange multipliers ξ according to the formula (54);
7: $\iota = \iota + 1$;
8: **end while**
9: Output the optimal solution;

Algorithm 2: Resource allocation scheme based on convex optimization algorithm.

Input: $\mathcal{M}$ Set of BSs, $\mathcal{U}$ Set of players, $\mathcal{K}$ Set of groups, $\mathcal{T}$ Set of consecutive time slots;
Output: The variable $\mathcal{A}, \mathcal{S}, \mathcal{B}, \mathcal{K}$, and the minimum value of the objective function $Value_{best}$;
1: Initialization the variable s_{h*}^t = rand $[0, M]$, a_{u*}^t = rand $[0, M]$, k_{mh}^t, B_{mu}^t, and $Value_{best} = +\infty$
2: **for** iter=1:ι_3 **do**
3: **for** $\alpha \in \mathcal{M}, \sum_{h=1}^{H} \lambda_{\alpha h} s_{\alpha h} \le \Pi_\alpha$ **do**
4: run Algorithm 1, obtain $s_{h*}^{t\alpha}$, $a_{u*}^{t\alpha}$ and $Cost_I$
5: **for** iter=1:T **do**
6: run Algorithm 2, obtain k_{mh}^t, B_{mu}^t and $Value_{current}$
7: **end for**
8: **if** $Value_{current} < Value_{best}$ **then**
9: $Value_{bset} = Value_{current}$;
10: $s_{h*}^t = \alpha$, $a_{u*}^t = \alpha$;
11: **else**
12: $Value_{best} = Value_{best}$;
13: $s_{h*}^t = s_{h*}^t$, $a_{u*}^t = a_{u*}^t$;
14: **end if**
15: **end for**
16: **end for**

Algorithm 3: Two-stage iterative algorithm based on α expansion.

optimal solution to the original problem, we solve the original problem in two steps. So, we need to integrate the above two subalgorithms. Firstly, we input the result of Algorithm 1 as a fixed value into Algorithm 2 to solve Algorithm 2, and then, we compared the results of Algorithm 2 with the historical optimal results and updated the related variables. The above process is summarized in Algorithm 3.

Since we traverse for each MEC (Line 3 in Algorithm 3), the caching size can be restricted under Π_α at each round of α expansion.

4.4. Algorithm Complexity Analysis. Since Algorithms 1 and 2 are the modules invoked by Algorithm 3 for $M \times \iota_3$ times, where M is the number of MECs and ι_3 is the maximum number of iterations in Algorithm 3, we, respectively, investigate the complexity of Algorithms 1 and 2. According to paper [44], the complexity of the Algorithm 1 can be expressed as $\mathcal{O}(|\mathscr{E}||\mathscr{V}|^2)$, where $|\mathscr{V}|$ is the number of vertices and $|\mathscr{E}|$ is the number of edges in the constructed graph. In our case, $|\mathscr{V}| = T(U+H)+2$ is bounded by $\mathcal{O}(T(U+H))$; $|\mathscr{E}| = 3HUT + H(T+1) + TH$ is bounded by $\mathcal{O}(TUH)$.

Table 2: The simulation parameters.

Simulation parameters	Value
The total bandwidth	[0.8–1.2]GHz
The number of players	100
The number of BSs	10
The number of VR service modules	40
The downlink path loss exponent	[2.75–4.75]
The power spectral density of noise	-174 dBm/Hz
The transmission power of the players	0.1 W
The storage capability of the MEC	[400–900]G
The computing capability of the BS	[30–80]GHz

Therefore, the complexity of Algorithm 1 is $\mathcal{O}(T^3U^4)$, due to $H \le U$. For Algorithm 2, the variables a_{mu}^t and s_{mu}^t have been fixed, and the remaining question can be broken down into solving local optimization problem (48) at each BS by using ADMM algorithm, whose complexity is $\mathcal{O}(UH)$. $\iota_{\max}$ is the number of iterations required for Algorithm 2

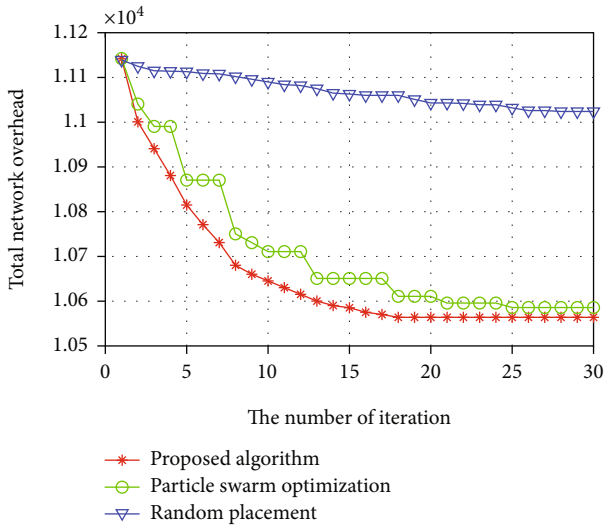


Figure 4: Total network overhead versus iteration times.

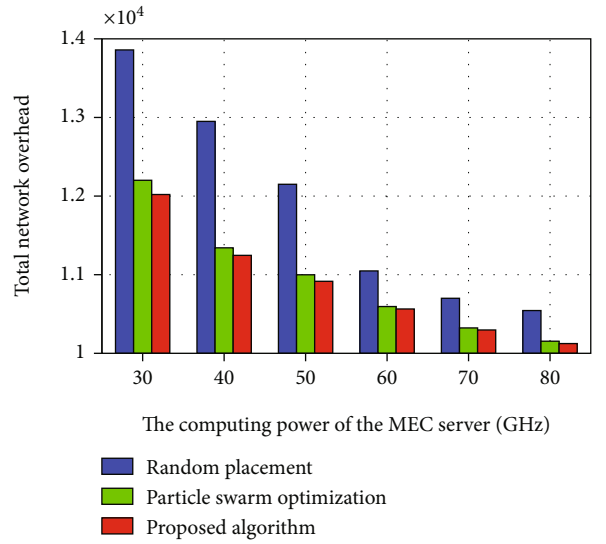


Figure 6: Total network overhead versus computing power of MEC server.

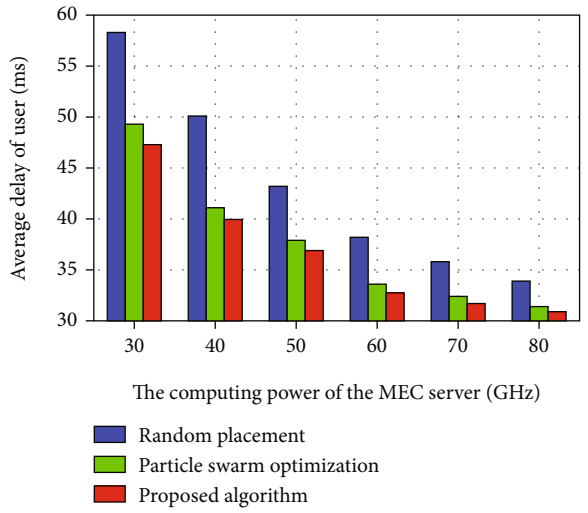


Figure 5: Average delay of user versus computing power of MEC server.

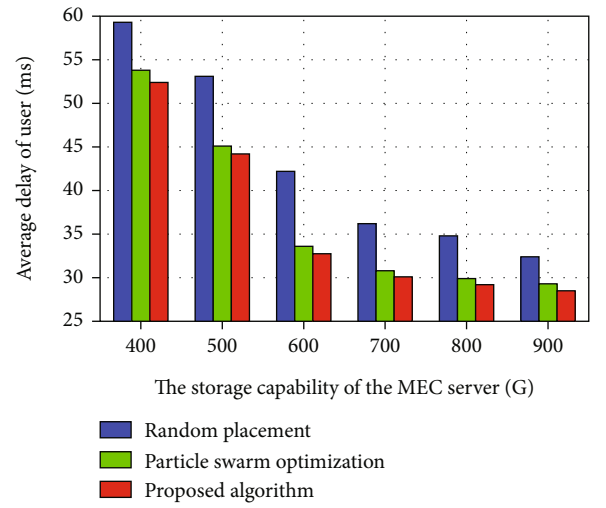


Figure 7: Average delay of user versus storage capacity of MEC server.

convergence; the total computational complexity is $\mathcal{O}(\iota_{\max} UH)$. Therefore, the overall complexity of Algorithm 3 is $\mathcal{O}(\iota_3 M(T^3U^4 + \iota_{\max} U^2 T))$.

5. Simulation Results and Discussions

In a wireless cellular network, it is assumed that 100 players and 10 base stations are randomly distributed in a circle with a radius of 100 m; other major simulation parameters are shown in Table 2.

To evaluate the performance of our proposed approach, we compare our proposed α expansion-based two-stage approach to two other approaches: (1) placing each VR service module randomly on a MEC at each time slot, as labeled as "random placement," and (2) particle swarm optimization was used to solve the objective function, as labeled as "particle swarm optimization."

In Figure 4, we iteratively find the minimum value of the total network overhead under the condition that the maximizing computing capacity of each MEC server is 60 GHz and the maximizing storage capacity of each MEC server is 600 G, where total network overhead is the sum of the adjusted placement cost, communication cost, migration cost, and rendering cost, i.e., this paper's object function $\sum_{t=1}^{T} \varepsilon_1 \text{Cost}_C^t + \varepsilon_2 \text{Cost}_R^t + \varepsilon_3 \text{Cost}_P^t + \varepsilon_4 \text{Cost}_M^t$. As shown above, the total network overhead of our proposed scheme and particle swarm optimization decreases rapidly as the iteration increases at the beginning, and then, the total network overhead converges and remains at an almost constant value. Moreover, it can be seen from the iteration diagram that our proposed algorithm converges in about 18 generations, while particle swarm

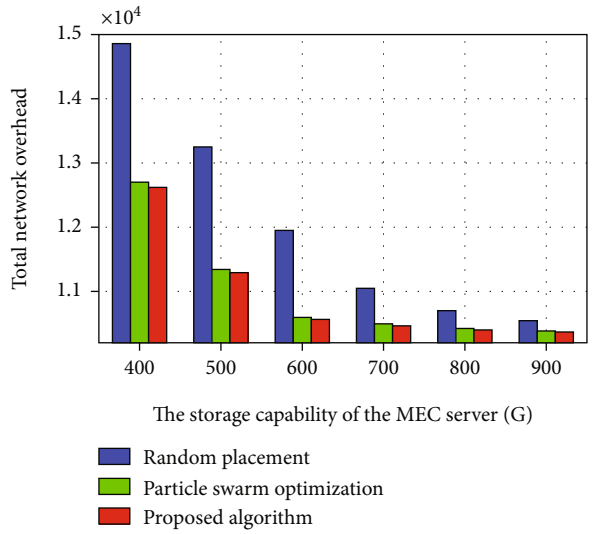


Figure 8: Total network overhead versus storage capacity of MEC server.

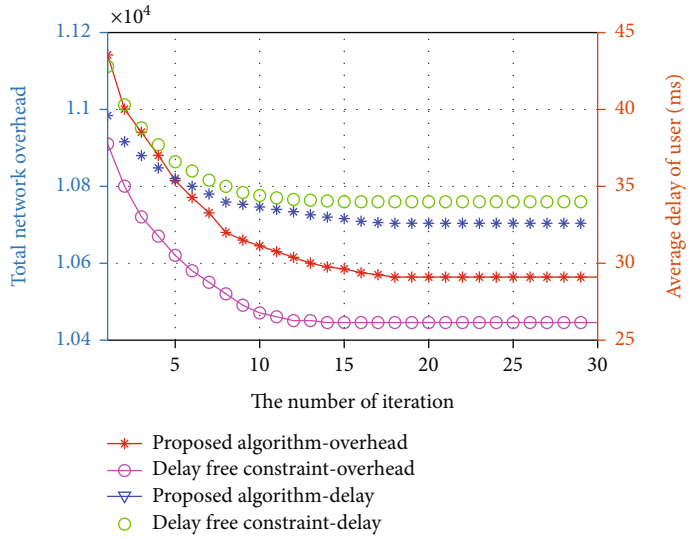


Figure 9: The influence of delay constraint on total network overhead and average delay of user.

optimization converges in about 25 generations. So, compared with other schemes, our proposed algorithm converges faster in the iteration process and keeps the lowest total network overhead.

Figure 5 shows the relationship between the computing power of the MEC server and the user average latency. Figure 6 shows the relationship between the computing power of the MEC server and the total network overhead. In the above two figures, as the computing power of the MEC server increases, the average latency and total network overhead of the user are greatly reduced. This is mainly because the more computing resources a MEC server can provide to the player, the less latency it needs to perform rendering. At the same time, the richer computing resources on the MEC server, the more MEC servers the system could be chosen to provide rendering services for a group of VR players, which saves the network cost of routing.

Figure 7 shows the relationship between the storage capability of the MEC server and the user average latency. Figure 8 shows the relationship between the storage capability of the MEC server and the total network overhead. As shown in the above two figures, the placement strategy proposed by us can effectively reduce the total network overhead. Moreover, with the storage capacity of the MEC server increasing, the average latency of user and total network overhead is greatly reduced. This is mainly because the larger the storage capacity of the MEC server, the more VR service modules can be placed on each edge node, which can reduce the migration costs between two base stations to a certain extent. Especially when the number of VR service modules that can be placed on the MEC server is small, in order to meet the video processing requirements of the constantly moving player, VR service modules need to migrate frequently between base stations. As shown in Figure 8, when

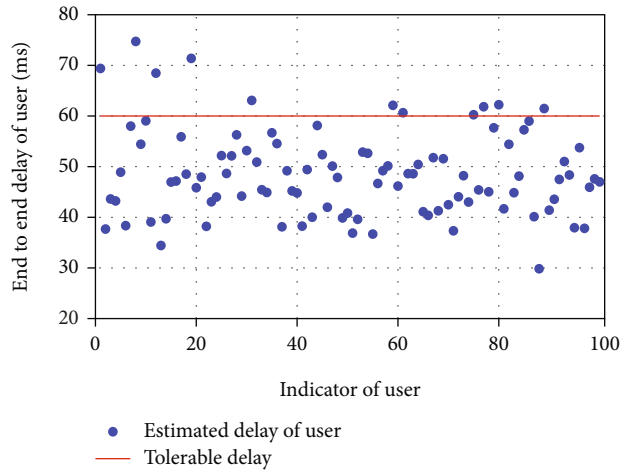


Figure 10: The actual delay of each player under the condition of no delay constraint and the tolerable delay.

the storage capability of the MEC server is less than 600 G, VR service module migration between base stations becomes frequent, and the total network overhead increases greatly.

In Figure 9, we compare the user average delay and total network overhead without delay constraint with the user average delay and total network overhead with delay constraint. The network parameter is the maximizing computing capacity of each MEC server is 60 GHz, and the storage capacity of each MEC server is 600 G. When there is no need to consider satisfying the delay constraint of each user, the feasible domain of the target problem becomes larger, and the total network cost is reduced compared with when the delay constraint is considered, but the average delay of the user will increase. At the same time, some users cannot complete their corresponding video processing tasks within the tolerable delay, as shown in Figure 10.

6. Conclusion

In this paper, we develop dynamical service module placement strategies based on rendering-aware to minimize the sum of the network costs over a long time under satisfying the delay constraint of each player. The strategies jointly consider the resource allocation scheme within each time slot and the service module migration scheme between different base stations in the adjacent time slot. Moreover, we propose a two-stage algorithm based on graph cut and convex optimization to solve the objective function. In future work, we will study the online placement strategy of VR service modules to further improve user experience and reduce network overhead in the process of VR video stream delivery and computing. In addition, we will extend our work to the security [45] and low-delay delivery of all kinds of superlarge video streams.

Data Availability

The simulation data used to support the findings of this study are included in the article. The research status data used to support the findings of this study are available in the references of this article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62171061).

References

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

[2] R. Fantacci and B. Picano, "End-to-end delay bound for wireless uVR services over 6G terahertz communications," *IEEE Internet of Things Journal*, vol. 8, no. 23, pp. 17090–17099, 2021.

[3] P. Bhattacharya, D. Saraswat, A. Dave et al., "Coalition of 6G and blockchain in AR/VR space: challenges and future directions," *IEEE Access*, vol. 9, pp. 168455–168484, 2021.

[4] P. Lin, Q. Song, F. R. Yu, D. Wang, and L. Guo, "Task offloading for wireless VR-enabled medical treatment with blockchain security using collective reinforcement learning," *IEEE Internet of Things Journal*, vol. 8, no. 21, pp. 15749–15761, 2021.

[5] C. Zheng, S. Liu, Y. Huang, and L. Yang, "Hybrid policy learning for energy-latency tradeoff in MEC-assisted VR video service," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 9, pp. 9006–9021, 2021.

[6] Y. Cao, R. Ji, L. Ji, G. Lei, H. Wang, and X. Shao, "l^2-MPTCP: a learning-driven latency-aware multipath transport scheme for industrial internet applications," *IEEE Transactions on Industrial Informatics*, 2022.

[7] G. Lei, L. Ji, R. Ji, Y. Cao, and X. Huang, "Extracting low-rate DDoS attack characteristics: the case of multipath TCP-based communication networks," *Wireless Communications and Mobile Computing*, vol. 2021, no. 4, Article ID 2264187, p. 10, 2021.

[8] L. Qin, Y. Cao, X. Shao et al., "A deep heterogeneous optimization framework for Bayesian compressive sensing," *Computer Communications*, vol. 178, pp. 74–82, 2021.

[9] Y. Ye, R. Q. Hu, G. Lu, and L. Shi, "Enhance latency-constrained computation in mec networks using uplink Noma," *IEEE Transactions on Communications*, vol. 68, no. 4, pp. 2409–2425, 2020.

[10] X. Liu and Y. Deng, "A decoupled learning strategy for mec-enabled wireless virtual reality (vr) network," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, Montreal, QC, Canada, 2021.

[11] J. Du, F. R. Yu, G. Lu, J. Wang, J. Jiang, and X. Chu, "MEC-assisted immersive vr video streaming over terahertz wireless networks: a deep reinforcement learning approach," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9517–9529, 2020.

[12] L. Wang, L. Jiao, J. Li, J. Gedeon, and M. Muhlhauser, "MOERA: mobility-agnostic online resource allocation for edge computing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1843–1856, 2019.

[13] H. Wu, L. Liu, X. Zhang, and H. Ma, "Vbargain: a market-driven quality oriented incentive for mobile video offloading," *IEEE Transactions on Mobile Computing*, vol. 18, no. 9, pp. 2203–2216, 2019.

[14] X. Shao, G. Hasegawa, M. Dong, Z. Liu, H. Masui, and Y. Ji, "An online orchestration mechanism for general-purpose edge computing," *IEEE Transactions on Services Computing*, pp. 1–1, 2022.

[15] I. Labriji, F. Meneghello, D. Cecchinato et al., "Mobility aware and dynamic migration of MEC services for the internet of vehicles," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 570–584, 2021.

[16] F. Guo, F. R. Yu, H. Zhang, H. Ji, V. C. M. Leung, and X. Li, "An adaptive wireless virtual reality framework in future wireless networks: a distributed learning approach," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8514–8528, 2020.

[17] A. Yousafzai, I. Yaqoob, M. Imran, A. Gani, and R. Md Noor, "Process migration-based computational offloading framework for IoT-supported mobile edge/cloud computing," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4171–4182, 2020.

[18] J. Feng, F. R. Yu, Q. Pei, J. Du, and L. Zhu, "Joint optimization of radio and computational resources allocation in blockchain-enabled mobile edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 19, no. 6, pp. 4321–4334, 2020.

[19] J. Liu and Q. Zhang, "Reliability and latency aware code-partitioning offloading in mobile edge computing," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–7, Marrakesh, Morocco, 2019.

[20] L. Pu, L. Jiao, X. Chen, L. Wang, Q. Xie, and J. Xu, "Online resource allocation, content placement and request routing for cost-efficient edge caching in cloud radio access networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1751–1767, 2018.

[21] L. Wang, L. Jiao, T. He, J. Li, and M. Muhlhauser, "Service entity placement for social virtual reality applications in edge computing," in *IEEE INFOCOM 2018- IEEE Conference on Computer Communications*, pp. 468–476, Honolulu, HI, USA, 2018.

[22] L. Guo, J. Pang, and A. Walid, "Joint placement and routing of network function chains in data centers," in *IEEE INFOCOM 2018 – IEEE Conference on Computer Communications*, pp. 612–620, Honolulu, HI, USA, 2018.

[23] P.-Y. Chou, W.-Y. Chen, C.-Y. Wang, R.-H. Hwang, and W.-T. Chen, "Deep reinforcement learning for mec streaming with joint user association and resource management," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–7, Dublin, Ireland, 2020.

[24] V. Farhadi, F. Mehmeti, T. He et al., "Service placement and request scheduling for dataintensive applications in edge clouds," in *IEEE INFOCOM 2019 – IEEE Conference on Computer Communications*, pp. 1279–1287, Paris, France, 2019.

[25] Y. Han, D. Guo, W. Cai, X. Wang, and V. Leung, "Virtual machine placement optimization in mobile cloud gaming through QoE-oriented resource competition," *IEEE Transactions on Cloud Computing*, pp. 1–1, 2020.

[26] X. Shao, H. Asaeda, M. Dong, and Z. Ma, "Cooperative inter-domain cache sharing for information-centric networking via a bargaining game approach," *IEEE Transactions on Network Science and Engineering*, vol. 6, no. 4, pp. 698–710, 2019.

[27] Y. Liu, J. Liu, A. Argyriou, L. Wang, and Z. Xu, "Rendering-aware VR video caching over multi-cell MEC networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 3, pp. 2728–2742, 2021.

[28] J. Dai, Z. Zhang, S. Mao, and D. Liu, "A view synthesis-based 360° VR caching system over MEC-enabled C-RAN," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 10, pp. 3843–3855, 2020.

[29] Y. Ye, L. Shi, X. Chu, R. Q. Hu, and G. Lu, "Resource allocation in backscatter-assisted wireless powered MEC networks with limited MEC computation capacity," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2022.

[30] W. Lu, X. Meng, and G. Guo, "Fast service migration method based on virtual machine technology for MEC," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4344–4354, 2019.

[31] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A survey on service migration in mobile edge computing," *IEEE Access*, vol. 6, no. 23, pp. 23511–23528, 2018.

[32] L. Yang, D. Yang, J. Cao, Y. Sahni, and X. Xu, "QoS guaranteed resource allocation for live virtual machine migration in edge clouds," *IEEE Access*, vol. 8, pp. 78441–78451, 2020.

[33] L. Liang, J. Xiao, Z. Ren, Z. Chen, and Y. Jia, "Particle swarm based service migration scheme in the edge computing environment," *IEEE Access*, vol. 8, pp. 45596–45606, 2020.

[34] P. Fang, Y. Zhao, Z. Liu, J. Gao, and Z. Chen, "Resource allocation strategy for MEC system based on vm migration and rf energy harvesting," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pp. 1–6, Antwerp, Belgium, 2020.

[35] K. Liu, J. Peng, J. Wang, W. Liu, Z. Huang, and J. Pan, "Scalable and adaptive data replica placement for geo-distributed cloud storages," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 7, pp. 1575–1587, 2020.

[36] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *IEEE INFOCOM 2019- IEEE Conference on Computer Communications*, pp. 10–18, Paris, France, 2019.

[37] S. Li, P. Lin, J. Song, and Q. Song, "Computing-assisted task offloading and resource allocation for wireless VR systems," in *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pp. 368–372, Chengdu, China, 2020.

[38] C. Zheng, S. Liu, Y. Huang, and L. Yang, "MEC-enabled wireless VR video service: a learning-based mixed strategy for energy-latency tradeoff," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, Seoul, Korea (South), 2020.

[39] H. Xiao, C. Xu, Z. Feng et al., "A transcoding-enabled 360° VR video caching and delivery framework for edge-enhanced next-generation wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 5, pp. 1615–1631, 2022.

[40] K. Long, Y. Cui, C. Ye, and Z. Liu, "Optimal wireless streaming of multi-quality 360 VR video by exploiting natural, relative smoothness-enabled, and transcoding-enabled multicast opportunities," *IEEE Transactions on Multimedia*, vol. 23, pp. 3670–3683, 2021.

[41] T. Bai and R. W. Heath, "Coverage and rate analysis for millimeter-wave cellular networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 2, pp. 1100–1114, 2015.

[42] S. Singh, M. N. Kulkarni, A. Ghosh, and J. G. Andrews, "Tractable model for rate in self-backhauled millimeter wave cellular

networks," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 10, pp. 2196–2211, 2015.

[43] P. Zhang, "A new approximation algorithm for the unbalanced min *s-t* cut problem," *Theoretical Computer Science*, vol. 609, pp. 658–665, 2016.

[44] J. R. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *Journal of the ACM (JACM)*, vol. 19, no. 2, article 248C264, pp. 248–264, 1972.

[45] Y. Cao, R. Ji, L. Ji, X. Shao, G. Lei, and H. Wang, "MPTCP-*me*Learning: a multi-expert learning-based MPTCP extension to enhance multipathing robustness against network attacks," *Transactions on Information and Systems*, vol. E104.D, no. 11, pp. 1795–1804, 2021.