This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

Dynamic Service Placement for Virtual Reality Group Gaming on Mobile Edge Cloudlets

Yuan Zhang, Lei Jiao, *Member, IEEE,* Jinyao Yan, and Xiaojun Lin, *Fellow, IEEE*

***Abstract*—To realize mobile Virtual Reality (VR) group gaming services which are currently hampered by the prohibitive bandwidth and the stringent delay requirements, we investigate the problem of provisioning such services using the emerging Mobile Edge Cloudlet (MEC) networks with a distributed content rendering architecture. The underlying dynamic rendering-module placement problem requires to optimize the service's operational cost and the users' end-to-end performance, involving multiple intertwined conflicting system objectives which are discrete, nonconvex, and higher-degree polynomial functions, with coupled decisions and arbitrary user dynamics over time. We solve this online placement problem by leveraging Model Predictive Control (MPC) and overcoming the aforementioned challenges over each prediction window. We explore the connection between the placement problem and the minimal s-t cut problem in graph theory, and solve the former via solving a series of instances of the latter. We formally prove the performance guarantee of our approach. We also conduct extensive trace-driven evaluations and demonstrate the superior practical performance of our MPC-based approach compared to the de facto practices and the state-of-the-art alternatives.**

***Index Terms*—Virtual Reality; Group Gaming; Mobile Edge Computing; Service Placement; Model Predictive Control; Graph Cut.**

I. Introduction

WHILE the global Virtual Reality (VR) gaming market size is projected to reach $45 billion by 2025 [1], provisioning VR gaming services "anywhere, anytime" to large-scale untethered players (wearing Google Cardboard [2] or Samsung Gear VR [3], for example) imposes significant challenges to today's mobile network infrastructures. The prohibitive bandwidth for streaming panoramic VR frames and the stringent delay for responding to players' control actions requires to push the gaming computation, especially the content rendering, close to players [25]. This seamlessly matches the emerging paradigm of mobile edge computing [12], where telecom carriers and service providers place Mobile Edge Cloudlets (MECs), e.g., micro datacenters or servers, to run computation at user neighborhoods, cellular base stations, and WiFi hotspots.

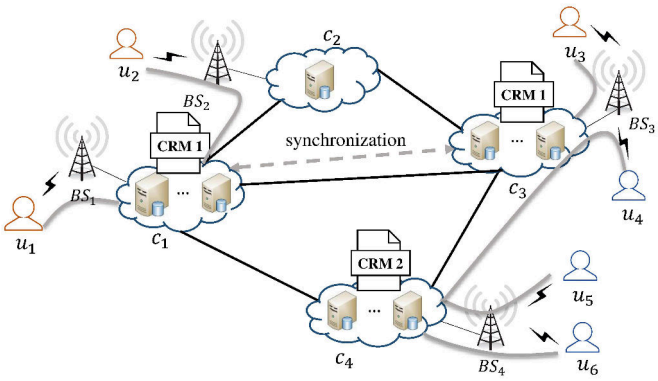


Fig. 1: An overview of the system architecture. u_1 through u_6 are players, where u_1, u_2, and u_3 are in Group 1, and u_4, u_5, and u_6 are in Group 2. c_1 through c_4 are MECs with base stations BS_1 through BS_4, respectively. CRM 1 is placed on c_1 for u_1 and u_2, and on c_3 for u_3. CRM 2 is placed on c_4 for u_4, u_5, and u_6.

Unlike existing research on exploiting the MEC to facilitate single-user VR gaming [25], [26], in this paper, we consider provisioning VR group gaming services using distributed MEC networks to a large number of mobile players. Group gaming allows multiple players to form a group and play a common game, such as completing a task or having a battle, which is extremely popular, occupying 8 seats out of the top 10 highest earning games in 2017 [4]. For group gaming, we notice that scenes of group players, such as the environments of the avatars in the games, are often heavily overlapped. In fact, for the Multi-player Online Battle Arena (MOBA) game "Arena of Valor" [6] and the Role-Playing Game (RPG) "Onmyoji" [5], we collected replays from 20 5v5 battles and 20 3v1 battles, respectively, and analyzed their frame patterns. We replayed the recordings, took screenshots every 2 minutes, and obtained more than 4000 screenshots. We found that for Arena of Valor the average ratio of the different pixels across all the players was 12.25% and for Onmyoji was only 3.64%, confirming that players in the same battle shared a large proportion of views.

We consider a distributed rendering architecture, as exhibited in several existing efforts where the common background environments and the player-specific foregrounds of VR games can be rendered separately [25], [26]. We call the correspond-

Manuscript received December 15, 2018; revised June 27, 2019; accepted June 28, 2019. This work was partially supported by the "Fundamental Research Funds for the Central Universities", and by National Science Foundation under Grants CNS-1717493 and CNS-1703014. *(Corresponding Authors: Lei Jiao; Jinyao Yan.)*

Yuan Zhang is with the Key Laboratory of Media Audio and Video, Communication University of China, Beijing 100024, China (e-mail: yuanzhang@cuc.edu.cn).

Lei Jiao is with the Department of Computer and Information Science, University of Oregon, Eugene, OR 97403, USA (e-mail: jiao@cs.uoregon.edu).

Jinyao Yan is with the Key Laboratory of Media Audio and Video, Communication University of China, Beijing 100024, China (e-mail: jyan@cuc.edu.cn).

Xiaojun Lin is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA (e-mail: linx@ecn.purdue.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier XXX

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

ing modules the Common Rendering Module (CRM) and the Individual Rendering Module (IRM), respectively. For mobile VR group gaming, we can use the MECs to run the CRMs of the players while letting each player run the IRM on one's own mobile VR device. The benefits of such distributed rendering are two-fold: for one, players in the same group can share a common CRM at an MEC in order to reduce the rendering workload on the infrastructure; for the other, as the background views often contain large percentage of infrequent changes between frames, video encoding/decoding techniques can be leveraged to further reduce the network workload. Figure 1 examplifies this system architecture, where players send control actions to the MECs and receive frames from them through nearby cellular base stations.

We focus on the fundamental problem of dynamically placing the CRMs over the distributed and heterogenous MECs in order to optimize the service' operational cost and the players' service quality over a long time span. We consider a time-slotted system. At each time slot, we decide on which MEC to place each player's CRM. Our optimization is characterized by the multiple intertwined, conflicting objectives, including the following: (1) the computational overhead of the CRM execution, which encourages placing the CRMs on the most computationally-efficient MECs; (2) the communication traffic and delay of the in-group CRM synchronization (e.g., for gaming information like avatar's location, blood volume, and score) and the pairwise player-to-player interaction (e.g., via text, voice) regardless of groups, which encourages placing the CRMs close to one another; (3) the colocation interference due to CRM scheduling and computation slow-down, which encourages placing each CRM on a different MEC; (4) the migration traffic and delay of CRMs between MECs as players move around, or join and leave the system, which encourages keeping each CRM's location unchanged over time. We collectively call the first three types of cost the "service" cost, and the last type of cost the "switching" cost. This optimization problem features two primary challenges:

First, service cost involves nonconvex higher-degree functions. Defined at every time slot, the service cost depends on the joint placement of CRMs. For instance, CRM synchronization depends on the placement of the CRMs of the same group; inter-player interaction depends on the placement of the CRMs of the pair of interacting players; colocation performance degradation depends on the placement of all the CRMs in the system. Such cost functions are discrete and nonconvex, and can be higher-degree polynomials, collectively hard to optimize [41]–[44]. With the service cost alone, the problem is already NP-hard [41]–[44].

Second, switching cost couples decisions over time. Defined across every pair of consecutive time slots, the switching cost depends on the placement of every CRM over time. Each CRM may need to be migrated across MECs to accommodate the arbitrary player dynamics, while avoiding moving it back and forth too frequently and incurring excessive costs. Placement decisions at each time slot influence the migration cost between that current time slot and the future time slots; it is particularly hard to make such decisions in an online manner with zero or limited knowledge of future player dynamics [19], [21], [24], [29], [32]–[34].

Existing works on mobile VR gaming [20], [22], [25], [26] often focus on individual VR systems of a small and local user scale, without considering the management of the distributed infrastructure to provide VR services to large-scale mobile users. In contrast, for non-VR services, despite previous efforts have extensively studied large-scale online service placements over clouds [15], [16], [31], they do not always capture user dynamics and the associated switching cost; for those that accommodate user mobilities via cloudlets [19], [21], [24], [29], [32]–[34], they fail to capture the VR-gaming-specific factors such as end-to-end user interaction, in-group multicast, per-group common rendering, and user joining/leaving the system. In addition, the works for non-VR services [15], [16], [19], [21], [24], [29], [31]–[34] often target linear/convex costs and other settings, and are thus inapplicable to our problem.

To overcome the aforementioned challenges, we propose a combinatorial, polynomial-time algorithm based on "graph cuts" for the problem over a prediction window, and embed it into the Model Predictive Control (MPC) framework to construct an online algorithm which solves our problem over time. Our approach, at each time slot, leverages the predicted player dynamics in a prediction window, solves the problem over that prediction window via our graph-cuts-based combinatorial algorithm, and applies the placement decisions for the current time slot. Our main contributions are listed as follows:

- We propose techniques to transform the dynamic CRM placement problem over the prediction window into the graph model to connect to the graph-cut technique. To be specific, (1) we show that the dynamic switching cost can be considered as self-to-self interactions (versus inter-player interactions) from a static view, and can be treated uniformly as the static service cost; (2) we propose to consider the same player at different time slots as different players, in order to construct a single graph, rather than multiple graphs, for the entire prediction window; (3) we introduce a virtual MEC to host the CRMs of all the players that are not in the system currently to represent the player joining/leaving actions.
- We develop algebraic conversions to transform our placement problem into the graph-theoretic minimal s-t cut problem, and exhibit that our problem can be solved via solving a series of corresponding graph cuts using existing polynomial-time algorithms.
- We demonstrate the weighted graph construction, establish the problem equivalence, and rigorously prove that within each MPC prediction window, the total cost of our placement result is bounded by a constant approximation factor times the theoretical optimum. We also derive the time complexity of our proposed solution.
- We conduct extensive trace-driven experiments, contrast our approach to multiple algorithms, and observe the following results: (1) our approach overwhelmingly outperforms baseline dynamic placement methods such as random placement and nearest placement, achieving $2.88\times$ and $3.25\times$ less cost; (2) compared to the "greedy" approach which uses the state-of-the-art discrete optimization solver Gurobi [8] to solve the problem optimally

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

in each prediction window, our approach achieves around 20% more cost with around 60% less running time for small-scale experiments, and more benefit for large-scale experiments (as Gurobi takes an unacceptably long time to finish); (3) as the prediction window size increases, our approach rapidly approaches the offline optimum; (4) our approach converges fast, scales well, and is robust when using inaccurate predictions of player dynamics.

II. Related Work

We summarize existing research from the following two aspects: (1) mobile VR with gaming, and (2) dynamic service placement on clouds and cloudlets. We point out how they fall insufficient compared to our work in this paper.

A. Mobile Virtual Reality with Gaming

Several recent works focus on improving VR rendering to offer immerse mobile VR experience via commodity untethered products such as Google Cardboard [2] and Samsung Gear VR [3]. Flashback [20] pre-renders all possible scenes for different user positions and orientations, and caches them on the mobile device. MoVR [22] uses mmWave radios as the transmit channel to enable multi-Gbps wireless communication between mobile VR devices and the server using mmWave smart mirrors. Some other works explore mobile edge servers to support VR rendering. CloudVR [27] proposes to use the server-client paradigm to augment the VR rendering. To reduce the network transmission delay, it directly uses the TCP socket to transmit the H.264 stream after rendering, rather than using the HTTP-based streaming protocols such as DASH and HLS. Furion [25] uses edge servers to render the background panorama of the VR game and transmits the compressed data to the mobile devices. LTE-VR [26], on the other hand, renders all the scenes on the edge server and uses side channels to transmit the signaling operations to reduce the network delay. ITEM [29] provisions distributed VR services over the MEC network with user interactions considered. EC+ [30] proposes to facilitate Massively Multiplayer Online Games by using MEC networks to handle the "view change event" rendering in order to satisfy the low latency requirement of VR gaming.

Among these works, [20], [22], [25]–[27] focus on developing individual VR systems that leverage specific hardware settings to enhance the VR experiences. They lack the consideration of managing and provisioning the VR gaming services as a whole over a large-scale distributed infrastructure in a cost-efficient manner, as what we study in this paper. From this perspective, ITEM [29] and EC+ [30] might be the most related to our work. However, the problem space is different in that we consider a group gaming scenario which introduces the unique in-group synchronization and per-group common rendering problem. [29] focuses on a static scenario and does not account for the switching cost caused by user dynamics. [30] addresses the dynamic service placement problem, but only considers traffic migration with player movements. We consider a more general case with multiple interwind, conflicting cost functions including computation cost, communication cost, colocation cost, and switching (migration) cost. The unique in-group synchronization and per-group common rendering problem also makes the Markov Decision Process method in [30] inapplicable to our problem.

B. Dynamic Service Placement on Clouds and Cloudlets

Dynamic service placement has been extensively studied in the cloud environment, involving virtual machine consolidation [15], load balancing [16], network function chaining [31], and so on. However, such studies do not typically target the operational cost and/or service quality driven by *user mobility*, and are thus insufficient compared to our work in general.

Recent research has started to focus on addressing the key challenge of user dynamics in the MEC environment via online service placement and migration. A branch of such works require no prediction of user dynamics, but formulate a Markov decision process based on the assumption that user mobility can be approximated by Markov chains [19], [21]. There also exist works that can accommodate arbitrary user mobilities online without using prediction. For example, Ouyang et al. [33] migrate services in the MEC network to follow the users for optimal user-perceived latency, subject to long-term cost budget constraints; Wang et al. [32] distribute users' workload in response to user movement around the MEC network to optimize the operational and migration cost over time; Chen et al. [34] place service instances over selected MECs, "learn" the resulting benefit, and improve the placement decisions over time; [35]–[37] consider virtual function placement in the Evolved Packet Core to provide deployment flexibility while reducing cost. In terms of exploring the usage of the predictions of user dynamics, Wang et al. [24] might be the closest to us, and propose an algorithm for each prediction window and embed it into an online algorithm, despite both their optimization objective and algorithms differ from ours.

This group of works lack the consideration of all the following factors that feature our problem in this paper: user-to-user interactions, in-group multicast, the MEC occupation effect, and user joining/leaving. Moreover, their proposed algorithmic techniques are also inapplicable to our problem due to the two primary challenges (nonconvex, higher-degree functions and time coupling switching cost) resulting from our unique problem space. For example, [19] and [21] are restricted by Markov assumptions; [24] and [33] target the time-averaged limit of the total cost; [32], [35]–[37] work with linear cost functions; [34] is dedicated to the cost-oblivious settings. Such differences and insufficiencies all motivate us to design novel predictive algorithms.

III. Model and Problem Formulation

A. Settings, Assumptions, and Notations

We consider a geographically distributed MEC network. Since the VR gaming services that we study often handle dynamic workloads and inputs, and require dynamic decisions to orchestrate computing/communication resources, we model our problem as a time-slotted system, where we use T to denote the set of consecutive time slots under consideration. We use I to represent the set of the MECs in the system. We use U_t, $\forall t \in T$ to represent the set of all the players

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

in the system at the time slot t, and use the two terms "user" and "player" interchangeably in this paper. Since we are considering the group gaming scenario, the player set U_t can be further represented as $U_t = \bigcup_{j \in J_t} U_{j,t}$, $\forall t \in T$, where $U_{j,t}$ is the set of the players in the group j at the time slot t and J_t is the set of the groups in the system at the time slot t. A player can only join one group at any time; we do not consider the case where a player participates in multiple groups simultaneously. It is common for modern games to allow players to interact with each other (e.g., voice/text chatting, "borrowing" avatars [5]) while playing the game even if they are not in the same group. Thus, we use $L_t \subseteq U_t \times U_t$, $\forall t \in T$ to denote the set of the pairs of interacting users at time t. We also use $p_{u,t}$ to denote the MEC that hosts the CRM of player u at time t, and use $p^{\dagger}_{u,t}$ to denote the MEC that is the closest to player u at time t, via which u connects to the MEC network to access u's CRM. Note that if there are multiple players of the same group that have their CRMs on the same MEC, we keep only one copy of the CRM on that MEC for that group.

Table. I summarizes the notations that will be used throughout this paper. In the following sections, we will continue to explain the notations as we proceed to our formulations.

TABLE I: Notations

Symbol	Definition
I	the set of all the MECs
U_t	the set of all the players at time (slot) t
J_t	the set of all the groups at time t
$U_{j,t}$	the set of all the players of group j at time t
L_t	the set of all the pairs of interacting players at time t
$p_{u,t}$	the MEC that hosts the CRM of player u at time t
$p^{\dagger}_{u,t}$	the MEC that is the nearest to player u at time t
$p^{\alpha}_{u,t}$	the MEC for player u after the α expansion at time t
$R_{j,t}$	the workload of the CRM of group j at time t
h_i	the processing capability of MEC i
$d(p,q)$	the network delay between MECs p and q
$f_{u,v,t}$	the interaction rate between players u and v at time t
$f_{u,t}$	the frame rate of player u at time t
$f_{j,t}$	the synchronization rate of group j at time t
$\delta_{i,j,t}$	the binary indicator of whether there exists at least one player u in $U_{j,t}$ such that $p_{u,t} = i$ at time t
$x_{u,t}$	the binary variable of whether to place the CRM of player u on MEC α in an α expansion at time t

B. Modeling Computation Overhead

We model the computation overhead of all CRMs in the system. Such overhead is incurred by running CRMs at MECs for rendering and encoding/decoding the VR frames. We let $R_{j,t}$ be the computation workload of the CRM of group j at time t, and let h_i be the processing capability of MEC i. $\delta_{i,j,t}$ is a binary indicator to represent whether there is at least one user of group j, who has the corresponding CRM placed on MEC i at time t:

$$\delta_{i,j,t} \triangleq \begin{cases} 1, & \exists u \in U_{j,t},\, p_{u,t} = i\,; \\ 0, & \text{otherwise.} \end{cases}$$

The total computation overhead over all MECs and over all groups at t is thus calculated as

$$E_{1,t} = \sum\nolimits_{i \in I} \sum\nolimits_{j \in J_t} \frac{R_{j,t}}{h_i} \delta_{i,j,t}.$$

C. Modeling Communication Delay

We model the total communication delay as the metric for the end-to-end users' quality of experience for mobile VR gaming. Notice that in our system, one time slot is in the order of minutes, while the network delay between two MECs, or between MECs to end users should be under tens of milliseconds. Therefor, the time length of one time slot is much larger than the delays.There are two types of communication in the system. The first is the synchronization among the CRMs of a group. Since players in the same group should communicate and exchange data (e.g., realtime scores, blood volume, hits and kills), all the MECs that host the CRMs for the same group should thus keep synchronized. This is what we call the *in-group synchronization*. Such in-group synchronization could have various choices for implementation, such as peer-to-peer and client-server. Here, we assume the client-server pattern, which is adopted by the majority of current games on the market [7]. We thus assume there is one "central server"[1], and CRMs are synchronized with one another via sending traffic to and receiving traffic from this central server. The central server is logically centralized, and can be implemented as multiple distributed MECs or remote servers if needed. The second type of communication is that the players can interact with each other within or outside a group. Players can do instant messaging, voice chatting, or some gaming-specific interactions such as "borrowing" one's avatar across groups. This is what we call the *pairwise interaction*. Without loss of generality, and to maintain a single service module for each player, we assume such interactions are implemented as data exchange between the CRMs of the two interacting players.

We denote the communication delay between two MECs p and q as $d(p,q)$, the interaction rate between two players u and v at time t as $f_{u,v,t}$, the frame rate of player u at time t as $f_{u,t}$, and the synchronization rate of group j at time t as $f_{j,t}$. Since we are considering VR gaming services, which is regarded as the killer App in 5G era, the bandwidth from MEC to user end can be considered large. As for the network that connects the distributed MECs, which often adopts the huge bandwidth optical network, the bandwidth is also huge. Therefore, we assume the delay of two MECs does not change with the transmission (interaction/frame/synchronization) rates, which is reasonable under network conditions of sufficient bandwidth. The total communication delay at t can be formulated as

$$E_{2,t} = \sum\nolimits_{(u,v) \in L_t} f_{u,v,t} d(p_{u,t}, p_{v,t}) + \sum\nolimits_{u \in U_t} f_{u,t} d(p_{u,t}, p^{\dagger}_{u,t}) + \sum\nolimits_{i \in I} \sum\nolimits_{j \in J_t} f_{j,t} d(i,c) \delta_{i,j,t},$$

where $f_{u,v,t} d(p_{u,t}, p_{v,t})$ is the total pairwise communication delay between two interacting players u and v at time t; $f_{u,t} d(p_{u,t}, p^{\dagger}_{u,t})$ is the total pairwise communication delay between the MEC that has player u's CRM and the MEC that is closest to u at time t; $f_{j,t} d(i,c) \delta_{i,j,t}$ equals the in-group synchronization delay between MEC i and the central server

[1]We assume the central server already exists and is selected before the player groups are formed and CRMs are placed. The selection of the central server is an orthogonal problem out of the scope of this paper.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

c if at time t the former holds the CRM of group j, and equals zero otherwise.

D. Modeling Colocation Interference

When multiple CRMs are placed on the same MEC, the system would slow down due to resource contention, and the response time of each CRM would inflate due to CRM scheduling. We model such interference effects among colocated CRMs by the "dilation factor", as introduced in [14], generally defined as the makespan of the CRMs running concurrently on the MEC divided by its individual completion time without contention. The dilation factor is approximately linear to the number of colocated CRMs [14]. Let $a_{i,1}$ and $a_{i,2}$ be the parameters of the dilation factor of MEC i. We can formulate the colocation interference at time t as

$$\begin{aligned} E_{3,t} &= \sum\nolimits_{i\in I}\left(a_{i,1}\sum\nolimits_{j\in J_t}\delta_{i,j,t}+a_{i,2}\right) \\ &= \sum\nolimits_{i\in I}\sum\nolimits_{j\in J_t}a_{i,1}\delta_{i,j,t}+\sum\nolimits_{i\in I}a_{i,2}. \end{aligned}$$

E. Modeling Switching Cost

To accommodate the dynamics of player's moving, joining and leaving, the system incurs cost. For example, as a player moves from one location to another within the system at two consecutive time slots, her CRM may need to be migrated from the old MEC to a new MEC correspondingly, where the new MEC would need to be reconfigured to host that CRM. We adopt the term "switching cost" to represent such migration and configuration cost incurred by CRMs between two consecutive time slots. We calculate the total switching cost at time t as

$$E_{4,t} = \sum\nolimits_{u\in U_t} g(p_{u,t}, p_{u,t-1}),$$

where $g(p_{u,t}, p_{u,t-1})$ is defined as

$$g(p_{u,t}, p_{u,t-1}) \triangleq \begin{cases} g_u, & p_{u,t} \neq p_{u,t-1}; \\ 0, & p_{u,t} = p_{u,t-1}, \end{cases}$$

where g_u is the cost of migrating the CRM for player u.

F. Problem Formulation and Challenges

We consider a comprehensive cost model that incorporates the above four types of costs. The total cost at time t is

$$E_t(p_{u,t}) = E_{1,t} + E_{2,t} + E_{3,t} + E_{4,t},$$

where the variables are the placement decisions $p_{u,t}$ for each player u at each time slot t. We minimize the total cost over time:

$$\min_{p_{u,t}} \sum\nolimits_{t=1}^{T} E_t(p_{u,t}). \tag{1}$$

In equation (1), all the costs depend on the placement decisions, and are essentially related to computation/communication delays so we can add them up together. Nevertheless, our model is actually more general than only capturing delays, e.g., the switching cost can be easily adjusted to capturing the migration traffic. In this case, the costs are of different dimensions (e.g., units), and one can associate different weights to different costs, which is a common approach to handle multi-objective optimizations (note that the weights can be set according to the operator's preferences/policies, and can be tuned using standard approaches and are thus out of the scope of our paper). Having the weights does not affect the core design of our proposed algorithms.

The problem of (1) is difficult to solve due to the following challenges:

1) *$E_{1,t}, E_{2,t}$ and $E_{3,t}$ are discrete, nonconvex, and higher-degree functions.*
   For one, $E_{1,t}, E_{2,t}$ and $E_{3,t}$ all contain the binary indicator $\delta_{i,j,t}$, which can be rewritten as
   $$\delta_{i,j,t} = 1 - \prod\nolimits_{u\in U_{j,t}} \mathbf{1}\left(p_{u,t} \neq i\right),$$
   where $\mathbf{1}(\cdot)$ is a binary function that equals 1 if the specified condition holds and 0 otherwise. Thus, $\delta_{i,j,t}$ depends on the production of multiple binary functions which further depend on the joint placement of the CRMs of the users in a group, often considered as a "higher-order clique" [42] and hard to optimize. For the other, $E_{2,t}$ is not easy to optimize on its own, even without the term that involves $\delta_{i,j,t}$, as the function $d(\cdot,\cdot)$ also depends on the joint placement of the CRMs of a pair of users, and varies for different user pairs. The primary obstacle is where to place each CRM cannot be independently determined. This challenge escalates if we consider the exponentially-many possibilities to place all CRMs over all MECs in the system. We can prove that our problem is NP-hard (in Section IV-C).
2) *$E_{4,t}$ couples the decisions over time.*
   $E_{4,t}$ is the fundamental challenge for solving the problem online. Without $E_{4,t}$, the problem of minimizing $E_{1,t} + E_{2,t} + E_{3,t}$ over time would be equivalent to minimizing them at each individual time slot t; with $E_{4,t}$, the decisions $p_{u,t-1}$ made at one time slot $t-1$ will influence the total cost by influencing the cost at both time $t-1$ and t. Note that we will make the decisions sequentially as time goes. Due to such coupling, any decisions made at t will influence the decisions that are to be made at $t+1$.

IV. The Approach of Model Predictive Control

To construct an online algorithm, we leverage the framework of Model Predictive Control (MPC). Applying MPC requires to solve the problem of (1) over every prediction window. Thus, the aforementioned challenges still exist. In this section, we describe the general approach of MPC, introduce our virtual MEC idea for arbitrary user dynamics, and show the problem over each prediction window and its NP-hardness; in the next section, we will focus on solving the problem over the prediction window with performance-provable approximation algorithms.

A. Model Predictive Control with Virtual MEC

MPC is not about how to do prediction, but about how to exploit prediction to make online decisions. As time goes, at

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

each time slot, MPC uses the predicted information or inputs in a "prediction window" to make *better* decisions for the current time slot, compared to the decisions that could have been made using no prediction at all. This is true, particularly as we have the switching cost in our problem. As decisions at the current time slot will influence the switching cost between it and the next time slot, having the predicted information about the future enables us to avoid making bad decisions blindly for the current time slot. Specifically, MPC solves the problem over each prediction window based on the predicted inputs, and applies the decisions only for the first time slot, i.e., the current one, while dropping the decisions for the rest time slots of the prediction window. Given that substantial existing studies have demonstrated that user mobilities are largely predictable via various means [17], [28], [48][2],we consider MPC a reasonable framework for us to construct an online algorithm for our problem. Noting that MPC often assumes accurate predictions, we also assume so in this paper, while as we conduct evaluations as described later, we will evaluate the case of inaccurate predictions as well.

We introduce additional notations here. Let W be the length of the prediction window. Then, at t, the time slots in the prediction window are $t, t+1, ..., t+W$. Let $\tilde{U}_{t'}$ be the set of players that are predicted to appear in the system at t', where $t' \in \{t+1, ..., t+W\}$. We define $\tilde{U}_{t..t+W} \triangleq U_t \bigcup \tilde{U}_{t+1} \bigcup \tilde{U}_{t+2} ... \bigcup \tilde{U}_{t+W}$. $\tilde{U}_{t..t+W}$ is the set of players that will appear in the system in the prediction window. $\tilde{U}_{t..t+W} \backslash U_t$ is the set of players that are not present at t but will appear in the rest time slots of the prediction window; $\tilde{U}_{t..t+W} \backslash \tilde{U}_{t'}, t' = t+1, ..t+W$ is the set of players that are not present at t' but will appear in the other time slots of the prediction window.

To represent the player dynamics such as joining and leaving the system, we introduce a virtual MEC $\mathcal{F}$ to host the CRMs of all the players that are in $\tilde{U}_{t..t+W}$ but are not present at the time slot in question. That is, for time t, the virtual MEC hosts $\tilde{U}_{t..t+W} \backslash U_t$, while for time $t', t' = t+1, ..., t+W$, the virtual MEC hosts $\tilde{U}_{t..t+W} \backslash \tilde{U}_{t'}$. For the virtual MEC, let its processing speed be infinity, i.e., $h_{\mathcal{F}} = \infty$, the delay between it and any real MEC p be 0, i.e., $d(\mathcal{F}, p) = 0$, and its dilation factors be 0, i.e., $a_{\mathcal{F},1} = 0, a_{\mathcal{F},2} = 0$. Thus, the cost of hosting CRMs of absent players at the virtual MEC is 0. Setting these parameters like this enables to model player joining and leaving the sytem the same way as moving within the system, i.e., moving from the virtual MEC to a real MEC or vice versa, so that we can treat all player dynamics of joining, leaving, and moving in a unified manner in our models and algorithms.

B. Problem over a Prediction Window

At each t, MPC makes the placement decisions to minimize the cost over the current prediction window, given all the user dynamics in that window. We have this optimization problem:

$$\min_{p_{u,t'}} \sum_{t'=t}^{t+W} E_t \left(p_{u,t'}\right). \tag{2}$$

Note that the cost of hosting non-present players on the virtual MEC is 0 at any time $t' \in \{t, ..., t+W\}$. From now on, we write $E_W \left(p_{u,t'}\right) \triangleq \sum_{t'=t}^{t+W} E_t \left(p_{u,t'}\right)$ for simplicity.

C. Problem Hardness

Theorem 1 For each t, the minimization problem of (2) is NP-hard.

Proof. The problem in (2) can be easily transformed in polynomial time to the uncapacitated facility location (UFL) problem [40]. The UFL problem considers the placement of multiple facilities to minimize the opening and the transportation cost to a set of sites. The UFL problem is NP-hard and can be reduced to the set cover problem, which is one of the NP-complete problem [39]. To reduce (2) to UFL, one can keep the computation and the communication cost from CRM to player's nearest MEC and set all the other part of the objective in (2) to zero. Note that (2) contains (1) as a special case when the length of the prediction window is the entire time horizon, so the problem of (1) is also NP-hard. □

V. Cost Minimization via Graph Cuts

For the problem over each prediction window, we propose a polynomial-time approximation algorithm with a provably bounded performance guarantee. We design an iterative algorithm that consists of multiple rounds and performs an operation called "α expansion" [41] which updates the placement decisions of the CRMs in each round in batch. We exhibit the equivalence between α expansion and the graph-theoretic minimal s-t cut problem, elaborate the construction of such a corresponding graph, and prove the approximation performance guarantee.

A. α Expansion and Algorithm

An "α expansion" is a binary optimization that tries to move each CRM from its current MEC $p_{u,t}$ to the MEC α, thus trying to "expand" the number of CRMs placed on the MEC α if such expansion leads to cost reduction. We use $p^{\alpha}_{u,t}$ to denote the MEC for player u after α expansion, then $p^{\alpha}_{u,t}$ equals either α or $p_{u,t}$. We propose Algorithm 1 to iterate through all the MECs, and implement the optimal α expansion that minimizes $E_W \left(p_{u,t'}\right)$. To solve the α expansion in Line 5 of Algorithm 1, we first reformulate the problem using binary variables in Section V-B. In Section V-C, we use an example to illustrate how to construct a graph from the binary variables. After that, we construct the graph for our MPC-based CRM placement problem of equation (2). In order to better illustrate the whole structure of the solution, we show how to solve the α expansion via minimal cut in Algorithm 2 in Appendix.

[2]User joining and leaving should be predicted and serve as input to our work. Details of prediction method can be found in [17], [28], [48]. For example, [48] groups users into interpretable clusters based on their activities on the platform and ego-network structures; afterwards, it designs a novel deep learning pipeline based on Long Short-Term Memory (LSTM), a popular Recurrent Neural Network (RNN) technique, to accurately predict user churn by leveraging the correlations among users activities and the underlying user types.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

Algorithm 1 The α-Expansion Algorithm

Input:

I: the set of all the MECs;

$p_{u,t'}$: current placement at time t' before α expansion, $\forall u \in \tilde{U}_{t..t+W}, t' \in \{t, ..., t+W\}$.

Output:

$p_{u,t'}$: updated placement at time t' after α expansion, $\forall u \in \tilde{U}_{t..t+W}, t' \in \{t, ..., t+W\}$.

$flag \leftarrow 1$
while $flag == 1$ **do**
  $flag \leftarrow 0$
  for each $\alpha \in I$ **do**
    $p^{\alpha}_{u,t'} \leftarrow \alpha$ expansion of $p_{u,t'}$, $\forall u \in \tilde{U}_{t..t+W}, t' \in \{t, ..., t+W\}$
    $p^{\alpha*}_{u,t'} \leftarrow \arg\min_{p^{\alpha}_{u,t'}} E_W\left(p^{\alpha}_{u,t'}\right), \forall u \in \tilde{U}_{t..t+W}$, $t' \in \{t, ..., t+W\}$
    if $E_W\left(p^{\alpha*}_{u,t'}\right) < E_W\left(p_{u,t'}\right)$ **then**
      $flag \leftarrow 1, p_{u,t'} \leftarrow p^{\alpha*}_{u,t'}, \forall u \in \tilde{U}_{t..t+W}$, $t' \in \{t, ..., t+W\}$
    end if
  end for
end while

B. Problem Reformulation via α Expansion

We highlight that $E_W\left(p_{u,t'}\right)$ essentially contains three types of costs: the *unary cost* of $E_a\left(p_{u,t'}\right)$, the *pairwise cost* of $E_{b1}\left(p_{u,t'}, p_{v,t'}\right)$ and $E_{b2}\left(p_{u,t'}\right)$, and the *occupation cost* of $E_c\left(p_{u,t'}\right)$. The unary cost refers to the type of cost that relies on the placement the CRM of a single player. The pairwise cost refers to the type of cost that relies on the joint placement of the CRMs of a pair of players at the same time slot, i.e., $E_{b1}\left(p_{u,t'}, p_{v,t'}\right)$, or the joint placement of the CRM of a single player at two consecutive time slots, i.e., $E_{b2}\left(p_{u,t'}\right)$. The occupation cost refers to the type of cost that relies on the total number of the MECs that have been currently occupied to host the CRMs of all the players. We reformulate $E_W\left(p_{u,t'}\right)$ in terms of the three types of costs, where $A_1 \triangleq W \cdot \sum_{i=1}^{n} a_{i,2}$.

$$
\begin{aligned}
E_W\left(p_{u,t'}\right) = & \underbrace{\sum_{t'=t}^{t+W} \sum_{u \in \tilde{U}_{t..t+W}} f_{u,t} d(p_{u,t'}, p^{\dagger}_{u,t'})}_{E_a\left(p_{u,t'}\right)} \\
& + \underbrace{\sum_{t'=t}^{t+W} \sum_{(u,v) \in L_{t'}} f_{u,v,t} d(p_{u,t'}, p_{v,t'})}_{E_{b1}\left(p_{u,t'}, p_{v,t'}\right)} \\
& + \underbrace{\sum_{t'=t}^{t+W} \sum_{u \in \tilde{U}_{t..t+W}} g(p_{u,t'}, p_{u,t'-1})}_{E_{b2}\left(p_{u,t'}\right)} \\
& + \underbrace{\sum_{t'=t}^{t+W} \sum_{i \in I} \sum_{j \in J_{t'}} \left[\frac{R_{j,t'}}{h_i} + f_{j,t} d(i,c) + a_{i,1}\right] \delta_{i,j,t'} + A_1}_{E_c\left(p_{u,t'}\right)}
\end{aligned}
$$

We also note that α expansion can be represented by a set of binary variables $\mathbf{x} = \{x_{1,t}, x_{2,t}, ..., x_{M,t}, ..., x_{1,t+W}, x_{2,t+W}, ..., x_{M,t+W}\}$, where M is the total number of players in $\tilde{U}_{t..t+W}$ and $x_{u,t'}$ is defined as

$$
x_{u,t'} \triangleq \begin{cases} 1, & p^{\alpha}_{u,t'} = \alpha\,; \\ 0, & p^{\alpha}_{u,t'} \neq \alpha \left(\text{i.e., } p^{\alpha}_{u,t'} = p_{u,t'} \wedge p^{\alpha}_{u,t'} \neq \alpha\right). \end{cases}
$$

Therefore, we can further reformulate $E_W\left(p_{u,t'}\right)$ using binary variables. First, let us represent $E_a\left(p_{u,t'}\right)$, $E_{b1}\left(p_{u,t'}, p_{v,t'}\right)$, and $E_{b2}\left(p_{u,t'}\right)$ using $x_{u,t'}$ and $\bar{x}_{u,t'}$, where we define $\bar{x}_{u,t'} \triangleq 1 - x_{u,t'}$. We have

$$
\begin{aligned}
E_a\left(p_{u,t'}\right) &= \sum_{t'=t}^{t+W} \sum_{u \in \tilde{U}_{t..t+W}} f_{u,t} d(p_{u,t'}, p^{\dagger}_{u,t'}) \\
&= \sum_{t'=t}^{t+W} \sum_{u \in \tilde{U}_{t..t+W}} f_{u,t} \left[d(\alpha, p^{\dagger}_{u,t'}) x_{u,t'} + d(p_{u,t'}, p^{\dagger}_{u,t'}) \bar{x}_{u,t'}\right], \\
E_{b1}\left(p_{u,t'}, p_{v,t'}\right) &= \sum_{t'=t}^{t+W} \sum_{(u,v) \in L_{t'}} f_{u,v,t} d(p_{u,t'}, p_{v,t'}) \\
&= \sum_{t'=t}^{t+W} \sum_{(u,v) \in L_{t'}} f_{u,v,t} \big[d(p_{u,t'}, p_{v,t'}) \bar{x}_{u,t'} \bar{x}_{v,t'} + \\
&\quad d(\alpha, p_{v,t'}) x_{u,t'} \bar{x}_{v,t'} + d(p_{u,t'}, \alpha) \bar{x}_{u,t'} x_{v,t'}\big], \\
E_{b2}\left(p_{u,t'}\right) &= \sum_{t'=t}^{t+W} \sum_{u \in \tilde{U}_{t..t+W}} g(p_{u,t'}, p_{u,t'-1}) \\
&= \sum_{t'=t}^{t+W} \sum_{u \in \tilde{U}_{t..t+W}} \big[g(p_{u,t'}, p_{u,t'-1}) \bar{x}_{u,t'} \bar{x}_{u,t'-1} + \\
&\quad g(\alpha, p_{u,t'-1}) x_{u,t'} \bar{x}_{u,t'-1} + g(p_{u,t'}, \alpha) \bar{x}_{u,t'} x_{u,t'-1}\big].
\end{aligned}
$$

Next, let us represent $E_c\left(p_{u,t'}\right)$ using $x_{u,t'}$ and $\bar{x}_{u,t'}$, with $\theta_{i,j,t'} \triangleq \frac{R_{j,t'}}{h_i} + f_{j,t} d(i,c) + a_{i,1}$:

$$
E_c\left(p_{u,t'}\right) = \sum_{t'=t}^{t+W} \sum_{i \in I} \sum_{j \in J_{t'}} \theta_{i,j,t'} \delta_{i,j,t'},
$$

where, based on the definition of $\delta_{i,j,t'}$, we can rewrite

$$
\delta_{i,j,t'} = \begin{cases} 1 - \prod_{u \in U_{j,t'}, p_{u,t'} = i} x_{u,t'}, & i \neq \alpha; \\ 1 - \prod_{u \in U_{j,t'}, p_{u,t'} = i} \bar{x}_{u,t'}, & i = \alpha. \end{cases} \tag{3}
$$

In (3), we consider whether there exists at least one player in group $U_{j,t'}$, who moves her CRM from MEC i to MEC α. When $i \neq \alpha$, if all players in group $U_{j,t'}$ are not using i, then $\prod_{u \in U_{j,t'}, p_{u,t'} = i} x_{u,t'} = 1$ and $\delta_{i,j,t'} = 0$; otherwise, suppose the player u in group $U_{j,t'}$ leaves her CRM on i, then $\prod_{u \in U_{j,t'}, p_{u,t'} = i} x_{u,t'} = 0$ and $\delta_{i,j,t'} = 1$. When $i = \alpha$, if $\exists u \in \{u | u \in U_{j,t'}, p_{u,t'} = i\}$, then $\delta_{i,j,t'} = 1$; otherwise, $\delta_{i,j,t'} = 1$.

Last, we further introduce some auxiliary binary variables to (3) in order to transform the *product* of binary variables to the *sum* of binary variables for the purpose of graph construction, as described later. We introduce $y_{i,j,t'}, 1 \leq i \leq n, 1 \leq j \leq m_{t'}, t' \in \{t, ..., t+W\}$ and use $|C_{i,j,t'}|$ to denote the number

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

of players in group $U_{j,t'}$ that have their CRM on MEC i before the α expansion. Then, for (3), we have

$$-\prod_{u\in U_{j,t'},p_{u,t'}=i} x_{u,t'} = \min_{y_{i,j,t}\in\{0,1\}} \left[\left(|C_{i,j,t'}|-1\right) y_{i,j,t'} - \sum_{u\in U_{j,t'},p_{u,t'}=i} x_{u,t'} y_{i,j,t'} \right]. \tag{4}$$

Here is a brief proof for (4). If there is at least one $x_{u,t'}=0$, then the left-hand side of (4) equals 0; on the right-hand side, we have $\sum_{u\in U_{j,t'},p_{u,t'}=i} x_{u,t'} \le |C_{i,j,t'}|-1$ and further have the entire right-hand side equal 0 as $y_{i,j,t'}=0$. On the other hand, if all $x_{u,t'}=1$ and the left-hand side of (4) equals -1 correspondingly, then we have $\sum_{u\in U_{j,t'},p_{u,t'}=i} x_{u,t'} = |C_{i,j,t'}|$ and further have the entire right-hand side equal -1 as $y_{i,j,t'}=1$.

C. Binary Optimization and Graph Cut: A Mini Example

We observe that, after reformulating $E_W(p_{u,t'})$ in the previous section, we are minimizing a function of binary variables, where $E_a(p_{u,t'})$ corresponds to a weighted sum of binary variables; $E_{b1}(p_{u,t'},p_{v,t'})$ and $E_{b2}(p_{u,t'})$ correspond to a weighted sum of the products of pairs of binary variables; and $E_c(p_{u,t'})$ corresponds to a weighted sum of the products of multiple binary variables. If we consider the *simplest* cases, then $E_a(p_{u,t'})+E_{b1}(p_{u,t'},p_{v,t'})+E_{b2}(p_{u,t'})$ may correspond to $\beta x_u x_v+\rho x_u+\phi x_v$, and $E_c(p_{u,t'})$ may correspond to $\gamma(1-x_u x_v x_k)$, where x_u, x_v, x_k are binary variables, $\beta\le 0$ and $\gamma\ge 0$. We will see in Section V-D that "$\beta\le 0, \gamma\ge 0$" are indeed satisfied in our CRM placement problem. We note that for $\gamma(1-x_u x_v x_k)$, the product can contain as many binary variables as desired, but we only consider three as an example.

We demonstrate how we minimize $\beta x_u x_v+\rho x_u+\phi x_v$ and $\gamma(1-x_u x_v x_k)$, respectively but uniformly, via "graph cut". The basic idea is that we construct a graph and seek the minimal cut of the graph, which is a polynomial-time solvable problem, and satisfies two requirements: (1) the sum of the weights of the edges being cut equals the corresponding value of the objective function to be optimized, and (2) the nodes on one side of the cut correspond to the binary variables that take the value of 0 and those on the other side of the cut correspond to the binary variables that take the value of 1. Below, we show how to construct such a graph for our example.

First, we consider the graph construction for $\beta x_u x_v+\rho x_u+\phi x_v$. For each variable, we have a corresponding node; then, we additionally have the *source* node and the *terminal* node. To construct the edge, we reformulate the expression

$$\begin{aligned}&\beta x_u x_v+\rho x_u+\phi x_v\\ =&\tfrac{\beta}{2}x_u x_v+\tfrac{\beta}{2}x_u x_v+\rho x_u+\phi x_v\\ =&-\tfrac{\beta}{2}x_u(1-x_v)-\tfrac{\beta}{2}(1-x_u)x_v+\left(\tfrac{\beta}{2}+\rho\right)x_u+\left(\tfrac{\beta}{2}+\phi\right)x_v.\end{aligned}$$

For the first two terms, as $\beta\le 0$, an edge between u and v with weight $-\frac{\beta}{2}$ would satisfy the above mentioned two requirements. For the last two terms, there are four cases:

$$\left(\tfrac{\beta}{2}+\rho\right)x_u=\begin{cases}\left(\tfrac{\beta}{2}+\rho\right)x_u, & \text{if } \tfrac{\beta}{2}+\rho\ge 0;\\ -\left(\tfrac{\beta}{2}+\rho\right)(1-x_u)+\left(\tfrac{\beta}{2}+\rho\right), & \text{if } \tfrac{\beta}{2}+\rho<0.\end{cases}$$

$$\left(\tfrac{\beta}{2}+\phi\right)x_v=\begin{cases}\left(\tfrac{\beta}{2}+\phi\right)x_v, & \text{if } \tfrac{\beta}{2}+\phi\ge 0;\\ -\left(\tfrac{\beta}{2}+\phi\right)(1-x_v)+\left(\tfrac{\beta}{2}+\phi\right), & \text{if } \tfrac{\beta}{2}+\phi<0.\end{cases}$$

Case 1: $\frac{\beta}{2}+\rho\ge 0$ and $\frac{\beta}{2}+\phi\ge 0$, an edge between u and s with the weight $\frac{\beta}{2}+\rho$, and an edge between v and s with the weight $\frac{\beta}{2}+\phi$ would satisfy the two requirements.

Case 2: $\frac{\beta}{2}+\rho\ge 0$ and $\frac{\beta}{2}+\phi<0$, an edge between u and s with the weight $\frac{\beta}{2}+\rho$, and an edge between v and t with the weight $-\frac{\beta}{2}-\phi$ would satisfy the two requirements.

Case 3: $\frac{\beta}{2}+\rho<0$ and $\frac{\beta}{2}+\phi\ge 0$, an edge between u and t with the weight $-\frac{\beta}{2}-\rho$, and an edge between v and s with the weight $\frac{\beta}{2}+\phi$ would satisfy the two requirements.

Case 4: $\frac{\beta}{2}+\rho<0$ and $\frac{\beta}{2}+\phi<0$, an edge between u and t with the weight $-\frac{\beta}{2}-\rho$, and an edge between v and t with the weight $-\frac{\beta}{2}-\phi$ would satisfy the two requirements.

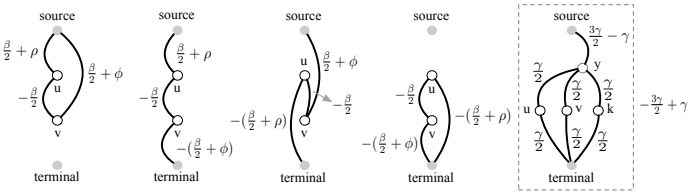


Fig. 2: Graph construction. The first four figures correspond to $\beta x_u x_v+\rho x_u+\phi x_v$; the last figure corresponds to $\gamma(1-x_u x_v x_k)$. For the latter, there is also a constant $-\frac{\gamma}{2}$, i.e., the cut result needs to be added to this constant to obtain the corresponding objective function value.

The first four figures in Fig. 2 illustrate the above four cases, respectively.

Next, we consider the graph construction for $\gamma(1-x_u x_v x_k)$. According to what we have described previously, we introduce an auxiliary node y as in (4):

$$-x_u x_v x_k = \min_{y\in\{0,1\}} \left[2y-(x_u+x_v+x_k)\,y\right].$$

There are only two possible results for $-x_u x_v x_k$: when $x_u=0$, or $x_v=0$, or $x_k=0$, then $y=0$ and $\gamma(1-x_u x_v x_k)=\gamma$; or, when $x_u=1, x_v=1, x_k=1$, then $y=1$ and $\gamma(1-x_u x_v x_k)=0$. We consider the two cases, respectively.

When $x_u=0$, or $x_v=0$, or $x_k=0$, then $y=0, \gamma(1-x_u x_v x_k)=\gamma$. It means that if any of x_u, x_v, x_k is on the same side of the cut with *source*, y is also on that side. This result indicates that the connection between x_u, x_v, x_k and *source* is *through* y. Therefore, the cut is between x_u or x_v or x_k, and y; or, between x_u or x_v or x_k, and *terminal*. As $\gamma(1-x_u x_v x_k)$ equals the same value for all the possible cuts, the weights between x_u, x_v, x_k and y and the weights between x_u, x_v, x_k and *terminal* should be identical. Let's denote this identical weight as η^1, then

$$3\eta^1+\eta^0=\gamma,$$

where η^0 is a constant to make the cut of the graph equals the cost (to satisfy requirement (1)).

When $x_u=1, x_v=1, x_k=1$, then $y=1, \gamma(1-x_u x_v x_k)=0$. In this case, the cut is between *source* and y. Let η^2 be the weight between *source* and y, then

$$\eta^2+\eta^0=0.$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

Therefore, any $\eta^0 < 0, \eta^1 > 0, \eta^2 > 0$ that satisfy the above equations could be the weights of our graph. In this paper, we take the values as in the last figure of Fig. 2 to satisfy the above two equations, i.e., $\eta^0 = -\frac{\gamma}{2}$ and $\eta^1 = \eta^2 = \frac{\gamma}{2}$.

D. Solving α Expansion via Graph Cut

We transform α expansion into graph cut. Here, the nodes on one side of the cut correspond to $x_{u,t'} = 0$, meaning that these CRMs stay at their current MECs, and the nodes on the other side of the cut correspond to $x_{u,t'} = 1$ meaning that these CRMs move to the MEC α. We show how to construct such a weighted, undirected graph so that the sum of the weights of the edges being cut equals the total cost incurred by the the corresponding placement decisions. Thus, any α expansion $p^{\alpha*}_{u,t'}$ in each iteration within Algorithm 1 can be solved by any existing polynomial-time minimal cut algorithm. Below, we construct the nodes and the weighted edges, respectively. Fig. 3 is an example.

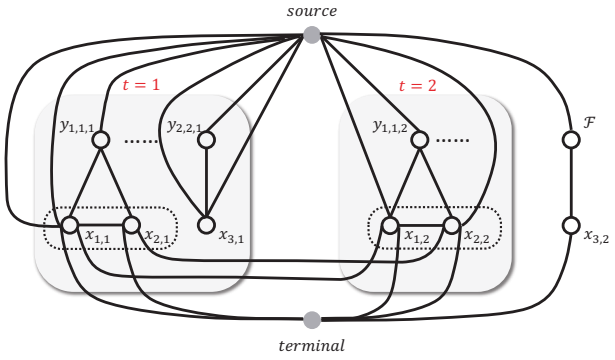


Fig. 3: Graph construction for our problem in a prediction window of two time slots. $x_{1,1}, x_{2,1}, x_{3,1}, x_{1,2}, x_{2,2}, x_{3,2}$ represent the users, where users 1 and 2 are in the same group at both $t = 1$ and $t = 2$. User 3 is in a different group from users 1 and 2 at $t = 1$, and leaves the system at $t = 2$. Here, we consider two MECs. $y_{1,1,1}, y_{2,2,1}, y_{,1,2}$ are the auxiliary nodes for the MECs. $\mathcal{F}$ represents the virtual MEC. Note that we do not show all the auxiliary nodes (i.e., $y_{2,1,1}, y_{1,2,1}, y_{2,1,2}$) and the edge weights for the ease of presentation.

1) ***Construction of Nodes****:* We construct a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ that has $\left|\tilde{U}_{t..t+W}\right|$ nodes corresponding to the players and $\tilde{U}_{t..t+W}$, $n\sum_{t'=t}^{t+W} m_{t'}$ nodes corresponding to the auxiliary binary variables $y_{i,j,t'}$, one node $\mathcal{F}$ corresponding to the virtual MEC, one source node $source$, and one destination node $terminal$. That is, $\mathcal{V} = \left\{x_{u,t'} \middle| u \in \tilde{U}_{t..t+W}\right\} \bigcup \left\{y_{i,j,t'} \middle| 1 \le i \le n, 1 \le j \le m_{t'}, t' \in \{t, ..., t+W\}\right\} \bigcup \{\mathcal{F}\} \bigcup \{source, terminal\}$.

2) ***Construction of Edges and Weights****:* We associate the node $source$ with the meaning of each CRM's current MEC (except for the CRMs that the current MEC is already α), and also associate the node $terminal$ with the meaning of being on the MEC α. In the following, we work with $E_{b1}(p_{u,t'}, p_{v,t'})$, $E_{b2}(p_{u,t'})$, $E_c(p_{u,t'})$, and $E_a(p_{u,t'})$, respectively, to elaborate the construction of the edges and their weights.

Firstly, we encode the quadratic terms into the graph. We have

$$\begin{aligned}
&E_{b1}(p_{u,t'}, p_{v,t'}) \\
&= \sum_{t'=t}^{t+W} \sum_{(u,v)\in L_{t'}} f_{u,v,t}\left(B_{1,u,t'} x_{u,t'} + B_{2,v,t'} x_{v,t'}\right) + \\
&\sum_{t'=t}^{t+W} \sum_{(u,v)\in L_{t'}} f_{u,v,t} \frac{d(\alpha, p_{v,t'}) + d(p_{u,t'}, \alpha) - d(p_{u,t'}, p_{v,t'})}{2} \times \\
&\left(x_{u,t'}\bar{x}_{v,t'} + \bar{x}_{u,t'} x_{v,t'}\right) + A_2,
\end{aligned}$$

where $B_{1,u,t'} = \frac{d(\alpha,p_{v,t'}) - d(p_{u,t'},\alpha) + d(p_{u,t'},p_{v,t'})}{2}$, $B_{2,v,t'} = \frac{d(p_{u,t'},\alpha) - d(\alpha,p_{v,t'}) + d(p_{u,t'},p_{v,t'})}{2}$, and analogously, we have

$$\begin{aligned}
&E_{b2}(p_{u,t'}) \\
&= \sum_{t'=t}^{t+W} \sum_{u\in \tilde{U}_{t..t+W}} \left(B_{3,u,t'} x_{u,t'} + B_{4,u,t'-1} x_{u,t'-1}\right) + \\
&\sum_{t'=t}^{t+W} \sum_{u\in \tilde{U}_{t..t+W}} \frac{g(\alpha, p_{u,t'-1}) + g(p_{u,t'}, \alpha) - g(p_{u,t'}, p_{u,t'-1})}{2} \times \\
&\left(x_{u,t'}\bar{x}_{u,t'-1} + \bar{x}_{u,t'} x_{u,t'-1}\right) + A_3,
\end{aligned}$$

where $B_{3,u,t'} = \frac{g(\alpha,p_{u,t'-1}) - g(p_{u,t'},\alpha) + g(p_{u,t'},p_{u,t'-1})}{2}$, $B_{4,u,t'-1} = \frac{g(p_{u,t'},\alpha) - g(\alpha,p_{u,t'-1}) + g(p_{u,t'},p_{u,t'-1})}{2}$. A_2 and A_3 are constants that can be calculated very easily and will not be encoded into the graph, which does not affect the placement decision of the α expansion algorithm.

The weight of the edge between $x_{u,t'}$ and $x_{v,t'}$ is thus $\frac{1}{2} f_{u,v,t}\left[d(\alpha, p_{v,t'}) + d(p_{u,t'}, \alpha) - d(p_{u,t'}, p_{v,t'})\right]$. Assuming triangle inequality, i.e., $d(\alpha, p_{v,t'}) + d(p_{u,t'}, \alpha) - d(p_{u,t'}, p_{v,t'}) \ge 0$, the weight is nonnegative. Similarly, for $E_{b2}(p_{u,t'})$, which is the pairwise cost between nodes $x_{u,t'}$ and $x_{u,t'-1}$, we add an edge between $x_{u,t'}$ and $x_{u,t'-1}$ with the weight of $\frac{1}{2}\left[g(\alpha, p_{u,t'-1}) + g(p_{u,t'}, \alpha) - g(p_{u,t'}, p_{u,t'-1})\right]$.

Secondly, we encode the higher-degree term of $E_c(p_{u,t'})$. Note that, for each $\theta_{i,j,t'}\delta_{i,j,t'}$, we only need to consider encoding $\min_{y_{i,j,t'}\in\{0,1\}} \theta_{i,j,t'}\left[\left(|C_{i,j,t'}| - 1\right) y_{i,j,t'} - \sum_{u\in U_{j,t'}, p_{u,t'}=i} x_{u,t'} y_{i,j,t'}\right]$ into the graph, since the rest part is a constant and won't affect the placement decision. To minimize this expression, there are two cases as mentioned earlier. The first case is that $\forall x_{u,t'} = 1, u \in U_{j,t'}, p_{u,t'} = i$, then the minimum is obtained when $y_{i,j,t'} = 1$. In this case, $\theta_{i,j,t'}\delta_{i,j,t'} = 0$. The second case is that $\exists x_{u,t'} = 0, u \in U_{j,t'}, p_{u,t'} = i$, then the minimum is obtained when $y_{i,j,t'} = 0$. In this case $\theta_{i,j,t'}\delta_{i,j,t'} = \theta_{i,j,t'}$. It means that when $\exists x_{u,t'}$ that remains on its current MEC after the α expansion, i.e., when $\exists x_{u,t'} = 0, u \in U_{j,t'}, p_{u,t'} = i$ is on the same side of the cut with $source$, $y_{i,j,t'}$ is also on that side. In other words, *if we only consider the occupation cost*, there is no direct edge between $x_{u,t'}$ and $source$, i.e., the connection between $x_{u,t'}$ and $source$ is through auxiliary node $y_{i,j,t'}$. Therefore, we add one edge between $x_{u,t'}$ and $y_{i,j,t'}$, $\forall x_{u,t'} = 0, u \in U_{j,t'}, p_{u,t'} = i$, and one edge between $y_{i,j,t'}$ and $source$. Then, we add one edge between $x_{u,t'}$ and $terminal$, since for $\forall x_{u,t'}, u \in U_{j,t'}, p_{u,t'} = i$, it can

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

be assigned 1 without affecting the value of $y_{i,j,t'}$ (unless $\forall x_{u,t'}, u \in U_{j,t'}, p_{u,t'} = i,\ x_{u,t'} = 1$).

Now, we consider how to configure the weights of the subgraph that involves the occupation cost for group $U_{j,t'}$ on MEC i. The requirement is that when $\forall x_{u,t'} = 1, u \in U_{j,t'}, p_{u,t'} = i$, then $y_{i,j,t'}$ should be 1 and the occupation cost is 0. Otherwise $\exists x_{u,t'} = 0, u \in U_{j,t'}, p_{u,t'} = i$, then $y_{i,j,t'} = 0$ and the occupation cost is $\theta_{i,j,t}$. Since occupation cost is the same when $\exists x_{u,t'} = 0$, therefore all the weights of the edges between $x_{u,t'}$ and $y_{i,j,t'}$ as well as between $x_{u,t'}$ and $terminal$ are identical. Let's denote this identical weight between $x_{u,t'}$ and $y_{i,j,t'}$ (i.e., between $x_{u,t'}$ and $terminal$) as $\eta^1_{i,j,t'}$, and the weight between $source$ and $y_{i,j,t'}$ as $\eta^2_{i,j,t'}$. Then, we have

$$|U_{j,t'}|\eta^1_{i,j,t'} = \theta_{i,j,t'} + \eta^2_{i,j,t'}.$$

The total cost is the sum of weights on the cut minus $\eta^2_{i,j,t'}$. Actually, there are many settings of weights that could satisfy the above equation. In this paper, we select the configuration of weights as suggested in [44], where $\eta^1_{i,j,t'} = \frac{1}{2}\theta_{i,j,t'}$, and $\eta^2_{i,j,t'} = \frac{1}{2}\theta_{i,j,t'}|U_{j,t'}| - \theta_{i,j,t'}$. The subgraph needs to add a constant value $-\frac{1}{2}\theta_{i,j,t'}|U_{j,t'}| + \theta_{i,j,t'}$ to adjust to the occupation cost. As a result, the second part of the weight of the edge between $x_{u,t'}$ and $terminal$ is the sum of $\eta^1_{i,j,t'}$ over all MECs i, which is $\sum_{i=1}^{n} \frac{1}{2}\theta_{i,j,t'}$. The above subgraph of the occupation cost also explains the weights of the edges between $source$ and $y_{i,j,t'}$, and between $x_{u,t'}$ and $y_{i,j,t'}$.

Thirdly, we encode the linear terms. At this circumstance, we not only need to consider the linear terms in unary cost $E_a(p_{u,t'})$, but also take the linear terms produced by the former transformation in the encoding process of the pairwise cost into account,

$$\begin{aligned} E_a(p_{u,t'}) = &\sum_{t'=t}^{t+W} \sum_{(u,v)\in L_{t'}} f_{u,v,t}\big[B_{1,u,t'}x_{u,t'} + B_{2,v,t'}x_{v,t'}\big] + \\ &\sum_{t'=t}^{t+W} \sum_{u\in\tilde{U}_{t..t+W}} \big[\Big(f_{u,t}d(\alpha, p^{\dagger}_{u,t'}) + B_{3,u,t'}\Big)x_{u,t'} + \\ &f_{u,t}d(p_{u,t'}, p^{\dagger}_{u,t'})\bar{x}_{u,t'} + B_{4,u,t'-1}x_{u,t'-1}\big]. \end{aligned}$$

When accumulated over time, $B_{4,u,t'}$ would be added to the coefficient of $x_{u,t'}$. Since we assume triangle inequality, then $B_{1,u,t'} \geq 0, B_{2,v,t'} \geq 0, B_{3,u,t'} \geq 0, B_{4,u,t'} \geq 0$. As a result, $\forall x_{u,t'}, u \in \tilde{U}_{t..t+W}, t' \in \{t, ..., t+W\}$, we add an edge between $x_{u,t'}$ and $source$ with the weight of $f_{u,t}d(\alpha, p^{\dagger}_{u,t'}) + B_{3,u,t'} + B_{4,u,t'}$, and add the weight of edge between $x_{u,t'}$ and $terminal$ by $f_{u,t}d(p_{u,t'}, p^{\dagger}_{u,t'})$. Therefore, the entire weight of the edge between $x_{u,t'}$ and $terminal$ is then $f_{u,t}d(p_{u,t'}, p^{\dagger}_{u,t'}) + \sum_{i=1}^{n} \frac{1}{2}\theta_{i,j,t'}$ considering all costs. Furthermore, $\forall (u,v) \in L_{t'}, t' \in \{t, ..., t+W\}$, the weight between $x_{u,t'}$ and $source$ needs to add $f_{u,v,t}B_{1,u,t'}$, and that between $x_{v,t'}$ and $source$ needs to add $f_{u,v,t}B_{2,v,t'}$.

E. Time Complexity Analysis

Since Algorithm 2 is a module invoked by Algorithm 1 for N times, where N is the number of iterations times the number of MECs, we firstly investigate the time complexity of Algorithm 2 which is provided in the Appendix.

Algorithm 2 contains two parts. It constructs the graph first and then invokes the s-t min-cut algorithm using the constructed graph as input. As can be seen from Algorithm 2, the time complexity for the graph construction is $O(WM \cdot \max\{Nm_{t'}, M\})$, where W is the length of the prediction window, M is the total number of players in $\tilde{U}_{t..t+W}$, and $m_{t'}$ is the number of groups at time slot t'. Let $|\mathcal{V}|$ be the number of vertices and $|\mathcal{E}|$ be the number of edges in the constructed graph. The time complexity of the s-t min-cut is $O(|\mathcal{V}|^2|\mathcal{E}|)$ according to [45]. In our specific case, $|\mathcal{V}| = W + WNm_{t'} + 3$ is bounded by $O(WNm_{t'})$; $|\mathcal{E}| = 3WM + WMNm_{t'} + WM^2 + WNm_{t'}$ is bounded by $O(WM \cdot \max\{Nm_{t'}, M\})$. As a result, the complexity of Algorithm 2 is dominated by the s-t min-cut. Therefore, the complexity of Algorithm 2 is $O(|\mathcal{V}|^2|\mathcal{E}|) = O(W^3N^2{m_{t'}}^2M \cdot \max\{Nm_{t'}, M\})$.

Besides Line 5, the other part in Algorithm 1 is $O(1)$, and therefore, the overall time complexity of our algorithm is $O(W^3N^3\widehat{m}^2M \cdot \max\{N\widehat{m}, M\})$, where $\widehat{m}$ is the maximum number of groups.

F. Performance Analysis

Theorem 2 If $p^*_{u,t'}$ is the optimal solution to minimize $E_W(p_{u,t'})$ in the prediction window, $E_c(I)$ is a constant representing the occupation cost when all MECs in I are occupied, and $\hat{p}_{u,t'}$ is the solution found by Algorithm 1, then we have $E_W(\hat{p}_{u,t'}) \leq 2\lambda E_W\left(p^*_{u,t'}\right) + E_c(I)$, where $\lambda \triangleq \max\{\lambda_1, \lambda_2\}$, with

$$\lambda_1 = \max_{(u,v)\in L_{t'}} \left\{ \frac{\max\limits_{\alpha_1,\beta_1:p_{u,t'}=\alpha_1,p_{v,t'}=\beta_1} E_{b1}(\alpha_1,\beta_1)}{\min\limits_{\gamma_1,\epsilon_1:p_{u,t'}=\gamma_1,p_{v,t'}=\epsilon_1} E_{b1}(\gamma_1,\epsilon_1)} \right\},$$

$$\lambda_2 = \max_{u\in\tilde{U}_{t..t+W}} \left\{ \frac{\max\limits_{\alpha_2,\beta_2:p_{u,t'}=\alpha_2,p_{u,t'-1}=\beta_2} E_{b2}(\alpha_2,\beta_2)}{\min\limits_{\gamma_2,\epsilon_2:p_{u,t'}=\gamma_2,p_{u,t'-1}=\epsilon_2} E_{b2}(\gamma_2,\epsilon_2)} \right\}.$$

Proof. Let's first fix an $\alpha \in P$ and define the set P_α as

$$P_\alpha \triangleq \left\{ u \in \tilde{U}_{t..t+W} : p^*_{u,t'} = \alpha \right\}.$$

Let $p^\alpha_{u,t'}$ be an α expansion to the solution $\hat{p}_{u,t'}$ in the following way:

$$p^\alpha_{u,t'} = \begin{cases} \alpha, & \langle u, t' \rangle \in P_\alpha; \\ \hat{p}_{u,t'}, & \text{otherwise.} \end{cases}$$

Since $\hat{p}_{u,t'}$ is a feasible solution, we have $E_W(\hat{p}_{u,t'}) \leq E_W\left(p^\alpha_{u,t'}\right)$.

We divide $E_a(p_{u,t'})$, $E_{b1}(p_{u,t'}, p_{v,t'})$, $E_{b2}(p_{u,t'}, p_{u,t'-1})$ into three areas w.r.t. P_α, namely, internal area, external area

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

and boundary area:

$$\begin{aligned}
&\mathcal{I}_\alpha = P_\alpha,\\
&\mathcal{I}_\alpha^L = \{(u,v) \in L_{t'}, \wedge\langle u,t'\rangle, \langle v,t'\rangle \in P_\alpha\},\\
&\mathcal{I}_\alpha^M = \{\langle u,t'\rangle, \langle u,t'-1\rangle \in P_\alpha\},\\
&\mathcal{O}_\alpha = P \setminus P_\alpha,\\
&\mathcal{O}_\alpha^L = \{(u,v) \in L_{t'}, \wedge\langle u,t'\rangle, \langle v,t'\rangle \notin P_\alpha\},\\
&\mathcal{O}_\alpha^M = \{\langle u,t'\rangle, \langle u,t'-1\rangle \notin P_\alpha\},\\
&\mathcal{B}_\alpha = \varnothing,\\
&\mathcal{B}_\alpha^L = \{(u,v) \in L_{t'}, \wedge\langle u,t'\rangle \in P_\alpha, \langle v,t'\rangle \notin P_\alpha\},\\
&\mathcal{B}_\alpha^M = \{\langle u,t'\rangle \in P_\alpha, \langle u,t'-1\rangle \notin P_\alpha\}.
\end{aligned}$$

Let $E(\cdot)|_{\mathcal{A}} \triangleq E_a(u)|_{u\in\mathcal{A}} + E_{b1}(u,v)|_{(u,v)\in\mathcal{A}^L} + E_{b2}(u)|_{u\in\mathcal{A}^M}$, then the following equations hold:

$$\begin{aligned}
E\left(p_{u,t'}^\alpha\right)|_{\mathcal{I}_\alpha} &= E\left(p_{u,t'}^*\right)|_{\mathcal{I}_\alpha},\\
E\left(p_{u,t'}^\alpha\right)|_{\mathcal{O}_\alpha} &= E\left(\hat{p}_{u,t'}\right)|_{\mathcal{O}_\alpha},\\
E\left(p_{u,t'}^\alpha\right)|_{\mathcal{B}_\alpha} &\le \lambda E\left(p_{u,t'}^*\right)|_{\mathcal{B}_\alpha}.
\end{aligned}$$

Hence,

$$\begin{aligned}
&\because E_W\left(\hat{p}_{u,t'}\right) \le E_W\left(p_{u,t'}^\alpha\right)\\
&\therefore E\left(\hat{p}_{u,t'}\right)|_{\mathcal{I}_\alpha} + E\left(\hat{p}_{u,t'}\right)|_{\mathcal{O}_\alpha} + E\left(\hat{p}_{u,t'}\right)|_{\mathcal{B}_\alpha} + E_c\left(\hat{p}_{u,t'}\right)\\
&\le E\left(p_{u,t'}^\alpha\right)|_{\mathcal{I}_\alpha} + E\left(p_{u,t'}^\alpha\right)|_{\mathcal{O}_\alpha} + E\left(p_{u,t'}^\alpha\right)|_{\mathcal{B}_\alpha} + E_c\left(p_{u,t'}^\alpha\right)\\
&\le E\left(p_{u,t'}^*\right)|_{\mathcal{I}_\alpha} + E\left(\hat{p}_{u,t'}\right)|_{\mathcal{O}_\alpha} + \lambda E\left(p_{u,t'}^*\right)|_{\mathcal{B}_\alpha} + E_c\left(p_{u,t'}^\alpha\right)\\
&\therefore E\left(\hat{p}_{u,t'}\right)|_{\mathcal{I}_\alpha} + E\left(\hat{p}_{u,t'}\right)|_{\mathcal{B}_\alpha} + E_c\left(\hat{p}_{u,t'}\right)\\
&\le E\left(p_{u,t'}^*\right)|_{\mathcal{I}_\alpha} + \lambda E\left(p_{u,t'}^*\right)|_{\mathcal{B}_\alpha} + E_c\left(p_{u,t'}^\alpha\right)\\
&\therefore \sum_{\alpha\in P}\left[E\left(\hat{p}_{u,t'}\right)|_{\mathcal{I}_\alpha} + E\left(\hat{p}_{u,t'}\right)|_{\mathcal{B}_\alpha}\right] + E_c\left(\hat{p}_{u,t'}\right)\\
&\le \sum_{\alpha\in P}\left[E\left(p_{u,t'}^*\right)|_{\mathcal{I}_\alpha} + \lambda E\left(p_{u,t'}^*\right)|_{\mathcal{B}_\alpha}\right] + E_c\left(p_{u,t'}^\alpha\right).
\end{aligned}$$

When accumulated over all MECs, $\sum_{\alpha\in P} E\left(\hat{p}_{u,t'}\right)|_{\mathcal{I}_\alpha} = E_a\left(\hat{p}_{u,t'}\right) + E_{b1}\left(\hat{p}_{u,t'},\hat{p}_{v,t'}\right)|_{\bigcup\mathcal{I}_\alpha} + E_{b2}\left(\hat{p}_{u,t'},\hat{p}_{u,t'-1}\right)|_{\bigcup\mathcal{I}_\alpha}$, while $\sum_{\alpha\in P} E\left(\hat{p}_{u,t'}\right)|_{\mathcal{B}_\alpha} = 2E_{b1}\left(\hat{p}_{u,t'},\hat{p}_{v,t'}\right)|_{\bigcup\mathcal{B}_\alpha} + 2E_{b2}\left(\hat{p}_{u,t'},\hat{p}_{u,t'-1}\right)|_{\bigcup\mathcal{B}_\alpha}$. The first equation is obvious. As for the second equation, it is because when summed over the boundary area $\bigcup\mathcal{B}_\alpha$ (i.e. when $\hat{p}_{u,t'} \ne \hat{p}_{v,t'}, (u,v) \in L_{t'}$ and $\hat{p}_{u,t'} \ne \hat{p}_{u,t'-1}, u \in \tilde{U}_{t..t+W}$), each pairwise cost between two players with different placement decisions is added up twice, once on the $\hat{p}_{u,t'}$ side and once on the $\hat{p}_{v,t'}$ or $\hat{p}_{u,t'-1}$ side. We also have $\sum_{\alpha\in P} E\left(p_{u,t'}^*\right)|_{\mathcal{I}_\alpha} = E_a\left(p_{u,t'}^*\right) + E_{b1}\left(p_{u,t'}^*,p_{v,t'}^*\right)|_{\bigcup\mathcal{I}_\alpha} + E_{b2}\left(p_{u,t'}^*,p_{u,t'-1}^*\right)|_{\bigcup\mathcal{I}_\alpha}$, and $\sum_{\alpha\in P}\lambda E\left(p_{u,t'}^*\right)|_{\mathcal{B}_\alpha} = 2\lambda\left[E_{b1}\left(p_{u,t'}^*,p_{v,t'}^*\right)|_{\bigcup\mathcal{B}_\alpha} + E_{b2}\left(p_{u,t'}^*,p_{u,t'-1}^*\right)\right]|_{\bigcup\mathcal{B}_\alpha}$. Therefore,

$$\begin{aligned}
&E_a\left(\hat{p}_{u,t'}\right) + E_{b1}\left(\hat{p}_{u,t'},\hat{p}_{v,t'}\right)|_{\bigcup\mathcal{I}_\alpha} + E_{b2}\left(\hat{p}_{u,t'},\hat{p}_{u,t'-1}\right)|_{\bigcup\mathcal{I}_\alpha} +\\
&2E_{b1}\left(\hat{p}_{u,t'},\hat{p}_{v,t'}\right)|_{\bigcup\mathcal{B}_\alpha} + 2E_{b2}\left(\hat{p}_{u,t'},\hat{p}_{u,t'-1}\right)|_{\bigcup\mathcal{B}_\alpha} + E_c\left(\hat{p}_{u,t'}\right)\\
&\le E_a\left(p_{u,t'}^*\right) + E_{b1}\left(p_{u,t'}^*,p_{v,t'}^*\right)|_{\bigcup\mathcal{I}_\alpha} + E_{b2}\left(p_{u,t'}^*,p_{u,t'-1}^*\right)|_{\bigcup\mathcal{I}_\alpha} +\\
&2\lambda\left[E_{b1}\left(p_{u,t'}^*,p_{v,t'}^*\right)|_{\bigcup\mathcal{B}_\alpha} + E_{b2}\left(p_{u,t'}^*,p_{u,t'-1}^*\right)|_{\bigcup\mathcal{B}_\alpha}\right] + E_c\left(p_{u,t'}^\alpha\right).
\end{aligned}$$

Since,

$$\begin{aligned}
&E_a\left(\hat{p}_{u,t'}\right) + E_{b1}\left(\hat{p}_{u,t'},\hat{p}_{v,t'}\right)|_{\bigcup\mathcal{I}_\alpha} + E_{b2}\left(\hat{p}_{u,t'},\hat{p}_{u,t'-1}\right)|_{\bigcup\mathcal{I}_\alpha} +\\
&E_{b1}\left(\hat{p}_{u,t'},\hat{p}_{v,t'}\right)|_{\bigcup\mathcal{B}_\alpha} + E_{b2}\left(\hat{p}_{u,t'},\hat{p}_{u,t'-1}\right)|_{\bigcup\mathcal{B}_\alpha} + E_c\left(\hat{p}_{u,t'}\right) = E\left(\hat{p}_{u,t'}\right)\\
&E_a\left(p_{u,t'}^*\right) + E_{b1}\left(p_{u,t'}^*,p_{v,t'}^*\right)|_{\bigcup\mathcal{I}_\alpha} + E_{b2}\left(p_{u,t'}^*,p_{u,t'-1}^*\right)|_{\bigcup\mathcal{I}_\alpha} +\\
&E_{b1}\left(p_{u,t'}^*,p_{v,t'}^*\right)|_{\bigcup\mathcal{B}_\alpha} + E_{b2}\left(p_{u,t'}^*,p_{u,t'-1}^*\right)|_{\bigcup\mathcal{B}_\alpha} + E_c\left(p_{u,t'}^*\right) = E\left(p_{u,t'}^*\right)
\end{aligned}$$

We have,

$$\begin{aligned}
&E\left(\hat{p}_{u,t'}\right) + E_{b1}\left(\hat{p}_{u,t'},\hat{p}_{v,t'}\right)|_{\bigcup\mathcal{B}_\alpha} + E_{b2}\left(\hat{p}_{u,t'},\hat{p}_{u,t'-1}\right)|_{\bigcup\mathcal{B}_\alpha}\\
&\le E\left(p_{u,t'}^*\right) + (2\lambda-1)\left[E_{b1}\left(p_{u,t'}^*,p_{v,t'}^*\right)|_{\bigcup\mathcal{B}_\alpha} +\right.\\
&\left.E_{b2}\left(p_{u,t'}^*,p_{u,t'-1}^*\right)|_{\bigcup\mathcal{B}_\alpha}\right] + E_c\left(p_{u,t'}^\alpha\right) - E_c\left(p_{u,t'}^*\right).
\end{aligned}$$

Since $p_{u,t'}^\alpha$ is an α expansion to the solution $\hat{p}_{u,t'}$, it would use no more MECs than the $\hat{p}_{u,t'}$, thus $E_c\left(p_{u,t'}^\alpha\right) \le E_c\left(\hat{p}_{u,t'}\right)$. Then, we have

$$E\left(\hat{p}_{u,t'}\right) \le 2\lambda E\left(p_{u,t'}^*\right) + E_c(I).$$

□

G. α Expansion with Capacity Constraint

We have focused on the uncapacitated version of the dynamic placement problem so far, which indeed works if the number of CRMs placed at an MEC, depending on the number of groups rather than the number of users, is obviously smaller than the capacity of that MEC. If it is desired that the capacity of each MEC needs to be explicitly respected, then we can still do our α-expansion algorithm and construct the graph—the only difference is that we do not seek the minimal s-t cut, but seek the k-size minimal s-t cut [47] which ensures the number of nodes on one side of the cut not to exceed k. The execution of the α expansion algorithm for the CRM placement problem with capacity constraints is the same as the problem without capacity constraints, except that in the capacity constraint case, we invoke the k-size s-t min-cut algorithm to replace the s-t min-cut algorithm in Line 24 of Algorithm 2. Unlike the s-t min-cut problem which can be solved in polynomial time, the k-size s-t min-cut is NP-hard [46]. The algorithm in [47] keeps the set of the $source$ under the size of k on an undirected graph with approximation ratio of $\frac{k+1}{k+1-k^*}$, where k^* is the size of the $source$ side of an optimal solution. Since it works on an undirected graph, we can restrict our $terminal$ side size under k and regard it as the $source$ in [47]. This way, the algorithm and properties in [47] can still be applied to our problem, which ensures that the number of CRMs on MEC α is under k after each α expansion. Since we traverse for each MEC (Line 4 in Algorithm 1), the number of CRMs on the original MEC is also restricted to less than k during execution.

In this case, we may lose our performance analysis; however, through evaluations, we show that our algorithm still works well practically.

VI. Evaluation

A. Data and Settings

*1) **User Dynamics and MECs**:* For user dynamics, we use anonymized data traces provided by one of the largest telecom carriers in China. We have obtained the permission to use the data in our research, on the condition that the carrier's name is anonymized. The dataset contains 5,000 users' signaling data logs in the city of Guangzhou, one of the largest cities in China, during the week of November 15th through 21st, 2015. Each log entry contains an anonymized user ID, user's joining time, leaving time, connected cell (i.e., base station) ID, uplink/downlink throughput, etc.

For the MECs, we extract all unique cell IDs from the same data traces, and view each cell as an MEC, following the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

5G standard and the mobile edge computing paradigm [11]. We get 358 unique cells, and for each cell, we also get its associated longitude and latitude. In our data, the longest distance between two cells is 27.03 km, and the majority of such inter-cell distances are less than 10 km. We assume the transmission delay d between two MECs is proportional to the distance between the two cells, calculated by the longitudes and latitudes. We assume MECs' processing capabilities are at three levels: high, medium, and low. For each MEC, the processing capability h_i is drawn from a normal distribution with an expectation (representing the processing capability level) and a standard deviation (allowing the heterogeneity of MECs). We treat 5 minutes as a time slot, and have about 2000 consecutive time slots in total.

*2) **User Groups and VR Gaming**:* We consider 5v5 MOBA games, where each group contains 10 players at the beginning, and organize users in groups by the ladder tournament matchmaking algorithm [9], [10] that has been used in real-world games such as StarCraft II, DOTA, and League of Legends. The matchmaking algorithm for game group formation [23] takes into account the factors such as player geo-location, network latency, waiting time, and fairness. To use the matchmaking algorithm, we randomly assign each player with a level of junior, middle, senior, or ultimate [23]. At each time slot, the matchmaking algorithm runs to form the groups.

We also assign the gaming parameters. We assume that the CRM workload $R_{j,t}$ follows a uniform distribution. The switching cost g between two consecutive time slots of a single CRM is also drawn from a uniform distribution. The frame rate $f_{u,t}$ is drawn from a uniform distribution of $[50, 70]$ frame-per-second [20]. The in-group synchronization rate $f_{j,t}$ is set to be equal to the frame rate. The interaction rate $f_{u,v,t}$ of two players of different groups is of 99% probability to be 0, and of 1% probability to be drawn from a uniform distribution of $(0, 3]$ times per second; the interaction rate $f_{u,v,t}$ of two players of the same group is drawn from a uniform distribution of $[0, 10]$ times per second. All our simulations are conducted on a DELL PowerEdge R740 server with 32 GB RAM and a 2.1 GHz Intel Xeon Silver 4110 CPU.

The initial (in the first time slot) CRM placement is shown in Fig. 4. The dots on the map are the locations of MECs. There are 1983 players active in this time slot. We zoom in to 5 selected MECs and list their group settings and user connections to other users of different groups.

B. Evaluation Results

We note that we conduct two scales of experiments. As for "small scale" experiments, the number of users is under 2000. As for "large scale" experiments, the number of users is above 4000. We state at the caption of each figure the scale at which the simulation is conducted; otherwise, the simulation is done at both scales and the result shows similar trends.

*1) **Running Time and Cost Comparison**:* We compare our MPC-based and α-expansion-based approach to four other placement approaches: greedy placement (i.e., placing CRMs optimally using the prediction information within the prediction window), random placement (i.e., placing each CRM randomly on an MEC at each time slot), nearest placement (i.e., placing the CRM of u to her nearest MEC $p^{\dagger}_{u,t}$ at each time slot, or "following the user"), and offline optimal placement over the entire time horizon (i.e., placing CRMs optimally over time as if it was an offline optimization problem), where both greedy and offline optimum are obtained using the Gurobi [8] optimization solver.

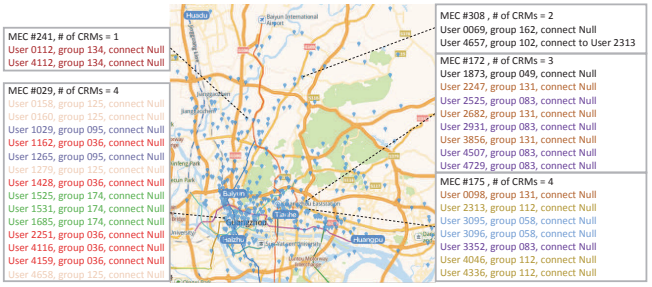


Fig. 4: The initial CRM placement. Users of the same color are in the same group. "Connect Null" means that the user does not have communication to users of a different group; otherwise, we directly show the ID(s) of her connected user(s).

Fig. 5 compares the running time and Fig. 6 compares the total cost over time, respectively. The y-axis of Fig. 5 is the actual time our algorithms takes to run in our test environment, with units of milliseconds. The y-axis of Fig. 6 is the normalized total cost w.r.t. the offline optimal solution. T represents the length of the total time horizon, and W represents the length of the prediction window. Here, we set each prediction window as 20 time slots with accurate predictions with total time horizon of 100 time slots. Due to randomness in our inputs, each result in Fig. 5 and Fig. 6 is averaged over 5 runs. We show weekdays and weekends separately in Fig. 6, due to the different mobility and traffic patterns [18]. We make the following observations. Despite random and nearest methods run faster, our approach performs overwhelmingly better than both of them in terms of the total cost. When the user size is at small scale, our approach achieves around $1.2\times$ the cost on both weekdays and weekends with around 60% less running time compared to greedy solution, and can achieve around 27% cost degradation on weekdays and around 23% cost degradation on weekends compared to the offline optimal method which runs around 12 hours. When the user population is large, both greedy and offline optimum cannot return any result within a reasonable amount of time, and thus we are not able to show their running time in Fig. 5. In Fig. 6, no matter on weekdays or weekends with different user dynamics, our approach achieves consistent results.

*2) **Capacity Constraint**:* Fig. 7 and Fig. 8 visualize the total cost when it is desired to strictly enforce MECs' capacity constraints. In order to observe the impact of capacity constraint compared to offline optimal solution, we conduct the simulation at small user scale. We assume all MECs are homogeneous, and in both two figures, the x-axis is the capacity of the MEC. In Fig. 7, we assume the size of the CRM is always 1, while in Fig. 8 the size of the CRM is drawn randomly from uniform distribution of $(0, 2]$. As can be seen, even without any theoretical performance guarantee in this case, our proposed algorithm can still effectively reduce

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

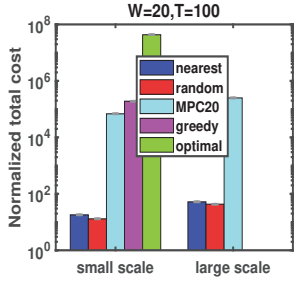


Fig. 5: Running Time of Different Methods

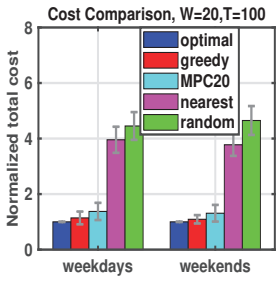


Fig. 6: Cost of Different Methods (Small Scale)

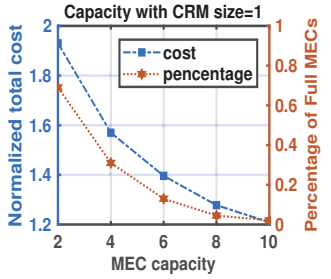


Fig. 7: Capacity with Fixed CRM Size (Small Scale)

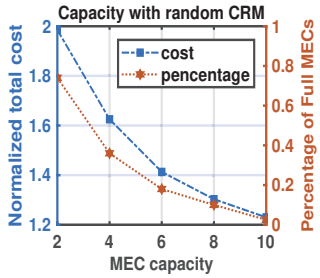


Fig. 8: Capacity with Random CRM Size (Small Scale)

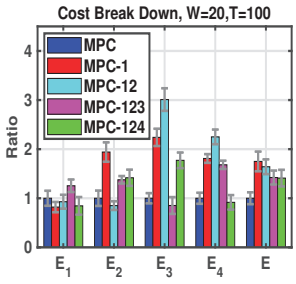


Fig. 9: Cost Breakdown of MPC Approach

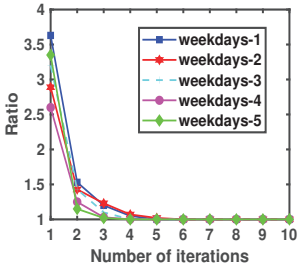


Fig. 10: Convergence vs. Number of Iterations

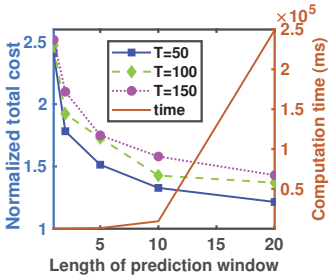


Fig. 11: Convergence vs. Window Length (Large Scale)

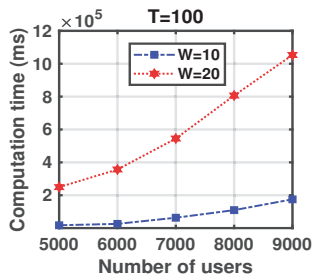


Fig. 12: Running time vs. Number of Users (Large Scale)

the total cost. We start from the MEC capacity of 2. In this case, the normalize total cost for both the fixed CRM size (i.e., size = 1) and the random CRM size is less than 2; the percentage of MECs that are full (represented by orange line) is around 75% for the case of the fixed CRM size and around 80% for the random CRM size. As the capacity of each MEC becomes larger, we reduce more cost as there is more room for optimization. When the MEC capacity reaches 10, the percentage of MECs that are full approaches 0, and the normalized total cost for both cases is around 1.2. The trends in the two figures are similar, while the fixed CRM size setting always shows a little better performance than the random CRM size setting in terms of both the total cost and the percentage of full MECs.

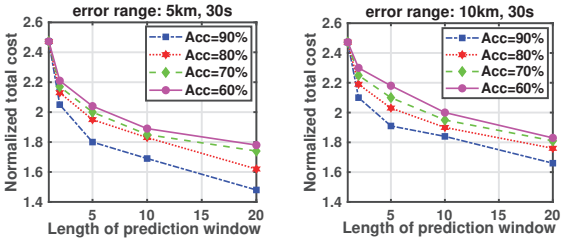
*3) **Cost Breakdown**:* Fig. 9 breaks down the total cost for our approach and digs into the impact of each type of the cost, where "MPC" represents the optimization of all types of costs, "MPC1" only optimizes computation, "MPC12" optimizes both computation and communication, "MPC123" optimizes computation, communication, and colocation, and "MPC124" optimizes computation, communication, plus the switching cost. E_1, E_2, E_3, E_4 and E represent the cost of computation overhead, communication delay, colocation interference, switching cost and total cost, respectively. The y-axis is the normalized ratio of the corresponding cost of each target solution (i.e., MPC, MPC-1, MPC-12, MPC-123, MPC-124) w.r.t. the cost of using the overall optimization (i.e., MPC). As can be seen, our approach allows for optimizing different types of costs selectively, and the overall optimization achieves the best performance when all types of costs are considered.

*4) **Convergence**:* Fig. 10 and Fig. 11 illustrate the convergence speed of our approach. In Fig. 10, we normalize the total cost w.r.t. the solution provided by our algorithm when it become stable (i.e., after several rounds of iterations). In order to not confuse with the term of "normalized total cost" (used in other figures), which is normalized to the offline optimal solution, we choose to use the term "ratio" for the y-axis of Fig. 10 instead.In Fig. 10, we see that our α-expansion algorithm reduces most of the cost in the first few iterations, and terminates only after a smalle number of iterations. In our experiments, the total cost does not decrease after 6 iterations at most; in other words, our algorithm converges very fast. In Fig. 11, the lines of T=50, T=100, and T=100 align with the left y-axis, which illustrate the normalized total cost; the line of time/ms aligns with the right y-axis, which shows the computation time.We vary the length of the entire time horizon, normalize the total cost over the corresponding offline optimal cost, respectively, and observe that our algorithm has better total cost over time, as the size of the prediction window becomes larger. This aligns with the expectation for the MPC framework: the result becomes better as more future information is known.

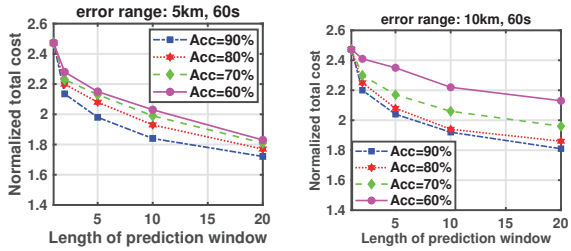
*5) **Scalability**:* Fig. 12 demonstrates the scalability of our algorithm. In particular, we show the amount of running time it takes to execute our algorithm to compute the placement decisions (the y-axis) as the size of the prediction window and the number of users vary (the x-axis).The cases of the enlarged number of users are done by concatenating different pieces of data traces and treating it as a single day with different users. The result exhibits that in general our algorithm scales well—just a little bit superlinear in the execution time as the number of users increases.

*6) **Robustness**:* Fig. 13 depicts the total cost achieved by our MPC-based algorithm when fed with *inaccurate* predictions. MPC often assumes accurate predictions of the input data in each prediction window; however, in reality, it may not be easy to get perfect predictions, and one may only have inaccurate ones, depending on the methods of prediction

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications



(a) Error range: 5km, 30s (b) Error range: 10km, 30s



(c) Error range: 5km, 60s (d) Error range: 10km, 60s

Fig. 13: Results of Inaccurate Predictions, Small Scale

and the statistical nature of the inputs. To simulate inaccurate predictions, we randomly select players, disturb the location and the joining/leaving time within a geo-error range and a time-error range, and feed such disturbed predictions into each prediction window. Fig. 13 shows the total cost with various error ranges. The total cost of our MPC-based placement is still much better than the random and nearest placements, even when the prediction accuracy is 60% with the error range of $(10\,\text{km}, 60\,\text{s})$, i.e., $2.1\times$ improvement compared to random and $1.8\times$ improvement compared to nearest. Besides, in each subfigure, as the prediction becomes more accurate, and across subfigures, as the error range decreases, we all obtain more cost benefit by leveraging the inaccurate predictions.

VII. Conclusions and Discussions

In this paper, we investigate the dynamic service placement problem for VR group gaming in the distributed mobile edge cloudlets environment. As in our models, this problem is an optimization problem involving discrete, nonconvex, and higher-order objectives with coupled decisions over time, which is very challenging to solve. While adopting the model predictive control framework to construct an online algorithm, we focus on designing approximation algorithms for the problem over each prediction window, where we solve the problem via solving a series of α-expansion-based binary optimizations by graph-theoretic minimal cuts and prove the bounded performance guarantee with this approach. Through extensive evaluations with real-world data, we validate the effectiveness, the efficiency, the scalability, and the robustness of our proposed scheme.

The scope of our work can be recapped as follows. Our work is focused on the mathematical modeling, the formal algorithm design, and the theoretical performance analysis for the problem of the predictive online placement of the CRMs over edge cloudlets for the mobile VR group gaming services. Our current paper is a self-contained piece of theoretical work with simulated numerical results. We also have ongoing engineering efforts on placing our algorithm implementations into real-world systems for measurement and validation, and we would like to postpone any discussion there to the future.

Within this scope, we have identified two limitations of our work as follows. First, our current provable performance guarantee is for every prediction window alone; we have not been able to prove the overall performance guarantee, if it exists, for the MPC framework that invokes our algorithm repetitively over overlapped prediction windows as time goes. Fortunately, via extensive numerical evaluations, we have found that the overall empirical performance of our algorithm is already advantageous compared against existing methods. Second, our current provable performance guarantee assumes exact predictions; we have not been able to prove the corresponding performance guarantee, if it exists, for inexact or error-prone predictions. Doing so would require incorporating error models into our algorithm design and can be of future interest. Analogously, through evaluations, we have demonstrated that our approach is empirically robust to prediction errors and can still be better than existing methods when evaluated under the same error-prone predictions.

References

[1] "Virtual Reality (VR) In Gaming Market Size Worth $45.09 Billion By 2025," https://www.grandviewresearch.com/press-release/global-virtual-reality-in-gaming-market. Access date: Jun. 21, 2019.
[2] "Google Cardboard - Google VR," https://vr.google.com/cardboard/. Access date: Jun. 21, 2019.
[3] "Samsung Gear VR," https://www.samsung.com/global/galaxy/gear-vr/. Access date: Jun. 21, 2019.
[4] "Top 10 highest earning mobile games in the world," http://www.myzaker.com/article/599240721bc8e0706a00002f/. Access date: Jun. 21, 2019.
[5] "Onmyoji," http://www.onmyojigame.com. Access date: Jun. 21, 2019.
[6] "Arena of Valor," https://www.arenaofvalor.com. Access date: Jun. 21, 2019.
[7] "Fast-Paced Multiplayer (Part I): Client-Server Game Architecture," https://gabrielgambetta.com/client-server-game-architecture.html. Access date: Jun. 21, 2019.
[8] "Gurobi Optimization - The State-of-The-Art Mathematical Programming Solver," http://www.gurobi.com. Access date: Jun. 21, 2019.
[9] "GitHub - dmatch," https://github.com/loveisasea/dmatch. Access date: Jun. 21, 2019.
[10] "Ladder tournament," https://en.wikipedia.org/wiki/Ladder_tournament. Access date: Jun. 21, 2019.
[11] "3GPP enables MEC over a 5G core," http://www.3gpp.org/news-events/partners-news/1969-mec/. Access date: Jun. 21, 2019.
[12] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal et al., "Mobile-edge computing introductory technical white paper," ETSI White Paper, 2014.
[13] C. Wilson, B. Boe, A. Sala, K. P. N. Puttaswamy, and B. Y. Zhao, "User Interactions in Social Networks and Their Implications," Proceedings of EuroSys, 2009.
[14] S. H. Lim, J. S. Huh, Y. Kim, G. M. Shipman, and C. R. Das, "D-factor: A Quantitative Model of Application Slow-down in Multi-resource Shared Systems," Proceedings of ACM SIGMETRICS, 2012.
[15] Y. Guo, A. L. Stolyar, and A. Walid, "Shadow-routing Based Dynamic Algorithms for Virtual Machine Placement in a Network Cloud," Proceedings of IEEE INFOCOM, 2013.
[16] Y. Le, J. Liu, F. Ergun, and D. Wang, "Online Load Balancing for MapReduce with Skewed Data Input," Proceedings of IEEE INFOCOM, 2014.
[17] Y. Chon, E. Talipov, H. Shin, and H. Cha,"SmartDC: Mobility Prediction-Based Adaptive Duty Cycling for Everyday Location Monitoring," IEEE Transactions on Mobile Computing, 13(3):512-525, 2014.

[18] H. Wang, F. Xu, Yong Li, P. Zhang, and D. Jin, "Understanding Mobile Traffic Patterns of Large Scale Cellular Towers in Urban Environment", Proceedings of ACM IMC, 2015.

[19] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. Leung, "Dynamic Service Migration in Mobile Edge-Clouds," Proceedings of IFIP Networking, 2015.

[20] K. Boos, D. Chu, and E. Cuervo, "FlashBack: Immersive Virtual Reality on Mobile Devices via Rendering Memoization," Proceedings of ACM MobiSys, 2016.

[21] T. Taleb, A. Ksentini, and P. Frangoudis, "Follow-Me Cloud: When Cloud Services Follow Mobile Users," IEEE Transactions on Cloud Computing, 2016.

[22] O. Abari, D. Bharadia, A. Duffield, and D. Katabi, "Enabling High-Quality Untethered Virtual Reality," Proceedings of USENIX NSDI, 2017.

[23] Z. Chen, Su. Xue, J. Kolen, N. Aghdaie, K. A. Zaman, Y. Sun, and M. S. Nasr, "EOMM: An Engagement Optimized Matchmaking Framework," Proceedings of WWW, 2017.

[24] S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer, and K. Leung, "Dynamic Service Placement for Mobile Micro-clouds with Predicted Future Costs," IEEE Transactions on Parallel and Distributed Systems, 28(4):1002-1016, 2017.

[25] Z. Lai, Y. C. Hu, Y. Cui, L. Sun, and N. Dai, "Furion: Engineering High-Quality Immersive Virtual Reality on Today's Mobile Devices," Proceedings of ACM MOBICOM, 2017.

[26] Z. Tan, Y. Li, Q. Li, Z. Zhang, Z. Li, and S. Lu, "Supporting Mobile VR in LTE Networks: How Close Are We?" Proceedings of ACM SIGMETRICS, 2018.

[27] T. Kämäräinen, M. Siekkinen, J. Eerikäinen, and A. Ylä-Jääski, "CloudVR: Cloud Accelerated Interactive Mobile Virtual Reality," Proceedings of ACM Multimedia 2018.

[28] J. Feng, Y. Li, C. Zhang, F. Sun, F. Meng, A. Guo, and D. Jin, "DeepMove: Predicting Human Mobility with Attentional Recurrent Networks," Proceedings of WWW, 2018.

[29] L. Wang, L. Jiao, T. He, J. Li, and M. Mühlhäuser, "Service Entity Placement for Social Virtual Reality Applications in Edge Computing," Proceedings of IEEE INFOCOM, 2018.

[30] W. Zhang, J. Chen, Y. Zhang, and D. Raychaudhuri,"Towards Efficient Edge Cloud Augmentation for Virtual Reality MMOGs," Proceedings of ACM/IEEE SEC, 2017.

[31] L. Guo, J. Pang, and A. Walid, "Joint Placement and Routing of Network Function Chains in Data Centers," Proceedings of IEEE INFOCOM, 2018.

[32] L. Wang, L. Jiao, J. Li, J. Gedeon, and M. Mühlhäuser, "MOERA: Mobility-agnostic Online Resource Allocation for Edge Computing," IEEE Transactions on Mobile Computing, 2018.

[33] T. Ouyang, Z. Zhou, and X. Chen, "Follow Me at the Edge: Mobility-Aware Dynamic Service Placement for Mobile Edge Computing,", IEEE Journal on Selected Areas in Communications, 36(10):2333-2345, 2018.

[34] L. Chen, J. Xu, S. Ren, and P. Zhou, "Spatio-Temporal Edge Service Placement: A Bandit Learning Approach," IEEE Transactions on Wireless Communications, 17(12):8388-8401, 2018.

[35] A. Gupta, M. Tornatore, B. Jaumard and B. Mukherjee, "Virtual-Mobile-Core Placement for Metro Network", Proceedings of IEEE NetSoft 2018;

[36] A. Baumgartner, V. S. Reddy, and T. Bauschert, "Mobile core network virtualization: A model for combined virtual core network function placement and topology optimization", Proceedings of IEEE NetSoft, 2015.

[37] D. Dietrich, C. Papagianni, P. Papadimitriou, and J. S. Baras, "Network function placement on virtualized cellular cores", Proceedings of IEEE COMSNETS, 2017.

[38] X. Hou, S. Dey, J. Zhang, and M. Budagavi,"Predictive View Generation to Enable Mobile 360-degree and VR Experiences", ACM SIGCOMM Workshop 2018.

[39] S. A. Cook, "The Complexity of Theorem-proving Procedures," Proceedings of ACM STOC, 1971.

[40] S. H. Owen, and M. S. Daskin, "Strategic Facility Location: A Review," European Journal of Operational Research, 111(3):423-447, 1998.

[41] Y. Boykov, O. Veksler, and R. Zabih, "Fast Approximate Energy Minimization via Graph Cuts," IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(11):1222-1239, 2001.

[42] V. Kolmogorov and R. Zabih, "What Energy Functions Can Be Minimized via Graph Cuts?" IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(2):147-159, 2004.

[43] D. Freedman and P. Drineas, "Energy Minimization via Graph Cuts: Settling What is Possible," Proceedings of IEEE CVPR, 2005.

[44] A. Delong, A. Osokin, H. N. Isack, and Y. Boykov, "Fast Approximate Energy Minimization with Label Costs," International Journal of Computer Vision, 96(1):1-27, 2012.

[45] J. R. Edmonds and R. M. Karp. "Theoretical improvements in algorithmic efficiency for network flow problems," Journal of the ACM, 1972. 19 (2): 248C264.

[46] A. Li, P. Zhang, "Unbalanced Graph Partitioning," Proceedings of International Symposium on Algorithms and Computation, 2010.

[47] P. Zhang, "A New Approximation Algorithm for the Unbalanced Min s–t Cut Problem," Theoretical Computer Science, 609(3):658-665, 2016.

[48] C. Yang, X. Shi, L. Jie, J. Han: "I Know You'll Be Back: Interpretable New User Clustering and Churn Prediction on a Mobile Social Application", Proceedings of ACM SIGKDD, 2018.

APPENDIX

Algorithm 2 Solving α-Expansion via Min s-t Graph Cut

Input:

L_t: the set of all the pairs of interacting players at time t;
$R_{j,t}$: the workload of the CRM of group j at time t;
h_i: the processing capability of MEC i;
$d(p,q)$: the network delay between MECs p and q;
$g(p_{u,t}, p_{u,t-1})$: switching cost of player u at time t;
$f_{u,v,t}$: interaction rate between players u and v at time t;
$f_{u,t}$: the frame rate of player u at time t;
$f_{j,t}$: the synchronization rate of group j at time t;
$a_{i,1}$: the parameters of the dilation factor of MEC i;
$\theta_{i,j,t'} = \frac{R_{j,t'}}{h_i} + f_{j,t} d(i,c) + a_{i,1}$;
$B_{1,u,t'} = \frac{d(\alpha, p_{v,t'}) - d(p_{u,t'}, \alpha) + d(p_{u,t'}, p_{v,t'})}{2}$,
$B_{2,v,t'} = \frac{d(p_{u,t'}, \alpha) - d(\alpha, p_{v,t'}) + d(p_{u,t'}, p_{v,t'})}{2}$,
$B_{3,u,t'} = \frac{g(\alpha, p_{u,t'-1}) - g(p_{u,t'}, \alpha) + g(p_{u,t'}, p_{u,t'-1})}{2}$,
$B_{4,u,t'-1} = \frac{g(p_{u,t'}, \alpha) - g(\alpha, p_{u,t'-1}) + g(p_{u,t'}, p_{u,t'-1})}{2}$.
$\delta_{i,j,t}$: binary indicator of whether $\exists u \in U_{j,t}$ s.t. $p_{u,t} = i$.

Output:

$x_{u,t'}$: binary variable of whether to place the CRM of player u on MEC α at time t' in α expansion, $\forall u \in \tilde{U}_{t..t+W}, t' \in \{t, ..., t+W\}$;
$y_{i,j,t'}$: auxiliary binary variables, $1 \le i \le n, 1 \le j \le m_{t'}, t' \in \{t, ..., t+W\}$;
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$: the constructed graph.

$\mathcal{V} = \{x_{u,t'} | u \in \tilde{U}_{t..t+W}\} \bigcup \{y_{i,j,t'} | 1 \le i \le n, 1 \le j \le m_{t'}, t' \in \{t, ..., t+W\}\} \bigcup \{\mathcal{F}\} \bigcup \{source, terminal\}$
for each $t' \in \{t, ..., t+W\}$ **do**
  for each $u \in \tilde{U}_{t..t+W}$ **do**
    Weight of edge between $x_{u,t'}$ and $x_{u,t'-1}$ $\leftarrow$ $\frac{1}{2}[g(\alpha, p_{u,t'-1}) + g(p_{u,t'}, \alpha) - g(p_{u,t'}, p_{u,t'-1})]$
    Weight of edge between $x_{u,t'}$ and $source$ $\leftarrow$ $f_{u,t} d(\alpha, p^{\dagger}_{u,t'}) + B_{3,u,t'} + B_{4,u,t'}$
    Weight of edge between $x_{u,t'}$ and $terminal$ $\leftarrow$ $f_{u,t} d(p_{u,t'}, p^{\dagger}_{u,t'}) + \sum_{i=1}^{n} \frac{1}{2}\theta_{i,j,t'}$
    for each $1 \le i \le n$ **do**
      for each $1 \le j \le m_{t'}$ **do**
        Weight of edge between $x_{u,t'}$ and $y_{i,j,t'}$ $\leftarrow$ $\frac{1}{2}\theta_{i,j,t'}$
      end for
    end for
  end for

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2019.2927071, IEEE Journal on Selected Areas in Communications

for each $(u, v) \in L_{t'}$ **do**
  Weight of edge between $x_{u,t'}$ and $x_{v,t'}$ $\leftarrow \frac{1}{2} f_{u,v,t}\big[d(\alpha, p_{v,t'}) + d(p_{u,t'}, \alpha) - d(p_{u,t'}, p_{v,t'})\big]$
  Add weight of edge between $x_{u,t'}$ and $source$ by $f_{u,v,t} B_{1,u,t'}$
  Add weight of edge between $x_{v,t'}$ and $source$ by $f_{u,v,t} B_{2,v,t'}$
end for
for each $1 \le i \le n$ **do**
  for each $1 \le j \le m_{t'}$ **do**
    Weight of edge between $source$ and $y_{i,j,t'}$ $\leftarrow \frac{1}{2} f_{u,v,t}\big[d(\alpha, p_{v,t'}) + d(p_{u,t'}, \alpha) - d(p_{u,t'}, p_{v,t'})\big]$
  end for
end for
end for
s-t min cut of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

Yuan Zhang received the B.Eng. degree in Telecommunications from Beijing University of Posts and Telecommunications in 2008, M.Eng. degree in Electronic Engineering from Tsinghua University, Beijing, China, in 2011 and the Ph.D. degree in Computer Science from University of Göttingen in Germany in 2015. She is now an assistant professor in Communication University of China. Her research interest includes mobile cloud computing, cloud resource scheduling, and SDN architecture.

Lei Jiao received the Ph.D. degree in computer science from University of Göttingen in Germany in 2014. He was a researcher at IBM Research in Beijing, China in 2010 prior to his Ph.D. study. He was also a member of technical staff at Bell Labs in Dublin, Ireland from 2014 to 2016. He is currently an assistant professor at the Department of Computer and Information Science, University of Oregon, USA. He is broadly interested in the mathematical and algorithmic sciences (e.g., optimization, control, mechanism design) for systems and networks. His research has been published in journals such as IEEE/ACM ToN, IEEE JSAC, and TMC, and in conferences such as ACM SIGMETRICS, MOBIHOC, IEEE INFOCOM, and ICNP. He is on the technical program committees of conferences including IEEE INFOCOM (Distinguished Member) and ICDCS. He is the recipient of the Best Paper Awards in IEEE CNS 2019 and IEEE LANMAN 2013, and the Recognition Award of Alcatel-Lucent Bell Labs UK and Ireland in 2016.

Jinyao Yan received the B.S. degree from Tianjin University, the M.S. and Doctor (in engineering) degrees from Beijing Broadcasting Institute, the Ph.D. (in science) degree from Swiss Federal Institute of Technology (ETH Zurich). Since 2010, he has been a Professor at Communication University of China, Beijing, P. R. China. He is a guest professor in communication system group at ETH Zurich. His research interests are in the areas of future network, multimedia communication, and cloud computing.

Xiaojun Lin received his B.S. degree from Zhongshan University, Guangzhou, China, in 1994, and his M.S. and Ph.D. degrees from Purdue University in 2000 and 2005, respectively. He is currently a Professor at School of Electrical and Computer Engineering at Purdue University. Dr. Lin's research interests are in the analysis, control and optimization of large and complex networked systems, including both communication networks and power grid. He is a Fellow of IEEE. He received the IEEE INFOCOM 2008 best paper award and the 2005 best paper of the year award from Journal of Communications and Networks. He received the NSF CAREER award in 2007. He was the Workshop co-chair for IEEE GLOBECOM 2007, the Panel co-chair for WICON 2008, the TPC co-chair for ACM MobiHoc 2009, the Mini-Conference co-chair for IEEE INFOCOM 2012, and the General co-chair for ACM e-Energy 2019. He is currently serving as an Area Editor for (Elsevier) Computer Networks journal, and has served as an Associate Editor for IEEE/ACM Transactions on Networking and a Guest Editor for (Elsevier) Ad Hoc Networks journal.