Enabling QoE Support for Interactive Applications over Mobile Edge with High User Mobility

Xiaojun Shang*, Yaodong Huang†, Yingling Mao*, Zhenhua Liu‡, and Yuanyuan Yang*,
*Department of Electrical and Computer Engineering, Stony Brook University, USA
†College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China
‡Department of Applied Mathematics and Statistics, Stony Brook University, USA

IEEE INFOCOM 2022 - IEEE Conference on Computer Communications | 978-1-6654-5822-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/INFOCOM48880.2022.9796811

***Abstract*—The fast development of mobile edge computing (MEC) and service virtualization brings new opportunities to the deployment of interactive applications, e.g., VR education, stream gaming, autopilot assistance, at the network edge for better performance. Ensuring quality of experience (QoE) for such services often requires the satisfaction of multiple quality of service (QoS) factors, e.g., short delay, high throughput rate, low packet loss. Nevertheless, existing mobile edge networks often fail to meet these requirements due to the mobility of end users and the volatility of network conditions. In this paper, we propose a novel scheme that both reduces delay and adjusts data throughput rate for QoE enhancement. We design an online service placement and throughput rate adjustment (SPTA) algorithm which coordinately migrates virtual services while tuning their data throughput rates based on real-time bandwidth fluctuation. By implementing a small-scale prototype supporting stream gaming at the edge, we show the necessity and feasibility of our work. Based on data from the experiments, we conduct real-world trace driven simulations to further demonstrate the advantages of our scheme over existing baselines.**

I. Introduction

The emerging mobile edge computing (MEC) has made high-demanding virtual services, e.g., internet of things applications, virtual reality, stream gaming, smart surveillance, possible for mobile end users [1], [2]. The central concept of mobile edge computing is to deploy computing infrastructures at the network edge within the proximity of end users. Virtual services are then processed on these edge infrastructures and outputs are sent to end users through mobile networks. Mobile edge computing can thus achieve much lower delay compared with cloud-centric computing paradigms while overcoming computational resource shortage of mobile devices limited by their physical sizes [3], [4]. Many conceptual models of such edge computing infrastructures have been proposed, e.g., FMC [5], micro datacenter [6], and Cloudlet [7]. In this paper, we call these edge computing facilities Edge Nodes (ENs). The mobile edge computing market is rapidly developing, e.g., predicted to grow by 26.4% annually from 2021 to 2026 [8].

In this paper, we focus on supporting good quality of experience (QoE) for users of interactive applications, e.g., stream gaming, virtual reality education, and autopilot assistance, in MEC. Based on existing work [9]–[12], QoE is becoming an important metric in evaluating the perceived quality of many high-demanding services. Good QoE of users is often the result of joint efforts of multiple quality of service (QoS) metrics, e.g., short delay, high throughput rate, low packet loss rate. For interactive applications considered in this paper specifically, they often rely on up-to-date data instead of data cached in advance. This means ensuring QoE of such services in MEC needs not only short delay but also high data throughput rate. For instance, players of an edge-centric stream game require real-time rendered frames responding to their operations with high frame per second (FPS) and good perceived visual quality [13]. Unfortunately, it is very challenging for a mobile edge network to provide satisfactory delay and data throughput rate. The challenges mainly come from high mobility of end users and volatile edge network conditions in MEC.

First, the movement of users often leads to long delay of services. Users may travel from the coverage area of one wireless Access Point (AP) to another. With fixed virtual service placement, such movement may increase the propagation distance and the number of transmission hops between service hosts and users, thus introducing extra communication delay that degrades QoE of interactive services [1], [2]. Currently, service placement and migration schemes, e.g., [1]–[5], [14]–[20]. have been widely adopted in MEC to solve such delay problems. Nevertheless, the online nature of service migration with unknown future information makes it very hard to yield good solutions.

Moreover, even with judicious service migration to reduce delay, QoE of interactive applications needs further enhancement since their data throughput rates are seriously affected by fluctuating real-time bandwidth in MEC. As shown in our small-scale experiment in Section V, when end users move between APs with their virtual services located in the nearest ENs, the real-time bandwidth, i.e., the maximal data throughput rate can be reached, still fluctuates violently over time and impacts the QoE significantly. Such fluctuation always exists and is affected by user speed, wireless signal interference, network congestion, hardware failure, etc. Since service migration fails to respond to such real-time bandwidth fluctuation, a bandwidth-aware throughput rate adjustment mechanism is necessary to guarantee QoE in MEC. The mechanism should provide sufficient data throughput rate for good QoE of end users. Besides, since edge nodes often have limited resource and energy budgets, the mechanism should carefully plan data throughput rates on each EN to optimize performance and prevent wastage.

Nevertheless, designing a good scheme considering both service migration and throughput rate adjustment for QoE support in MEC is very challenging. First, the online problem

of service migration is often addressed by predictive methods, e.g., [21]–[24]. However, prediction errors normally seen in the dynamic and volatile environment of MEC may have adverse effects on the final results. In addition, bandwidth variation in mobile edge networks is influenced by many unexpected factors and very hard to predict in advance. Thus, predictive methods may not be applicable in online throughput rate adjustment. Besides, service migration may keep moving virtual services around different ENs for lower delay. This dynamic procedure will notably impact the throughput rate adjustment on each EN, making it very difficult to spend the limited budget of each EN for performance optimization. Moreover, according to QoE studies [9]–[12], QoE improvement is often not linearly related to data throughput rate thus making the problem even more complicated.

To address the above-mentioned challenges and support high QoE of interactive virtual services in MEC, we propose a novel QoE enhancement scheme that considers both service migration and throughput rate adjustment. The scheme utilizes a predictive method to dynamically place virtual services for delay minimization, which is robust under prediction errors. Meanwhile, without predicting future bandwidth variation, the scheme adjusts throughput rates of virtual services based on the real-time bandwidth of each user to further improve QoE. The throughput rate adjustment takes ongoing service migration into consideration and judiciously spends the limited budget of each EN. Modeling the QoE improvement by general increasing concave functions, our throughput rate adjustment mechanism is suitable for various virtual services with diverse relations between throughput rate and QoE improvement. In addition to the mechanisms for service placement and throughput rate adjustment, we also implement a small-scale prototype to demonstrate how our scheme addresses remaining practical problems in QoE enhancement such as connection switching among different accessing points without interrupting ongoing interactive services. Our main contributions are summarized as follows.

- We formulate a two-tiered service placement and throughput rate adjustment model for QoE enhancement of interactive virtual services in MEC. It consists of an online service placement problem among ENs for delay minimization and an online throughput rate adjustment problem on each EN for QoE maximization.
- We propose a novel algorithm named SPTA to solve the formulated two problems. SPTA takes advantage of predictable information to achieve service placement close to the offline optimal even under prediction errors. SPTA can tolerate unpredictable bandwidth fluctuation in throughput rate adjustment while preserving a theoretical guarantee to the offline optimal solution.
- We implement a small-scale prototype supporting stream gaming at the edge to present the necessity and feasibility of our scheme. The prototype demonstrates technical details besides the aforementioned algorithmic design as supplements of our QoE enhancement scheme. According to data collected from the prototype, we further conduct simulations based on real-world traces to show that the proposed SPTA algorithm performs better than existing baselines.

The remainder of this paper is organized as follows. Section II presents an overview of related work. Section III proposes the QoE enhancement model while Section IV continues with the design of the STPA algorithm solving the formulated problem and proof of its theoretical guarantees. The small-scale prototype supporting a stream game at the edge is demonstrated in Section V and simulation results are presented in Section VI. Finally, Section VII concludes this paper.

II. Related Work

The high mobility of users and the volatile conditions of edge networks have been major obstacles towards deploying high-demanding virtual services in MEC. Dynamic service placement is proved to be an efficient solution and much work for various objectives has been proposed, e.g., [1]–[5], [14], [15], [17], [19], [25], [26]. When it comes to papers focusing on QoS optimization specifically, Ouyang et al. propose a novel service migration strategy in [4], which minimizes users' delay under long-term budget. Tang et al. [27] design efficient deep reinforcement learning algorithms for minimization of communication delay and computational power. Zhang et al. [28] propose a novel dynamic rendering-module placement strategy to trade off end-to-end delay and cost for virtual reality group gaming in MEC. Deng et al. [29] raise a dynamic throughput maximum algorithm managing the allocation of multiple resources under task and energy queue stability constraints. Despite their contribution to improving QoS, as far as we are concerned, no previous work considers supporting QoE of real-time interactive applications in MEC. Our work fills this gap by considering both service migration among ENs and throughput rate adjustment within each EN.

Recently, quality of experience describing the users' satisfaction of services has drawn more and more attention. As illustrated by Mushtaq et al. in [11], QoE of users is more subjective and comprehensive than QoS in evaluating the performance of a service. When it comes to relations between QoE and QoS metrics, Reichl et al. point out in [9] that QoE often follows Logarithmic laws. Belmudez et al. also show by experiments that perceived visual quality of a video and the received bit rate are concavely related [30]. Xu et al. in [12] propose a unified QoE scoring framework by calculating the QoE score through the QoS metrics including transmission delay, packet loss rate, and throughput rate. In this paper, we propose the QoE enhancement scheme in MEC by considering both delay and data throughput rate and modeling QoE improvement with concave functions. Different from existing data-driven approaches, e.g., [21], [22], [24], which focus mainly on learning-based prediction, our scheme takes advantages of both predictive method and online optimization to yield theoretically guaranteed solutions which are robust even in the worst cases.

III. QoE Enhancement Model

In this section, we formulate the two-tiered QoE enhance model dealing with service migration and throughput rate adjustment.

A. Dynamic Service Placement among ENs

We consider S, a set of interactive virtual services in a mobile edge network, $S = \{1, ..., s, ..., |S|\}$. Each service s corresponds to an end user in the network. I is the set of ENs holding these virtual services, $I = \{1, ..., i, ..., |I|\}$. We denote by $T = \{1, ..., t, ..., |T|\}$ the set of time slots for decision operations, i.e., service migration and throughput rate adjustment. Here, the scale of each time slot is considered as several minutes [28]. The reason is that service migration often takes time and multiple steps [2]. Although throughput rate adjustment can be done within seconds as shown in our prototype, frequent throughput rate adjustment may also bother the users and negatively impact QoE. We utilize the decision variable $x_{s,i}(t)$ to represent whether virtual service s is placed on EN i at time t.

$$x_{s,i}(t) = \begin{cases} 1 & \text{service } s \text{ is on EN } i \text{ at } t \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

It is clear that each service s can be only placed on one EN at each time slot t. Thus,

$$\sum_{i \in I} x_{s,i}(t) = 1, \ \forall s \in S, \forall t \in T. \tag{2}$$

With the decision variable defined, we now construct the objective function considering delay reduction. It is worth noting that reducing communication delay via service migration may introduce extra drawbacks, e.g., computational delay, migration overhead, which will be jointly considered in our objective function.

Communication delay: We define a parameter $d_{s,i}(t)$ as the communication delay between the service host and the end user if service s is placed on EN i at time slot t. This parameter will change over time with the movement of end users. Therefore, at any time t, future $d_{s,i}(\tau)$ is unknown with $\tau > t$. Fortunately, communication delay $d_{s,i}(\tau)$ within an upcoming time window is rather predictable as mentioned in work [1], [2], [28], [31]. Detailed prediction methods involve predicting users' future locations, estimating corresponding propagation distance and transmission hops, which are out of the scope of our paper. With $d_{s,i}(t)$ defined, the total communication delay of all virtual services at each time slot t is summarized as

$$U_1(t) = \sum_{s \in S} \sum_{i \in I} d_{s,i}(t) \cdot x_{s,i}(t).$$

Computational delay: Since ENs in a mobile edge network may have various computational capacities. Thus, diverse computational delay may occur even for the same virtual service when it is executed on different ENs. We denote C_i as the computational capacity of EN i. We also define c_s as the workload of virtual service i to generate the basic data throughput rate. The total computational delay of all virtual services at time t can then be formulated as

$$U_2(t) = \sum_{s \in S} \sum_{i \in I} \frac{c_s}{C_i} \cdot x_{s,i}(t).$$

Collocation interference: Collocating multiple virtual services on one EN may introduce extra delay because of resource contention and network congestion. Thus, collocation interference should be considered during dynamic virtual service placement [28]. Such interference is modeled by "dilation factor" in [32], which is approximately linear. We thus define the collocation interference of all ENs at time slot t as follows. Here, $\alpha_{1,i}$ and $\alpha_{2,i}$ are the parameters of the dilation factor of EN i.

$$U_3(t) = \sum_{i \in I} \left(\alpha_{1,i} \cdot \sum_{s \in S} x_{s,i}(t) + \alpha_{2,i} \right).$$

Service migration overhead: Migrating virtual services between ENs will result in additional delay and expense [1]–[3]. Offline migration often pauses ongoing services and causes severe experience decline. Online migration needs the same service running on both source and target severs, introducing complex synchronization procedures and additional resource occupation. It is essential for service migration strategies to consider the tradeoff between delay and migration overhead. We denote $\beta_{s,i}$ as the overhead of migrating service s on EN i. The total migration overhead at time t is thus formed as

$$U_4(t) = \sum_{s \in S} \sum_{i \in I} \beta_{s,i} \cdot |x_{s,i}(t) - x_{s,i}(t-1)|.$$

Dynamic service placement problem: With four parts of the objective function defined, we formulate the dynamic virtual service placement problem for delay reduction below and name it $P1$.

$$\begin{aligned} \min_{x_{s,i}(t)} \quad & \sum_{t \in T} (U_1(t) + U_2(t) + U_3(t) + U_4(t)) \\ \text{s.t.} \quad & (1), \ (2). \end{aligned} \tag{P1}$$

The formulated $P1$ leads to an online integer optimization problem. The challenge of getting close-to-optimal solution of $P1$ mainly comes from unknown future information $d_{s,i}(t)$. Even if $d_{s,i}(t)$ is predictable, the prediction window is often limited and prediction errors may occur. Moreover, the integral variable and the convex expression of migration overhead also increase the complexity.

It is also worth noting that, the four components of the objective function related to delay may apply different units. The common coordination method in related work is to assign different weights to different components as mentioned in [28]. Since the weights are all constants, the convexness of $P1$ is always preserved. Detail values of weights can be set and tuned according to different service providers' specific policies or preferences. Nevertheless, different settings of weights will not affect the design and theoretical proof of our algorithms.

B. Online Throughput Rate Adjustment on Each EN

As mentioned in Section I, besides service migration, we tend to further enhance QoE of interactive virtual services by adjusting their data throughput rate on each EN. We define a decision variable $y_{s,i}(t)$ as the amount of data throughput rate improved from the basic data throughput rate y_s^{bsc}, which is considered to be supported by any EN. We further denote by $b_{s,i}(t)$ as the bandwidth between service s on EN i and its end user at time t. Here, we assume if a service s is deployed on EN i at t by the aforementioned service migration mechanism for delay reduction, the service is rather close to its user and $b_{s,i}(t) \geq y_s^{bsc}$ always stands. If $y_{s,i}(t) > b_{s,i}(t) - y_s^{bsc}$, the exceeded amount of data cannot be transmitted to the end user and corresponding resources are wasted. It is also worth noting that, if service s is not placed on EN i at time t, i.e., $x_{s,i}(t) = 0$, $y_{s,i}(t) = 0$ as well. Therefore, we have

$$y_{s,i}(t) \in [0, (b_{s,i}(t) - y_s^{bsc}) \cdot x_{s,i}(t)], \forall s \in S, \forall i \in I, \forall t \in T. \quad (3)$$

With the decision variable defined, we then formulate the online throughput rate adjustment problem for QoE improvement as follows.

QoE improvement: As mentioned in Section I, the quality experience of an interactive service is often positively related to data throughput rate but not necessarily in a linear way. Without loss of generality, we utilize $Q_s(y_{s,i}(t))$ to represent the quality improvement of service s at t if data throughput rate is improved by $y_{s,i}(t)$. Based on [9], [30], we require that Q_s is concave, increasing, and continuously differentiable for $y_{s,i}(t)$. The total QoE improvement of services on EN i at each time slot t is then formulated as

$$V_i(t) = \sum_{s \in S} Q_s(y_{s,i}(t)).$$

Throughput rate improvement budget: Enhancing QoE by improving data throughput rate will incur extra costs on each EN. We define $\gamma_{s,i}(t)$ as the cost of improving one unit of data throughput rate for service s on EN i at time t. $\gamma_{s,i}(t)$ also fluctuates over time and is hard to predict due to instant conditions of EN i such as changing electricity price, possible resource contention, and available renewable energy. Since EN often has limited energy and resource budget for services running on it [1], we denote by B_i the cost limitation of EN i which can be utilized to improve data throughput rate of services. Budget B_i can be spent within a budget period T_i^{bgt}. Meaning

$$\sum_{t \in T_i^{bgt}} \sum_{s \in S} \gamma_{s,i}(t) \cdot y_{s,i}(t) \leq B_i. \quad (4)$$

The detailed values of B_i and T_i^{bgt} are often determined by the policy and preference of the service provider of EN i and not necessarily related to T.

Online throughput rate adjustment for QoE improvement: For each EN i, we formulate an online throughput rate adjustment model as follows and name it P2.

$$\max_{y_{s,i}(t)} \quad \sum_{t \in T_i^{bgt}} V_i(t)$$
$$\text{s.t.} \quad (3), (4). \quad \text{(P2)}$$

The formulated $P2$ is an online convex optimization problem. It is rather difficult to solve $P2$ with a theoretical guarantee to the offline optimal solution considering that both $b_{s,i}(t)$ and $\gamma_{s,i}(t)$ are very hard to predict. Proactive methods commonly used are thus not applicable. Moreover, throughput rate adjustment is affected by service migration as shown by Constraint 3. Therefore, an algorithm that considers service migration and throughput rate adjustment coordinately is necessary.

IV. Online Service Placement and Throughput Rate Adjustment Algorithm

In this section, we present the online service placement and throughput rate adjustment (SPTA) algorithm which solves the formulated problems in the previous section coordinately. Theoretical analysis demonstrates that SPTA can take full utilization of predictable data for delay reduction when solving $P1$ and preserve performance guarantee under unpredictable parameters when dealing with $P2$.

A. Design of the SPTA Algorithm

Since service migration takes non-negligible time to proceed, the decision of service placement at each time slot t should be made before t. In this way, at each time slot $t-1$, SPTA first decides $x_{s,i}(t)$ for the upcoming time slot t. For every $t-1$, predictions of end-user locations in the future time interval $[t, t+w-1]$ are available, where w is the prediction window size. Thus, communication delay $d_{s,i}(\tau), \tau \in [t, t+w-1]$ can be predicted based on propagation distance and transmission hops. Detailed prediction methods are considered in work, e.g., [28], [31], and out of the scope of our paper.

With the predicted values, SPTA transforms $P1$ within $[t, t+w-1]$ into problem $\tilde{P}1$ by relaxing the integer variable $x_{s,i}(\tau) \in \{0,1\}$ to $\tilde{x}_{s,i}(\tau) \in [0,1]$. It then utilizes a convex solver, e.g., CVXPY [33], to solve $\tilde{P}1$. Here, we claim the optimal solution $\tilde{x}^*_{s,i}(\tau)$ from the convex solver are all integers, which will be proved in Lemma 1. The SPTA algorithm uses $\tilde{x}^*_{s,i}(t)$ at the specific time t as the service placement decision for the upcoming time slot t. SPTA is efficient in solving $P1$, since the computational complexity at each time slot t is the complexity of solving a convex problem $\tilde{P}1$ within the small window size w using well-developed convex solvers. Detailed complexity is determined by the solver applied, e.g., [33], and the actual running time is often small. For instance, the running time is always less than one second for situations with 16 edge servers, 219 users, and a prediction window 5 in our simulations. Since calculating service placement of t is carried out at $t-1$ and the size of each time slot is often set as several minutes [28], the time of solving $P1$ is often completely covered by the time waiting for the next time slot.

With the service placement decided for time t, SPTA now needs to decide the throughput rate improvement $y_{s,i}(t)$ for virtual services on each EN i at time t. Since bandwidth $b_{s,i}(t)$ is hard to predict, we instead measure it at the beginning of each time slot t. The detailed measurement method is

presented in our prototype design in Section V. The cost of improving throughput rate $\gamma_{s,i}(t)$ is also determined by each EN i at the beginning of t. With both parameters, SPTA is then applied on each EN i to determine $y_{s,i}(t)$. To simplify the expression, we utilize $\tilde{y}_{s,i} = \gamma_{s,i}(t) \cdot y_{s,i}(t)$ to substitute $y_{s,i}(t)$ in the formulated problem $P2$. Then, each element of the objective function becomes $Q_s(\frac{\tilde{y}_{s,i}(t)}{\gamma_{s,i}(t)})$. We further use $\tilde{Q}_{s,t}(\tilde{y}_{s,i}(t))$ to substitute it. Each $\tilde{Q}_{s,t}$ is concave, increasing, continuously differentiable for $\tilde{y}_{s,i}(t)$, and related to the parameter $\gamma_{s,i}(t)$. Since the placement of virtual services $x_{s,i}(t)$ is already known at $t-1$, we utilize $m_{s,i}(t)$ to substitute $\gamma_{s,i}(t) \cdot x_{s,i}(t) \cdot (b_{s,i}(t) - y_s^{bsc})$ and have $\tilde{y}_{s,i}(t) \in [0, m_{s,i}(t)]$. In this way, $P2$ is now denoted as follows.

$$\begin{aligned} \max_{\tilde{y}_{s,i}(t)} \quad & \sum_{t \in T_i^{bgt}} \sum_{s \in S} \tilde{Q}_{s,t}(\tilde{y}_{s,i}(t)) \\ \text{s.t.} \quad & \sum_{t \in T_i^{bgt}} \sum_{s \in S} \tilde{y}_{s,i}(t) \leq B_i, \\ & \tilde{y}_{s,i}(t) \in [0, m_{s,i}(t)], \forall s \in S, \forall t \in T_i^{bgt}. \end{aligned} \tag{P2}$$

The SPTA algorithm further converts solving $P2$ to finding the solution of a minimization problem $P3$ at each time slot t. The problem tends to minimize the quality improvement but with a minimal cost budget $B_i^{min}(t)$ has to be spent at each time slot t.

$$\begin{aligned} \min_{\tilde{y}_{s,i}(t)} \quad & \sum_{s \in S} \tilde{y}_{s,i}(t) \\ \text{s.t.} \quad & \sum_{s \in S} \tilde{Q}_{s,t}(\tilde{y}_{s,i}(t)) \geq B_i^{min}(t), \\ & \tilde{y}_{s,i}(t) \in [0, m_{s,i}(t)], \forall s \in S. \end{aligned} \tag{P3}$$

Here, we define the minimal budget needed to spend as $B_i^{min}(t) = \theta \cdot (g^*(t) - g^*(t-1))$, where θ is a positive parameter determined in the following theoretical analysis and $g^*(t)$ is the optimal value of the following problem $P4$. Here, $t_{i,0}$ is the first time slot in the current budget period T_i^{bgt}.

$$\begin{aligned} \max_{\tilde{y}_{s,i}(\tau)} \quad & \sum_{\tau \in [t_{i,0}, t]} \sum_{s \in S} \tilde{Q}_{s,\tau}(\tilde{y}_{s,i}(\tau)) \\ \text{s.t.} \quad & \sum_{\tau \in [t_{i,0}, t]} \sum_{s \in S} \tilde{y}_{s,i}(\tau) \leq B_i, \\ & \tilde{y}_{s,i}(\tau) \in [0, m_{s,i}(\tau)], \forall s \in S, \forall \tau \in [t_{i,0}, t]. \end{aligned} \tag{P4}$$

Throughput rate adjustment can be done in time at the beginning of each time slot. The reason is that, adjusting the data rate of virtual services can often be carried out within seconds as shown by our prototype, which is much smaller than the common size of a time slot. Furthermore, getting $\tilde{y}_{s,i}(t)$ at each time slot only demands each EN to solve two small-scale convex problems $P3$ and $P4$. At each t, ENs calculate their own $\tilde{y}_{s,i}(t)$ with a limited number of services on each EN in a parallel manner thus reducing the time complexity significantly.

With the above steps, service placement and data throughput rate improvement are all determined by the SPTA algorithm for each t. Details of the algorithm are presented in Algorithm 1. The theoretical analysis of SPTA will then be demonstrated.

Algorithm 1 Service Placement and Throughput Rate Adjustment Algorithm

Input: $d_{s,i}(\tau)$ with $\tau \in [t, t+w-1]$ at each time slot t.
Output: Service placement $x_{s,i}(t)$ and data throughput rate improvement $y_{s,i}(t)$ at each time slot t.

1: **for all** $t \in T$ **do**
2: At $t-1$, do lines 3-5.
3: Relax constraint (1) of $P1$ from $x_{s,i}(t) \in \{0,1\}$ to $\tilde{x}_{s,i}(t) \in [0,1]$ and get $\tilde{P1}$.
4: Use a convex solver to solve problem $\tilde{P1}$ within the prediction window $[t, t+w-1]$ and achieve $\tilde{x}^*_{s,i}(\tau), \tau \in [t, t+w-1]$.
5: Determine virtual service placement at t according to the solution $\tilde{x}^*_{s,i}(t)$.
6: At the beginning of t, do lines 7-15.
7: **for all** $i \in I$ **do**
8: **if** $t = t_{i,0}$, the beginning of a new budget period T_i^{bgt}. **then**
9: $g^*(t-1) = 0$.
10: **end if**
11: Utilize the convex solver to solve the problem $P4$ within the time interval $[t_{i,0}, t]$ and get the optimal value $g^*(t)$.
12: With $g^*(t-1)$ from the previous time slot, calculate $B_i^{min}(t) = \theta \cdot (g^*(t) - g^*(t-1))$.
13: With the minimal budget needed to spend at t, we solve $P3$ with the convex solver.
14: Based on solutions $\tilde{y}_{s,i}(t)$, we calculate $y_{s,i}(t) = \frac{\tilde{y}_{s,i}}{\gamma_{s,i}(t)}$ and determines the data throughput rate improvement.
15: **end for**
16: **end for**

B. Theoretical Analysis of the SPTA Algorithm

In this section, we first demonstrate the feasibility of the SPTA algorithm, i.e., line 4 in Algorithm 1 has integer solutions, by proving Lemma 1. We further prove the theoretical guarantees of the SPTA algorithm in solving $P1$ and $P2$.

Lemma 1. *The optimal offline solution of the relaxed problem $\tilde{P1}$ over a window of size w, is achieved at margins of the variable domain $\tilde{x}_{s,i}(t) \in [0,1]$.*

Proof. For the problem $\tilde{P1}$, Constraint (2) can be transformed to $x_{s,\hat{i}}(t) = 1 - \sum_{i \in I \setminus \hat{i}} \tilde{x}_{s,i}(t), \ \forall s \in S, \forall t \in T$. Then, this constraint can be merged into the objective function through variable substitution. Then, the updated problem has an objective function consists of linear and absolute value expressions of $\tilde{x}_{s,i}(t)$ under the constraint of $\tilde{x}_{s,i}(t) \in [0,1], \forall t \in T, \forall s \in S, \forall i \in I$. Thus, the optimal solution is achieved at margins of variable domain or on the extreme point for each $\tilde{x}_{s,i}(t)$. Moreover, we make the following *claim: when achieving the optimal solution, the extreme points of the absolute value expressions also locate at margins of the variable domain if exist.* Therefore, the optimal solution is achieved with all $\tilde{x}_{s,i}(t) \in \{0,1\}$. In this way, Lemma 1 is proved.

Claim proof: For each absolute value expression $\beta_{s,i} \cdot |x_{s,i}(t) - x_{s,i}(t-1)|$, $x_{s,i}(t) = x_{s,i}(t-1)$ is the condition to reach the extreme point. Then, we can utilize $x_{s,i}(t)$ to represent $x_{s,i}(t-1)$ thus eliminating it. The absolute value expression is then transformed to a linear expression. Since

$x_{s,i}(t)$ with only linear expressions get optimum at its margins and $x_{s,i}(t-1)$ has the same margins as $x_{s,i}(t)$, the claim is proved. □

Based on Lemma 1, we can further prove that, when solving the online integer problem $P1$, our SPTA algorithm works at least as good as the Receding Horizon Control (RHC) method [34] in solving the online convex optimization (OCO) problem $\tilde{P}1$ in the following Theorem 1. Here, RHC is a widely applied algorithm in solving OCO problems with the prediction window of size w. According to [34], the competitive ratio of RHC to the offline optimal solution is $1+\mathcal{O}(\frac{1}{w})$ in one-dimensional setting and $1+\Omega(1)$ in general cases with perfect predictions. In addition, RHC is more resistant to prediction noises compared to other algorithms such as AFHC according to [35]. However, RHC is designed for fractional problems and is often not suitable for problems with integer variables. With RHC introduced, we now prove Theorem 1 as follows.

Theorem 1. *The competitive ratio of SPTA to the offline integer solution in solving $P1$ is smaller or equal to that of RHC to the offline fractional solution in solving $\tilde{P}1$.*

Proof. With Lemma 1 proved and the detailed procedure of RHC, it is clear that the solution of SPTA at line 5 from solving the problem $\tilde{P}1(\tau), \tau \in [t, t+w-1]$ is also the solution of utilizing RHC at each $t-1$ to solve the entire problem $\tilde{P}1$ within the time interval $|T|$. That is $SPTA(P1) = SPTA(\tilde{P}1)) = RHC(\tilde{P}1)$. Since the problem $\tilde{P}1$ is transformed from $P1$ by relaxing $x_{s,i}(t) \in \{0,1\}$ to $\tilde{x}_{s,i}(t) \in [0,1]$, we have $OPT(\tilde{P}1) \leq OPT(P1)$. In this way, $\frac{RHC(\tilde{P}1)}{OPT(P1)} \leq \frac{RHC(\tilde{P}1)}{OPT(\tilde{P}1)}$. Then, we can achieve $\frac{SPTA(P1)}{OPT(P1)} \leq \frac{RHC(\tilde{P}1)}{OPT(\tilde{P}1)}$. Theorem 1 is thus proved. □

Now, we go on to prove that, even with no future information of $b_{s,i}(\tau)$ and $\gamma_{s,i}(\tau), \tau > t$, the SPTA algorithm can solve $P2$ with a theoretical bound to the offline optimal solution. To prove the theoretical performance guarantee, we first need to know the maximal value of θ which makes SPTA feasible for $P2$. However, it is hard to directly find the value of θ^{max}. We thus construct two assisting problems $P5$ and $P6$ as shown below.

$$\begin{aligned} \max_{\tilde{y}_{s,i}(t)} \quad & \sum_{s\in S} \tilde{Q}_{s,t}(\tilde{y}_{s,i}(t)) \\ \text{s.t.} \quad & \sum_{s\in S} \tilde{y}_{s,i}(t) \leq |S| \cdot z, \\ & \tilde{y}_{s,i}(t) \in [0, m_{s,i}(t)], \forall s \in S. \end{aligned} \tag{P5}$$

We denote by $A_{i,t}(z)$ as the optimal value of the problem $P5$ for each EN i at time slot t. With $A_{i,t}(z)$, we further define $P6$ as follows.

$$\begin{aligned} \max_{\tilde{y}_{s,i}(t)} \quad & \sum_{t\in T_i^{bgt}} \sum_{s\in S} A_{i,t}(\tilde{y}_{s,i}(t)) \\ \text{s.t.} \quad & \sum_{t\in T_i^{bgt}} \sum_{s\in S} \tilde{y}_{s,i}(t) \leq B_i, \\ & \tilde{y}_{s,i}(t) \in [0, \frac{\sum_{s\in S} m_{s,i}(t)}{|S|}], \forall s \in S, \forall t \in T_i^{bgt}. \end{aligned} \tag{P6}$$

With $P6$ formulated, we refer to another online optimization algorithm called CR-Pursuit (CRP) [36] to solve it. CRP also contains a parameter similar to θ. In this paper, we always keep the consistency of the two parameters in CRP and SPTA. So they can be both denoted by θ. With $A_{i,t}(z)$ and CRP, we now want to present the following lemmas.

Lemma 2. *Consider applying CRP to P6 and SPTA to P2 with the same θ. If the solution of CRP is feasible for P6, then the solution of SPTA is feasible for P2.*

Lemma 2 is proved using an intermediate solving process between CRP and our SPTA. We omit the details due to the page limitation. Now, we only need to find θ^{max} which makes the solution of CRP feasible for $P6$. To achieve this goal, we need the following two lemmas.

Lemma 3. *When utilizing CRP with parameter θ to solve P6, we get solutions $\tilde{y}_{s,i}^{CRP}(t)$, Which satisfy $\tilde{y}_{s,i}^{CRP}(t) \leq \delta \cdot \frac{A_{i,t}(\tilde{y}_{s,i}^{CRP}(t))}{h_i(t)}$. Here, $\delta = \max_{b_{s,i}(t),\gamma_{s,i}(t)} \frac{k(t)}{\tilde{Q}_{s,t}(\tilde{y}_{s,i}(t))/m_{s,i}(t)}$, $k_i(t) = \lim_{\tilde{y}_{s,i}(t)\to 0^+} \frac{\tilde{Q}_{s,t}(\tilde{y}_{s,i}(t))}{\tilde{y}_{s,i}(t)}$, and $h_i(t) = \lim_{\tilde{y}_{s,i}(t)\to 0^+} \frac{A_{i,t}(\tilde{y}_{s,i}(t))}{\tilde{y}_{s,i}(t)}$.*

Lemma 4. *When utilizing CRP with parameter θ to solve P6, the solutions $\tilde{y}_{s,i}^{CRP}(t)$ also satisfies*

$$\sum_{t\in T_i^{bgt}} \sum_{s\in S} \frac{A_{i,t}(\tilde{y}_{s,i}^{CRP}(t))}{h_i(t)} \leq \theta \cdot B_i \left(1 + (1-\eta^{-\frac{1}{|T_i^{bgt}|-1}})(|T_i^{bgt}|-1)\right).$$

It is worth noting that the constant η equals $\frac{\max\{k_i(t)\}}{\min\{k_i(t)\}}$, with $k_i(t)$ defined in Lemma 3. The details of proving above lemmas are also omitted here due to the page limitation. With Lemma 2, 3, and 4, we can finally prove the theoretical performance guarantee of the SPTA algorithm in solving $P2$ in Theorem 2.

Theorem 2. *The competitive ratio of the SPTA algorithm to the offline optimal solution in solving $P2$ is $1/(\delta + \delta \cdot (1-\eta^{-\frac{1}{|T_i^{bgt}|-1}})(|T_i^{bgt}|-1))$.*

Proof. We denote by $SPTA(\theta)$ as the solution of $P2$ using SPTA with parameter θ. Since SPTA actually solves $P3$, according to the formulation of $P3$ and $P4$, we have $SPTA(\theta) \geq \sum_{t\in T_i^{bgt}} B_i^{min}(t) = \sum_{t\in T_i^{bgt}} \theta \cdot (g^*(t) - g^*(t-1)) = \theta \cdot g^*(|T_i^{bgt}|) = \theta \cdot OPT(P2)$. Thus, the maximal θ which makes $SPTA(\theta)$ feasible is the competitive ratio. Based on Lemma 2, if $CRP(\theta)$ is feasible for $P6$, then $SPTA(\theta)$ is feasible for $P2$.

Based on Lemma 3 and 4, we can conclude that $\sum_{t\in T_i^{bgt}} \sum_{s\in S} \tilde{y}_{s,i}^{CRP}(t) \leq \delta \cdot \sum_{t\in T_i^{bgt}} \sum_{s\in S} \frac{A_{i,t}(\tilde{y}_{s,i}^{CRP}(t))}{h_i(t)}$ and $\sum_{t\in T_i^{bgt}} \sum_{s\in S} \frac{A_{i,t}(\tilde{y}_{s,i}^{CRP}(t))}{h_i(t)} \leq \theta \cdot B_i(1 + (1-\eta^{-\frac{1}{|T_i^{bgt}|-1}})(|T_i^{bgt}|-1))$. Thus, $\sum_{t\in T_i^{bgt}} \sum_{s\in S} \tilde{y}_{s,i}^{CRP}(t) \leq \theta \cdot \delta \cdot B_i(1 + (1-\eta^{-\frac{1}{|T_i^{bgt}|-1}})(|T_i^{bgt}|-1))$. Then, it is

obvious that if $\theta \leq 1/(\delta + \delta \cdot (1 - \eta^{-\frac{1}{|T_i^{bgt}|-1}})(|T_i^{bgt}| - 1))$, $\sum_{t \in T_i^{bgt}} \sum_{s \in S} \tilde{y}_{s,i}^{CRP}(t) \leq B_i$ and $CRP(\theta)$ is feasible. Therefore, $\theta^{max} = 1/(\delta + \delta \cdot (1 - \eta^{-\frac{1}{|T_i^{bgt}|-1}})(|T_i^{bgt}| - 1))$ is the competitive ratio between the result of SPTA and the offline optimal solution. □

Although the expression of θ^{max} is complex and contains $|T_i^{bgt}|$, it is actually larger than the logarithmic bound related to the parameter η in existing work such as [36]. This is because, $\eta^a > 1 + a \cdot \log(\eta)$ for $|a| < 1$, which leads to $1 - \eta^{-\frac{1}{|T_i^{bgt}|-1}} < \frac{1}{|T_i^{bgt}|-1} \log(\eta)$. Therefore, $(1 - \eta^{-\frac{1}{|T_i^{bgt}|-1}})(|T_i^{bgt}| - 1) < \log(\eta)$. After some simple transformations, we have $1/(\delta + \delta \cdot (1 - \eta^{-\frac{1}{|T_i^{bgt}|-1}})(|T_i^{bgt}| - 1)) > \frac{1}{\delta \cdot (\log(\eta)+1)}$.

With proposed SPTA and corresponding theoretical analysis, dynamic service placement and throughput rate adjustment have algorithmic directions. We further design a small-scale prototype in the following section to discuss more details about how to practically support interactive services at mobile edge.

V. Prototype of an Stream Game at Mobile Edge

In this section, we demonstrate a small-scale prototype supporting a stream game at the network edge. It is a software platform deployed on mobile devices, APs and edge servers. We utilize the prototype to demonstrate how our scheme can address the remaining practical problems in QoE enhancement.

A. Major Features of the Prototype

Interactive application at the edge: The prototype supports a simple stream game called "Soaring Bird" developed by Pygame [37]. The main body of the game runs on edge servers while sending rendered frames to its end users through a Wi-Fi network. Players need to control the flying direction of the bird to avoid collisions and the bird will keep speeding up to increase the difficulty.

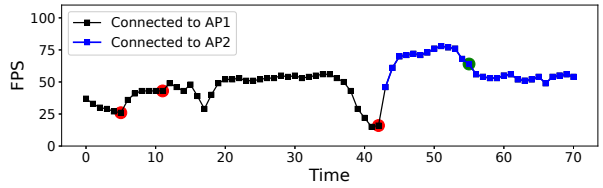
Mobile application supporting game streaming: An android application is realized, which enables mobile users to play the game in the streaming mode. Instead of installing the game, the android app keeps communicating with the edge server, requiring rendered frames and sending the user's operations back to the game. Since the game is highly interactive with timely users' feedback, the application does not cache any rendered frames in advance. The mobile application can be easily modified to fit other interactive games and services.

Connection switching among APs: The android application on the user's device keeps a list of connected APs. The connection list is refreshed every 5 seconds. In every second, each connected AP will send a standard package to the user's device. The application can thus measure communication delay of different APs. If multiple APs are connected, the app will require game frames from a new AP having the lowest delay in a 5-second period. Such a period is called the switching-trigger (ST) period. In this way, when the end user is moving from the coverage of one base station to that of another, the connection switching is carried out automatically. Since service migration strategies should be specifically designed for different interactive services, we leave them to our future work. Currently, APs are wired to the edge server and managed by a router.

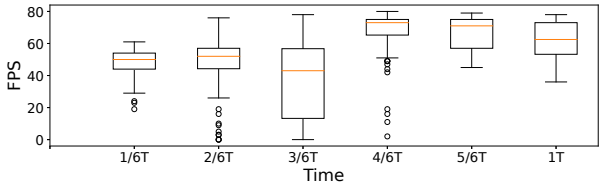
Throughput rate adjustment for better QoE: The android app keeps track of the stream game's FPS and frame capacity. Bandwidth can be calculated by multiplying both parameters. Due to the racing nature of the particular stream game, we suppose that players emphasize more on fluency (FPS) than perceived visual quality. Therefore, if FPS of the game goes below 50 or above 75 for 5 continuous seconds, it will send a throughput rate adjustment request to the edge server running the game. The frame capacity of the game is then downgraded or upgraded based on the request. Thus, the prototype provides guaranteed fluency while improving visual quality as much as possible.

B. Experiment Details and Results

In the experiment, we apply two raspberry Pi 4 with Wi-Fi 5 as wireless APs. They are powered by 5V/3A power banks and connected to the edge service by Cat 6 cables through a router. Considering the power limitation, the two APs are located 40 meters apart. A ThinkPad X1 Carbon laptop with 16G memory works as the edge server holding the stream game. The android application is installed on a OnePlus 8 mobile phone with 8G memory. The end user plays the "soaring bird" on the phone while moving from the coverage of one AP (AP1) to that of the other (AP2). When throughput rate adjustment is triggered, the capacity of each frame is degraded or upgraded by 50%. During the user's movement, we record the game's FPS, the time slots when connection switching happen, and the time slots when frame capacity is upgraded or downgraded. Detailed experiment results are shown in Fig. 1.



(a)



(b)

Fig. 1. Experiment results. (a) FPS change of the stream game when the end user moves from AP1 to AP2. The colors of the line show the connections to different APs. The red and green spots represent frame capacity downgrade and upgrade, respectively. (b) Box plot showing the FPS of the game during different time periods of the movement in multiple experiments.

Fig. 1(a) shows an example of FPS fluctuation of the stream game when end user moves from AP1 to AP2 within a 70 seconds time period. The black line (blue line) represents the user is currently served by AP1 (AP2, respectively). The red

spot (green spot) marks the time slot when the capacity of each frame is downgraded (upgraded, respectively). It is clear that the downgrades make sure the game's fluency (FPS) is often above the preset threshold (50 FPS) unless the FPS drop recovers within 5 seconds. When the user is rather far away from AP1 and approaching AP2 at around the 40^{th} second, FPS goes down sharply and triggers the third quality downgrade. Since the delay from AP2 is now better than that of AP1, the connection switching mechanism is activated after a 5-second period and switches the connection to AP2. FPS of the game after connection switching quickly recovers and is above the second threshold (75 FPS), the upgrade is thus triggered to achieve better visual quality. Further experiments show that the FPS downgrade during connection switching can be further mitigated by choosing a switching-trigger (ST) period smaller than 5 seconds. With smaller ST, the connection switches to the new AP immediately after FPS starts to reduce. We only present results with a 5-second ST period due to the page limitation. Fig. 1(b) utilizes a box plot to show the average performance of 10 individual experiments. Since throughput rate adjustment is triggered only when FPS is below 50 for 5 seconds continuously, it is possible that the median FPS in some time period is below 50. Nevertheless, FPS of the game is above 50 FPS in most cases. Even in the third time period (3/6T in Fig. 1(b)) when the connection switching is most likely to happen, the medium value is above 40 FPS, meaning that the fluency of the game is still maintained at the coverage border of APs.

VI. Performance Evaluation

Based on experimental results of the small-scale prototype, we further conduct real-world trace driven simulations. We compare our QoE enhancement scheme with baselines to show its advantages in both delay reduction and quality experience improvement.

A. Simulation Settings

To simulate end users with relatively high mobility, we utilize the mobility data traceset of taxi cabs in San Francisco, USA [38]. We consider a rectangular area of roughly 30km^2. Since the coverage radius of a 4G base station is around $0.5 \sim 1$km, we suppose there are 4×4 edge nodes with APs serving this area ($|I| = 16$). We utilize mobility data of in total 219 taxis traveling in this area during 8am-9am PDT, 2008-5-29. In each independent simulation, we consider a time interval of 40 minutes with each time slot of 1 minute ($|T| = 40$) and randomly pick 30 taxis as mobile users of interactive applications ($|S| = 30$).

For parameters in $P1$, we suppose the wireless delay to a user within the AP coverage of an EN is randomly distributed in $[0, 2]$. If data is transmitted from a distant EN, each additional transmission hop increases the communication delay by a number uniformly distributed in $[1, 5]$. Thus, $d_{s,i}(t)$ can be calculated using locations of end users. We also suppose the capacity of each EN, C_i, ranges between $[5, 10]$ randomly and the workload of providing basic service throughput rate, c_s varies between $[1, 5]$. The dilation factors of ENs are randomly distributed in $[0, 2]$. The relative values of other parameters in $P1$, e.g., migration cost of virtual services $\beta_{s,i}$, prediction window size w, change during different simulations and will be discussed in each figure, respectively. For problem $P2$, the bandwidth $b_{s,i}(t)$ varies in the range of $[1, 5]$ following the distribution of our experiment observation. The cost of improving data throughput rate, $\gamma_{s,i}(t)$, is randomly generated between $[1, 5]$. Other parameters, e.g., cost budget B_i and budget period T_i^{bgt}, vary in different simulations.

B. Performance of the SPTA Algorithm

Delay reduction of SPTA: We utilize Fig. 2 to show the performance of our SPTA algorithm in reducing delay. We compare SPTA with the offline optimal solution, OPT, and two widely-adopted benchmarks, FOLLOW and FIX (denoted by always migrate strategy and static strategy in [1]). FOLLOW always deploys services on the nearest EN to users. FIX, instead, remains services at the original EN. Total delay (objective value of $P1$) in each subplot of Fig. 2 is the average result of 10 simulations and normalized by the value of OPT in the first column.

Fig. 2(a) shows the performance of SPTA with a perfect prediction window $w = 5$ and baselines when the relative service migration cost increases. When the relative migration cost is 1, migrating a service weighs on average the same as communication delay increased by adding one transmission hop. Although the relative distance between SPTA and OPT increases slightly with larger migration costs, SPTA is still close to the optimal solution and far better than the benchmarks. Fig. 2(b) shows the performance of SPTA with different prediction window sizes with relative migration cost equal to 2. It is clear that the more future information available, the better SPTA performs. In addition, SPTA can achieve delay close enough to the optimum with relatively small window sizes, e.g., $w = 4$. Fig. 2(c) further evaluates SPTA with relative migration cost 2 and $w = 5$ under prediction errors. As mentioned in Section IV, the similar nature of SPTA to the Receding Horizon Control (RHC) method makes it robust to prediction errors. As shown by the figure, SPTA still preserves good performance even when the predicted location of each user is on average 800 meters away from the actual location (the length of the simulation area is around 5km). Considering the diverse mobility of end users, we also apply SPTA and the baselines to users randomly moving in the area with different speeds. Fig. 2(d) shows corresponding results. It is obvious that total delay increases with higher speed. Nevertheless, the relative distance between SPTA and the optimal solution remains close despite the users' speed.

Quality improvement of SPTA: We then present Fig. 3 to evaluate the SPTA algorithm in improving QoE of interactive services on each EN. Each data point counted in the figure is the QoE improvement of services on an EN i over one budget period T_i^{bgt} (objective value of $P2$). The box plots demonstrate the distribution of such data under various conditions where each box contains data points of 16 ENs

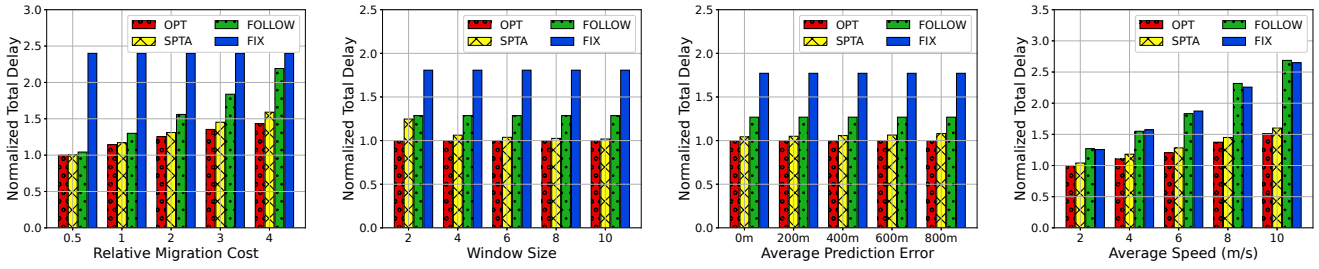


Fig. 2. Performance of SPTA in reducing delay. Values in each subplot are normalized by the OPT value in the first column. (a) Total delay of SPTA and baselines with different service migration costs. (b) Performance of SPTA with different prediction window sizes. (c) Influence of prediction errors to SPTA. (d) Performance of SPTA and baselines under random-walk mobile users with different speeds.

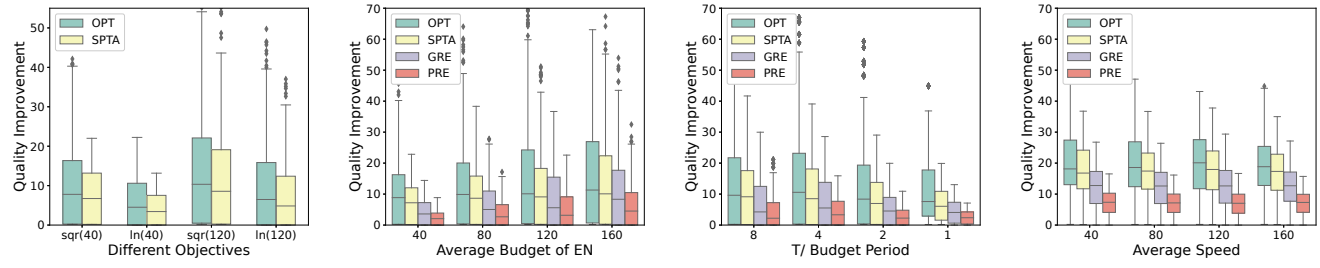


Fig. 3. Performance of SPTA in improving QoE. (a) Performance of SPTA under different objective functions $Q_s(y_{s,i}(t))$. (b) QoE improvement of the algorithms under different cost budgets B_i. (c) Influence of the length of budget period to SPTA. (d) Performance of different algorithms under random-walk movement model with different speeds.

over 10 experiments. In Fig. 3, we compare SPTA with the offline optimal solution (OPT) and two other commonly used baselines, i.e., a greedy algorithm (GRE) and a prediction algorithm (PRE). The GRE algorithm measures $b_{s,i}(t)$ at the beginning of each t as SPTA does. It then adjusts throughput rate of each service s to the maximal bandwidth to improve QoE as much as possible. Since bandwidth is hard to predict, the PRE algorithm predicts the average bandwidth of each service over T_i^{bgt} instead and utilizes the predicted bandwidth as throughput rate. Here, we suppose the prediction is 100% accurate. During the throughput rate adjustment process of all four algorithms, dynamic service placement is carried out by SPTA similar to those in Fig. 2.

Since the relation between QoE and data throughput rate is often modeled by logarithmic or square root expressions [9], [30], we assign the concave, non-decreasing $Q_s(y_{s,i}(t))$ to $\sqrt{y_{s,i}(t)}$ and $ln(y_{s,i}(t)+1)$, respectively in Fig. 3(a). The figure shows the quality improvement of OPT and SPTA with different objective functions under average cost budgets 40 and 120 with budget period $T_i^{bgt} = 10$. It is obvious that the performance of SPTA is close to that of OPT despite diverse object functions. Fig. 3(b) further shows the performance of SPTA and other baselines with a wider range of cost budgets. Here, $Q_s(y_{s,i}(t)) = \sqrt{y_{s,i}(t)}$ and $T_i^{bgt} = 10$. We observe that, when the budget is highly limited, SPTA is way better than baselines GRE and PRE. When the average budget is sufficient, e.g., $B_i = 160$, it is rational to see that distance between SPTA and GRE decreases. This is because such an amount of budget can on average afford all services to have the maximal throughput rate over the entire T_i^{bgt}. Fig. 3(c) illustrates QoE improvement of the algorithms with different length of budget period T_i^{bgt} when average cost budget is 80. Results are normalized to $T_i^{bgt} = 10$, e.g., the quality improvement of $T_i^{bgt} = 40$ is divided by 4, for a better comparison. As demonstrated by Theorem 2, smaller T_i^{bgt} indeed leads to better performance of SPTA. Nevertheless, even when the budget period is large and equal to $|T|$, the median value of SPTA is still close to that of OPT and the overall performance is still much better than the other baselines. We further apply the algorithms under random walk mobility models with different speeds shown in Fig. 3(d). Here, average cost budget is 80 and $T_i^{bgt} = 10$. Fig. 3(d) shows that the QoE improvement of SPTA is relatively robust under different movement patterns and speeds of users.

VII. Conclusion

In this paper, we tackle the challenges of supporting QoE of interactive virtual services in mobile edge computing due to high user mobility and volatile network conditions. We propose an online algorithm SPTA handling both service migration and throughput rate adjustment for delay reduction and QoE improvement. We implement a small-scale prototype providing stream gaming at the edge supplementing technical details to our QoE enhancement scheme. Both theoretical guarantees and extensive simulation results highlight that our scheme is efficient in supporting QoE of interactive applications over mobile edge networks.

Acknowledgments

This research work was supported in part by the U.S. National Science Foundation under grant numbers CCF-1526162, CCF-1717731, CNS-1617698, CNS-1717588, CNS-1730128, CNS-1919752.

REFERENCES

[1] Z. Rejiba, X. Masip-Bruin, and E. Marín-Tordera, "A survey on mobility-induced service migration in the fog, edge, and related computing paradigms," *ACM Computing Surveys (CSUR)*, vol. 52, no. 5, pp. 1–33, 2019.

[2] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A survey on service migration in mobile edge computing," *IEEE Access*, vol. 6, pp. 23 511–23 528, 2018.

[3] A. Machen, S. Wang, K. K. Leung, B. J. Ko, and T. Salonidis, "Live service migration in mobile edge clouds," *IEEE Wireless Communications*, vol. 25, no. 1, pp. 140–147, 2017.

[4] 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*, vol. 36, no. 10, pp. 2333–2345, 2018.

[5] T. Taleb, A. Ksentini, and P. A. Frangoudis, "Follow-me cloud: When cloud services follow mobile users," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 369–382, 2016.

[6] V. Bahl, "Cloud 2020: The emergence of micro datacenter for mobile computing," *Microsoft, Redmond, WA, USA*, 2015.

[7] K. Ha, Y. Abe, Z. Chen, W. Hu, B. Amos, P. Pillai, and M. Satyanarayanan, "Adaptive vm handoff across cloudlets," *Technical Report CMU-CS-15-113*, 2015.

[8] Mordor Intelligence, "Moble edge computing market: Growth, trends, covid-19 impact, and forecast (2021-2026)," Online, https://www.mordorintelligence.com/industry-reports/mobile-edge-computing-market.

[9] P. Reichl, B. Tuffin, and R. Schatz, "Logarithmic laws in service quality perception: where microeconomics meets psychophysics and quality of experience," *Telecommunication Systems*, vol. 52, no. 2, pp. 587–600, 2013.

[10] K. Miller, A.-K. Al-Tamimi, and A. Wolisz, "Qoe-based low-delay live streaming using throughput predictions," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 13, no. 1, pp. 1–24, 2016.

[11] M. S. Mushtaq and A. Mellouk, *Quality of Experience Paradigm in Multimedia Services: Application to OTT Video Streaming and VoIP Services*. Elsevier, 2017.

[12] Z. Xu and A. Zhang, "Network traffic type-based quality of experience (qoe) assessment for universal services," *Applied Sciences*, vol. 9, no. 19, p. 4107, 2019.

[13] Z. Lai, Y. C. Hu, Y. Cui, L. Sun, N. Dai, and H.-S. Lee, "Furion: Engineering high-quality immersive virtual reality on today's mobile devices," *IEEE Transactions on Mobile Computing*, vol. 19, no. 7, pp. 1586–1602, 2019.

[14] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," *Performance Evaluation*, vol. 91, pp. 205–228, 2015.

[15] W. Zhang, Y. Hu, Y. Zhang, and D. Raychaudhuri, "Segue: Quality of service aware edge cloud service migration," in *2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE, 2016, pp. 344–351.

[16] H. Feng, J. Llorca, A. M. Tulino, and A. F. Molisch, "Optimal dynamic cloud network control," *IEEE/ACM Transactions on Networking*, vol. 26, no. 5, pp. 2118–2131, 2018.

[17] R. Bruschi, F. Davoli, P. Lago, C. Lombardo, and J. F. Pajo, "Personal services placement and low-latency migration in edge computing environments," in *2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. IEEE, 2018, pp. 1–6.

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

[19] M. Chen, W. Li, G. Fortino, Y. Hao, L. Hu, and I. Humar, "A dynamic service migration mechanism in edge cognitive computing," *ACM Transactions on Internet Technology (TOIT)*, vol. 19, no. 2, pp. 1–15, 2019.

[20] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Service placement and request routing in mec networks with storage, computation, and communication constraints," *IEEE/ACM Transactions on Networking*, vol. 28, no. 3, pp. 1047–1060, 2020.

[21] S.-R. Yang, Y.-J. Tseng, C.-C. Huang, and W.-C. Lin, "Multi-access edge computing enhanced video streaming: Proof-of-concept implementation and prediction/qoe models," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1888–1902, 2018.

[22] P. Zhou, Y. Xie, B. Niu, L. Pu, Z. Xu, H. Jiang, and H. Huang, "Qoe-aware 3d video streaming via deep reinforcement learning in software defined networking enabled mobile edge computing," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 419–433, 2020.

[23] X. Shang, Z. Liu, and Y. Yang, "Online service function chain placement for cost-effectiveness and network congestion control," *IEEE Transactions on Computers*, 2020.

[24] X. Liu and Y. Deng, "Learning-based prediction, rendering and association optimization for mec-enabled wireless virtual reality (vr) network," *IEEE Transactions on Wireless Communications*, 2021.

[25] Y. Liu, X. Shang, and Y. Yang, "Joint sfc deployment and resource management in heterogeneous edge for latency minimization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 8, pp. 2131–2143, 2021.

[26] X. Shang, Y. Huang, Z. Liu, and Y. Yang, "Reducing the service function chain backup cost over the edge and cloud by a self-adapting scheme," *IEEE Transactions on Mobile Computing*, 2021.

[27] Z. Tang, X. Zhou, F. Zhang, W. Jia, and W. Zhao, "Migration modeling and learning algorithms for containers in fog computing," *IEEE Transactions on Services Computing*, vol. 12, no. 5, pp. 712–725, 2018.

[28] Y. Zhang, L. Jiao, J. Yan, and X. Lin, "Dynamic service placement for virtual reality group gaming on mobile edge cloudlets," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 8, pp. 1881–1897, 2019.

[29] X. Deng, J. Li, L. Shi, Z. Wei, X. Zhou, and J. Yuan, "Wireless powered mobile edge computing: Dynamic resource allocation and throughput maximization," *IEEE Transactions on Mobile Computing*, 2020.

[30] B. Belmudez and S. Moller, "An approach for modeling the effects of video resolution and size on the perceived visual quality," in *2011 IEEE International Symposium on Multimedia*. IEEE, 2011, pp. 464–469.

[31] H. Wang, J. Xie, and T. Han, "A smart service rebuilding scheme across cloudlets via mobile ar frame feature mapping," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[32] 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," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 1, pp. 271–282, 2012.

[33] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[34] M. Lin, Z. Liu, A. Wierman, and L. L. Andrew, "Online algorithms for geographical load balancing," in *2012 international green computing conference (IGCC)*. IEEE, 2012, pp. 1–10.

[35] N. Chen, J. Comden, Z. Liu, A. Gandhi, and A. Wierman, "Using predictions in online optimization: Looking forward with an eye on the past," *ACM SIGMETRICS Performance Evaluation Review*, vol. 44, no. 1, pp. 193–206, 2016.

[36] Q. Lin, H. Yi, J. Pang, M. Chen, A. Wierman, M. Honig, and Y. Xiao, "Competitive online optimization under inventory constraints," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 1, pp. 1–28, 2019.

[37] Pete Shinners, "Python pygame introduction," Online, http://www.pygame.org/docs/tut/PygameIntro.html.

[38] M. Piorkowski, N. Sarafijanovoc-Djukic, and M. Grossglauser, "A Parsimonious Model of Mobile Partitioned Networks with Clustering," in *The First International Conference on COMmunication Systems and NETworkS (COMSNETS)*, January 2009. [Online]. Available: http://www.comsnets.org