2017 IEEE 37th International Conference on Distributed Computing Systems

Online Resource Allocation for Arbitrary User Mobility in Distributed Edge Clouds

Lin Wang
TU Darmstadt
wang@tk.tu-darmstadt.de

Lei Jiao
University of Oregon
jiao@cs.uoregon.edu

Jun Li
University of Oregon
lijun@cs.uoregon.edu

Max Mühlhäuser
TU Darmstadt
max@tk.tu-darmstadt.de

***Abstract*—As clouds move to the network edge to facilitate mobile applications, edge cloud providers are facing new challenges on resource allocation. As users may move and resource prices may vary arbitrarily, resources in edge clouds must be allocated and adapted continuously in order to accommodate such dynamics. In this paper, we first formulate this problem with a comprehensive model that captures the key challenges, then introduce a gap-preserving transformation of the problem, and propose a novel online algorithm that optimally solves a series of subproblems with a carefully designed logarithmic objective, finally producing feasible solutions for edge cloud resource allocation over time. We further prove via rigorous analysis that our online algorithm can provide a parameterized competitive ratio, without requiring any *a priori* knowledge on either the resource price or the user mobility. Through extensive experiments with both real-world and synthetic data, we further confirm the effectiveness of the proposed algorithm. We show that the proposed algorithm achieves near-optimal results with an empirical competitive ratio of about 1.1, reduces the total cost by up to $4\times$ compared to static approaches, and outperforms the online greedy one-shot optimizations by up to 70%.**

I. Introduction

Pushing the cloud frontier to the Internet edge has attracted tremendous interest not only from cloud operators of the IT service/software industry but also from cellular carriers of the telecom industry [1]. Connected by dedicated networks or the Internet, edge clouds may not have a huge amount of resources, but they are in close proximity to end users at various locations such as metropolitan centers, residential neighborhoods, cellular base stations, and WiFi access points [2]–[6]. Serving end users from edge clouds has many advantages such as lower or even bounded delay, reduced wide-area traffic, dedicated security or reliability, *etc.*

A core problem in such an edge cloud computing paradigm is the dynamic resource allocation for the services operated in edge clouds for serving *mobile* users, which often involves multiple challenges: First, resource allocation for individual users may not be a single-location transaction in terms of both cost and performance. When a user accesses the service in an edge cloud, she may eventually have resources allocated for her in multiple nearby and heterogeneous edge clouds as a result of cost optimization or capacity constraints, as long as her service quality can be guaranteed. The user's perceived service quality in terms of the total access delay may include the network delay between the user and the edge cloud she connects to and also between this edge cloud and all the other related edge clouds that hold her workload.

Further, resource allocation is not a one-time operation and needs to be continuously adapted to accommodate user movements, incurring the "adaptation cost" over time. Every user can move arbitrarily in the system, and, from a time-slotted view, a user may connect to one edge cloud in one time slot and switch to another in the next. Each time slot may have its own optimal resource allocation, which may, however, become suboptimal if the adaptation cost during time-slot transition is considered. The adaptation cost refers to hardware wear-and-tear (such as switching on/off a server) or the resource leading time (such as booting up or shutting down a virtual machine) [7], [8]; it can also account for the bandwidth cost in the case of workload migration [9].

Finally, resource allocation needs to be performed on the fly, without any knowledge about future resource price and user location dynamics. It is usually hard or even impossible to predict how the resource price at each edge cloud will vary [8] and how each user will move over time [3]. Without such prediction, it is difficult to make an informed and good decision of resource allocation at each time slot; it is even more difficult to make decisions that are competitive, with guaranteed approximation towards the best decisions that can ever be made when assuming perfect knowledge about the future, which, however, is under our consideration.

Despite the extensive existing research on resource allocation in the cloud context in general [10], [11], only a few have studied online resource allocation in edge clouds, falling short in addressing the three aforementioned challenges. Most of them often assume statistical knowledge about user mobility [12]–[14], or rely on prediction of future costs [15]. In addition, the resource adaptation cost has not been well considered until recently in the cloud in general [7] and in edge clouds in particular [8], [16]; nevertheless, none of them consider user mobility or its influence on resource allocation and adaptation. To the best of our knowledge, we are the first to study the online and user-mobility-driven optimization of the costs of allocation, reconfiguration, service quality, and migration altogether in edge clouds, under unpredictable resource prices and user movements. We make three contributions:

We build a comprehensive model to capture the optimization problem of online resource allocation in edge clouds. Our model focuses on four types of costs, but can capture a wide

1063-6927/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDCS.2017.30

range of different types of costs in general. The first two are static costs associated with every independent time slot: the allocation cost which captures the usage of cloud resources such as severs, virtual machines, and energy, and the service quality cost which captures the total access delay between users and clouds and between clouds themselves. The other two are dynamic costs related to every pair of consecutive time slots: the reconfiguration cost which reflects the changes of the amount of allocated resources, and the migration cost which reflects the movements of users' workloads. The static costs adopt affine models; the reconfiguration cost accounts for adding resources only, as removing resources is often fast and incurs negligible cost; the migration cost accounts for both incoming and outgoing migrations for each edge cloud due to user mobility. We pursue the optimization of the total cost over time while serving each user's workload and respecting the capacity limit of each edge cloud.

We transform our problem and propose an efficient online algorithm, for which, via rigorous competitive analysis, we prove a parameterized competitive ratio. We first apply a transformation to the migration cost in our originally formulated problem to make it easier to work with, while guaranteeing that any online algorithm for the resulting problem with a certain competitive ratio would provide the same competitive ratio for the original problem. Then, our major contribution is the design of an online algorithm based on the regularization technique [17], which decouples our original problem into a series of subproblems that are solvable in each independent time slot, only using the solution obtained for the previous time slot as input. The series of solutions generated in each time slot thus constitute a feasible solution to our original problem. Our online algorithm assumes no priori knowledge on either the user mobility or the resource price, thus allowing for arbitrary dynamics on them. By relaxation and primal-dual properties, we are able to demonstrate that our algorithm always outputs resource allocation decisions for each mobile user in each time slot, with a provable competitive gunrantee even for the worst-case inputs.

We carry out extensive experiments to validate the performance of our proposed online algorithm. We first test it with the real-world data, where we select 15 metro stations from the Rome metro system as the locations of edge clouds and we use the Rome taxi dataset [18] to emulate user mobility. The results show that up to $4\times$ reduction on the total cost can be achieved compared to the static approaches which are typically employed in edge clouds. Moreover, our algorithm produces near-optimal results with an empirical competitive ratio around 1.1 and outperforms the online greedy approach by up to 60%. We also test the algorithm with synthetic data and the results are consistent with that in the real-world data case, proving the effectiveness and generality of our algorithm.

II. Model Formulation

We describe our model for each of the components in an edge cloud system, based on which we formulate the resource allocation problem in this section.

A. Edge Cloud System

We consider a time-slotted system over T time slots that are denoted by the set $\mathcal{T} = \{1, 2, ..., T\}$. We envisage I edge clouds, denoted by the set $\mathcal{I} = \{1, 2, ..., I\}$, in the system. An edge cloud is defined as a pool of virtualized computing resources, which is usually collocated with a cellular base station or a WiFi access point. Each edge cloud is equipped with a certain number of servers and the maximum capacity of an edge cloud i is defined as C_i. All the edge clouds are connected to one another via dedicated carrier networks and the network delay between two edge clouds i and i', i.e., the inter-cloud delay, is given by $d(i, i')$, where we define $d(i, i) = 0, \forall i \in \mathcal{I}$. Each edge cloud is supposed to cover a small geographical area and any area will only receive coverage from a single edge cloud.

B. Users and Workload

We assume a set $\mathcal{J} = \{1, 2, ..., J\}$ of J users with mobile devices that are moving around in the system. In a certain time slot $t \in \mathcal{T}$, a user $j \in \mathcal{J}$ connects to an edge cloud $l_{j,t} \in \mathcal{I}$ that covers the vicinity of the user and accesses the service or offloads computation tasks, incurring a total amount λ_j of workload in the system. For the purpose of cost reduction by taking advantage of the heterogeneity of the edge clouds, the edge cloud operator may distribute the workload from each user to any of the edge clouds given that the service quality is guaranteed and workload redistribution is carried out in the form of workload migration. We denote by $x_{i,j,t}$ the amount of resources that will be allocated for user j in edge cloud i at time t and we enforce $\sum_{i\in\mathcal{I}} x_{i,j,t} \geq \lambda_j$ in order to successfully meet the demand from the user. The delay between the user and the connected edge cloud, i.e., the access delay, is denoted by $d(j, l_{j,t})$. As users are moving, $l_{j,t}$ and $d(j, l_{j,t})$ can vary across time slots. To stay generic, we make no assumption on the mobility pattern of the users, i.e., the movement of each user is arbitrary.

C. Costs

We consider four aspects of costs for the edge cloud system: the operation cost, the service quality cost, the reconfiguration cost, and the migration cost, where the former two fall into the category of static cost that are independent in each time slot while the latter two belong to the category of dynamic cost that is accounted across time slots. These costs are able to represent the most prominent expenditure from the perspective of the cloud operator.

1) The operation cost: This cost refers to server or virtual machine usage, regular hardware or service maintenance, energy consumption, and even carbon emission, all of which can be captured by the following function in general:

$$Cost_{op} = \sum_t \sum_i \sum_j a_{i,t} x_{i,j,t}, \tag{1}$$

where $a_{i,t}$ denotes the "operation price" (i.e., the unit cost) in time slot t. Note that we allow arbitrary variations on the

operation price over time, and such variations can be heterogeneous for different edge clouds due to different hardware or software specifications and energy prices.

2) The reconfiguration cost: This cost is charged in proportion to the amount of workload that has been increased across any two consecutive time slots in each edge cloud. As users move, the operator may redistribute the workload from each user, which results in adapting the amount of resources allocated in each edge cloud. Such adaptation involves powering up a physical server or booting a virtual machine on an active server, which would incur some inevitable delay for preparing the resources and would also bring wear-and-tear to the hardware. We assume that virtual machines are the smallest resource segment in the edge clouds. The reconfiguration price (i.e., the cost for increasing unit resource) is denoted by c_i for edge cloud i. By defining function $(x)^+ = \max\{x, 0\}$ for all $x \in \mathbb{R}$, the total reconfiguration cost is calculated as

$$Cost_{rc} = \sum_t \sum_i c_i \left(\sum_j x_{i,j,t} - \sum_j x_{i,j,t-1} \right)^+, \quad (2)$$

where $\left(\sum_j x_{i,j,t} - \sum_j x_{i,j,t-1}\right)^+$ captures the increase of the workload at edge cloud i when transitioning from time slot $(t-1)$ to time slot t. The cost associated with reducing the amount of resources is omitted as usually that can be completed very fast and the cost is usually negligible.

3) The service quality cost: This cost is defined in proportion to the network delay experienced by each user. From the operator's view, the workload from each user may be distributed to any of the edge clouds in the system for cost optimization. However, the quality of service has to be guaranteed. For a given edge cloud i and a user j, the service quality cost is calculated as the sum of the access delay and the weighted inter-cloud delay, i.e., $d(j, l_{j,t}) + \sum_i x_{i,j,t} d(l_{j,t}, i)/\lambda_j$. Straightforwardly, the total service quality cost is given by

$$Cost_{sq} = \sum_t \sum_j \left(d(j, l_{j,t}) + \sum_i \frac{x_{i,j,t}}{\lambda_j} d(l_{j,t}, i) \right). \quad (3)$$

4) The migration cost: This cost represents the "price" being paid when we migrate some workload from one edge cloud to another due to the bandwidth cost associated with data movement. Such cost is dynamic and is usually counted at both ends of a migration. We denote by b_i^{out} and b_i^{in} the unit migration cost associated with data moving out of and into edge cloud i, respectively. Denoting by $z_{i,t}^{out}$ and $z_{i,t}^{in}$ the amount of workload being migrated out of and into edge cloud i at time t, respectively, we have

$$z_{i,t}^{out} = \sum_j \left(x_{i,j,t-1} - x_{i,j,t}\right)^+, z_{i,t}^{in} = \sum_j \left(x_{i,j,t} - x_{i,j,t-1}\right)^+.$$

The total migration cost $Cost_{mg}$ can be captured by

$$Cost_{mg} = \sum_t \sum_i b_i^{out} z_{i,t}^{out} + b_i^{in} z_{i,t}^{in}. \quad (5)$$

Note that the reconfiguration cost is dictated by the change of the collective workload of all the users at the same edge cloud, while the migration cost is only related to the workload change of every individual user.

D. Problem Formulation

Our goal is to design an online optimizer which takes the user's workload and location as input and continuously decides where to distribute the workload for each user and allocate resources at each edge cloud accordingly such that the total cost is minimized over time. The total cost is the weighted sum of all the aforementioned costs, as given by $Cost_{op} + Cost_{rc} + Cost_{sq} + Cost_{mg}$. For the simplicity of expression, we omit the weights here but we will keep them during our evaluation. In each time slot $t \in \mathcal{T}$, the resource allocation decision $x_{i,j,t-1}$ for the previous time slot $(t-1)$ might become suboptimal due to the variation of $a_{i,t}$ and the change of $l_{j,t}$ as a result of user movement. Therefore, the optimizer will need to redistribute the workload among all the edge clouds in order to maintain optimal cost efficiency. However, the redistribution of workload comes with additional costs, i.e., $Cost_{rc}$ for reconfiguring the edge clouds and $Cost_{mg}$ for migrating the workload. Ideally, the optimizer would make the best tradeoff between the static and the dynamic costs.

Having all the aforementioned notations, the resource allocation problem can further be formalized into the following linear program, denoted as $\mathbb{P}_0$.

$$\begin{aligned}
\min \quad & P_0 = \overbrace{Cost_{op} + Cost_{sq}}^{\text{static}} + \overbrace{Cost_{rc} + Cost_{mg}}^{\text{dynamic}} \\
\text{s.t.} \quad & \sum_i x_{i,j,t} \geq \lambda_j, \ \forall j, \ \forall t, && \text{(6a)} \\
& \sum_j x_{i,j,t} \leq C_i, \ \forall i, \ \forall t, && \text{(6b)} \\
& x_{i,j,t} \geq 0, \ \forall i, \ \forall j, \ \forall t. && \text{(6c)}
\end{aligned}$$

Constraint (6a) ensures sufficient resources are allocated at each edge cloud; constraint (6b) guarantees the capacity limit of each edge cloud. Note that all the costs have time-varying factors corresponding to the uncertainties or dependencies across consecutive time slots. As a result, there is no once-for-all solution for the problem in each separate time slot from an online perspective.

E. Example

We observe that the problem can be easily solved by directly applying a linear program solver (e.g., GLPK) if we are given all the input data including the operation prices and the user mobility patterns in all time slots. However, this is impossible in the online setting, where the input data is revealed step by step over time. Without any a priori knowledge, a nature solution would be greedily adopting the best decision in each independent time slot. However, we show by examples that this online greedy approach is suboptimal in many cases.

We first consider the example in Figure 1(a), where we have a system with two edge clouds and one user is moving

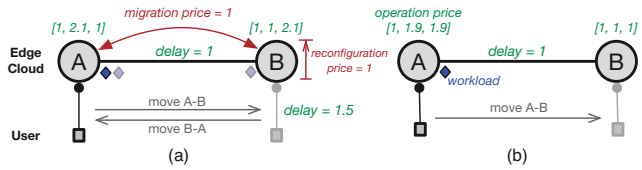

Fig. 1: Simple examples to illustrate the cost calculation and to show the major drawbacks: (a) aggressiveness and (b) conservativeness, of the nature online greedy solution to the problem.

around them. The four types of prices are given in the figure and the user is assumed to have one unit of workload. We consider three time slots where in the first time slot the user is connected to edge cloud A and then, it moves to B in the second time slot and moves back to A in the third time slot. We now show that greedy can be too aggressive. Following the greedy approach, the user workload would be migrated from A to B in the second time slot (as $2.1+1+1.5 > 1+1+1+1.5$) and then will be migrated back to A in the third time slot (as $2.1+1+1.5 > 1+1+1+1.5$). The total cost is then calculated as $(1+1.5)+(1+1+1+1.5)+(1+1+1+1.5) = 11.5$, where both migration and reconfiguration costs are incurred in the last two time slots while the service quality cost is minimized as the workload is following the user all the time. However, with a holistic view on all the time slots, the optimal solution would keep the user workload in A throughout the three time slots, resulting in a total cost of $(1+1.5)+(2.1+1+1.5)+(1+1.5) = 9.6$. The second example in Figure 1(b) shows that greedy can also be too conservative, where greedy would keep the workload all the time at A (as $1.9+1+1.5 < 1+1+1+1.5$) with a total cost of 11.3, while the optimal solution would migrate the workload to B in the second time slot and brings a total cost of only 9.5.

III. Online Algorithm Design

We discuss the design of an online algorithm for resource allocation in edge clouds in this section. Before presenting the algorithm, we first carry out a gap-preserving transformation to simplify the original problem. The proposed algorithm is then based on solving in each time slot a subproblem with a carefully designed logarithmic objective over time, and the solutions for all the subprograms will finally constitute a feasible solution for the original resource allocation problem.

A. Problem Transformation

We notice that the migration cost in $\mathbb{P}_0$ is counted bidirectionally, which is too complicated to handle. To simplify this expression, we first make a transformation on the migration cost in the objective of $\mathbb{P}_0$, from which we generate the following new program $\mathbb{P}_1$:

$$\min \quad P_1 = Cost_{op} + Cost_{rc} + Cost_{sq} + \sum_t \sum_i b_i z_{i,t}^{in}$$

$$\text{s.t.} \quad \text{(6a), (6b), (6c)}$$

where we define $b_i \triangleq b_i^{out} + b_i^{in}$. The intuition behind this transformation is to combine the migration costs counted in both directions into a single direction. The transformation is gap-preserving while improving the tractability of the problem. More specifically, by following similar techniques used in [9], we are able to show that

Lemma 1. *Any r-competitive online algorithm that solves $\mathbb{P}_1$ also yields a r-competitive online algorithm for $\mathbb{P}_0$.*

Proof. For each edge cloud $i \in \mathcal{I}$, we have that the accumulative workload x_i during the whole time period $[1, T]$ is bounded by the capacity C_i of the edge cloud, i.e.,

$$x_i = |\sum_t z_{i,t}^{in} - \sum_t z_{i,t}^{out}| \le C_i.$$

Consequently, the following result can be derived.

$$\begin{aligned} P_0 &= \sum_t \sum_i (b_i^{out} z_{i,t}^{out} + b_i^{in} z_{i,t}^{in}) \\ &= \sum_t \sum_i b_i^{out} \left(\sum_t z_{i,t}^{in} \pm x_i \right) + \sum_i b_i^{in} z_{i,t}^{in} \\ &\ge \sum_t \sum_i (b_i^{out} + b_i^{in}) z_{i,t}^{in} - \sum_i b_i^{out} C_i \\ &\ge P_1 - \sum_i b_i^{out} C_i \end{aligned}$$

Define $\sigma = \sum_i b_i^{out} C_i$ as a constant and we have $P_1 \le P_0 + \sigma$, which indicates that P_1 is upper bounded by P_0 plus a constant σ. This completes the proof as any online algorithm that produces a solution with objective value bounded by r times the optimal of $\mathbb{P}_1$ will also be a solution that is bounded by r times the optimal of $\mathbb{P}_0$ within a constant $r\sigma$. □

The above result provides us the convenience to consider only the problem $\mathbb{P}_1$ hereafter. We also observe that the migration cost can be decomposed for individual users. By introducing auxiliary variables $z_{i,j,t}$ where $z_{i,j,t} = (x_{i,j,t} - x_{i,j,t-1})^+$ and $z_{i,t}^{in} = \sum_j z_{i,j,t}$, we rewrite the objective function of $\mathbb{P}_1$ as follows:

$$P_1 = Cost_{op} + Cost_{rc} + Cost_{sq} + \sum_t \sum_i \sum_j b_i z_{i,j,t}. \quad (9)$$

B. Online Algorithm

We now present the design of an online algorithm for resource allocation. In the online setting, *competitive ratio* is defined as the ratio of the objective of an online algorithm for a given online optimization problem where the input is revealed over time and the optimal objective obtained assuming all the input for the problem is pre-given, i.e., offline optimal. The competitive ratio is used to quantify the quality of the solutions produced by an online algorithm. To simplify the presentation, we denote by $x_{i,t}$ the total amount of resources allocated in edge cloud i at time t, i.e., $x_{i,t} = \sum_j x_{i,j,t}$.

Our online algorithm is based on regularization [17], i.e., solving a series of regularized versions of $\mathbb{P}_1$. At the beginning of each time slot $t \in \mathcal{T}$, observing $l_{j,t}$ and taking $x_{i,j,t-1}^*$ ($x_{i,j,0}^* \triangleq 0$) which is the workload assignment decision made in time slot $(t-1)$ as input, we solve the following problem

$\mathbb{P}_2$ and obtain the resource allocation decisions $x^*_{i,j,t}$ for the current time slot.

$$\begin{aligned}\min \quad & P_2(t) = \sum_i \sum_j a_{i,t} x_{i,j,t} \\ & + \sum_j \left(d(j, l_{j,t}) + \sum_i \frac{x_{i,j,t}}{\lambda_j} d(l_{j,t}, i) \right) \\ & + \sum_i \frac{c_i}{\eta_i} \left((x_{i,t} + \varepsilon_1) \ln \frac{x_{i,t} + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} - x_{i,t} \right) \\ & + \sum_i \sum_j \frac{b_i}{\tau_{i,j}} \left((x_{i,j,t} + \varepsilon_2) \ln \frac{x_{i,j,t} + \varepsilon_2}{x^*_{i,j,t-1} + \varepsilon_2} - x_{i,j,t} \right) \\ \text{s.t.} \quad & \sum_i x_{i,j,t} \geq \lambda_j \ \forall j, \qquad (10a) \\ & \sum_{k \in \mathcal{I} \setminus i} \sum_j x_{k,j,t} \geq \sum_j \lambda_j - C_i, \ \forall i, \qquad (10b) \\ & x_{i,j,t} \geq 0, \ \forall i, \ \forall j, \qquad (10c)\end{aligned}$$

where $\eta_i = \ln(1 + C_i/\varepsilon_1)$, $\tau_{i,j} = \ln(1 + \lambda_j/\varepsilon_2)$, and $\varepsilon_1 > 0$, $\varepsilon_2 > 0$ are parameters. Note that the objective function $P_2(t)$ is convex and the constraints are all linear. As a result, $\mathbb{P}_2$ can be optimally solved by any solver for convex programming. Combining the optimal solution to $\mathbb{P}_2$ in every time slot, denoted by $x^*_{i,j,t}$, we construct an approximate solution to the original problem $\mathbb{P}_1$ by simply following exactly the same resource allocation decisions. We show in the following that the produced solution is feasible to $\mathbb{P}_1$ inherently.

Theorem 1 (**Feasibility**). *The optimal solution $x^*_{i,j,t}$ to $\mathbb{P}_2$ in every time slot constitutes a feasible solution to $\mathbb{P}_1$ over time.*

Proof. The proof is conducted by showing that the optimal solution obtained for $\mathbb{P}_2$ also satisfies the constraints (6a), (6b), and (6c) in $\mathbb{P}_1$. It can be easily checked that (6a) and (6c) are satisfied by the fact that (10a) and (10c) are enforced in the optimal solution to $\mathbb{P}_2$ over time. To prove the validity of the constraint (6b), we first show that $P_2(t)$ increases monotonically with $x_{i,j,t}$ in $[x^*_{i,j,t-1}, +\infty)$ due to that

$$\begin{aligned}\frac{\partial P_2(t)}{\partial x_{i,j,t}} =& a_{i,t} + \frac{d(l_{j,t}, i)}{\lambda_j} + \frac{c_i}{\eta_i} \ln \frac{x_{i,t} + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} \\ & + \frac{b_i}{\tau_{i,j}} \ln \frac{x_{i,j,t} + \varepsilon_2}{x^*_{i,j,t-1} + \varepsilon_2} > 0,\end{aligned}$$

where $x^*_{i,j,t} \geq x^*_{i,j,t-1}$ for any $i \in \mathcal{I}$, $j \in \mathcal{J}$, and $t \in \mathcal{T}$. Since we have $x_{i,j,0} = 0$, as well as $x^*_{i,0} = \sum_j x^*_{i,j,0} = 0$, if we assume the optimal solution satisfies $x^*_{i,j,1} > C_i$ or $x^*_{i,1} > C_i$, we can always find another solution where we let $x^*_{i,j,1} = C_i$ and $x^*_{i,1} = C_i$. The new solution provides a smaller value for the objective $P_2(t)$ and consequently, a contradiction is reached as there exists no solution that performs better than the optimal solution. Similarly, the same result can be applies to any $2 \leq t \leq T$ by induction, which completes the proof. □

IV. Competitive Analysis

In this section, we carry out rigorous analysis on the performance of the proposed online resource allocation algorithm. Particularly, we prove that our algorithm has guaranteed performance by achieving a parameterized competitive ratio. We first present a sketch for the proof, followed by the detailed derivations step by step.

A. Sketch

The competitive analysis will follow three high-level steps: i) We first relax the program $\mathbb{P}_1$ by linearizing the objective function via introducing auxiliary variables $u_{i,t}$ and $v_{i,j,t}$, from which we obtain a linear program $\mathbb{P}_3$. ii) We then derive the dual problem of $\mathbb{P}_3$ to obtain a program $\mathbb{D}$. iii) We finally construct a feasible solution for $\mathbb{D}$ from the optimal solution $x^*_{i,j,t}$ generated by optimally solving $\mathbb{P}_2$. The rationale of adopting the regularization-based approach is that the optimal solution for $\mathbb{P}_2$ shares some common properties with the solution we constructed for $\mathbb{D}$, which can be exploited by deriving the KKT conditions for $\mathbb{P}_2$ and compare them with the constraints in $\mathbb{D}$. Therefore, a connection between the optimal solution to $\mathbb{P}_2$ and the constructed solution to $\mathbb{D}$ is established. More formally, we aim to derive the following inequalities:

$$P_1 \geq P_3 \geq D \geq \frac{1}{r} P_2, \qquad (12)$$

where the first inequality follows by the fact that $\mathbb{P}_3$ is relaxed from $\mathbb{P}_1$, as a result of which it produces the optimal solution no larger than that of $\mathbb{P}_1$. The second inequality is obtained by applying the Weak Duality Theorem, and the third inequality follows by comparing the solutions to $\mathbb{D}$ with that to $\mathbb{P}_2$. Those inequalities directly lead to the result that the proposed online algorithm is r-competitive, where r will be determined later.

B. Auxiliary Programs

Following the above sketch, we first provide the formulation for the relaxed program $\mathbb{P}_3$. As already mentioned, we introduce auxiliary variables $u_{i,t}$ and $v_{i,j,t}$ to reformulate the nonlinear terms in the objective function of $\mathbb{P}_2$. We also enforce lower bounds on the new variables in the constraints. The formal formulation of $\mathbb{P}_3$ is given below.

$$\begin{aligned}\min \quad & P_3(t) = \sum_t \sum_i \sum_j a_{i,t} x_{i,j,t} \\ & + \sum_t \sum_j \left(d(j, l_{j,t}) + \sum_i \frac{x_{i,j,t}}{\lambda_j} d(l_{j,t}, i) \right) \\ & + \sum_t \sum_i c_i u_{i,t} + \sum_t \sum_i \sum_j b_i v_{i,j,t} \\ \text{s.t.} \quad & u_{i,t} \geq \sum_j x_{i,j,t} - \sum_j x_{i,j,t-1}, \ \forall i, \ \forall t, \qquad (13a) \\ & v_{i,j,t} \geq x_{i,j,t} - x_{i,j,t-1}, \ \forall i, \ \forall j, \ \forall t, \qquad (13b) \\ & \sum_{k \in \mathcal{I} \setminus i} \sum_j x_{k,j,t} \geq \left(\sum_j \lambda_j - C_i \right)^+, \ \forall i, \ \forall t, \qquad (13c) \\ & u_{i,t} \geq 0, \ \forall i, \ \forall t, \qquad (13d) \\ & v_{i,j,t} \geq 0, \ \forall i, \ \forall j, \ \forall t, \qquad (13e) \\ & \text{(6a), (6c),}\end{aligned}$$

where function $(x)^+$ can be applied to the right-hand term of (13c) due to the fact that $x_{i,j,t} \geq 0$. We now derive the Lagrange dual of $\mathbb{P}_3$ to generate program $\mathbb{D}$. To this end, we introduce dual variables for each of the constraints in $\mathbb{P}_3$: let $\alpha_{i,t}$, $\beta_{i,j,t}$, $\rho_{i,t}$, and $\theta_{j,t}$ be the dual variables associated with (13a), (13b), (13c), and (6a), respectively. The dual program $\mathbb{D}$ can be derived as follows.

$$\max \quad D = \sum_t \sum_j \lambda_j \theta_{j,t} + \sum_t \sum_i \left(\sum_j \lambda_j - C_i \right)^+ \rho_{i,t}$$

$$\text{s.t.} \quad - a_{i,t} - \frac{d(l_{j,t}, i)}{\lambda_j} + \alpha_{i,t+1} - \alpha_{i,t} + \beta_{i,j,t+1}$$
$$- \beta_{i,j,t} + \sum_{k \in \mathcal{I} \setminus i} \rho_{k,t} + \theta_{j,t} \leq 0, \ \forall i, \ \forall j, \ \forall t, \quad (14a)$$
$$- c_i + \alpha_{i,t} \leq 0, \ \forall i, \ \forall t, \quad (14b)$$
$$- b_i + \beta_{i,j,t} \leq 0, \ \forall i, \ \forall j, \ \forall t \quad (14c)$$
$$\alpha_{i,t} \geq 0, \rho_{i,t} \geq 0 \ \forall i, \ \forall t, \quad (14d)$$
$$\beta_{i,j,t} \geq 0, \theta_{j,t} \geq 0, \ \forall i, \ \forall j, \ \forall t, \quad (14e)$$

On the other hand, we derive the KKT conditions of the program $\mathbb{P}_2$. We associate dual variables $\theta'_{j,t}$, $\rho'_{i,t}$, and $\delta'_{i,j,t}$ to constraints (10a), (10b), and (10c), respectively. Consequently, we have

$$a_{i,t} + \frac{d(l_{j,t}, i)}{\lambda_j} + \frac{c_i}{\eta_i} \ln \frac{x_{i,t} + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} + \frac{b_i}{\tau_{i,j}} \ln \frac{x_{i,j,t} + \varepsilon_2}{x^*_{i,j,t-1} + \varepsilon_2}$$
$$- \theta'_{j,t} - \sum_{k \in \mathcal{I} \setminus i} \rho'_{k,t} - \delta'_{i,j,t} = 0, \ \forall i, \ \forall j, \quad (15a)$$
$$\theta'_{j,t} (\lambda_j - \sum_i x_{i,j,t}) = 0, \ \forall j, \quad (15b)$$
$$\rho'_{i,t} \left(\sum_j \lambda_j - C_i - \sum_{k \in \mathcal{I} \setminus i} \sum_j x_{k,j,t} \right) = 0, \ \forall i \quad (15c)$$
$$- x_{i,j,t} \delta'_{i,j,t} = 0, \ \forall i, \ \forall j, \quad (15d)$$
$$\theta'_{j,t} \geq 0, \rho'_{i,t} \geq 0, \ \forall i, \ \forall j, \quad (15e)$$

where equation (15a) is the optimality condition, while equation (15b), (15c), and (15d) are due to complementary slackness. Using the optimal solutions obtained from solving $\mathbb{P}_3(t)$ in time slot t, i.e., $x^*_{i,j,t}$ and the dual variables $\theta'_{j,t}$ and $\rho'_{i,t}$, we construct a solution S_D for program $\mathbb{D}$ by following the mappings shown below.

$$\alpha_{i,t} = \frac{c_i}{\eta_i} \ln \frac{C_i + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1}, \beta_{i,j,t} = \frac{b_i}{\tau_{i,j}} \ln \frac{C_i + \varepsilon_2}{x^*_{i,j,t-1} + \varepsilon_2}$$
$$\theta_{j,t} = \theta'_{j,t}, \rho_{i,t} = \rho'_{i,t}.$$

We are able to show that

Lemma 2. *The solution S_D is feasible for program $\mathbb{D}$.*

Proof. The proof is conducted by showing that when substituted in $\mathbb{D}$, S_D satisfies all the constraints. To this end, we first derive some preliminaries. For $\alpha_{i,t}$ we have

$$\alpha_{i,t+1} - \alpha_t = \frac{c_i}{\eta_i} \ln \frac{C_i + \varepsilon_1}{x^*_{i,t} + \varepsilon_1} - \frac{c_i}{\eta_i} \ln \frac{C_i + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1}$$
$$= \frac{c_i}{\eta_i} \ln \frac{x^*_{i,t-1} + \varepsilon_1}{x^*_{i,t} + \varepsilon_1}.$$

Similarly, for $\beta_{i,j,t}$ we have

$$\beta_{i,j,t+1} - \beta_{i,j,t} = \frac{b_i}{\tau_{i,j}} \ln \frac{x^*_{i,j,t-1} + \varepsilon_2}{x^*_{i,j,t} + \varepsilon_2}.$$

Based on the above equations, we have

$$\begin{aligned} &- \alpha_{i,t} - \frac{d(l_{j,t}, i)}{\lambda_j} + \alpha_{i,t+1} - \alpha_{i,t} \\ &+ \beta_{i,j,t+1} - \beta_{i,j,t} + \sum_{k \in \mathcal{I} \setminus i} \rho_{k,t} + \theta_{j,t} \\ =& - \alpha_{i,t} - \frac{d(l_{j,t}, i)}{\lambda_j} - \frac{c_i}{\eta_i} \ln \frac{x^*_{i,t} + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} \\ &- \frac{b_i}{\tau_{i,j}} \ln \frac{x^*_{i,j,t} + \varepsilon_2}{x^*_{i,j,t-1} + \varepsilon_2} + \sum_{k \in \mathcal{I} \setminus i} \rho_{k,t} + \theta_{j,t} \leq 0, \end{aligned}$$

where the inequality in the last line follows from equation (15a). The above inequality indicates that the constraint (14a) is satisfied by the solution S_D. Constraint (14b) is satisfied by the following inequality.

$$\begin{aligned} \alpha_{i,t} &= \frac{c_i}{\eta_i} \ln \frac{C_i + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} \\ &= \frac{c_i}{\ln(1 + C_i/\varepsilon_1)} \left(\ln(1 + C_i/\varepsilon) - \ln\left(1 + x^*_{i,t}/\varepsilon_1\right) \right) \\ &= c_i \left(1 - \frac{\ln(1 + x^*_{i,t}/\varepsilon_1)}{\ln(1 + C_i/\varepsilon_1)} \right) \leq c_i, \end{aligned}$$

where the inequality in the last line follows by $x^*_{i,t} \geq 0$ as $x^*_{i,j,t} \geq 0$ given in constraint (10c). Analogously, to prove constraint (14c) we verify that $\beta_{i,j,t} \leq b_i$. Combining with the fact that constraints in (14d) and (14e) follow naturally by definition, we complete the proof. □

C. Competitive Ratio

We now focus on the last inequality in (12) and derive the competitive ratio r. We first show the following preliminary results that will be used later.

Lemma 3. $\sum_t x^*_{i,t} \ln \frac{x^*_{i,t} + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} \geq 0$ *for all* $i \in \mathcal{I}$.

Proof. We separate $x^*_{i,t} \ln(x^*_{i,t} + \varepsilon_1)/(x^*_{i,t-1} + \varepsilon_1)$ into two parts $(x^*_{i,t} + \varepsilon_1) \ln(x^*_{i,t} + \varepsilon_1)(x^*_{i,t-1} + \varepsilon_1)$ and $\varepsilon_1 \ln(x^*_{i,t} + \varepsilon_1)/(x^*_{i,t-1} + \varepsilon_1)$. Then, we show for each of them a lower bound can be obtained, which together will form a lower bound for the original expression. The detailed derivation is as follows.

$$\begin{aligned} &\sum_t (x^*_{i,t} + \varepsilon_1) \ln \frac{x^*_{i,t} + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} + \sum_t \varepsilon_1 \ln \frac{x^*_{i,t} + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} \\ \geq& \left(\sum_t (x^*_{i,t} + \varepsilon_1) \right) \ln \frac{\sum_t (x^*_{i,t} + \varepsilon_1)}{\sum_t (x^*_{i,t-1} + \varepsilon_1)} + \sum_t \varepsilon_1 \ln \frac{x^*_{i,t} + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} \\ \geq& \sum_t (x^*_{i,t} + \varepsilon_1) - \sum_t (x^*_{i,t-1} + \varepsilon_1) + (x^*_{i,0} + \varepsilon_1) \ln \frac{x^*_{i,0} + \varepsilon_1}{x^*_{i,T} + \varepsilon} \\ \geq& x^*_{i,T} - x^*_{i,0} + x^*_{i,0} - x^*_{i,T} = 0, \end{aligned}$$

where the first inequality follows from $\sum_i m_i \ln m_i/n_i \geq (\sum_i m_i) \ln \sum_i m_i / \sum_i n_i$ for any $m, n > 0$, while the second and the last inequalities follow by applying $m \ln m/n \geq m-n$ for any $m, n > 0$. □

Similarly, the same result can be obtained for $x^*_{i,j,t}$ and we have the following lemma.

Lemma 4. *$\sum_t x^*_{i,j,t} \ln \frac{x^*_{i,j,t}+\varepsilon_1}{x^*_{i,j,t-1}+\varepsilon_1} \geq 0$ for all $i \in \mathcal{I}$, $j \in \mathcal{J}$.*

Lemma 5. *The total static cost, i.e., the sum of the operation cost and the service quality cost in $\mathbb{P}_1$, is upper bounded by D when evaluated at $x^*_{i,j,t}$,* i.e., *$Cost_{op} + Cost_{sq} \leq D$.*

Proof. The proof is conducted by applying the equations (15a)-(15d) obtained from the KKT conditions of $\mathbb{P}_2$. Note that we omit the constant part of the service quality cost that does not depend on the resource allocation decision, i.e., $\sum_t \sum_j d(j, l_{j,t})$, as it will appear identically in the original objective function.

$$
\begin{aligned}
&\sum_t \sum_i a_{i,t} x^*_{i,t} + \sum_t \sum_i \sum_j \frac{d(l_{j,t}, i)}{\lambda_j} x^*_{i,j,t} \\
=&\sum_t \sum_i x^*_{i,t} (-\frac{c_i}{\eta_i} \ln \frac{x_{i,t}+\varepsilon_1}{x^*_{i,t-1}+\varepsilon_1} \\
&- \frac{b_i}{\tau_{i,j}} \ln \frac{x_{i,j,t}+\varepsilon_2}{x^*_{i,j,t-1}+\varepsilon_2} + \theta_{j,t} + \sum_{k \in \mathcal{I} \setminus i} \rho_{k,t} + \delta_{i,j,t}) \\
\leq&\sum_t \sum_i \sum_j x_{i,j,t} (\theta_{j,t} + \sum_{k \in \mathcal{I} \setminus i} \rho_{k,t} + \delta_{i,j,t}) \\
=&\sum_t \sum_j \lambda_j \theta_{j,t} + \sum_t \sum_i \left(\sum_j \lambda_j - C_i \right) \rho_{i,t} \\
\leq&\sum_t \sum_j \lambda_j \theta_{j,t} + \sum_t \sum_i \left(\sum_j \lambda_j - C_i \right)^+ \rho_{i,t} = D,
\end{aligned}
$$

where the equation in the first line is obtained by applying (15a), the inequality in the second line follows by Lemma 3 and Lemma 4, the equality in the third line is obtained by applying equations (15b), (15c), and (15d), and the inequality in the last line follows by $(x)^+ \geq x$ according to the definition of function $(x)^+$. □

Lemma 6. *The total dynamic cost, i.e., the sum of the reconfiguration cost and the migration cost in $\mathbb{P}_1$, is upper bounded by constant times of D when evaluated at $x^*_{i,j,t}$, i.e., $Cost_{rc} + Cost_{mg} \leq \gamma |\mathcal{I}| D$ where*

$$
\gamma = \max_{i \in \mathcal{I}} \left\{ (C_i + \varepsilon_1) \ln(1 + \frac{C_i}{\varepsilon_1}), (C_i + \varepsilon_2) \ln(1 + \frac{C_i}{\varepsilon_2}) \right\}.
$$

Proof. We first introduce the following set definitions.

$$
\mathcal{I}^+_t = \{ i \mid i \in \mathcal{I} \wedge x^*_{i,t} > x^*_{i,t-1} \}, \tag{23}
$$

$$
\mathcal{Y}^+_t = \{ (i,j) \mid i \in \mathcal{I} \wedge j \in \mathcal{J} \wedge x^*_{i,j,t} > x^*_{i,j,t-1} \}. \tag{24}
$$

Then, we focus on the sum of the reconfiguration cost and the migration cost, where we show

$$
\begin{aligned}
&\sum_t \sum_i c_i (x^*_{i,t} - x^*_{i,t-1})^+ + \sum_t \sum_i \sum_j (x^*_{i,j,t} - x^*_{i,j,t-1})^+ \\
=&\sum_t \sum_{i \in \mathcal{I}^+_t} c_i (x^*_{i,t} - x^*_{i,t-1}) + \sum_t \sum_{(i,j) \in \mathcal{Y}^+_t} (x^*_{i,j,t} - x^*_{i,j,t-1}) \\
\leq&\max_i \{(C_i + \varepsilon_1)\eta_i\} \sum_t \sum_{i \in \mathcal{I}^+_t} \frac{C_i}{\eta_i} \ln \frac{x^*_{i,t} + \varepsilon_1}{x^*_{i,t-1} + \varepsilon_1} \\
&+ \max_{i,j} \{(C_i + \varepsilon_2)\tau_{i,j}\} \sum_t \sum_{(i,j) \in \mathcal{Y}^+_t} \frac{b_i}{\tau_{i,j}} \ln \frac{x^*_{i,j,t} + \varepsilon_2}{x^*_{i,j,t-1} + \varepsilon_2} \\
\leq&\gamma \sum_t \sum_i \sum_j \left(\theta_{j,t} + \sum_{k \in \mathcal{I} \setminus i} \rho_{k,t} \right) \\
\leq&\gamma \sum_t \sum_i \sum_j \left(\theta_{j,t} + \sum_{k \in \mathcal{I} \setminus i} \rho_{k,t} \right) \lambda_j \\
\leq&\gamma |\mathcal{I}| \sum_t \sum_j \theta_{j,t} \lambda_j + \sum_t \sum_i \left(\sum_j \lambda_j - C_i \right)^+ \rho_{i,t} \\
=&\gamma |\mathcal{I}| D,
\end{aligned}
$$

where the equality in the first line follows by the provided set definitions (23) and (24), the inequality in the second line follows by $m-n \leq m \ln m/n$ for any $m, n > 0$, the inequality in the third line follows by applying equation (15a) and $\delta'_{i,j,t} = 0$ due to equation (15d) and the fact that $x^*_{i,j,t} > x_{i,j,0} = 0$ due to the definition of set $\mathcal{Y}^+_t$, the inequality in the forth line follows due to $\lambda_j > 1$, given that $\lambda_j \in \mathbb{Z}^+$ for all $j \in \mathcal{J}$, and the inequality in the fifth line follows due to equations (15b) and (15c). Therefore, the proof is completed. □

Combining the results in Theorem 1, Lemma 5, and Lemma 6, the following theorem on the competitive ratio can be directly obtained for the proposed online algorithm.

Theorem 2. *The solution obtained by optimally solving $\mathbb{P}_2$ in every time slot will constitute a feasible solution to $\mathbb{P}_0$ over time, with a competitive ratio $r = 1 + \gamma |\mathcal{I}|$.*

Remark. We observe that r is monotonically decreasing with the parameters ε_1 and ε_2 so the competitive ratio can be improved by carefully tuning the values for the parameters. We will further evaluate the empirical competitive ratios of the algorithm with real case experiments in the next section. The lower bounds on the competitive ratio will be explored as a future work.

V. Evaluation

We built a discrete-time simulator in Python to validate the performance of the proposed online resource allocation algorithm. We conducted experiments using both real-world and synthetic data and we report the experimental results in this section. All the measurements were performed on a Linux server equipped with Intel Xeon CPU E5-2687W (3.0GHz)

and 512 GB of RAM. We modeled the linear and convex programs by Pyomo and solved them by invoking IPOPT.

A. Experimental Settings

The major dataset we use for the real-world case is the Roma taxi trajectory traces [18]. We envision an edge cloud system deployed in the center area of Roma city with 15 edge clouds deployed that are located at 15 selected metro stations. The edge clouds in the system will be used by the customers (termed as users hereafter) in the taxis, whose movement records are provided by the traces. The number of users varies from hour to hour but is generally around 300.

Operation price. We generate the operation price following the process: For each edge cloud, we first determine its base operation price reversely proportional to its capacity. This is reasonable due to the economy-of-scale effect on both energy and maintenance. The real-time operation price for each edge cloud follows Gaussian distributions, where we set the the mean value as the base price we just generated and the standard deviation as half of the base price [8].

Network delay. The network delay is used to calculate the service quality cost and for each user it can be partitioned into two parts: the delay between the user and the access point and the weighted average delay between the access point and the recruited edge clouds by the user. The delay in our model is measured by the geographical distance between any two entities based on their GPS locations. The GPS location data for the taxis are provided by the dataset and we collect the GPS locations for the 15 edge clouds (i.e., metro stations) manually on Google Maps. The service quality price is set to be proportional to the measured delay.

Bandwidth price. The migration cost is associated with the bandwidth price and the bandwidth usage during the migration. In our model the bandwidth price is not assumed to be time-varying. However, different edge clouds may connect to the Internet via different Internet providers. We categorize all the edge clouds in three clusters, each of which is subscribed to one of the three Internet providers: Tiscali Italia, Vodafone Italia, and Infostrada-Wind. The per-month flat rate prices averaged for 1Mbps connection are 2.49 euro, 4.86 euro, and 1.25 euro, respectively [19]. We will use this relative ratios between them to set the bandwidth prices for the three categories of edge clouds.

Reconfiguration price. The reconfiguration price is assumed to be static over time and it varies among different edge clouds. We generate the reconfiguration prices following a Gauss distribution with the negative tail cutted.

User workload. To understand the impact of the distribution of user workload on the effectiveness of our algorithm, we use three different distributions: power, uniform, and normal. the power distribution represents highly screwed workload, which can be typically seen in online social network services, e.g., the number of friends of each user on the social network satisfies the power law.

Capacity. The total capacity of the edge clouds is assumed to be slightly larger than the total workload in the system by design. More specifically, we assume that the utilization of the system keeps at the level of 80%. Consequently, the total capacity is set to be 1.25 times the total workload. The capacity will be distributed to all the edge clouds proportionally to the frequency of users being attached to them, i.e., the total number of direct user connection in all the relevant time slots.

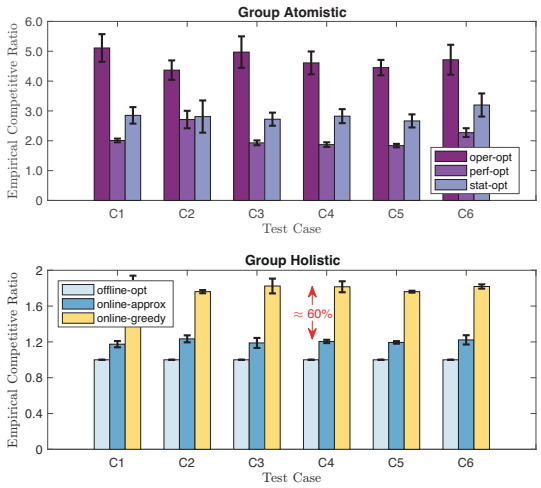

Fig. 2: Comparison on the empirical competitive ratio achieved by the two groups of algorithms with user workloads generated following a power distribution.

B. Measuring the Competitive Ratio

The theoretical analysis has already provided an upper bound on the competitive ratio for the online algorithm. We now validate how the algorithm would perform in reality. We carry out experiments using the above settings and we compare the results of our algorithm with the two groups of algorithms: atomistic and holistic. Atomistic algorithms only consider the static part in the total cost and they include:

- The `perf-opt` algorithm aims at minimizing only the service quality cost $Cost_{sq}$ in every time slot.
- The `oper-opt` algorithm minimizes only the operation cost $Cost_{op}$ in each time slot.
- The `stat-opt` algorithm minimizes the total static cost $Cost_{op} + Cost_{sq}$ in each time slot and ignores the dynamic costs for reconfiguration and migration.

While the algorithms in the holistic group include:

- The `offline-opt` algorithm minimizes P_0 assuming a global view over all the time slots in advance. This is considered impractical and only serves as a baseline.
- The `online-greedy` algorithm directly takes the objective value of $\mathbb{P}_0$ and minimizes P_0 in every time slot. Decision making is based on the outcome of the previous time slot, but considers no future possibilities.

The experimental results are shown in Figure 2. From the Roma taxi traces, we select the data from date Feb 12, 2014 and we choose six hours from 3pm to 8pm as six independent test cases. We set the length of a time slot as one minute and thus each of the test cases consists of 60 time slots. All the values are normalized by the offline optimal objective. The

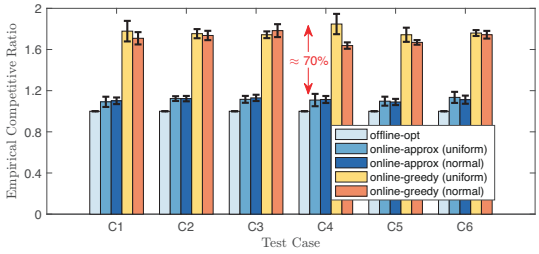

Fig. 3: Comparison on the empirical competitive ratio under uniformly and normally distributed user workloads.

experiments are repeated independently for five times and the plots show the mean values as well as the standard deviations. As we can see from the figure that the algorithms from the atomistic group perform poorly as expected. Among them, the `perf-opt` performs the best, thanks to the reduced frequency of workload migration because of the moderate mobility in the Roma taxi dataset. The `online-greedy` algorithm in the holistic group performs better than any of the atomistic algorithms. However, we still notice a considerable gap to the offline optimal, which is mainly due to the reasons we already discussed at the end of Section II. In contrast, our online algorithm (denoted as `online-approx`) can produce near-optimal results, achieving an improvement of up to 60% compared to the online greedy algorithm.

C. Identifying the Impact of Parameters

Figure 3 illustrates the performance of our algorithm under different workload scenarios, where we generate the user workload using uniform and normal distributions in addition to the power distribution. As we can see that our algorithm preserves similar properties, i.e., producing near-optimal solution and up to 70% improvement compared to `online-greedy`, under any of the workload distributions and our algorithm performs even slightly better under uniform workloads.

Figure 4 shows the impact of the parameters ε_1 and ε_2 on the performance of our algorithm, we set $\varepsilon_1 = \varepsilon_2 = \varepsilon > 0$ and we vary ε from 10^{-3} to 10^3 in a logarithmic scale in all the above test cases. It is interesting to notice that with the increase of ε, the empirical competitive ratio of our algorithm declines slightly at the beginning and then increases to a stable level. We report also in Figure 4 the impact of the ratio between the weight of the dynamic cost and the weight of the static cost (denoted as μ) in the objective by varying its value from 10^{-3} to 10^3 in a logarithmic scale. We observe that when μ is small, i.e., the static cost is negligible, our algorithm can roughly achieve optimal results. When the dynamic cost is dominant, our algorithm can still achieve a stable yet reasonably good competitive ratio.

D. Testing with Synthetic Mobility Patterns

Figure 5 illustrates the experimental results with synthetic mobility data under various numbers of users, which validates the generality of our algorithm. The synthesis mobility data is generated following a random walk process: We assume each user starts from an arbitrary metro station equipped with

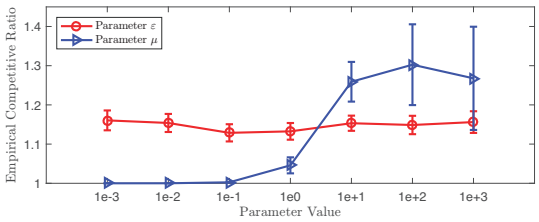

Fig. 4: The impact of the parameter ε and μ on the empirical competitive ratio.

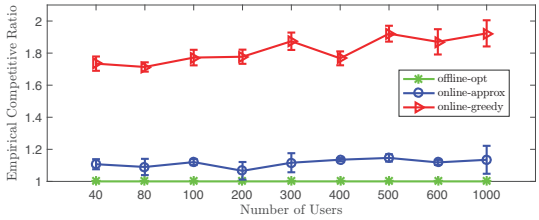

Fig. 5: Comparison on the empirical competitive ratio where user mobility is generated following a random-walk process.

an edge cloud and is traveling with the metro. In each time slot, each user determiners its location for the next time slot by choosing randomly from the neighbor stations with an edge cloud equipped or just staying at the same metro station. Assume in a certain time slot the user is at a location with three neighbors so the probably of moving to any of the three neighbors, as well as of staying at the same location, in the next time slot, would be 25%. Following the above process we generate the movement traces of the users. We vary the number of users from 40 to 1000 and we compare our algorithm with the `offline-opt` and `online-greedy`. We observe that our algorithm performs in a similar way as in the real-world mobility scenario, i.e., the empirical competitive ratio is around 1.1, which is very close to the optimal, while the `online-greedy` has empirical competitive ratios up to 1.8. In addition, our algorithm performs stably regardless of the number of users.

VI. Related Work

The concept of edge computing was initially inspired by the idea of deploying computing facilities at the network edge to enhance the performance of mobile devices [1], [2]. While numerous novel architectures for edge computing [6], [20]–[23] have been proposed, the resource allocation problem in such systems remains as a critical challenge.

A bundle of existing research in this area are on allocating edge cloud resources to computational tasks offloaded from mobile devices. COSMOS [24] is a system that efficiently manages cloud resources for offloading requests to both improve the mobile performance and reduce the provider's monetary cost. Deng et al. [25] study online scheduling policies to maximize data offloading under unpredictable user mobility patterns. Chen et al. [26] focus on game-theoretical mechanisms for offloading decision making in the presence of multiple users, taking into account the energy consumption and the delay. Hou et al. [16] study the reconfiguration in

edge clouds and propose an efficient online algorithm for configuration updating. However, all of them are focused on resource allocation in a single edge cloud environment.

On the other hand, attentions have been paid very recently on resource management in an edge cloud computing system with multiple edge clouds [3], [4], [13], [27]–[29]. Jia et al. [4] study the optimal placement of cloudlets in wireless Metropolitan Area Networks and design an algorithm for user to cloudlet allocation. In a follow-up work, they further propose an efficient algorithm for load balancing among multiple edge clouds [28]. Mukherjee et al. [29] proposed an optimal cloudlet selection strategy to reduce power and latency in multi-cloudlet environments. Tong et al. advocate a hierarchical architecture for edge clouds and develop workload placement algorithms for minimized delay. The most relevant works to ours are from Wang et al. [13] and Urgaonkar et al. [3], where they propose stochastic frameworks for dynamic workload migration based on the Lyapunov optimization technique. However, all of them either require statistics information on user mobility or assumes a Markov chain model for user movement, which is not necessary in our model.

There are also research on workload distribution and resource allocation in geo-distributed data centers [8], [11]. While sharing some common objectives with our problem, they are intrinsically different from edge cloud environments as neither delay sensitivity nor user mobility is considered. In contrast, our research addresses the challenge of the allocation and the continuous adaptation of resources in edge clouds, accommodating arbitrary resource price and user mobility dynamics. Our model captures multiple types of important costs, including static and dynamic ones; our online algorithm, without any knowledge on the future, makes resource allocation decisions on the fly while guaranteeing a parameterized competitive ratio for the worst-case inputs.

VII. Conclusion

In this paper, we studied the online resource allocation problem in edge cloud systems. We identified the major challenges in this problem. We further captured all of them by a comprehensive model, where we incorporated as the optimization objective the costs associated with edge cloud operation, delay, server reconfiguration, as well as service migration. We proposed an online algorithm that could guarantee a parameterized competitive ratio. The effectiveness of the algorithm was also validated by extensive experiments using both real-world and synthetic data.

Acknowledgement

This work has been funded by the German Research Foundation (DFG) Collaborative Research Center (CRC) 1053 – MAKI, and partially by the National Science Foundation under Grant No. 1564348.

References

[1] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos, "Edge analytics in the internet of things," *IEEE Pervasive Computing*, vol. 14, no. 2, pp. 24–31, 2015.

[2] M. Satyanarayanan, P. Bahl, R. Cáceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

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

[4] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, vol. PP, no. 99, pp. 1–1, 2015.

[5] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *INFOCOM*, 2016.

[6] J. Cho, K. Sundaresen, R. Mahindra, J. V. der Merwe, and S. Rangarajan, "ACACIA: context-aware edge computing for continuous interactive applications over mobile networks," in *CoNEXT*, 2016, pp. 1–15.

[7] M. Lin, A. Wierman, L. L. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," *IEEE/ACM Trans. Netw.*, vol. 21, no. 5, pp. 1378–1391, 2013.

[8] L. Jiao, A. Tulino, J. Llorca, Y. Jin, and A. Sala, "Smoothed online resource allocation in multi-tier distributed cloud network," in *IPDPS*, 2016.

[9] N. Buchbinder, N. Jain, and I. Menache, "Online job-migration for reducing the electricity bill in the cloud," in *IFIP Networking*, 2011.

[10] X. Qiu, H. Li, C. Wu, Z. Li, and F. C. Lau, "Cost-minimizing dynamic migration of content distribution services into hybrid clouds," *IEEE Trans. Para. and Dist. Comp.*, vol. 26, no. 12, pp. 3330–3345, 2015.

[11] S. Ren, Y. He, and F. Xu, "Provably-efficient job scheduling for energy and fairness in geographically distributed data centers," in *ICDCS*, 2012.

[12] A. Ksentini, T. Taleb, and M. Chen, "A markov decision process-based service migration procedure for follow me cloud," in *ICC*, 2014.

[13] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. S. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," in *IFIP Networking*, 2015.

[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] S. Wang, R. Urgaonkar, K. Chan, T. He, M. Zafer, and K. K. Leung, "Dynamic service placement for mobile micro-clouds with predicted future costs," *IEEE Trans. Para. and Dist. Comp.*, p. in press, 2016.

[16] I. Hou, T. Zhao, S. Wang, and K. Chan, "Asymptotically optimal algorithm for online reconfiguration of edge-clouds," in *MobiHoc*, 2016.

[17] N. Buchbinder, S. Chen, and J. Naor, "Competitive analysis via regularization," in *SODA*, 2014.

[18] Roma taxi dataset. http://crawdad.org/roma/taxi/20140717/.

[19] Roma network prices. http://www.tempobox.it/en/index.htm.

[20] M. Chen, Y. Hao, Y. Li, C. Lai, and D. Wu, "On the computation offloading at ad hoc cloudlet: architecture and service modes," *IEEE Communications Magazine*, vol. 53, no. 6-Supplement, pp. 18–24, 2015.

[21] A. Bhattcharya and P. De, "Computation offloading from mobile devices: Can edge devices perform better than the cloud?" in *ARMS-CC*, 2016.

[22] R. Stoenescu, V. A. Olteanu, M. Popovici, M. Ahmed, J. Martins, R. Bifulco, F. Manco, F. Huici, G. Smaragdakis, M. Handley, and C. Raiciu, "In-net: in-network processing for the masses," in *EuroSys*, 2015.

[23] M. Jang, H. Lee, K. Schwan, and K. Bhardwaj, "SOUL: an edge-cloud system for mobile applications in a sensor-rich world," in *IEEE/ACM Symposium on Edge Computing*, 2016.

[24] C. Shi, K. Habak, P. Pandurangan, M. H. Ammar, M. Naik, and E. W. Zegura, "COSMOS: computation offloading as a service for mobile devices," in *MobiHoc*, 2014.

[25] H. Deng and I. Hou, "Online scheduling for delayed mobile offloading," in *INFOCOM*, 2015.

[26] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, 2016.

[27] L. Wang, L. Jiao, D. Kliazovich, and B. Pascal, "Reconciling task assignment and scheduling in mobile edge clouds," in *ICNP*, 2016.

[28] M. Jia, W. Liang, Z. Xu, and M. Huang, "Cloudlet load balancing in wireless metropolitan area networks," in *INFOCOM*, 2016.

[29] A. Mukherjee, D. De, and D. G. Roy, "A power and latency aware cloudlet selection strategy for multi-cloudlet environment," *IEEE Transactions on Cloud Computing*, vol. PP, no. 99, pp. 1–1, 2016.