Dynamic Distributed Edge Resource Provisioning via Online Learning across Timescales

Wencong You[1], Lei Jiao[1*], Sourav Bhattacharya[2], Yuan Zhang[3]
[1]University of Oregon, Eugene, OR, USA [2]Samsung AI Center, Cambridge, UK
[3]Communication University of China, Beijing, China

***Abstract*—The strategic management of distributed resources of mobile edge computing networks often requires managing different system components over different timescales. In this paper, we formulate a nonlinear mixed-integer program to capture the online optimization of the edge network's long-term cost, where we distribute workload more frequently on the fast timescale and provision resources less frequently on the slow timescale. We design a novel online learning framework consisting of three algorithms to make fast-timescale and slow-timescale fractional decisions, respectively, and round such decisions into integers. Our algorithms run in polynomial time in an online manner, jointly solving the original NP-hard problem that can contain arbitrary and unpredictable inputs. Via rigorous formal analysis, we prove a parameterized-constant competitive ratio as the performance guarantee for our approach. We conduct extensive evaluations with real-world data and confirm our approach's superiority over existing practices and state-of-the-arts.**

I. Introduction

Telecom carriers, network operators, and service providers today are increasingly interested in building and deploying computing resources at the network edge in closer proximity to end users. Doing so brings multiple benefits, such as shorter communication delay, less core-network traffic, and better data localization and user privacy, just to name a few. The cornerstone to realizing all such potentials lies in the strategic management of distributed edge resources, often in the form of micro clouds or "cloudlets", which can be co-located with cellular base stations and local communities [1], connecting to one another via high-speed wired networks.

While adapting to the dynamics of operational expenses (e.g., electricity cost) and service workload (e.g., user requests) has been a major task in provisioning edge resources, a critical factor that has been largely overlooked is managing different components of the system over different *timescales*. Consider an edge cloudlet network in Fig. 1 as an example. Users' (e.g., HTTP) requests, often small in size and varying in volume (e.g., due to user movements), can be easily moved around and distributed at different cloudlets for execution [2]–[5]; in contrast, the resources (e.g., virtual machines) [6] at each cloudlet cannot be adjusted (e.g., booted or reconfigured) fast enough to catch up with the workload variation and distribution—booting new virtual machines and/or preparing the runtime environment often takes time. In this case, the workload and the resources to serve such workload need to be

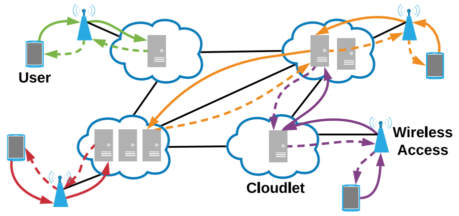

Fig. 1: Edge network with workload distribution

managed on two different timescales, i.e., the workload could be distributed more frequently on a fast timescale and the resources could be adjusted less frequently on a slow timescale. Compared to a single-timescale scheme [7]–[11] that forces the synchronization of management, multi-timescale methods [5], [12]–[15] can lead to greater management flexibility with better performance and more cost savings.

However, making optimal decisions on the fly to provision resources and distribute workload within the cloudlet network on two timescales is non-trivial, facing multiple challenges. First, resources provisioned in one time period need to accommodate multiple workload distribution decisions, before the next time period when the resource decision can be updated [16]. As such workload decisions are only made after the resource decision, it is difficult to provision resources beforehand without knowing how the workload will be distributed (and without knowing the workload itself, because the workload may fluctuate unpredictably) [5]. Second, the difficulty for making resource decisions escalates when the *switching cost* is involved [9], incurred by the change between two consecutive decisions which often leads to system instability and performance degradation. Thus, in contrast to encouraging the resources to follow the fluctuating workload to save operational expense, the switching cost encourages stable, constant resource decisions over time. It is challenging to strike this trade-off wisely, as any resource decision made at any moment can potentially impact the switching cost between itself and the next resource decision yet to be made. Third, other constraints in the cloudlet network adds extra complexities, such as the time-varying operational price [12], the heterogeneous network delay between cloudlets [10], and the limited capacity of each cloudlet [4], which all need to be taken into account when pursuing the optimal cost over time.

Existing research works fall insufficient for our problem. The vast majority are single-timescale resource management [4], [8]; some have switching cost [9], [10], [17] or learning

*The corresponding author is Lei Jiao (jiao@cs.uoregon.edu).

978-1-7281-6630-8/20/$31.00 ©2020 IEEE

of uncertainties [7], [11], but are inapplicable to our two-timescale setting. Those two-timescale approaches are often based on different assumptions, such as time-averages [12]–[14], stochastic knowledge [15] or predictions [16] of inputs, which do not match our case. To our best knowledge, this is the first paper to jointly address all the aforementioned challenges.

In this paper, firstly, we formulate the problem of resource provisioning and workload distribution at edge as an online mixed-integer program, featuring the long-run cumulative minimization of the workload migration cost, the operational expense of the resources, the switching cost of changing resource decisions, and the performance degradation due to the possible mismatch between the slow-timescale resources and the fast-timescale workload. Our formulation generally captures any arbitrary dynamics of the inputs, and our problem is provably NP-hard even in an offline setting.

Next, we design a novel online learning algorithmic framework which consists of three polynomial-time algorithms. Our fast-timescale online algorithm observes the dynamic workload in each *time slot* and optimizes the workload distribution in real time. Our slow-timescale online algorithm, based on existing workload distribution of each current *time frame*, transforms the original problem, decomposes it into a series of convex sub-problems corresponding to each time frame, and solves such sub-problems to obtain the fractional solutions to the original problem. Our rounding algorithm converts the fractional decisions into integers (i.e., the number of virtual machines) by iteratively rounding pairs of fractions to compensate each other, and violates no constraints [18].

Our algorithmic framework features two types of decomposition. On one hand, we decompose the two-timescale joint problem into a fast-timescale sub-problem and a slow-timescale sub-problem which are solvable separately, based on our key observation that the optimal workload distribution in the joint problem and that in the fast-timescale sub-problem make the same constraints tight. On the other hand, we further decompose the slow-timescale sub-problem into a series of convex sub-problems solvable without worrying about incurring excessive switching cost due to not knowing the future inputs, via a learning approximation transformation that we develop and the regularization technique [19].

Further, we perform rigorous formal analysis to quantify the overall performance of our approach. Particularly, via multiple theorems, we prove the *competitive ratio* of our approach as a function of the key parameters of our problem. We highlight that, unlike classic and standard online learning algorithms whose performance is often characterized by the *regret* against the optimal *invariant* offline decisions [7], [11], we use the competitive ratio to compare our algorithms against the best possible *dynamic* offline decisions, which makes more sense in our setting that naturally allows dynamic decisions.

Finally, we conduct extensive evaluations using real-world data in realistic settings to validate the practical performance of our proposed approach. We highlight the following findings: (1) Our two-timescale approach outperforms one-timescale algorithms significantly, and remains better even when the one-timescale algorithms are adapted to the two-timescale settings; (2) Our approach can achieve more total cost savings as more frequent fast-timescale decisions are allowed for each slow-timescale decision; (3) Our approach becomes more advantageous as the switching cost and the learning loss becomes more important in the system; (4) Our approach executes efficiently, and scales well as the problem size increases.

II. Model Formulation

A. Settings and Notations

Cloudlets and Timescales. We consider an edge network that consists of a set of geographically distributed cloudlets J. Without loss of generality, we assume that each cloudlet is co-located with a different cellular tower [1], and different cloudlets connect to one another via wireline backhaul networks. Users access their nearest cloudlet, to which they submit their requests for processing. We study the system over a time horizon that consists of a number of consecutive *time frames*, denoted as T, where we use $t \in T$ to index a single time frame. Each time frame t further consists of a number of consecutive *time slots*, denoted as S_t, where we use $\tau \in S_t$ to index a single time slot. The time frames and the time slots are of consistent durations, and correspond to the slow timescale and the fast timescale, respectively.

Workload and Distribution. We use $\lambda_{jt\tau}$ to denote the aggregated workload (e.g., user requests) received at cloudlet j at time slot τ of time frame t. We make no assumption on how $\lambda_{jt\tau}$ varies at different cloudlets or fluctuates over time. We allow moving workload across cloudlets, and thus the workload received at one cloudlet may be processed at a different cloudlet with the results routed back to the corresponding users. We use $a_{ijt\tau}$ to denote the price (i.e., unit cost) of routing the workload from cloudlet i to cloudlet j at time slot τ of time frame t. Such routing cost can represent the bandwidth consumption, the network delay, etc.

Resources and Provisioning. It is common to adopt virtualization techniques in cloudlets, and without loss of generality, we consider a virtual machine, or VM, as the unit of resource provisioning. We denote by e_j the amount of workload that can be processed by virtual machines at cloudlet j, and by C_j the capacity of cloudlet j in terms of the total number of virtual machines that can be hosted. We denote by b_{jt} the operational price (i.e., unit operational cost) of a virtual machine at cloudlet j at time frame t. Such operational cost may refer to electricity price, averaged software maintenance/license fee, etc. We further denote by c_j the unit switching cost at cloudlet j for changing resource decisions across consecutive time frames. Such switching cost may represent the performance penalty related to system instability due to decision changes.

Two-Timescale Decisions. We use $x_{ijt\tau}$ as the decision variable that represents the amount of workload moved from cloudlet i to cloudlet j at time slot τ of time frame t. We also use y_{jt} as the decision variable that represents the number of virtual machines provisioned at cloudlet j at time frame t. Note that we are modeling the decisions over two timescales, where we make each workload distribution decision along the fast

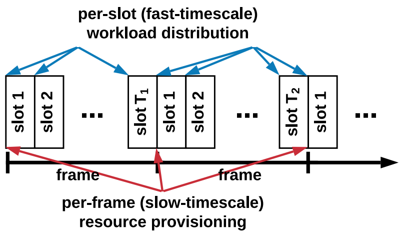
per-slot (fast-timescale) workload distribution
slot 1
slot 2
...
slot T₁
slot 1
slot 2
...
slot T₂
slot 1
...
frame
frame
per-frame (slow-timescale) resource provisioning

Fig. 2: Decisions of workload distribution and resource provisioning

timescale and make each resource provisioning decision along the slow timescale. We also treat making a decision at a time frame as making that decision at the first time slot of that time frame; one can also treat it as making that decision at the last time slot of the previous frame, which is essentially indifferent. This is illustrated in Fig. 2. As we provision the resources before distributing the workload, there might be performance degradation, or what we call as *learning loss*, if insufficient resources are provisioned. Here, we use d_j to denote the unit learning loss incurred at cloudlet j.

We summarize all the notations in Table I.

TABLE I: Notations

Inputs	
J	Set of cloudlets
T	Set of time frames
S_t	Set of time slots of time frame t
$\lambda_{jt\tau}$	Workload at cloudlet j at time slot τ of time frame t
$a_{ijt\tau}$	Delay from cloudlet i to j at time slot τ of time frame t
b_{jt}	Operational price of VMs at cloudlet j at time frame t
c_j	Unit switching cost of VMs at cloudlet j
d_j	Unit learning loss at cloudlet j
e_j	Workload that can be processed by one VM at cloudlet j
C_j	Capacity of cloudlet j
Decision Variables	
$x_{ijt\tau}$	Amount of workload distributed from cloudlet i to j at time slot τ of time frame t
y_{jt}	Number of VMs to provision at cloudlet j at time frame t

B. Problem Formulation

We formulate the long-term total cost minimization problem $\mathbf{P_L}$ as below:

$$\begin{aligned}
\min \quad & P_L = \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} x_{ijt\tau} + \sum_t \sum_j b_{jt+1} y_{jt} \\
& + \textstyle\sum_t \sum_j c_j (y_{jt} - y_{jt-1})^+ \\
& + \textstyle\sum_t \sum_\tau \sum_j d_j (\sum_i x_{ijt+1\tau} - e_j y_{jt})^+ \\
\text{s.t.} \quad & y_{jt} \le C_j, \quad \forall j, \forall t, & \text{(1a)} \\
& \textstyle\sum_j x_{ijt\tau} \ge \lambda_{it\tau}, \quad \forall i, \forall t, \forall \tau \in S_t, & \text{(1b)} \\
& x_{ijt\tau} \ge 0, \quad \forall i, \forall j, \forall t, \forall \tau \in S_t, & \text{(1c)} \\
& y_{jt} \in \{0, 1, 2, \cdots\}, \quad \forall j, \forall t. & \text{(1d)}
\end{aligned}$$

The objective consists of four terms: total workload routing cost $\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} x_{ijt\tau}$, total resource operational cost $\sum_t \sum_j b_{jt+1} y_{jt}$, total resource switching cost $\sum_t \sum_j c_j (y_{jt} - y_{jt-1})^+$, and total learning loss $\sum_t \sum_\tau \sum_j d_j (\sum_i x_{ijt+1\tau} - e_j y_{jt})^+$, where $(\cdot)^+ \overset{\text{def}}{=} \max\{\cdot, 0\}$. Each of the four terms can be associated to a different weight, which is omitted for the ease of the presentation. Constraint (1a) enforces that resources provisioned at each cloudlet is no greater than the capacity. Constraint (1b) ensures that at every cloudlet all the aggregated workload is fully distributed for processing. Constraints (1c) and (1d) ensure that all decision variables are non-negative, and resource decisions (e.g., number of virtual machines) always take integer values.

Challenges. To solve our problem in an online manner, we need to overcome the following challenges:

- *Uncertainty*. At t, it is non-trivial to determine y_{jt}, as it depends on unknown information: (1) for the value of $b_{jt+1} y_{jt}$, b_{jt+1} is unknown at t and will only be known at $t+1$; (2) for the value of $(y_{jt+1} - y_{jt})^+$, y_{jt+1} will only be determined and known at $t+1$ as well; and (3) for the value of $(\sum_i x_{ijt+1\tau} - e_j y_{jt})^+$, $x_{ijt+1\tau}$ will also only be determined and known at τ of $t+1$.
- *Intractability*. At t, determining y_{jt} as integers is NP-hard. Our problem is essentially NP-hard even in the offline setting (i.e., all the inputs are known at once beforehand)—by introducing auxiliary variables to replace the function of $(\cdot)^+$ as will be shown later, our problem can be proved to contain the *covering problem*, which is NP-hard, as a special case.

III. Two-timescale Online Learning Framework

We propose a novel polynomial-time online algorithmic framework to solve the resource provisioning optimization problem with provable performance guarantees. First, we propose an upper bound for the problem's objective to overcome the challenge of *uncertainty* using a transformation based only on current inputs and decisions. Second, we decompose our transformed problem and design two fractional online algorithms for the two timescales, respectively, to address the switching cost and a rounding algorithm to convert the fractional decisions to integers, in order to overcome the challenge of *intractability*.

A. Problem Transformation

We transform the optimization objective of

$$\begin{aligned}
P_L = & \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} x_{ijt\tau} + \sum_t \sum_j c_j (y_{jt} - y_{jt-1})^+ \\
& + \textstyle\sum_t \sum_j b_{jt+1} y_{jt} + \sum_t \sum_\tau \sum_j d_j (\sum_i x_{ijt+1\tau} - e_j y_{jt})^+
\end{aligned}$$

to

$$\begin{aligned}
P_{NL} = & \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} x_{ijt\tau} + \sum_t \sum_j \Delta_{jt} (y_{jt} - y_{jt-1})^+ \\
& + \textstyle\sum_t \sum_j 2 b_{jt} y_{jt} + \sum_t \sum_\tau \sum_j d_j (\sum_i x_{ijt\tau} - e_j y_{jt})^+,
\end{aligned}$$

where $\Delta_{jt} = c_j + b_{jt} + |S_t| d_j e_j$. Details of this transformation are in Section IV. The new objective is actually an upper bound of the original one. We highlight that, unlike the original objective, in the new objective, b_{jt} and $x_{ijt\tau}$ are in the same time frame as y_{jt}. That is, no information at $t+1$ is needed. We will hereafter optimize the new objective, and then compare the value of new objective with our proposed online algorithms against the offline optimum of the original objective.

Furthermore, we introduce some auxiliary variables: w_{jt} to replace $(y_{jt} - y_{jt-1})^+$, and $v_{jt\tau}$ to replace $(\sum_i x_{ijt\tau} - e_j y_{jt})^+$. We thus rewrite $\mathbf{P_{NL}}$:

$$\min \quad P_{NL} = \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} x_{ijt\tau} + \sum_t \sum_j \Delta_{jt} w_{jt}$$

$$
\begin{aligned}
& +\textstyle\sum_t \sum_j 2b_{jt}y_{jt} + \sum_t \sum_\tau \sum_j d_j v_{jt\tau} \\
\text{s.t.}\quad & \textstyle\sum_i \lambda_{it\tau} - \sum_j e_j C_j - \sum_j v_{jt\tau} \le 0, \forall t, \forall \tau \in S_t, && (4a)\\
& \lambda_{it\tau} - \textstyle\sum_j x_{ijt\tau} \le 0, \quad \forall i, \forall t, \forall \tau \in S_t, && (4b)\\
& y_{jt} - y_{jt-1} - w_{jt} \le 0, \quad \forall j, \forall t, && (4c)\\
& \textstyle\sum_i x_{ijt\tau} - e_j y_{jt} - v_{jt\tau} \le 0, \quad \forall j, \forall t, \forall \tau \in S_t, && (4d)\\
& \text{all variables} \ge 0, y_{jt} \in \{0, 1, \cdots\}, \forall i, \forall j, \forall t, \forall \tau \in S_t. && (4e)
\end{aligned}
$$

Consequently, Constraints (4c) and (4d) are added; Constraint (4e) is updated. Meanwhile, Constraint (4a) is rewritten from (1a) and (1b), combined with (4d). This involves an equivalent transformation which does not change the problem [20], and we adopt this transformation to facilitate our performance analysis later.

B. Decomposition across Timescales

We decompose $\mathbf{P_{NL}}$ which involves two timescales via solving for $\{x_{ijt\tau}, \forall i, j, \forall \tau \in S_t, \forall t\}$ using a fast-timescale algorithm, and solving for $\{y_{jt}, \forall j, \forall t\}$ using a slow-timescale algorithm and a rounding algorithm. The performance guarantees of these algorithms are formally analyzed in Section IV.

Here we introduce additional notations for the variables for brevity. We consider y_{jt} as an example: we use $\hat{y}_{jt}$ and $\bar{y}_{jt}$ to denote the fractional solution and its rounded value for cloudlet j at time frame t, respectively; we also use $\{\hat{\mathbf{y}}_\mathbf{t}\}$ and $\{\bar{\mathbf{y}}_\mathbf{t}\}$ to denote the set of $\hat{y}_{jt}, \forall j$ and the set of $\bar{y}_{jt}, \forall j$ at t, respectively. Similar notations also apply to other variables.

Algorithm 1: Fast-Timescale Online Algorithm, $\forall \tau \in S_t, \forall t$

```
Solve the below problem $\mathbf{P_{NL}(x)}$ to get the optimal $\{\hat{\mathbf{x}}_\mathbf{t}\}$:
    min    $P_{NL}(x) = \sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} x_{ijt\tau}$
    s.t.   $\lambda_{it\tau} - \sum_j x_{ijt\tau} \le 0$        $\forall i$
           $\sum_i x_{ijt\tau} - e_j C_j \le 0$        $\forall j$
           $x_{ijt\tau} \ge 0$                   $\forall i, \forall j$
```

Fast-Timescale Algorithm. First, we run Algorithm 1 at every time slot of every time frame to solve the workload decisions $\{\hat{\mathbf{x}}_\mathbf{t}\}$. We solve the problem $\mathbf{P_{NL}(x)}$ which is derived from $\mathbf{P_{NL}}$, with additional constraints $\sum_i x_{ijt\tau} \le e_j C_j, \forall j, \forall t, \forall \tau \in S_t$ which ensure distributed workload is within cloudlets' capacity. $\mathbf{P_{NL}(x)}$ is simply a linear program. Our key observation here is that, when the optimum is reached, $\{\hat{\mathbf{x}}_\mathbf{t}\}$ solved from $\mathbf{P_{NL}(x)}$ and the optimal $\{\mathbf{x_t}\}$ solved from $\mathbf{P_{NL}}$ make the same constraint, i.e., (4b), tight. Thus, we are able to derive $\mathbf{P_{NL}(x)}$ from $\mathbf{P_{NL}}$ and solve $\{\mathbf{x_t}\}$ independently on the fast timescale. This connection is used to analyze our proposed algorithms' performance as well.

Slow-Timescale Algorithm: Fractional. Next, we call Algorithm 2 at every time frame to solve the fractional resource decisions $\{\hat{\mathbf{y}}_\mathbf{t}\}$ with $\{\hat{\mathbf{x}}_\mathbf{t}\}$ as inputs. Our focus here is to address the switching cost. We construct the problem $\mathbf{P_{NL_t}}$ for time frame t by relaxing the integer constraints to the continuous domain and substituting $(y_{jt} - y_{jt-1})^+$ by a well-designed logarithmic term of $(y_{jt} + \varepsilon) \ln \frac{y_{jt}+\varepsilon}{y_{jt-1}+\varepsilon} - y_{jt}$,

Algorithm 2: Slow-Timescale Online Algorithm, $\forall t$

```
1 Insert $\{\hat{\mathbf{x}}_\mathbf{t}\}$ in the following problem $\mathbf{P_{NL_t}}$ as inputs;
2 Solve $\mathbf{P_{NL_t}}$ to obtain the optimal $\{\hat{\mathbf{y}}_\mathbf{t}, \hat{\mathbf{v}}_\mathbf{t}\}$;
    min    $P_{NL_t} = \sum_\tau \sum_i \sum_j a_{ijt\tau} x_{ijt\tau} + \sum_j 2b_{jt} y_{jt}$
             $+ \sum_\tau \sum_j d_j v_{jt\tau}$
             $+ \sum_j \frac{\Delta_{jt}}{\eta} \Big( (y_{jt} + \varepsilon) \ln \frac{y_{jt} + \varepsilon}{\hat{y}_{jt-1} + \varepsilon} - y_{jt} \Big)$
    s.t.   (4a), (4b), (4d)          without "$\forall t$"
           $y_{jt}, v_{jt\tau} \ge 0$               $\forall j, \forall \tau \in S_t$
  where $\varepsilon > 0$ and $\eta = \ln\big(1 + \frac{1}{\varepsilon}\big)$ are parameters.
```

where $\varepsilon > 0$ is a pre-specified parameter. This new term is convex and differentiable in y_{jt} given y_{jt-1} [19], and the property of logarithm helps us regularize the current decisions in a controlled manner to avoid excessive switching cost that might be incurred by potential future workload increase. It allows us to solve $\mathbf{P_{NL_t}}$ at t in polynomial time using any standard convex solver by taking solutions from $\mathbf{P_{NL_{t-1}}}$ and requiring no information beyond t. Moreover, it brings us proven performance guarantees.

Algorithm 3: Pairwise Rounding Algorithm, $\forall t$

```
   ▷ Round $\hat{\mathbf{y}}_\mathbf{t}$ to $\bar{\mathbf{y}}_\mathbf{t}$ in a randomized manner.
1  $\theta_{jt} \overset{\text{def}}{=} \hat{y}_{jt}, \forall j$;
2  $J' \overset{\text{def}}{=} J \setminus \{j | \theta_{jt} - \lfloor \theta_{jt} \rfloor = 0\}$;
3  while $|J'| > 1$ do
4      Select $j_1, j_2 \in J'$, where $j_1 \ne j_2$;
5      $\omega_1 \overset{\text{def}}{=} \min\{\lceil \theta_{j_1t} \rceil - \theta_{j_1t}, \theta_{j_2t}\}$;
6      $\omega_2 \overset{\text{def}}{=} \min\{\theta_{j_1t}, \lceil \theta_{j_2t} \rceil - \theta_{j_2t}\}$;
7      With the probability $\frac{\omega_2}{\omega_1+\omega_2}$,
       Set $\theta'_{j_1t} = \theta_{j_1t} + \omega_1, \theta'_{j_2t} = \theta_{j_2t} - \omega_1$;
8      With the probability $\frac{\omega_1}{\omega_1+\omega_2}$,
       Set $\theta'_{j_1t} = \theta_{j_1t} - \omega_2, \theta'_{j_2t} = \theta_{j_2t} + \omega_2$;
9      if $\theta'_{j_1t} - \lfloor \theta'_{j_1t} \rfloor = 0$ then Set $\bar{y}_{j_1t} = \theta'_{j_1t}, J' = J' \setminus \{j_1\}$;
10     else Set $\theta_{j_1t} = \theta'_{j_1t}$;
11     if $\theta'_{j_2t} - \lfloor \theta'_{j_2t} \rfloor = 0$ then Set $\bar{y}_{j_2t} = \theta'_{j_2t}, J' = J' \setminus \{j_2\}$;
12     else Set $\theta_{j_2t} = \theta'_{j_2t}$;
13 end
14 if $|J'| = 1$ then Set $\bar{y}_{jt} = \lceil \hat{y}_{jt} \rceil$ for the only $j \in J'$
```

Slow-Timescale Algorithm: Rounding. Lastly, we invoke Algorithm 3, inspired by the dependent rounding technique [18], at every time frame to convert the fractional solutions produced by Algorithm 2 to integers. In each iteration, the algorithm picks a pair of fractions and rounds at least one of the two values to an integer randomly. The main loop is Line 3 through Line 13 where either Line 7 or Line 8 is executed in each iteration based on the probability. The main loop has three properties: (1) In each iteration, at least one fraction is rounded to an integer; (2) After each iteration, there is $\theta_{j_1t} + \theta_{j_2t} = \theta'_{j_1t} + \theta'_{j_2t}$ that ensures the integral solutions after rounding do not violate the constraints of $\mathbf{P_{NL_t}}$; and (3) After rounding, the expectation of the integral value equals to the previous fractional value, i.e., $E(\bar{y}_{jt}) = \hat{y}_{jt}, \forall j \in J \setminus J'$. This will be used to derive the integrality gap for analysis.

IV. Competitive Analysis

We formally prove that the objective value of our original problem $\mathbf{P_L}$ evaluated with the solutions produced by our proposed online algorithms is upper-bounded by a parameterized constant times the offline optimum of $\mathbf{P_L}$. We will establish the following chain of inequalities:

$$\begin{aligned}
&E(P_L(\hat{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t})) && \text{(7a)}\\
&\leq E(P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t})) && \text{(7b)}\\
&\leq r_3 P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}) && \text{(7c)}\\
&\leq r_2 r_3 P_{NL_{OPT}} && \text{(7d)}\\
&\leq r_1 r_2 r_3 P_{L_{OPT}}, && \text{(7e)}
\end{aligned}$$

where $r = r_1 r_2 r_3$ is the overall *competitive ratio*. Regarding (7a) $\leq$ (7b), we exhibit that P_{NL} is an upper bound of P_L when evaluated with $\{\hat{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}\}$. We focus on the expected value "E", because we have introduced probability-based randomization in our rounding algorithm. Regarding (7b) $\leq$ (7c), we characterize the performance of our rounding algorithm, i.e., connecting the value of P_{NL} with rounded solutions to its value with fractional ones. Regarding (7c) $\leq$ (7d), we characterize the performance of our fractional online algorithms, i.e., connecting the value of P_{NL} evaluated with the online fractional solutions to its corresponding offline optimum. Regarding (7d) $\leq$ (7e), we connect the optimum of $\mathbf{P_{NL}}$ to that of $\mathbf{P_L}$, because the latter is the original problem that we aim to solve.

A. Learning Approximation

We prove (7a) $\leq$ (7b) in Theorem 1, then prove (7d) $\leq$ (7e) and show r_1 in Theorem 2.

Theorem 1. *$E(P_L(\hat{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t})) \leq E(P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}))$ is due to*

$$\sum_t \sum_j b_{jt+1} \bar{y}_{jt} \leq \sum_t \sum_j 2 b_{jt} \bar{y}_{jt} + \sum_t \sum_j b_{jt} (\bar{y}_{jt} - \bar{y}_{jt-1})^+$$

and

$$\begin{aligned}
&\sum_t \sum_\tau \sum_j d_j (\sum_i \hat{x}_{ijt+1\tau} - e_j \bar{y}_{jt})^+\\
&\leq \sum_t \sum_\tau \sum_j d_j (\sum_i \hat{x}_{ijt\tau} - e_j \bar{y}_{jt})^+\\
&+ \sum_t \sum_j |S_t| d_j e_j (\bar{y}_{jt} - \bar{y}_{jt-1})^+.
\end{aligned}$$

Proof. See Appendix A. □

Theorem 2. *$P_{NL_{OPT}} \leq r_1 P_{L_{OPT}}$, where*

$$r_1 = \frac{\max_{j,t} (\Delta_{jt} + 2 b_{jt} + |S_t| d_j e_j)}{\min_j c_j} + 2.$$

Proof. See Appendix B. □

B. Fractional Competitive Ratio

We prove (7c) $\leq$ (7d) and show r_2 in Theorem 3. We present the proof via a primal-dual method. That is, we show

$$P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}) \leq r_2 D_{NL}(\pi(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}, \hat{\mathbf{v}}_\mathbf{t})) \leq r_2 P_{NL_{OPT}}.$$

D_{NL} denotes the objective function of the problem $\mathbf{D_{NL}}$, which is the Lagrange dual problem of $\mathbf{P_{NL}}$. π denotes a mapping that maps $\mathbf{P_{NL}}$'s fractional solutions to the feasible solutions to $\mathbf{D_{NL}}$. Then, due to weak duality, we immediately have $D_{NL}(\{\pi(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}, \hat{\mathbf{v}}_\mathbf{t})\}) \leq P_{NL_{OPT}}$, and thus we only need to prove $P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}) \leq r_2 D_{NL}(\pi(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}, \hat{\mathbf{v}}_\mathbf{t}))$. Below, we formulate $\mathbf{D_{NL}}$, construct π, and establish the bound.

***Formulating Lagrange Dual of* $\mathbf{P_{NL}}$.** We derive the dual problem $\mathbf{D_{NL}}$ as follows, where $\alpha_{t\tau}, \beta_{it\tau}, \phi_{jt}, \varphi_{jt\tau}$ are the dual variables:

$$\begin{aligned}
\max\quad & D_{NL} = \sum_t \sum_\tau \alpha_{t\tau} (\sum_i \lambda_{it\tau} - \sum_j e_j C_j)\\
& + \sum_t \sum_\tau \sum_i \beta_{it\tau} \lambda_{it\tau}\\
\text{s.t.}\quad & a_{ijt\tau} - \beta_{it\tau} + \varphi_{jt\tau} \geq 0, \quad \forall i, \forall j, \forall t, \forall \tau \in S_t, && \text{(9a)}\\
& 2 b_{jt} + \phi_{jt} - \phi_{jt+1} - e_j \sum_\tau \varphi_{jt\tau} \geq 0, \quad \forall j, \forall t, && \text{(9b)}\\
& \Delta_{jt} - \phi_{jt} \geq 0, \quad \forall j, \forall t, && \text{(9c)}\\
& d_j - \alpha_{t\tau} - \varphi_{jt\tau} \geq 0, \quad \forall j, \forall t, \forall \tau \in S_t, && \text{(9d)}\\
& \alpha_{t\tau}, \beta_{it\tau}, \phi_{jt}, \varphi_{jt\tau} \geq 0, \quad \forall j, \forall t, \forall \tau \in S_t. && \text{(9e)}
\end{aligned}$$

Constructing the Mapping. We construct the mapping as follows, where $\hat{y}_{jt}$ are solved from $\mathbf{P_{NL_t}}$, and α_τ, $\beta_{i\tau}$, $\varphi_{j\tau}$ are $\mathbf{P_{NL_t}}$'s dual variables:

$$\begin{aligned}
&\alpha_{t\tau} = \alpha_\tau, && \forall \tau \in S_t,\\
&\beta_{it\tau} = \beta_{i\tau}, && \forall i, \forall \tau \in S_t,\\
&\varphi_{jt\tau} = \varphi_{j\tau}, && \forall j, \forall \tau \in S_t,\\
&\phi_{jt} = \frac{\Delta_{jt}}{\eta} \ln \frac{1+\varepsilon}{\hat{y}_{jt-1}+\varepsilon}, && \forall j.
\end{aligned}$$

To verify that such constructed solutions are indeed feasible for $\mathbf{D_{NL}}$, we need to show that they satisfy $\mathbf{D_{NL}}$'s constraints. This can actually be shown by using the Karush-Kuhn-Tucker (KKT) conditions of $\mathbf{P_{NL_t}}$, which they satisfy:

$$\begin{aligned}
& a_{ijt\tau} - \beta_{i\tau} + \varphi_{j\tau} = 0 && \text{(11a)}\\
& 2 b_{jt} + \frac{\Delta_{jt}}{\eta} \ln \frac{\hat{y}_{jt}+\varepsilon}{\hat{y}_{jt-1}+\varepsilon} - e_j \sum_\tau \varphi_{j\tau} = 0 && \text{(11b)}\\
& d_j - \alpha_\tau - \varphi_{j\tau} = 0 && \text{(11c)}\\
& \alpha_\tau (\sum_i \lambda_{it\tau} - \sum_j e_j C_j - \sum_j \hat{v}_{jt\tau}) = 0 && \text{(11d)}\\
& \beta_{i\tau} (\lambda_{it\tau} - \sum_j \hat{x}_{ijt\tau}) = 0 && \text{(11e)}\\
& \varphi_{j\tau} (\sum_i \hat{x}_{ijt\tau} - e_j \hat{y}_{jt} - \hat{v}_{jt\tau}) = 0 && \text{(11f)}\\
& \text{all primal and dual variables} \geq 0 && \text{(11g)}
\end{aligned}$$

Bounding. Consequently, we bound the non-switching cost and the switching cost, respectively. Combining them together, we have the following theorem:

Theorem 3. *$P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}) \leq r_2 D_{NL}(\pi(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}, \hat{\mathbf{v}}_\mathbf{t}))$, where $r_2 = 1 + \eta |J| (\max_j e_j C_j)$, and $\eta = \ln(1 + \frac{1}{\varepsilon}), \varepsilon \geq 0$.*

Proof. See Appendix C. □

C. Integrality Gap

We prove (7b) $\leq$ (7c) and show r_3 in Theorem 4.

Theorem 4. *$E(P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t})) \leq r_3 P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t})$ is due to*

$$\begin{aligned}
&E(\sum_t \sum_j 2 b_{jt} \bar{y}_{jt} + \sum_t \sum_\tau \sum_j d_j (\sum_i \hat{x}_{jit\tau} - e_j \bar{y}_{jt})^+)\\
&\leq \delta'_y P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}),
\end{aligned}$$

and $E(\sum_t \sum_j \Delta_{jt} (\bar{y}_{jt} - \bar{y}_{jt-1})^+) \leq \delta''_y P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t})$,

where

$$\begin{aligned}
\delta'_y &= \frac{2|J| \max_{j,t} b_{jt} C_j}{\min_t \sum_\tau \sum_i \lambda_{it\tau} \min_{i,j,t,\tau} a_{ijt\tau}} + \frac{\max_j d_j}{\min_{i,j,t,\tau} a_{ijt\tau}},\\
\delta''_y &= \frac{|J| \max_{j,t} \Delta_{jt} C_j}{\min_t \sum_\tau \sum_i \lambda_{jt\tau} \min_{i,j,t,\tau} a_{ijt\tau}},\\
r_3 &= \delta'_y + \delta''_y.
\end{aligned}$$

Proof. See Appendix D. □

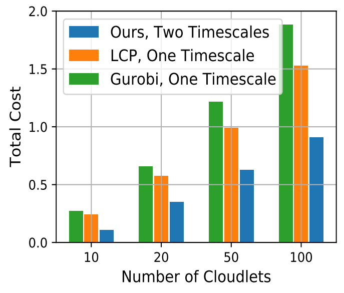

Fig. 3: Cost of single vs. multiple timescales

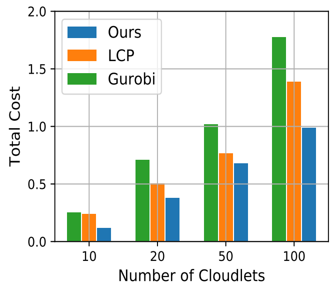

Fig. 4: Cost of algorithms in two timescales

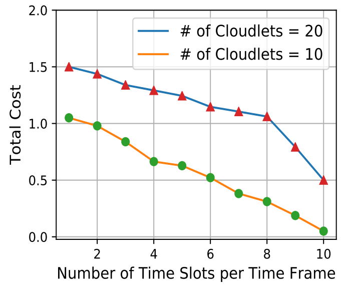

Fig. 5: Cost for different numbers of slots per frame

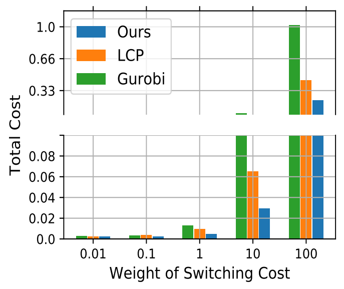

Fig. 6: Cost for different weights of switching cost

V. Numerical Evaluation

A. Evaluation Setup

Cloudlets and Timescales. We simulate the cloudlets as being co-located with the Tube stations of the Greater London Urban Area, where each station has one cloudlet. For scalability, we vary the cloudlets under consideration as the largest $10 \sim 100$ stations within the radius of $4 \sim 20$ miles centered around the Oxford Circus station. We observe the system for 24 hours. We set the length of a time frame as 5 minutes [21], and vary the length of a time slot as $0.5 \sim 5$ minutes.

Workload. We use the dynamic numbers of the passengers entering and exiting each selected Tube station in the 24 hours of November 10, 2019 [22] to mimic the workload received at each cloudlet.

Routing Cost, Resource Price, Weights, and Capacity. We calculate the unit routing cost between two cloudlets based on the geographic distance between the corresponding stations. We use Amazon EC2 Spot Instance VMs' prices (c3.2xlarge, Linux/UNIX, US West) [23] as the VM operational price in our system. We also vary the weights of the switching cost and that of the learning loss as $0.01 \sim 100$ in order to show the results for a spectrum of settings. We set the algorithmic parameter ε as 0.001. We set the capacity of each cloudlet based on a formula that captures the approximate number of users a website can host via Google Analytics [24].

Algorithms. We use Python 3 and A Mathematical Programming Language (AMPL) [25] for data processing and algorithm implementation. We invoke the interior-point-based IPOPT solver [26] to solve the underlying fractional optimization problems. In the results, we denote our approach as "`Ours`". We also implement two alternative algorithms. The first solves each one-shot sub-problem of our non-learning problem (i.e., $\mathbf{P_{NL}}$) via invoking the mixed-integer program solver Gurobi [27] to directly get the integer solutions. We denote this approach as "`Gurobi`". The second applies the "Lazy Capacity Provisioning" (LCP) algorithm to our non-learning problem, where LCP is a state-of-the-art online algorithm that tackles problems with switching costs [9], coupled with Python 3's default rounding method. We denote this approach as "`LCP`". All evaluations run on a laptop with an Intel(R) Core i7-7500U 2.7-GHz CPU and 16-GB RAM. All the total cost values in the figures have been normalized for better visualization.

B. Evaluation Results

Single Timescale vs. Two Timescales. We compare our approach which runs on two timescales against `Gurobi` and `LCP` which run on a single timescale (by setting the length of a time slot to that of a time frame). As we allow the system to react fast and slow corporately, our approach does not incur as frequent decision changes for resource provisioning as the single-timescale ones. Fig. 3 confirms that our approach produces significantly less total cost, saving on average 48% and 41% total cost compared to `Gurobi` and `LCP`, respectively.

Total Cost in Two Timescales. We compare our approach, `Gurobi` and `LCP`, all on two timescales. That is, `Gurobi` and `LCP` also invoke our fast-timescale algorithm to distribute the workload, but make their own decisions in the slow timescale to provision the resources. Fig. 4 depicts that, as the number of cloudlets varies, our approach outperforms `Gurobi` and `LCP` by 32% and 26% total cost on average. As more cloudlets and workload are involved in the system, the total cost goes up. Although the alternative approaches as well produce better results in the two-timescale setting compared to themselves in the single-timescale setting, our approach is always better.

Impact of Number of Slots per Frame. We verify the total cost incurred by our approach as the number of time slots per time frame varies, while fixing the length of each time frame. Fig. 5 presents a decreasing pattern of the total cost as each time frame contains more time slots. As the number of time slots per time frame grows, the system can react to user requests and distribute workload more frequently. Therefore, it has more opportunities to "learn", yielding a better resource provisioning decision for each time frame.

Impact of Weight of Switching Cost and Learning Loss. We check the total cost of the algorithms when different weights are associated to the switching cost and the learning loss. Fig. 6 shows that, as the weight goes up, the total cost incurred by all algorithms increases; our approach, however, yields the least total cost that are on average 53% and 41% less than `Gurobi` and `LCP`, respectively. The reasons are as follows: (1) `Gurobi` cannot address switching cost well, as it essentially ignores the switching cost; (2) Python 3's default

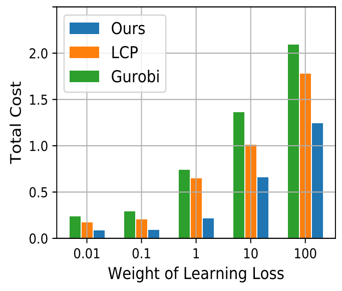

Fig. 7: Cost for different weights of learning loss

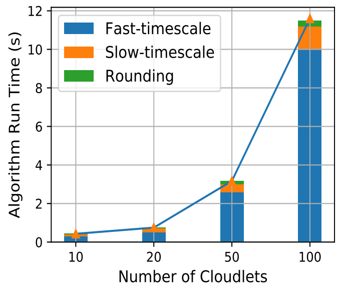

Fig. 8: Running time of different algorithms

rounding method with LCP does not ensure that our problem's constraints are satisfied, so some resource provisioning variables are rounded up and others are rounded down, without any guarantees. Fig. 7 exhibits the influence of the weight of learning loss on the total cost. It shows that as the weight grows, the total cost for all algorithms increases. Our approach still increases the slowest and outperforms Gurobi and LCP by 38% and 27% less total cost on average.

Algorithm Execution Time. We investigate our algorithms' running time of each round of decision making in Fig. 8. The line shows our approach's overall running time; the stacked bars show the running time of each algorithmic component. Our fast-timescale algorithm takes relatively much longer time to run than our slow-timescale algorithm, because it involves a larger number of variables and runs more frequently. Our rounding algorithm takes negligible time to finish, as it has a linear time complexity by design. Our approach is extremely efficient and scalable because the overall running times in all scenarios, where even a hundred cloudlets are considered, are always less than 2% of the length of a time frame.

VI. Related Work

We consider the existing research on single-timescale and multi-timescale cloud and edge management, and highlight their insufficiency compared to our work.

Single-Timescale Resource Management. He et al. studied service placement and request scheduling with shareable and non-shareable data across cloudlets [4]. Xu et al. designed a decoupled model to manage different resources independently based on a priori resource-sharing contracts [8]. Lin et al. proposed to save the energy consumption of data centers by "lazily" right-sizing servers via online algorithms [9]. Jiao et al. managed both the servers contained in the cloudlets and the cloudlets themselves for better overall performance [17]. Wang et al. addressed user mobility in edge networks by an online algorithm for allocating resources and shifting workloads [10]. Cai et al. designed a general online learning framework based on stochastic multi-armed bandits for a variety of network optimization problems [7]. Ouyang et al. jointly optimized user's perceived latency and service migration cost using a Thompson-sampling-based learning algorithm [11].

All of these works are for the one-timescale setting, rather than for two timescales. Although some of them have considered the switching cost between consecutive decisions [9], [10], [17], or the learning of the uncertainties [7], [11], we emphasize that it largely remains a question whether and how we could adapt such existing one-timescale approaches to two timescales with provable performance guarantees, due to the fundamental change in the settings. This also motivates us to design novel, intrinsic two-timescale approaches.

Multi-Timescale System Optimization. Goel et al. did a theoretical study on how to design control algorithms that can be decomposed across timescales [16]. Deng et al. and Yao et al. applied Lyapunov optimization to the power supply minimization for cloud data centers while purchasing energy on two timescales [12], and also to the joint cloud power and workload delay reduction by managing routing and servers over different timescales [13]. Chen et al. extended a stochastic dual gradient algorithm for placing virtual network functions while stabilizing the backlogs of network services [14]. Gao et al. proposed an MDP-based framework to maximize service profit while provisioning resources on a slow timescale and scheduling workload on a fast timescale [15]. Farhadi et al. formulated and solved a two-timescale online linear program for joint service placement and request scheduling [5].

Despite these works focus on the two-timescale approaches, some of them adopt offline instead of online algorithms [5]; those that are online often target the time-averaged objectives and constraints [12]–[14], or the stochastic models of the system [15]. Some rely on the capability of predicting future inputs [16]. None of these works matches our problem settings featured by the switching cost plus the deterministic models that need no stochastic knowledge or prediction of the system's inputs. To the best of our knowledge, this paper is the first to address these challenges in the two-timescale settings.

VII. Conclusion

In this paper, we investigate the online resource provisioning and workload distribution problem in distributed edge cloudlet networks. We propose and present a novel online learning framework which makes resource provisioning decisions and workload distribution decisions on two separate timescales. Our approach features the online fractional algorithm and the rounding algorithm that produce resource provisioning decisions on the fly, without worrying about the unknown upcoming dynamic inputs and the workload distribution decisions to be made in the future. We have formally proved a parameterized-constant competitive ratio for our framework, and also conducted numerical evaluations with real-world data to confirm the advantages of our approach in practice.

Appendix

A. Proof of Theorem 1

$$\begin{aligned}
&\textstyle\sum_t \sum_j b_{jt+1}\bar{y}_{jt} = \sum_t \sum_j b_{jt}\bar{y}_{jt-1} && \text{(15a)}\\
=&\textstyle\sum_t \sum_j b_{jt}\bar{y}_{jt} - \sum_t \sum_j b_{jt}(\bar{y}_{jt} - \bar{y}_{jt-1}) && \text{(15b)}\\
\le&\textstyle\sum_t \sum_j b_{jt}\bar{y}_{jt} + \sum_t \sum_j b_{jt}|\bar{y}_{jt} - \bar{y}_{jt-1}| && \text{(15c)}\\
\le&\textstyle\sum_t \sum_j 2b_{jt}\bar{y}_{jt} + \sum_t \sum_j b_{jt}(\bar{y}_{jt} - \bar{y}_{jt-1})^+ && \text{(15d)}
\end{aligned}$$

$$\textstyle\sum_t \sum_\tau \sum_j d_j(\sum_i \hat{x}_{ijt+1\tau} - e_j\bar{y}_{jt})^+$$

$$\begin{aligned}&\le \textstyle\sum_t\sum_\tau\sum_j d_j(\sum_i \hat{x}_{ijt+1\tau}-e_j\bar{y}_{jt+1})^+\\&\quad+\textstyle\sum_t\sum_\tau\sum_j d_je_j(\bar{y}_{jt+1}-\bar{y}_{jt})^+ &(16a)\\&=\textstyle\sum_t\sum_\tau\sum_j d_j(\sum_i \hat{x}_{ijt\tau}-e_j\bar{y}_{jt})^+\\&\quad+\textstyle\sum_t\sum_\tau\sum_j d_je_j(\bar{y}_{jt}-\bar{y}_{jt-1})^+ &(16b)\\&=\textstyle\sum_t\sum_\tau\sum_j d_j(\sum_i \hat{x}_{ijt\tau}-e_j\bar{y}_{jt})^+\\&\quad+\textstyle\sum_t\sum_j |S_t|d_je_j(\bar{y}_{jt}-\bar{y}_{jt-1})^+ &(16c)\end{aligned}$$

(15a) holds because of $b_{jT+1}=0$ and $\bar{y}_{j0}=0$ by definition. (15b) equals to (15a). (15c) holds because the absolute value is added. (15d) is reached due to the properties of the absolute value and $(\cdot)^+$.

(16a) holds due to the non-negative property of $(\cdot)^+$. (16b) holds because by definition we have $\hat{x}_{ijT+1\tau}=0$, $\bar{y}_{jT+1}=0$ and $\bar{y}_{j0}=0$. (16c) removes the irrelevant summation. Obviously, the expectations of these terms grant the same relations.

B. *Proof of Theorem 2*

Let us review the objective function of the non-learning problem and the original online learning problem.

$$\begin{aligned}P_{NL}=&\textstyle\sum_t\sum_\tau\sum_i\sum_j a_{ijt\tau}x_{ijt\tau}+\sum_t\sum_j 2b_{jt}y_{jt}\\&+\textstyle\sum_t\sum_j \Delta_{jt}(y_{jt}-y_{jt-1})^+\\&+\textstyle\sum_t\sum_\tau\sum_j d_j(\sum_i x_{ijt\tau}-e_jy_{jt})^+\end{aligned}$$

$$\begin{aligned}P_L=&\textstyle\sum_t\sum_\tau\sum_i\sum_j a_{ijt\tau}x_{ijt\tau}+\sum_t\sum_j b_{jt+1}y_{jt}\\&+\textstyle\sum_t\sum_j c_j(y_{jt}-y_{jt-1})^+\\&+\textstyle\sum_t\sum_\tau\sum_j d_j(\sum_i x_{ijt+1\tau}-e_jy_{jt})^+\end{aligned}$$

Assuming $\{\hat{\mathbf{x}}_\mathbf{t}^*,\bar{\mathbf{y}}_\mathbf{t}^*,\forall t\}$ are offline optimal solutions to $\mathbf{P_L}$, and let Q represents $\sum_t\sum_\tau\sum_i\sum_j a_{ijt\tau}x_{ijt\tau}$ that appears in both P_{NL} and P_L. The following chain of inequalities is built:

$$\begin{aligned}&P_{NL_{OPT}}\le P_{NL}(\{\hat{\mathbf{x}}_\mathbf{t}^*,\bar{\mathbf{y}}_\mathbf{t}^*,\forall t\}) &(19a)\\&=Q(\{\hat{\mathbf{x}}_\mathbf{t}^*,\bar{\mathbf{y}}_\mathbf{t}^*,\forall t\})+\textstyle\sum_t\sum_j\Delta_{jt}(\bar{y}_{jt}^*-\bar{y}_{jt-1}^*)^+\\&\quad+\textstyle\sum_t\sum_j 2b_{jt}\bar{y}_{jt}^*+\sum_t\sum_\tau\sum_j d_j(\sum_i\hat{x}_{ijt\tau}^*-e_j\bar{y}_{jt}^*)^+ &(19b)\\&\le Q(\{\hat{\mathbf{x}}_\mathbf{t}^*,\bar{\mathbf{y}}_\mathbf{t}^*,\forall t\})+\textstyle\sum_t\sum_j 2b_{jt}\bar{y}_{jt-1}^*\\&\quad+\textstyle\sum_t\sum_j(\Delta_{jt}+2b_{jt})(\bar{y}_{jt}^*-\bar{y}_{jt-1}^*)^+\\&\quad+\textstyle\sum_t\sum_\tau\sum_j d_j(\sum_i\hat{x}_{ijt\tau}^*-e_j\bar{y}_{jt-1}^*+e_j\bar{y}_{jt-1}^*-e_j\bar{y}_{jt}^*)^+ &(19c)\\&\le Q(\{\hat{\mathbf{x}}_\mathbf{t}^*,\bar{\mathbf{y}}_\mathbf{t}^*,\forall t\})+\textstyle\sum_t\sum_j 2b_{jt+1}\bar{y}_{jt}^*\\&\quad+\textstyle\sum_t\sum_j(\Delta_{jt}+2b_{jt}+|S_t|d_je_j)(\bar{y}_{jt}^*-\bar{y}_{jt-1}^*)^+\\&\quad+\textstyle\sum_t\sum_\tau\sum_j d_j(\sum_i\hat{x}_{ijt+1\tau}^*-e_j\bar{y}_{jt}^*)^+ &(19d)\\&\le r_1P_{L_{OPT}} &(19e)\end{aligned}$$

where $r_1=2+\frac{\max_{j,t}(\Delta_{jt}+2b_{jt}+|S_t|d_je_j)}{\min_j c_j}$.

(19a) and (19b) hold because $\{\hat{\mathbf{x}}_\mathbf{t}^*,\bar{\mathbf{y}}_\mathbf{t}^*,\forall t\}$ are not necessarily the optimal solutions to $\mathbf{P_{NL}}$. (19c) holds because the term $\sum_t\sum_j 2b_{jt}\bar{y}_{jt-1}^*$ is non-negative. (19d) holds due to the definition of $(\cdot)^+$, and $\hat{x}_{ijT+1\tau}=0$, $b_{jT+1}=0$ and $\bar{y}_{j0}=0$. (19e) is valid due to $r_1>1$.

C. *Proof of Theorem 3*

We first bound the non-switching cost terms:

$$\begin{aligned}&\textstyle\sum_t\sum_\tau\sum_i\sum_j a_{ijt\tau}\hat{x}_{ijt\tau}+\sum_t\sum_j 2b_{jt}\hat{y}_{jt}\\&+\textstyle\sum_t\sum_\tau\sum_j d_j\hat{v}_{jt\tau}\\&=\textstyle\sum_t\sum_\tau\sum_i\sum_j(\beta_{it\tau}-\varphi_{jt\tau})\hat{x}_{ijt\tau}\\&\quad+\textstyle\sum_t\sum_j(e_j\sum_\tau\varphi_{jt\tau}-\frac{\Delta_{jt}}{\eta}\ln\frac{\hat{y}_{jt}+\varepsilon}{\hat{y}_{jt-1}+\varepsilon})\hat{y}_{jt}\\&\quad+\textstyle\sum_t\sum_\tau\sum_j(\alpha_{t\tau}+\varphi_{jt\tau})\hat{v}_{jt\tau} &(20a)\\&\le\textstyle\sum_t\sum_\tau\sum_j(\sum_i\beta_{it\tau}\hat{x}_{ijt\tau}-\sum_i\varphi_{jt\tau}\hat{x}_{ijt\tau})\\&\quad+\textstyle\sum_t\sum_\tau\sum_j e_j\varphi_{jt\tau}\hat{y}_{jt}\\&\quad+\textstyle\sum_t\sum_\tau\sum_j(\alpha_{t\tau}\hat{v}_{jt\tau}+\varphi_{jt\tau}\hat{v}_{jt\tau}) &(20b)\\&=\textstyle\sum_t\sum_\tau\alpha_{t\tau}\sum_j\hat{v}_{jt\tau}\\&\quad-\textstyle\sum_t\sum_\tau\sum_j\varphi_{jt\tau}(\sum_i\hat{x}_{ijt\tau}-e_j\hat{y}_{jt}-\hat{v}_{jt\tau})\\&\quad+\textstyle\sum_t\sum_\tau\sum_i\beta_{it\tau}\sum_j\hat{x}_{ijt\tau} &(20c)\\&=\textstyle\sum_t\sum_\tau\alpha_{t\tau}(\sum_i\lambda_{it\tau}-\sum_j e_jC_j)+\sum_t\sum_\tau\sum_i\beta_{it\tau}\lambda_{it\tau} &(20d)\\&=D &(20e)\end{aligned}$$

In (20a), the parameters are substituted with terms obtained from KKT conditions (11a) $\sim$ (11c). In (20b), the log term is removed and the inequality still holds due to its non-negativity. Further explanation is given below. $\forall p,\forall q$, there are two facts:

$$p-q\le p\ln\tfrac{p}{q}$$
$$(\textstyle\sum_n p_n\ln\frac{\sum_n p_n}{\sum_n q_n})\le\sum_n p_n\ln\frac{p_n}{q_n}.$$

Therefore, $\forall j$,

$$\begin{aligned}&\textstyle\sum_t\hat{y}_{jt}\ln\frac{\hat{y}_{jt}+\varepsilon}{\hat{y}_{jt-1}+\varepsilon}\\&=\textstyle\sum_t(\hat{y}_{jt}+\varepsilon)\ln\frac{\hat{y}_{jt}+\varepsilon}{\hat{y}_{jt-1}+\varepsilon}-\sum_t\varepsilon\ln\frac{\hat{y}_{jt}+\varepsilon}{\hat{y}_{jt-1}+\varepsilon}\\&\ge(\textstyle\sum_t(\hat{y}_{jt}+\varepsilon))\ln\frac{\sum_t(\hat{y}_{jt}+\varepsilon)}{\sum_t(\hat{y}_{jt-1}+\varepsilon)}+(\hat{y}_{j0}+\varepsilon)\ln\frac{\hat{y}_{j0}+\varepsilon}{\hat{y}_{jT}+\varepsilon}\\&\ge\textstyle\sum_t(\hat{y}_{jt}+\varepsilon)-\sum_t(\hat{y}_{jt-1}+\varepsilon)+\hat{y}_{j0}-\hat{y}_{jT}\\&=0\end{aligned}$$

(20c) makes rearrangements in terms of the dual variables. (20d) is valid from (11d) $\sim$ (11f). (20e) follows from the objective function of the dual problem.

Then we bound the switching cost term. Let $\hat{J}=\{j|\hat{y}_{jt}\ge\hat{y}_{jt-1}\}$, because when $\hat{y}_{jt}\le\hat{y}_{jt-1}$, the switching cost equals zero thus there's no need to consider. We have

$$\begin{aligned}&\textstyle\sum_t\sum_j\Delta_{jt}(\hat{y}_{jt}-\hat{y}_{jt-1})^+\\&=\textstyle\sum_t\sum_{j\in\hat{J}}\Delta_{jt}(\hat{y}_{jt}-\hat{y}_{jt-1}) &(23a)\\&=\textstyle\sum_t\sum_{j\in\hat{J}}\Delta_{jt}((\hat{y}_{jt}+\varepsilon)-(\hat{y}_{jt-1}+\varepsilon)) &(23b)\\&\le\textstyle\sum_t\sum_{j\in\hat{J}}\Delta_{jt}(\hat{y}_{jt}+\varepsilon)\ln\frac{\hat{y}_{jt}+\varepsilon}{\hat{y}_{jt-1}+\varepsilon} &(23c)\\&\le\textstyle\sum_t\sum_{j\in\hat{J}}\Delta_{jt}C_j\ln\frac{\hat{y}_{jt}+\varepsilon}{\hat{y}_{jt-1}+\varepsilon} &(23d)\\&\le\eta\max_j C_j\textstyle\sum_t\sum_{j\in\hat{J}}(e_j\sum_\tau\varphi_{jt\tau}-2b_{jt}) &(23e)\\&\le\eta\max_j C_j\textstyle\sum_t\sum_\tau\sum_{j\in\hat{J}}e_j\beta_{it\tau} &(23f)\\&\le\eta\max_j e_jC_j\textstyle\sum_t\sum_\tau\sum_j\beta_{it\tau}\sum_i\lambda_{it\tau} &(23g)\\&\le\eta|J|(\max_j e_jC_j)D &(23h)\end{aligned}$$

where $\eta=\ln(1+\frac{1}{\varepsilon}),\varepsilon\ge0$.

(23a) holds due to the definition of $\hat{J}$. (23b) introduces the non-negative parameter ε. (23c) holds due to the fact that $\forall p, \forall q \geq 0, p - q \leq p \ln \frac{p}{q}$. (23d) is valid due to $\hat{y}_{jt} \leq C_j, \forall j, \forall t$. (23e) introduces parameter η and follows from (11b). (23f) removes the negative terms and follows from (11a). Moreover, per problem settings there would be at least one user request across all cloudlets in every time slot of every time frame, i.e., $\sum_i \lambda_{it\tau} \geq 1, \forall t, \forall \tau \in S_t$. (23g) holds due to (11e). (23h) follows from the objective function of the dual problem.

Combining the results from above derivations, we have proved that $P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}) \leq r_2 D_{NL}(\{\pi(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}, \hat{\mathbf{v}}_\mathbf{t})\})$ and obtained the fractional competitive ratio as $r_2 = 1 + \eta |J| (\max_j e_j C_j)$.

D. Proof of Theorem 4

Based on problem setting, $\{\hat{\mathbf{y}}_\mathbf{t}\}$ need to be rounded to integers while satisfying constraints (4c) and (4d). The non-switching cost terms and the switching cost term that contain $\hat{\mathbf{y}}_\mathbf{t}$ are treated separately.

Let's start with the expectation of the non-switching cost terms.

$$\begin{aligned}
&E(\textstyle\sum_t \sum_j 2b_{jt}\bar{y}_{jt} + \sum_t \sum_\tau \sum_j (\sum_i \hat{x}_{ijt\tau} - e_j \bar{y}_{jt})^+) \\
\leq& \textstyle\max_{j,t} 2b_{jt} \sum_t (\sum_{j \in J \setminus J'} \hat{y}_{jt} + \sum_{j \in J'} \lceil \hat{y}_{jt} \rceil) \\
&+ \frac{\max_j d_j}{\min_{i,j,t,\tau} a_{ijt\tau}} \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} \hat{x}_{ijt\tau} \qquad (24a) \\
\leq& 2 \textstyle\max_{j,t} b_{jt} \sum_t \sum_j C_j \\
&+ \frac{\max_j d_j}{\min_{i,j,t,\tau} a_{ijt\tau}} \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} \hat{x}_{ijt\tau} \qquad (24b) \\
=& 2 \textstyle\max_{j,t} b_{jt} \sum_t \sum_j (\frac{C_j}{\sum_\tau \sum_i \lambda_{it\tau}}) \sum_\tau \sum_i \lambda_{it\tau} \\
&+ \frac{\max_j d_j}{\min_{i,j,t,\tau} a_{ijt\tau}} \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} \hat{x}_{ijt\tau} \qquad (24c) \\
\leq& \frac{2|J| \max_{j,t} b_{jt} C_j}{\min_t \sum_\tau \sum_i \lambda_{it\tau} \min_{i,j,t,\tau} a_{ijt\tau}} \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} \hat{x}_{ijt\tau} \\
&+ \frac{\max_j d_j}{\min_{i,j,t,\tau} a_{ijt\tau}} \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} \hat{x}_{ijt\tau} \qquad (24d) \\
\leq& \delta'_y \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} \hat{x}_{ijt\tau} \qquad (24e) \\
\leq& \delta'_y P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}) \qquad (24f)
\end{aligned}$$

where $\delta'_y = \frac{2|J| \max_{j,t} b_{jt} C_j}{\min_t \sum_\tau \sum_i \lambda_{it\tau} \min_{i,j,t,\tau} a_{ijt\tau}} + \frac{\max_j d_j}{\min_{i,j,t,\tau} a_{ijt\tau}}$.

(24a) decomposes $\{\bar{\mathbf{y}}_\mathbf{t}\}$ based on Algorithm 3 and drops the negative term. (24b) holds due to that $\hat{y}_{jt}$ can never exceeds the capacity of the cloudlet, i.e., $\hat{y}_{jt} \leq C_j, \forall j, \forall t$. (24c) is valid based on the problem settings where there would be at least one user request across all cloudlets in every time slot of every time frame, i.e., $\sum_\tau \sum_i \lambda_{it\tau} > 1, \forall t$. (24d) and (24e) make some rearrangements of the coefficients. (24f) is reached per definition of P_{NL}.

Then we bound the expectation of the switching cost.

$$\begin{aligned}
&E(\textstyle\sum_t \sum_j \Delta_{jt} (\bar{y}_{jt} - \bar{y}_{jt-1})^+) \\
\leq& E(\textstyle\sum_t \sum_j \Delta_{jt} \bar{y}_{jt}) \qquad (25a) \\
\leq& \textstyle\max_{j,t} \Delta_{jt} E(\sum_t \sum_j \bar{y}_{jt}) \qquad (25b) \\
\leq& \delta''_y \textstyle\sum_t \sum_\tau \sum_i \sum_j a_{ijt\tau} \hat{x}_{ijt\tau} \qquad (25c) \\
\leq& \delta''_y P_{NL}(\hat{\mathbf{x}}_\mathbf{t}, \hat{\mathbf{y}}_\mathbf{t}) \qquad (25d)
\end{aligned}$$

where $\delta''_y = \frac{|J| \max_{j,t} \Delta_{jt} C_j}{\min_t \sum_\tau \sum_i \lambda_{it\tau} \min_{i,j,t,\tau} a_{ijt\tau}}$.

(25a) holds due to the definition of $(\cdot)^+$. (25b) $\sim$ (25d) follow from the previous progress as (24a) $\sim$ (24f) similarly. We have $r_3 = \delta'_y + \delta''_y$ in the end.

References

[1] "Tia position paper edge data centers," https://www.tiaonline.org/wp-content/uploads/2018/10/TIA_Position_Paper_Edge_Data_Centers-18Oct18.pdf.

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

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

[4] T. He, H. Khamfroush, S. Wang, T. L. Porta, and S. Stein, "It's hard to share: Joint service placement and request scheduling in edge clouds with sharable and non-sharable resources," in *IEEE ICDCS*, 2018.

[5] V. Farhadi, F. Mehmeti, T. He, T. L. Porta, H. Khamfroush, S. Wang, and K. S. Chan, "Service placement and request scheduling for data-intensive applications in edge clouds," in *IEEE INFOCOM*, 2019.

[6] K. Ha, Y. Abe, T. Eiszler, Z. Chen, W. Hu, B. Amos, R. Upadhyaya, P. Pillai, and M. Satyanarayanan, "You can teach elephants to dance: agile vm handoff for edge computing," in *ACM/IEEE SEC*, 2017.

[7] K. Cai, X. Liu, Y.-Z. J. Chen, and J. C. Lui, "An online learning approach to network application optimization with guarantee," in *IEEE INFOCOM*, 2018.

[8] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, "Zenith: Utility-aware resource allocation for edge computing," in *IEEE EDGE*, 2017.

[9] M. Lin, A. Wierman, L. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," in *IEEE INFOCOM*, 2011.

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

[11] T. Ouyang, R. Li, X. Chen, Z. Zhou, and X. Tang, "Adaptive user-managed service placement for mobile edge computing: An online learning approach," in *IEEE INFOCOM*, 2019.

[12] W. Deng, F. Liu, H. Jin, and C. Wu, "Smartdpss: Cost-minimizing multi-source power supply for datacenters with arbitrary demand," in *IEEE ICDCS*, 2013.

[13] Y. Yao, L. Huang, A. Sharma, L. Golubchik, and M. Neely, "Data centers power reduction: A two time scale approach for delay tolerant workloads," in *IEEE INFOCOM*, 2012.

[14] X. Chen, W. Ni, T. Chen, I. Collings, X. Wang, R. P. Liu, and G. B. Giannakis, "Multi-timescale online optimization of network function virtualization for service chaining," *IEEE Transactions on Mobile Computing*, vol. 18, no. 12, pp. 2899–2912, 2019.

[15] G. Gao, H. Hu, Y. Wen, and C. Westphal, "Resource provisioning and profit maximization for transcoding in clouds: A two-timescale approach," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 836–848, 2017.

[16] G. Goel, N. Chen, and A. Wierman, "Thinking fast and slow: Optimization decomposition across timescales," in *IEEE CDC*, 2017.

[17] L. Jiao, L. Pu, L. Wang, X. Lin, and J. Li, "Multiple granularity online control of cloudlet networks for edge computing," in *IEEE SECON*, 2018.

[18] A. A. Ageev and M. I. Sviridenko, "Pipage rounding: A new method of constructing algorithms with proven performance guarantee," *Journal of Combinatorial Optimization*, vol. 8, no. 3, pp. 307–328, 2004.

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

[20] R. D. C. L. K. Fleischer, V. J. Leung, and C. A. Phillips, "Strengthening integrality gaps for capacitated network design and covering problems," in *ACM-SIAM SODA*, 2000.

[21] "Api: Last 24 hours," https://hourlypricing.comed.com/api?type=5minutefeed.

[22] "London underground passenger counts data," https://tfl.gov.uk/info-for/open-data-users/.

[23] "Amazon ec2 spot instances pricing," https://aws.amazon.com/ec2/spot/pricing/.

[24] "How to check concurrent visitors with google analytics," https://www.cloudways.com/blog/checking-concurrent-visitors-with-google-analytics/.

[25] "Ampl: Streamlined modeling for real optimization," https://www.ampl.com/.

[26] "Coin-or interior point optimizer ipopt," https://www.coin-or.org/Ipopt/.

[27] "Gurobi optimizer," https://www.gurobi.com/.