Distributed Threshold-based Offloading for Large-Scale Mobile Cloud Computing

Xudong Qin* Bin Li* Lei Ying†
*Department of ECBE, University of Rhode Island, Kingston, Rhode Island, USA
†Department of EECS, University of Michigan, Ann Arbor, Michigan, USA

***Abstract*—Mobile cloud computing enables compute-limited mobile devices to perform real-time intensive computations such as speech recognition or object detection by leveraging powerful cloud servers. An important problem in large-scale mobile cloud computing is computational offloading where each mobile device decides when and how much computation should be uploaded to cloud servers by considering the local processing delay and the cost of using cloud servers. In this paper, we develop a distributed threshold-based offloading algorithm where it uploads an incoming computing task to cloud servers if the number of tasks queued at the device reaches the threshold, and processes it locally otherwise. The threshold is updated iteratively based on the computational load and the cost of using cloud servers. We formulate the problem as a symmetric game, and characterize the sufficient and necessary conditions for the existence and uniqueness of the Nash Equilibrium (NE) assuming exponential service times. Then, we show the convergence of our proposed distributed algorithm to the NE when the NE exists. Finally, we perform extensive simulations to validate our theoretical findings and demonstrate the efficiency of our proposed distributed algorithm under various practical scenarios such as general service times, imperfect server utilization estimation, and asynchronous threshold updates.**

I. Introduction

Real-time mobile cloud applications have grown rapidly over the last few years and have become ubiquitous. For example, in an international trade show such as Consumer Electronics Show, people in the same convention center may need real-time translation services on their mobile devices at the same time, making it challenging to provide low latency language translation with a low service cost. On one hand, computing limited devices may not have the required capability to process the data locally; and on the other hand, offloading the computing to a cloud-computing center incurs both communication and computing costs. Mobile cloud computing, which utilizes both mobile and cloud computing powers, is a vital solution to address this challenge. A central question in mobile cloud computing is: how much to offload and when? This paper addresses this important question and proposes a distributed offloading algorithm where each device aims at minimizing a cost function including both the local processing delay and offloading cost at the cloud computing center.

While mobile cloud computing has received great research interest in recent years (see [1], [2] for the most recent thorough survey), most of the prior work mainly focused on reducing the average energy consumption (e.g., [3], [4], [5], [6], [7], [8], [9]), they did not consider users' cost of using cloud servers. Some of the prior works studied the distributed offloading algorithm for mobile cloud computing (e.g., [10], [11], [12], [13], [14], [15]), where all computing tasks are available at the beginning of each time slot. However, they did not address the case with dynamic computing tasks arrival and service time, which is predominant in practical cloud computing systems.

In this paper, we consider a system in which computing tasks dynamically arrive at each device and each device needs to make a real-time decision on whether to offload an incoming task or process it locally, with the objective of minimizing a cost function including local task processing time and the cost of using the cloud service. We focus on a threshold-based offloading policy under which an incoming task is offloaded to the cloud if the number of tasks at the local device reaches a threshold, and queued at the local device to be processed otherwise. We choose threshold-based policies for the following reasons: (i) The optimal offloading shares the same spirit of the admission control in a single-queue system, where the optimal solution has a threshold structure (see [16]), (ii) Optimal policies for many complex Markov Decision Process (MDP) problems are threshold-based; and (iii) Threshold policies are low-complexity policies that are easy to implement in practice. To that end, the following five fundamental questions naturally arise:

(i) **Algorithm Design**: How should a device adapt its threshold to minimize its cost?
(ii) **Existence**: Does there exist an equilibrium point such that each device will settle on a threshold and have no incentive to deviate from it?
(iii) **Uniqueness**: Is the equilibrium point unique?
(iv) **Convergence**: Does the system converge to the equilibrium when each device adapts its threshold to minimize its own cost?
(v) **Efficiency (or Price of Anarchy)** How efficient is the distributed threshold-based policy compared with the optimal centralized solution?

There are two main challenges to answer the questions above: (i) Since each user's offloading decision (i.e., threshold) is discrete and unbounded, classical fixed point theorems, which have been used successfully for proving the existence

This research has been supported in part by NSF grants: CNS-1717108, CNS-1815563, CNS-1942383, CNS-2002608, and CNS-2001687.

and uniqueness of Nash Equilibrium in many applications (e.g. [17], [18], [19], [20], [21], [22]), do not directly apply in our model; (ii) Since thresholds take integer values, it is challenging to show that the integer sequences will converge to the equilibrium point. In fact, it is *not* clear whether a distributed threshold-based algorithm, where each device chooses the optimal threshold given the current state of the cloud service, converges. However, we are able to show that an incremental distributed threshold-based policy, where each device increases/decreases its threshold to move it closer to the current optimal threshold, converges under some minor conditions. The main results and contributions of this paper are listed below:

- Under the exponential service time assumption, we analytically characterize the sufficient and necessary conditions for the existence and uniqueness of the Nash Equilibrium (see Theorem 1).
- We develop a distributed implementation of the threshold-based offloading algorithm (see Section III-A) so that each user iteratively and incrementally updates its own threshold based on its own cost function. We prove the convergence of our proposed algorithm to the Nash Equilibrium offloading decision if it exists under the exponential service time distribution (See Theorem 2).
- We characterize the efficiency of the Nash Equilibrium offloading decision via the Price of Anarchy, capturing efficiency loss comparing with the optimal centralized offloading (cf. Theorem 3).
- We perform extensive simulations (see Section IV) to validate our theoretical findings. Under various practical scenarios (e.g., general service time distributions, imperfect server utilization estimation, and asynchronous threshold updates), we also demonstrate the convergence of our proposed distributed algorithm to the Nash Equilibrium offloading decision, which is computed via numerical calculations.

The remainder of this paper is organized as follows: We introduce our system model in Section II. In Section III, we propose a distributed threshold-based offloading algorithm and present our main theoretical results. In Section IV-B, we perform extensive simulations to validate our theoretical findings as well as the efficiency of our proposed distributed algorithm. Section V concludes our paper.

II. System Model

We consider a mobile cloud computing system of N users, as shown in Fig. 1. Tasks arrive at each user according to a Poisson process of rate $\lambda > 0$. Each user can process a task either locally or upload it to cloud servers with a total service rate of Nc, where $c > \lambda$ ensures that all tasks can be processed at cloud servers if necessary. The mean service time of a task at a local device is $1/\mu$, where $\mu > 0$. We assume that each user n $(n = 1, 2, \ldots, N)$ maintains a queue to hold tasks awaiting for processing locally and we use $q_n(t)$ to denote the queue-length at time t, i.e., the number of awaiting tasks of user n at time t.

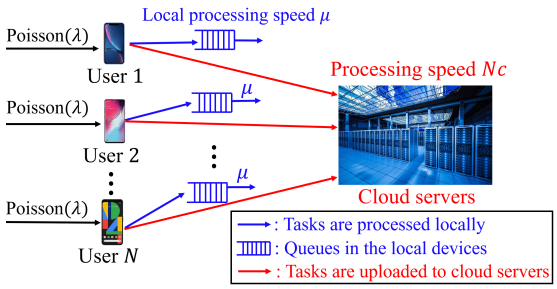


Fig. 1: System Model

For each incoming task, it experiences both queueing and processing delays when being processed locally. We assume the user will be charged with a service cost based on *server utilization* at the time if a task is offloaded to the cloud, where server utilization is the current load of the cloud servers. Here, we assume that cloud servers are high-capacity servers so that the delay of offloading to the cloud is negligible compared with local processing and queueing delays. Under these assumptions, each individual user makes an *offloading decision* that determines whether its incoming task is processed by itself or is uploaded to cloud servers with the goal of minimizing both its delay and service cost.

Since the offloading problem shares a similar spirit of the optimal admission control of a single queue whose solution has a threshold-based structure (see [16]), we focus on the following Threshold-Based Offloading (TBO) Policy.

Threshold-Based Offloading (TBO) Policy with integer parameters $\mathbf{B} \triangleq (B_n)_{n=1}^N$: For each user n, an incoming computing task will be processed by itself if its current queue-length $q_n(t)$ is less than B_n. Otherwise, the task is uploaded to cloud servers for computation.

In the TBO policy, B_n is the *threshold* of user n. If B_n is set to 0, then user n will upload all its incoming tasks to cloud servers. If $B_n \uparrow \infty$, then all computing tasks are processed by user n. Under the TBO policy, the queue-length of each user only depends on its own threshold B_n when the threshold is fixed. Therefore, we use $Q(B_n)$ to denote the average queue-length and $\pi(B_n)$ to denote probability that an incoming task is uploaded to cloud servers (also referred to as *offloading probability*). For each user n, an incoming task is processed by itself with probability $1-\pi(B_n)$. In such a case, it experiences the average delay of $\frac{Q(B_n)}{\lambda(1-\pi(B_n))}$ by Little's Law, where we use the fact that the average rate of tasks processed by user n is $\lambda(1-\pi(B_n))$. With offloading probability $\pi(B_n)$, an incoming task is uploaded to cloud servers and experiences a cost depending on the server utilization, i.e., $g(\beta(\mathbf{B}))$, where $\mathbf{B} \triangleq (B_n)_{n=1}^N$, $\beta(\mathbf{B}) \triangleq \lambda \sum_{n=1}^N \pi(B_n)/(Nc)$ is the utilization of cloud servers, $\lambda \sum_{n=1}^N \pi(B_n)$ is the average number of tasks that are uploaded to cloud servers, and $g(\cdot)$ is some convex, non-decreasing, and non-negative function. This is motivated by the fact that a large server utilization results in a high service cost in cloud services (see [23]). In the rest

of the paper, we assume that $g(x) = kx^2$, where $k > 0$ is some scaling parameter. Therefore, the average cost of user n can be expressed as

$$\frac{Q(B_n)}{\lambda} + k\beta^2(\mathbf{B})\pi(B_n). \tag{1}$$

In this paper, we focus on the *large-scale* mobile cloud computing system (i.e., N is large enough). Our goal is to develop a *distributed offloading algorithm* under which each device updates its own threshold, without knowing other users' thresholds, to minimize its cost function. The important questions to answer include whether such an algorithm can converge? If it does, where does it converges to and how efficient is the equilibrium point? We study this problem from a game perspective. In particular, each user n optimizes its own cost function $Q(B_n)/\lambda + k\beta^2\pi(B_n)$ given a fixed server utilization β. $\mathbf{B}^* \triangleq (B_n^*)_{n=1}^N$ is defined to be the *Nash Equilibrium (NE)* of the system (when it exists) if

$$B_n^* \in \arg\min_{B} Q(B)/\lambda + k\beta^2\pi(B) \tag{2}$$

$$\text{and} \quad \beta = \lambda \sum_{n=1}^{N} \pi(B_n^*)/(Nc).$$

Note the cost function in (2) is different from (1) because the N-player game defined by (1) is difficult to solve so we approach the problem using a mean-field approximation (or large-system approach) where we assume that each user's choice of the threshold has the minimal impact on the server utilization β, so each user views the server utilization as a fixed constant when optimizing its threshold. The NE has to satisfy two conditions: (i) the threshold is optimal given the server utilization (*optimality condition*) (ii) the server utilization is indeed the one under the chosen thresholds from all users (*consistency condition*).

We define the *Price of Anarchy (PoA)* to be the performance gap between cost under the NE offloading decision and the global minimum cost, i.e.,

$$\text{PoA} \triangleq 1 - \frac{\text{Global minimum cost}}{\text{Average cost under NE offloading decision}}.$$

Note that PoA $\in [0, 1]$. The smaller the PoA, the more efficient the system under the NE offloading decision.

III. Algorithm Design and Main Results

In this section, we first propose a distributed offloading algorithm that incrementally updates the threshold for each user. Then, we present our main theoretical results on the performance of the proposed algorithm.

A. Algorithm Description

In this subsection, we introduce an Iterative Threshold Update (ITU) algorithm that constantly updates each user's threshold. Let $B_n^{(m)}$ be the threshold of user n in the m^{th} iteration. Motivated by the fact that the server utilization asymptotically equal to $\beta(\mathbf{B}^{(m)})$ as $N \to \infty$ at the beginning of the $(m+1)^{th}$ iteration, we define the *approximate average cost* of user n given the server utility $\beta(\mathbf{B}^{(m)})$ in the m^{th} iteration as

$$T_n\left(B_n; \mathbf{B}^{(m)}\right) \triangleq \frac{Q(B_n)}{\lambda} + k\beta^2\left(\mathbf{B}^{(m)}\right)\pi(B_n),$$

where $\mathbf{B}^{(m)} \triangleq \left(B_n^{(m)}\right)_{n=1}^N$.

Algorithm 1 Iterative Threshold Update (ITU) Algorithm

Each user starts from some random threshold $B_n^{(0)}$, where $n = 1, 2, \cdots, N$;
for $m = 0, 1, 2 \cdots$, **do**
  for $n = 1, 2, \cdots, N$ **do**

$$\widehat{B}_n^{(m+1)} \in \arg\min_{B_n} T_n\left(B_n; \mathbf{B}^{(m)}\right). \tag{3}$$

    if $m = 0$ **then**
      $B_n^{(m)} \leftarrow \widehat{B}_n^{(m+1)}$
    else
      if $\widehat{B}_n^{(m+1)} < B_n^{(m)}$ **then**
        $B_n^{(m+1)} \leftarrow B_n^{(m)} - 1$;
      else if $\widehat{B}_n^{(m+1)} > B_n^{(m)}$ **then**
        $B_n^{(m+1)} \leftarrow B_n^{(m)} + 1$;
      else
        $B_n^{(m+1)} \leftarrow B_n^{(m)}$.
      end if
    end if
  end for
end for

We describe our proposed ITU algorithm in Algorithm 1, where each user greedily optimizes its own decision in the first iteration step to speed up the convergence of the ITU algorithm and then gradually adjusts its threshold. Here, the optimal solution to (3) requires the knowledge of the server utilization, which relies on all users' offloading decisions and thus is typically unavailable beforehand. However, the server utilization can be estimated via the ratio of the average offloading rate (i.e., the ratio of the total number of offloaded tasks and the total amount of time) to the total service rate of cloud servers. Moreover, users in the system may update their offloading decisions asynchronously. In Section IV, we demonstrate via simulations that our proposed ITU algorithm still performs well in the presence of imperfect server utilization estimation and asynchronous threshold updates.

We are interested in whether the proposed ITU algorithm converges and which offloading decisions it converges to if it does converge. We analytically answer these two questions when the service time of each task is independently and identically distributed (i.i.d) and exponentially distributed with mean $1/\mu$. In such a case, when the threshold B_n of user n is fixed, the queue at a device is an $M/M/1/B_n$ queue (see [24]), which has a Poisson arrival process with the rate λ, exponentially distributed service time with mean $1/\mu$, and a finite buffer size B_n. Therefore, the average queue-length $Q(B_n)$ and probability $\pi(B_n)$ that an incoming task

is uploaded to cloud servers (also referred to as *offloading probability*) have the following closed-forms:

$$Q(B_n) = \begin{cases} \frac{B_n+1}{\rho^{B_n+1}-1} + B_n + \frac{1}{1-\rho}, & \rho \neq 1, \\ \frac{B_n}{2}, & \rho = 1, \end{cases} \tag{4}$$

$$\text{and} \quad \pi(B_n) = \begin{cases} \frac{\rho^{B_n} - \rho^{B_n+1}}{1-\rho^{B_n+1}}, & \rho \neq 1, \\ \frac{1}{B_n+1}, & \rho = 1, \end{cases} \tag{5}$$

respectively, where $\rho \triangleq \lambda/\mu > 0$.

B. Main Results

In this subsection, we analyze the performance of our proposed ITU algorithm under the exponential service time distribution assumption. We first characterize the sufficient and necessary conditions for the existence and uniqueness of the NE. Then, we show that the proposed ITU algorithm converges to the unique NE within a finite time when it exists. Finally, we characterize the efficiency of NE offloading decision via the PoA performance metric in some scenarios.

Theorem 1: If $W(0) < k\lambda^2/c^2$ and $V_1(\lfloor \widetilde{x} \rfloor) < k\lambda^2/c^2 < V_2(\lceil \widetilde{x} \rceil)$, then there is no NE. Otherwise, there exists a unique NE, in particular,

(i) if $W(0) \geq k\lambda^2/c^2$, then the unique NE is $(0)_{N\times 1}$;
(ii) if $W(0) < k\lambda^2/c^2$ and $W(\lfloor \widetilde{x} \rfloor) < k\lambda^2/c^2 \leq V_1(\lfloor \widetilde{x} \rfloor)$, then the unique NE is $(\lfloor \widetilde{x} \rfloor)_{N\times 1}$;
(iii) if $W(0) < k\lambda^2/c^2$ and $V_2(\lceil \widetilde{x} \rceil) \leq k\lambda^2/c^2 < W(\lceil \widetilde{x} \rceil)$, then the unique NE is $(\lceil \widetilde{x} \rceil)_{N\times 1}$.

In the statement above, $\widetilde{x}$ is the unique solution to $W(\widetilde{x}) = k\lambda^2/c^2$ when $W(0) < k\lambda^2/c^2$, and $W(x)$, $V_1(x)$, and $V_2(x)$ are defined as follows:

$$W(x) = \begin{cases} \frac{(1-\rho^{x+1})^2}{\lambda(1-\rho)^3\rho^{2x-1}} \left(x + 1 - \frac{\rho^{x+1}-1}{\log(\rho)} \right), & \rho \neq 1, \\ \frac{(x+1)^4}{2\lambda}, & \rho = 1. \end{cases} \tag{6}$$

$$V_1(x) \triangleq \frac{k\lambda^2}{c^2} \left| \frac{C_L(x+1) - C_L(x)}{C_E(x+1;x) - C_E(x;x)} \right| \tag{7}$$

$$\text{and } V_2(x) \triangleq \frac{k\lambda^2}{c^2} \left| \frac{C_L(x) - C_L(x-1)}{C_E(x;x) - C_E(x-1;x)} \right|, \tag{8}$$

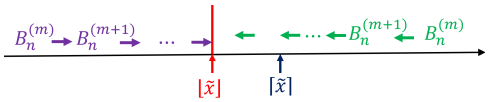
where $C_L(x) \triangleq Q(x)/\lambda$ denotes the average local computation cost and $C_E(x;y) \triangleq k\left(\lambda\pi(y)/c\right)^2 \cdot \pi(x)$ represents the average service cost.[1]

Proof: The proof is available in Appendix A. ∎

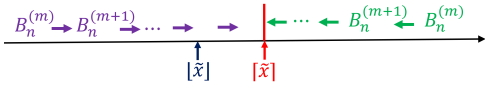
To apply Theorem 1, we only need to examine the value of $k\lambda^2/c^2$ to check whether the NE exists or not given the system parameters. Note that the term $k\lambda^2/c^2$ denotes the cost of using cloud servers when all the computing tasks are uploaded to cloud servers. Therefore, if $k\lambda^2/c^2 \leq W(0)$, then the cost of using cloud servers is small and thus it is better to upload all the tasks to cloud servers (i.e., $\widetilde{x} = 0$). Otherwise, each user partially uploads computing traffic to the cloud servers with the goal of minimizing its own cost (i.e., $\widetilde{x} > 0$). In addition, when the NE exists, we can further quantify the NE. The next theorem shows that the proposed ITU algorithm converges to the unique NE over a finite number of iterations when the NE exists.

Theorem 2: If the unique NE exists, then the proposed ITU algorithm converges to it over a finite number of iterations.

Proof: The proof relies on the following key property: after each iteration of the ITU algorithm, each user's threshold will get closer and closer to the NE, as it is shown in Fig. 2. Fig. 2 illustrates the convergence of each user's threshold in the case when $\lfloor \widetilde{x} \rfloor$ is the NE and the case when $\lceil \widetilde{x} \rceil$ is the NE, respectively, where we recall that $\widetilde{x}$ is the solution to (9) if it exists.



(a) $(\lfloor \widetilde{x} \rfloor)_{N\times 1}$ is NE



(b)$(\lceil \widetilde{x} \rceil)_{N\times 1}$is NE

Fig. 2: Convergence of the n^{th} user's thresholds.

Then, in both cases, the updated threshold of each user exhibits bisection property, i.e., if $B_n^{(m)} < \widetilde{x}$, then the threshold will increase by one in each iteration until it reaches the NE. If $B_n^{(m)} > \widetilde{x}$, then the threshold will decrease by one in each iteration until it converges to the NE. Therefore, we either increase or decrease the current threshold by one in each iteration to ensure the convergence of ITU algorithm. Please see Appendix B for the detailed proof. ∎

Finally, we characterize PoA of the NE when it exists, which captures the efficiency of our proposed ITU algorithm when it converges. In particular, we provide conditions under which the proposed ITU algorithm is optimal, i.e. the PoA is zero.

Theorem 3: When the NE exists, PoA is zero when $W(0) \geq k\lambda^2/c^2$, and converges to zero as $k \to \infty$ when $W(0) < k\lambda^2/c^2$ and $0 < \rho < 1$.

Proof: The proof consists of the following two cases:

(i) If $W(0) \geq k\lambda^2/c^2$, according to Theorem 1, the unique NE is $(0)_{N\times 1}$ and thus it is optimal to offload all the tasks to the cloud server. We can also show that in such a case, the thresholds of the global optimal solution for all users are 0 using a similar argument and is omitted here due to space limitations. Therefore, PoA $= 0$ in this case.

(ii) If $W(0) < k\lambda^2/c^2$ and $0 < \rho < 1$, we first obtain the following upper bound on PoA using its definition:

$$\text{PoA} \leq 1 - \frac{(1-\rho)\log(\rho)}{3(\rho\log(\rho) + \rho^{\widetilde{x}+2}(1-\rho))} \cdot \left(\frac{2x^*+2}{\rho^{x^*+1}-1} + 2x^* + \frac{2+\rho}{1-\rho} + \frac{\rho^{x^*+1}}{\log(\rho)} \right),$$

where x^* is the unique solution to the equation $k\lambda^2/c^2 = W(x^*)/3$. In Appendix C, we further show that this upper

[1] In this paper, $\lfloor y \rfloor$ and $\lceil y \rceil$ denote the maximum integer that is not greater than y and the minimum integer that is not less than y, respectively, $(y)_{N\times 1}$ denotes $N-$ dimensional vector with all y values.

bound converges to 0 as $k \to \infty$. This, together with the fact that PoA ≥ 0, implies the desired result. The detailed proof is in Appendix C. ∎

Note that these theoretical results are obtained under the assumption that the service time follows an exponential distribution. In the next section, we perform simulations to validate our theoretical results and to demonstrate the efficiency of our proposed ITU algorithm under various practical scenarios, such as general service time distribution, imperfect server utilization estimations, and asynchronous threshold updates.

IV. Simulations

In this section, we perform simulations to validate our theoretical findings, especially conditions for the existence and uniqueness of the NE (cf. Theorem 1) and the convergence of our proposed ITU algorithm (cf. Theorem 2) under the exponential service time distribution with $\mu = 4$. We also demonstrate the convergence property of the proposed ITU algorithm when the service time follows a hyperexponential distribution, i.e., it follows an exponential distribution with a rate of $8p$ with probability p and another exponential distribution with a rate of $8(1-p)$ otherwise. Note that the mean of the hyperexponential distribution is $1/4$ and the variance is $1/(8p(1-p)) - 1/16$. Finally, we evaluate the efficiency of the NE via the PoA performance that characterizes the gap between the cost under the NE and the global minimum cost. In our simulations, we consider $N = 1000$ users, each of which has the Poisson arrival process with the rate of $\lambda = 6$, and $c = 4$ unless we explicitly mention it.

A. Existence of the NE

In this subsection, we perform numerical simulations to validate the conditions such that the NE exists under the exponential service time distribution. We consider three different values of k, i.e., $k = 22$, $k = 30$, and $k = 40$. These corresponds to the case with $W(\lfloor \widetilde{x} \rfloor) < k\lambda^2/c^2 \leq V_1(\lfloor \widetilde{x} \rfloor)$, $V_1(\lfloor \widetilde{x} \rfloor) < k\lambda^2/c^2 \leq V_2(\lfloor \widetilde{x} \rfloor)$, and $V_2(\lceil \widetilde{x} \rceil) < k\lambda^2/c^2 \leq V_1(\lceil \widetilde{x} \rceil)$, respectively, under which the unique NE is $(2)_{N\times 1}$, NE does not exist, and the unique NE is $(3)_{N\times 1}$, according to Theorem 1. Fig. 3 shows the optimal threshold value of one user assuming users adopt the same threshold under above three different cases. From Fig. 3a, and Fig. 3c, we can see that there exists a unique NE $(2)_{N\times 1}$ when $k = 22$, and NE $(3)_{N\times 1}$ when $k = 40$, which means that the optimal threshold of one user is the same as all other users' threshold. However, we can observe from Fig. 3b that there does not exist a NE when $k = 30$. This validates the conditions for the existence and uniqueness of the NE, as shown in Theorem 1.

B. Convergence under the ITU Algorithm

In this subsection, we perform simulations to validate the convergence of the ITU algorithm. We randomly select 5 users to study their convergences. Fig. 4 shows the convergence property of the ITU algorithm when the calculation of the server utilization uses the exact offloading probability (cf. (5)). We can see from Fig. 4a and Fig. 4c that our proposed ITU algorithm can quickly converge to the corresponding NE. The NE does not exist in the setup for Fig. 4b, in which case the updated threshold under the ITU algorithm oscillates between 2 and 3. This indicates the bisection property of the updated threshold of ITU. This validates the convergence property of the ITU algorithm, as revealed in Theorem 2.

In practice, the service time may not follow the exponential distribution. In addition, the knowledge of the server utilization is not available beforehand and requires to estimate over time. As such, we use the ratio between the average offloading rate (i.e., the ratio of the total number of offloaded tasks and the total amount of time) and the total service rate of cloud servers. Moreover, each user may not synchronously update its threshold.

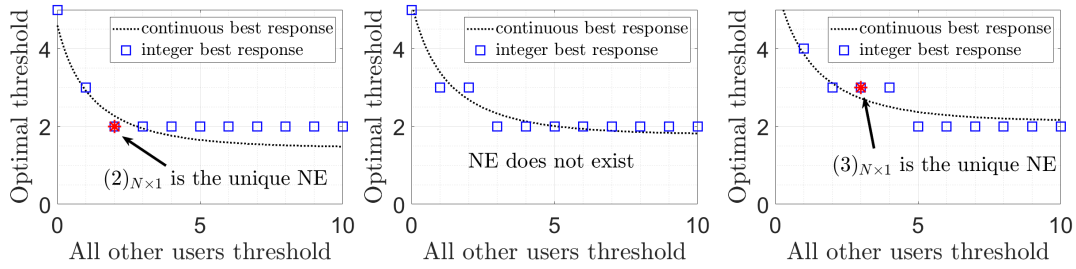
To this end, we consider a hyperexponential distribution service time distribution with $p = 1/8$. Each user updates its threshold asynchronously (i.e., updates with a fixed probability) to optimize its cost function in the presence of imperfect server utilization estimation. Note that we do not know the theoretical NE and thus we first perform numerical simulations to find the NE by plotting the optimal threshold value with respect to other users' threshold, as shown in Fig. 5. From Fig. 5, we can observe that the NE is $(3)_{N\times 1}$ when $k = 40$, while the NE does not exist when $k = 30$. Fig. 6 shows the convergence of the ITU algorithm under the hyperexponential distribution service time distribution together with asynchronous threshold update and imperfect server utilization estimation. From Fig. 6, we can observe that the updated threshold converges to the corresponding NE when the NE exists, and oscillates between 2 and 3 otherwise.

C. Price of Anarchy

In this subsection, we perform simulations to evaluate the efficiency of the NE via the PoA performance under three different cases: $\rho = 0.75$ (i.e., $\lambda = 3$), $\rho = 1$ (i.e., $\lambda = 4$), and $\rho = 1.25$ (i.e., $\lambda = 5$). We consider both exponential service time distribution and hyperexponential distribution service time distribution with $p \in \{1/4, 1/8, 1/16\}$. In Fig. 7, we plot PoA performance with respect to k varying from 0 to 100. Here, we ignore the trivial case with $k\lambda^2/c^2 \leq W(0)$, where PoA is always equal to zero. We can see from Fig. 7 that both the PoA under different service time distributions share the similar properties. In addition, PoA only exists in certain range of k since system parameters have a significant impact on the existence of the NE (see Theorem 1). From Fig. 7, we can also observe that PoA < 0.3 in all of our simulation scenarios, which implies that our proposed ITU algorithm is at least 70% efficient compared to the global optimal solution.

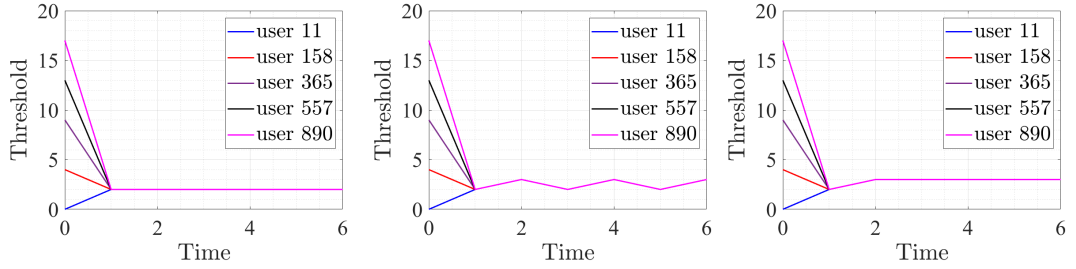
V. Conclusion

In this paper, we proposed a distributed threshold-based offloading algorithm so that each user gradually updates its own threshold with the goal of minimizing its own cost function consisting of average processing delay and the cost of using the cloud services depending on the server utilization in large-scale mobile cloud computing. We then characterized



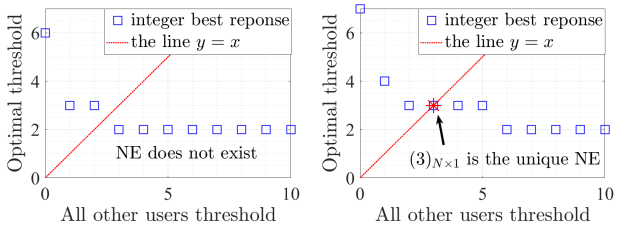
(a) $k = 22$ (i.e., $W(\lfloor \widetilde{x} \rfloor) < \frac{k\lambda^2}{c^2} \leq V_1(\lfloor \widetilde{x} \rfloor)$) (b) $k = 30$ (i.e., $V_1(\lfloor \widetilde{x} \rfloor) < \frac{k\lambda^2}{c^2} < V_2(\lceil \widetilde{x} \rceil)$) (c) $k = 40$ (i.e., $V_2(\lceil \widetilde{x} \rceil) \leq \frac{k\lambda^2}{c^2} < W(\lceil \widetilde{x} \rceil)$)

Fig. 3: Conditions for the existence and uniqueness of NE.



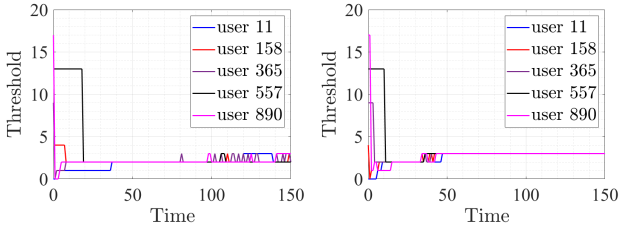
(a) $k = 22$ (i.e., $W(\lfloor \widetilde{x} \rfloor) < \frac{k\lambda^2}{c^2} \leq V_1(\lfloor \widetilde{x} \rfloor)$) (b) $k = 30$ (i.e., $V_1(\lfloor \widetilde{x} \rfloor) < \frac{k\lambda^2}{c^2} < V_2(\lceil \widetilde{x} \rceil)$) (c) $k = 40$ (i.e., $V_2(\lceil \widetilde{x} \rceil) \leq \frac{k\lambda^2}{c^2} < W(\lceil \widetilde{x} \rceil)$)

Fig. 4: Convergence of the ITU algorithm under exponential service time.



(a) $k = 30$ (b) $k = 40$

Fig. 5: The best response threshold under the hyperexponential distribution service time distribution.



(a) $k = 30$ (b) $k = 40$

Fig. 6: The convergence of the ITU algorithm under the hyperexponential distribution service time distribution.

the sufficient and necessary conditions for the existence and uniqueness of the Nash Equilibrium offloading decision under the exponential service time distribution. Furthermore, we showed the convergence of our proposed distributed algorithm to Nash Equilibrium when it exists. Finally, we perform extensive simulations to confirm our theoretical findings and exhibit the efficiency of our proposed algorithm under various practice scenarios such as general service time distributions, imperfect server utilization estimation, and asynchronous threshold updates.

Appendix A
Proof of Theorem 1

We first consider the cases when the NE exists and then the case when NE does not exist. Hence, we first consider the following three cases:

Case (i) $W(0) \geq k\lambda^2/c^2$: In such a case, given all other users threshold y, the best response is 0. This can be verified by monotonic increasing property of cost function $T(x; y)$ when $W(0) \geq k\lambda^2/c^2$. The proof follows from the basic calculus and thus is omitted due to the lack of space.

Then, we need two lemmas to prove the second case, whose proofs are available at the end of this section.

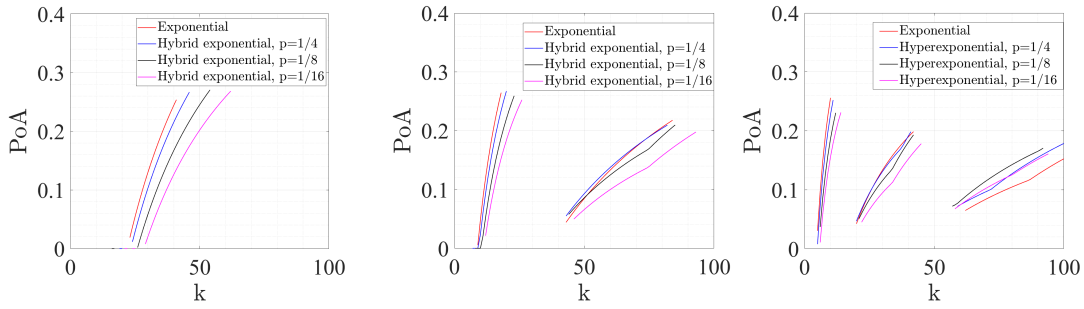
Lemma 1: If $W(0) < k\lambda^2/c^2$ and given all other user's offloading decisions $\widetilde{x}$, then the best response is also $\widetilde{x}$, where $\widetilde{x}$ is the unique solution to

$$W(\widetilde{x}) = k\lambda^2/c^2. \tag{9}$$

Lemma 2: $W(x)$, $V_1(x)$ and $V_2(x)$ satisfy the following relationship:

$$W(x) < V_1(x) < V_2(x+1) < W(x+1), \quad \forall x \geq 0.$$

Lemma 3: Functions $C_L(x)$ and $C_E(x; y)$ (their definitions are defined in Theorem 1) are strictly increasing and strictly decreasing on the interval $[0, \infty)$ independently of all other users' offloading decisions y, respectively.



(a) $\rho = 0.75$ (b) $\rho = 1$ (c) $\rho = 1.25$

Fig. 7: PoA performance.

Lemma 4: Given all other users' offloading decisions $\widetilde{B}$, if $\widetilde{B} \notin \{\lfloor \widetilde{x} \rfloor, \lceil \widetilde{x} \rceil\}$, then NE does not exist.

From Lemma 1, we can see that if $W(0) < k\lambda^2/c^2$, there exists the unique solution $\widetilde{x}$ to equation (9), i.e., $k\lambda^2/c^2 = W(\widetilde{x})$. Therefore, according to the monotonic increasing property of $W(x)$ (cf. Lemma 2), we have $W(\lfloor \widetilde{x} \rfloor) < k\lambda^2/c^2 < W(\lceil \widetilde{x} \rceil)$ when $\widetilde{x}$ is not an integer. Next, we characterize the conditions for the existence and uniqueness of the NE by considering a partition of the interval $(W(\lfloor \widetilde{x} \rfloor), W(\lceil \widetilde{x} \rceil))$. Therefore, we consider the following cases under the condition $W(0) < k\lambda^2/c^2$.

Case (ii) $W(\lfloor x \rfloor) < k\lambda^2/c^2 \leq V_1(\lfloor \widetilde{x} \rfloor)$: In such a case, we would like to show that $(\lfloor \widetilde{x} \rfloor)_{N\times 1}$ is the unique NE. From Lemma 4, we know that the NE must be either $(\lfloor \widetilde{x} \rfloor)_{N\times 1}$ or $(\lceil \widetilde{x} \rceil)_{N\times 1}$. Therefore, it is sufficient to show that $T(\lfloor \widetilde{x} \rfloor; \lfloor \widetilde{x} \rfloor) \leq T(\lceil \widetilde{x} \rceil; \lfloor \widetilde{x} \rfloor)$, i.e., the best response of an individual user is $\lfloor \widetilde{x} \rfloor$ given all other users' offloading decisions $\lfloor \widetilde{x} \rfloor$. Indeed, according to the condition $k\lambda^2/c^2 \leq V_1(\lfloor \widetilde{x} \rfloor)$ and the definition of $V_1(x)$ (cf. (7)), we have

$$\frac{k\lambda^2}{c^2} \leq \frac{k\lambda^2}{c^2}\left|\frac{C_L(\lceil \widetilde{x} \rceil) - C_L(\lfloor \widetilde{x} \rfloor)}{C_E(\lceil \widetilde{x} \rceil; \lfloor \widetilde{x} \rfloor) - C_E(\lfloor \widetilde{x} \rfloor; \lfloor \widetilde{x} \rfloor)}\right|,$$

By using Lemma 3, this immediately implies that

$$C_E(\lfloor \widetilde{x} \rfloor; \lfloor \widetilde{x} \rfloor) - C_E(\lceil \widetilde{x} \rceil; \lfloor \widetilde{x} \rfloor) \leq C_L(\lceil \widetilde{x} \rceil) - C_L(\lfloor \widetilde{x} \rfloor).$$

By rearranging items of the above inequality, we have

$$C_L(\lfloor \widetilde{x} \rfloor) + C_E(\lfloor \widetilde{x} \rfloor; \lfloor \widetilde{x} \rfloor) \leq C_L(\lceil \widetilde{x} \rceil) + C_E(\lceil \widetilde{x} \rceil; \lfloor \widetilde{x} \rfloor),$$

i.e., $T(\lfloor \widetilde{x} \rfloor; \lfloor \widetilde{x} \rfloor) \leq T(\lceil \widetilde{x} \rceil; \lfloor \widetilde{x} \rfloor)$.

Case (iii) $V_2(\lceil \widetilde{x} \rceil) \leq k\lambda^2/c^2 < W(\lceil \widetilde{x} \rceil)$: In such a case, we would like to show that $(\lceil \widetilde{x} \rceil)_{N\times 1}$ is the unique NE. Again following Lemma 4, the NE is either $(\lfloor \widetilde{x} \rfloor)_{N\times 1}$ or $(\lceil \widetilde{x} \rceil)_{N\times 1}$. Therefore, it is sufficient to show that $T(\lceil \widetilde{x} \rceil; \lceil \widetilde{x} \rceil) \leq T(\lfloor \widetilde{x} \rfloor; \lceil \widetilde{x} \rceil)$, i.e., the best response of an individual user is $\lceil \widetilde{x} \rceil$ given all other users' offloading decisions $\lceil \widetilde{x} \rceil$. Indeed, according to the condition $V_2(\lceil \widetilde{x} \rceil) \leq k\lambda^2/c^2$ and the definition of $V_2(x)$ (cf. (8)), we have

$$\frac{k\lambda^2}{c^2}\left|\frac{C_L(\lceil \widetilde{x} \rceil) - C_L(\lfloor \widetilde{x} \rfloor)}{C_E(\lceil \widetilde{x} \rceil; \lceil \widetilde{x} \rceil) - C_E(\lfloor \widetilde{x} \rfloor; \lceil \widetilde{x} \rceil)}\right| \leq \frac{k\lambda^2}{c^2}.$$

By using Lemma 3 again, we have

$$C_L(\lceil \widetilde{x} \rceil) - C_L(\lfloor \widetilde{x} \rfloor) \leq C_E(\lfloor \widetilde{x} \rfloor; \lceil \widetilde{x} \rceil) - C_E(\lceil \widetilde{x} \rceil; \lceil \widetilde{x} \rceil),$$

which immediately implies the desired result.

Finally, we will show the case when NE does not exist. In this case, we have $V_1(\lfloor \widetilde{x} \rfloor) < k\lambda^2/c^2 < V_2(\lceil \widetilde{x} \rceil)$. Indeed, by following the same arguments in the previous two cases, we are able to show that the best response of an individual user is $\lceil \widetilde{x} \rceil$ and $\lfloor \widetilde{x} \rfloor$ given all other users' offloading decisions $\lfloor \widetilde{x} \rfloor$ and $\lceil \widetilde{x} \rceil$, respectively. ■

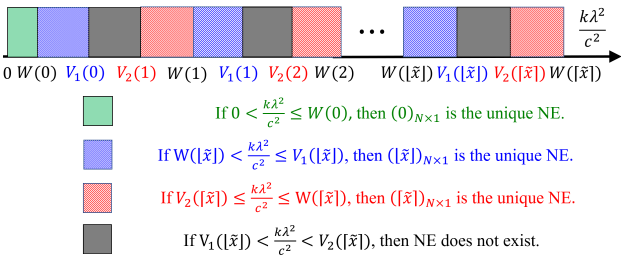


Fig. 8: Conditions for the existence and uniqueness of NE.

Fig. 8 summarizes the sufficient and necessary conditions of the existence and uniqueness of the NE.

The proofs of Lemma 2 and Lemma 3 follow from basic calculus and thus are omitted due to space limit. Next, we prove Lemma 1 and Lemma 4 to complete the proof.

Proof of Lemma 1: Here, we only provide the proof when $\rho > 1$ and omit the other cases for brevity since they share a similar proving procedure.

From the definition of $W(x)$ (cf. (6)), we have:

$$T(x;\widetilde{x}) = \frac{1}{\lambda}\left(\frac{x+1}{\rho^{x+1}-1} + x + \frac{1}{1-\rho}\right) + \frac{k\lambda^2}{c^2}\left(\frac{\rho^{\widetilde{x}} - \rho^{\widetilde{x}+1}}{1-\rho^{\widetilde{x}+1}}\right)^2 \cdot \frac{\rho^x - \rho^{x+1}}{1-\rho^{x+1}}. \quad (10)$$

Taking derivative of $T(x;\widetilde{x})$ with respect to x, then set $x = \widetilde{x}$ and let $dT(x;\widetilde{x})/dx = 0$, we have

$$\frac{k\lambda^3(1-\rho)^3}{c^2\rho^3}\left(\frac{\rho^{\widetilde{x}+1}}{\rho^{\widetilde{x}+1}-1}\right)^2 + \frac{\rho^{\widetilde{x}+1}-1}{\log(\rho)} = \widetilde{x}+1, \quad (11)$$

where $\log(\cdot)$ is the logarithm with the natural base e.

Next, we will find the condition such that the equation (11) has one unique solution. To simply the notations, we let $a = k\lambda^3(1-\rho)^3/(c^2\rho^3)$, $b = 1/\log(\rho)$ and $u = \rho^{x+1} - 1$. Then we rewrite (11) as follows.

$$h_1(u) = h_2(u),$$

where

$$h_1(u) \triangleq a\left(1+\frac{1}{u}\right)^2 + bu,\ u > -1$$
$$\text{and } h_2(u) \triangleq \log_\rho(u+1),\ u > -1.$$

Let $h_d(u)$ denote the difference between $h_1(u)$ and $h_2(u)$, i.e.,

$$h_d(u) \triangleq h_1(u) - h_2(u).$$

By taking derivative of $h_d(u)$ we can show that $h_d(u)$ is strictly increasing when $u \in [\rho-1, \infty)$.

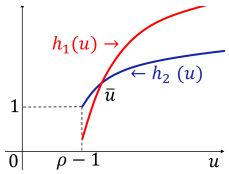


Fig. 9: Relations between $h_1(u)$ and $h_2(u)$: $\rho > 1$ and $k\lambda^2/c^2 > W(0)$

If $k\lambda^2/c^2 > W(0)$, then we have $h_1(\rho-1) < h_2(\rho-1)$, as shown in Fig. 9. Thus, we have $h_d(\rho-1) < 0$. Since $h_d(u)$ is strictly increasing in $u \in [\rho-1, \infty)$, then there must exist a unique $\widetilde{u} > \rho-1$ satisfying $h_d(\widetilde{u}) = 0$. Therefore, there exists some $\widetilde{x} = \log(\widetilde{u}+1) - 1$, which is the solution to

$$\frac{dT(x;\widetilde{x})}{dx} = \frac{u+1}{b\lambda u^2} h_d(\widetilde{u}) = 0.$$

Next, we will show that such $\widetilde{x}$ is the unique best response of $T(x;\widetilde{x})$ given all other users' offloading decisions $\widetilde{x}$. Indeed, if $x \in [0, \widetilde{x})$, then we have $u \in [\rho-1, \widetilde{u})$. Hence, we have

$$\frac{dT(x;\widetilde{x})}{dx} = \frac{u+1}{b\lambda u^2}\left(a\left(1+\frac{1}{\widetilde{u}}\right)^2 + bu - h_2(u)\right)$$
$$\overset{(a)}{<} \frac{u+1}{b\lambda u^2} h_d(u) \overset{(b)}{<} 0,$$

where step (a) follows from the fact that $b = 1/\log(\rho) > 0$ implies $(u+1)/(b\lambda u^2) > 0$ and $(1+1/\widetilde{u})^2 < (1+1/u)^2$; step (b) follows from the fact that $h_d(u) < 0$ for all $u \in [\rho-1, \widetilde{u})$, which follows from the fact that $h_d(u)$ is increasing in $[\rho-1, \infty)$ and the fact that $h_d(\widetilde{u}) = 0$.

Similarly, we can show that $dT(x;\widetilde{x})/dx > 0, \forall x \in [\widetilde{x}, \infty)$. Thus, $T(x;\widetilde{x})$ is decreasing in $x \in [0, \widetilde{x})$ and is increasing in $x \in [\widetilde{x}, \infty)$. Therefore, $\widetilde{x}$ is the unique solution to $k\lambda^2/c^2 = W(\widetilde{x})$. ■

Proof of Lemma 4: Here, we show the case when $\rho \neq 1$. The case when $\rho = 1$ follows from the similar arguments and thus omits here.

We want to show that $(\widetilde{B})_{N\times 1}$ is not the NE when $\widetilde{B} \notin \{\lfloor\widetilde{x}\rfloor, \lceil\widetilde{x}\rceil\}$. To that end, we consider the best response of an individual user given all other users' integer offloading decisions $\widetilde{B}$, denoted by $x_{\widetilde{B}}$, where $x_{\widetilde{B}}$ is real number and satisfies the following equation.

$$\rho\left(\rho^{x_{\widetilde{B}}+1} - 1\right) - (x_{\widetilde{B}}+1)\rho\log(\rho) = \frac{k\lambda^3(\rho-1)^3\log(\rho)}{c^2} \cdot \left(\frac{\rho^{\widetilde{B}}}{\rho^{\widetilde{B}+1}-1}\right)^2, \quad (12)$$

where $\rho \neq 1$, which is obtained by setting $dT(x_{\widetilde{B}};\widetilde{B})/dx_{\widetilde{B}} = 0$. It can be shown shortly that the best response $x_{\widetilde{B}}$ is decreasing with respect to $\widetilde{B}$. Since $\widetilde{B} \notin \{\lfloor\widetilde{x}\rfloor, \lceil\widetilde{x}\rceil\}$, we have $\widetilde{B} < \lfloor\widetilde{x}\rfloor$ or $\widetilde{B} > \lceil\widetilde{x}\rceil$. Then, we have the following two different cases:

- If $\widetilde{B} < \lfloor\widetilde{x}\rfloor < \widetilde{x}$, then according to the monotonic decreasing property of the best response $x_{\widetilde{B}}$, we have $x_{\widetilde{B}} > \widetilde{x}$, where we use the fact that the best response of an individual user is $\widetilde{x}$ given all other users' offloading decisions $\widetilde{x}$. This implies that $\lfloor x_{\widetilde{B}}\rfloor \geq \lfloor\widetilde{x}\rfloor > \widetilde{B}$ and hence $(\widetilde{B})_{N\times 1}$ is not a NE.
- If $\widetilde{B} > \lceil\widetilde{x}\rceil > \widetilde{x}$, then following the same arguments as in the case of $\widetilde{B} < \lfloor\widetilde{x}\rfloor < \widetilde{x}$, we again can show that $(\widetilde{B})_{N\times 1}$ is not a NE.

Next, we show the monotonic decreasing property of $x_{\widetilde{B}}$ with respect to $\widetilde{B}$ when $\rho \neq 1$ to complete the proof.

We first rearrange terms in (12),

$$f(x_{\widetilde{B}}) = g(\widetilde{B}), \quad (13)$$

where

$$f(x) \triangleq \rho^{x+1} - 1 - (x+1)\log(\rho),$$
$$\text{and } g(x) \triangleq \frac{k\lambda^3(\rho-1)^3\log(\rho)}{c^2\rho}\left(\frac{\rho^x}{\rho^{x+1}-1}\right)^2,$$

for all $x \geq 0$. It can be easily shown by calculus that $f(x)$ and $g(x)$ are strictly decreasing and increasing, respectively. The proofs are omitted due to the lack of space. Then, from (13), we have

$$f\left(x_{\widetilde{B}+1}\right) - f\left(x_{\widetilde{B}}\right) = g_2\left(\widetilde{B}+1\right) - g\left(\widetilde{B}\right) < 0,$$

where the last step follows from the monotonic decreasing and increasing property of $f(x)$ and $g(x)$, respectively. Hence, we have $x_{\widetilde{B}+1} < x_{\widetilde{B}}$ holds for any non-negative integer $\widetilde{B}$. ■

Appendix B
Proof of Theorem 2

From Theorem 1, we know that if the NE exists, it is either $(0)_{N\times 1}$, $(\lfloor\widetilde{x}\rfloor)_{N\times 1}$ or $(\lceil\widetilde{x}\rceil)_{N\times 1}$. Hence, We will consider these three cases, respectively.

(i) $(0)_{N\times 1}$ is the NE: In this case, we have $k\lambda^2/c^2 \leq W(0)$. (cf. Theorem 1) and $T(x;\widetilde{x})$ is increasing with respect to x, which can be easily verified by taking derivative with respect to x. Therefore, we have $\widehat{B}_n^{(m+1)} = 0$. Then, for any $B_n^{(1)} > 0$, the threshold will decrease by one in each iteration and goes to zero within $B_n^{(1)} + 1$ steps .

In order to prove the convergence in the other two cases, we need the following lemma that shows the bisection property of the updated threshold under the ITU algorithm.

Lemma 5: If $k\lambda^2/c^2 > W(0)$, then for any $\widetilde{x} > 0$, where $\widetilde{x}$ satisfies $k\lambda^2/c^2 = W(\widetilde{x})$, and $m \geq 1$, we have:

(i) If $B_n^{(m)} < \lfloor \widetilde{x} \rfloor$ or $B_n^{(m)} = \lfloor \widetilde{x} \rfloor$ but $(\lfloor \widetilde{x} \rfloor)_{N\times 1}$ is not NE, then $\widehat{B}_n^{(m+1)} \geq B_n^{(m)}$;

(ii) If $B_n^{(m)} > \lceil \widetilde{x} \rceil$ or $B_n^{(m)} = \lceil \widetilde{x} \rceil$ but $(\lceil \widetilde{x} \rceil)_{N\times 1}$ is not NE, then $\widehat{B}_n^{(m+1)} \leq B_n^{(m)}$.

From Lemma 5 we have that for any $B_n^{(m)} < \widetilde{x}$, then in the next iteration, $B_n^{(m+1)}$ will move closer to $\widetilde{x}$. Similarly, for any $B_n^{(m)} > \widetilde{x}$, $B_n^{(m+1)}$ will also move closer to $\widetilde{x}$. In either cases, in each iteration, the threshold will get closer and closer to $\widetilde{x}$, as shown in Fig. 2. Note that in the first iteration (i.e., $m = 0$), all users in the system solve the same optimization problem and obtain the same threshold, since the server utilization is the same for all users. Then, after the first iteration (i.e., $m > 1$), all users will adjust their threshold in the same way and thus we just need to focus on a particular user n. Now, we are ready to prove the convergence of the ITU algorithm when the NE is not $(0)_{N\times 1}$.

(ii) $(\lfloor \widetilde{x} \rfloor)_{N\times 1}$ is the NE: In this case , for any $B_n^{(1)} < \lfloor \widetilde{x} \rfloor < \widetilde{x}$, the threshold will increase by one in each iteration until reaching to $\lfloor \widetilde{x} \rfloor$. For any $B_n^{(1)} > \widetilde{x} > \lfloor \widetilde{x} \rfloor$, the threshold will decrease by one in each iteration until reaching to $\lfloor \widetilde{x} \rfloor$. Thus, it will take at most $\left| \lfloor \widetilde{x} \rfloor - B_n^{(1)} \right| + 1$ iterations for user n's threshold to converge to $\lfloor \widetilde{x} \rfloor$.

(iii) $(\lceil \widetilde{x} \rceil)_{N\times 1}$ is the NE: In this case , for any $B_n^{(1)} < \widetilde{x} < \lceil \widetilde{x} \rceil$, the threshold will increase in each iteration until reaching to $\lceil \widetilde{x} \rceil$. For any $B_n^{(1)} > \lceil \widetilde{x} \rceil > \widetilde{x}$, the threshold will decrease by one in each iteration until reaching to $\lceil \widetilde{x} \rceil$. Thus, it will take at most $\left| \lceil \widetilde{x} \rceil - B_n^{(1)} \right| + 1$ iterations for user n's threshold to converge to $\lceil \widetilde{x} \rceil$.

Next, we prove Lemma 5 to complete the proof.

Proof of Lemma 5: We first define the following two functions:

$$U_1(x) \triangleq \begin{cases} x, & \rho = 1, \\ \rho^{x+2} - (x+1)\rho \log \rho - \rho, & \rho \neq 1. \end{cases}$$

and $$U_2(x) \triangleq \begin{cases} \frac{\sqrt{2k\lambda}\lambda}{c(x+1)} - 1, & \rho = 1, \\ \frac{k\lambda^3(\rho-1)^3 \log \rho}{c^2} \cdot \left(\frac{\rho^x}{1-\rho^{x+1}} \right)^2, & \rho \neq 1. \end{cases}$$

It can be easily showed that function $U_1(x)$ is strictly increasing on $[0, \infty)$ and $U_2(x)$ is strictly decreasing on $[0, \infty)$. Therefore, $U_2(x) - U_1(x)$ is strictly decreasing. The detailed proofs are omitted due to space limit. Now we are ready to prove Lemma 5.

In the ITU algorithm, all users will solve the same optimization problem in the first iteration. Thus, all users will have the same threshold when $m = 1$. Then, for any $m \geq 1$, we can simplify the cost function in (3) as

$$T\left(x; B_n^{(m)}\right) = \frac{Q(x)}{\lambda} + k \left(\frac{\pi\left(B_n^{(m)}\right)}{c} \right)^2 \pi(x), \qquad (14)$$

where $x \geq 0$ is some real number and $B_n^{(m)}$ is the threshold of user n in the m^{th} iteration.

Since we have shown that cost function $T(x; \widetilde{x})$ is decreasing and increasing in $[0, \widetilde{x})$ and $[\widetilde{x}, \infty)$ when $k\lambda^2/c^2 > W(0)$, respectively (cf. Proof of Lemma 1). We notice that $T(x; \widetilde{x})$ (cf. (10)) and $T(x; B_n^{(m)})$ share the similar form. Therefore, we take derivative of $T(x; B_n^{(m)})$ with respect to x and set to zero. Then, we have $U_1(\widehat{x}) = U_2\left(B_n^{(m)}\right)$, where $\widehat{x}$ is a real number such that

$$\left(\frac{dT\left(x; B_n^{(m)}\right)}{dx} \right) \Bigg|_{x=\widehat{x}} = 0.$$

Therefore, we have

$$\widehat{B}_n^{(m+1)} \in \left\{ \lfloor \widehat{x} \rfloor, \lceil \widehat{x} \rceil \right\}. \qquad (15)$$

Note that $\widetilde{x}$ satisfies equation $W(\widetilde{x}) = k\lambda^2/c^2$ and through simple algebraic operations, we have $U_1(\widetilde{x}) = U_2(\widetilde{x})$. Next, we consider the following two different cases:

(i) If $B_n^{(m)} < \lfloor \widetilde{x} \rfloor$ or $B_n^{(m)} = \lfloor \widetilde{x} \rfloor$ but is not NE, then we have $B_n^{(m)} < \widetilde{x}$. Then we have

$$\begin{aligned} & U_1(\widehat{x}) - U_1\left(B_n^{(m)}\right) \\ =& U_2\left(B_n^{(m)}\right) - U_1\left(B_n^{(m)}\right) \\ \overset{(a)}{>}& U_2(\widetilde{x}) - U_1(\widetilde{x}) = 0, \end{aligned}$$

where step (a) follows from the fact $U_2(x) - U_1(x)$ is strictly decreasing. Therefore, we have $U_1(\widehat{x}) > U_1\left(B_n^{(m)}\right)$. Since $U_1(x)$ is strictly increasing, we have $\widehat{x} > B_n^{(m)}$. Thus, by (15), we have $\widehat{B}_n^{(m+1)} \geq B_n^{(m)}$.

(ii) If $B_n^{(m)} > \lceil \widetilde{x} \rceil$ or $B_n^{(m)} = \lceil \widetilde{x} \rceil$ but is not NE, then we have $B_n^{(m)} > \widetilde{x}$. Then we have

$$\begin{aligned} & U_1(\widehat{x}) - U_1\left(B_n^{(m)}\right) \\ =& U_2\left(B_n^{(m)}\right) - U_1\left(B_n^{(m)}\right) \\ \overset{(a)}{<}& U_2(\widetilde{x}) - U_1(\widetilde{x}) = 0, \end{aligned}$$

where step (a) follows from the fact that $U_2(x) - U_1(x)$ is strictly decreasing. Therefore, we have $U_1(\widehat{x}) < U_1\left(B_n^{(m)}\right)$. Since $U_1(x)$ is strictly increasing, we have $\widehat{x} < B_n^{(m)}$. Therefore, by (15), we have $\widehat{B}_n^{(m+1)} \leq B_n^{(m)}$. ■

APPENDIX C
PROOF OF THEOREM 3

Here, we show that the PoA converges to zero as $k \to \infty$. First, we will show that as $k \to \infty$, both $\widetilde{x} \to \infty$ and $x^* \to \infty$. By Theorem 1 and 3, we have $k\lambda^2/c^2 = W(\widetilde{x})$ and $3k\lambda^2/c^2 = W(x^*)$. As $k \to \infty$, we have both $W(\widetilde{x}) \to \infty$ and $W(x^*) \to \infty$. By Lemma 1 we know that $W(x)$ is strictly increasing on $[0, \infty)$. Therefore, both $\widetilde{x} \to \infty$ and $x^* \to \infty$ as $k \to \infty$. Hence, we have $\rho^{\widetilde{x}} \to 0$ and $\rho^{x^*} \to 0$. This combines with the upper bound on PoA, yielding

$$\text{PoA} \leq 1 - \frac{(1-\rho)\log(\rho)}{3(\rho \log(\rho))} \left(-2x^* - 2 + 2x^* + \frac{2+\rho}{1-\rho} \right) = 0. \blacksquare$$

REFERENCES

[1] A. Boukerche, S. Guan, and R. E. D. Grande, "Sustainable offloading in mobile cloud computing: algorithmic design and implementation," *ACM Computing Surveys (CSUR)*, vol. 52, no. 1, pp. 1–37, 2019.

[2] A. J. Ferrer, J. M. Marquès, and J. Jorba, "Towards the decentralised cloud: Survey on approaches and challenges for mobile, ad hoc, and edge computing," *ACM Computing Surveys (CSUR)*, vol. 51, no. 6, pp. 1–36, 2019.

[3] X. Qin, W. Xu, and B. Li, "Optimal joint offloading and wireless scheduling for parallel computing with deadlines," in *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2019.

[4] B. Li, "Optimal offloading for dynamic compute-intensive applications in wireless networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[5] K. Gai, K. Xu, Z. Lu, M. Qiu, and L. Zhu, "Fusion of cognitive wireless networks and edge computing," *IEEE Wireless Communications*, vol. 26, no. 3, pp. 69–75, 2019.

[6] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. S. Shen, "Energy efficient dynamic offloading in mobile edge computing for internet of things," *IEEE Transactions on Cloud Computing*, 2019.

[7] S. Guo, J. Liu, Y. Yang, B. Xiao, and Z. Li, "Energy-efficient dynamic computation offloading and cooperative task scheduling in mobile cloud computing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 2, pp. 319–333, 2018.

[8] Y. Geng, Y. Yang, and G. Cao, "Energy-efficient computation offloading for multicore-based mobile devices," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 46–54.

[9] W. Wang, Y. Jiang, and W. Wu, "Multiagent-based resource allocation for energy minimization in cloud computing systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 2, pp. 205–220, 2016.

[10] S. Guo, M. Chen, K. Liu, X. Liao, and B. Xiao, "Robust computation offloading and resource scheduling in cloudlet-based mobile cloud computing," *IEEE Transactions on Mobile Computing*, 2020.

[11] C. Yi, J. Cai, and Z. Su, "A multi-user mobile computation offloading and transmission scheduling mechanism for delay-sensitive applications," *IEEE Transactions on Mobile Computing*, vol. 19, no. 1, pp. 29–43, 2019.

[12] J. Zheng, Y. Cai, Y. Wu, and X. Shen, "Dynamic computation offloading for mobile cloud computing: A stochastic game-theoretic approach," *IEEE Transactions on Mobile Computing*, vol. 18, no. 4, pp. 771–786, 2018.

[13] L. Yang, H. Zhang, X. Li, H. Ji, and V. C. Leung, "A distributed computation offloading strategy in small-cell networks integrated with mobile edge computing," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2762–2773, 2018.

[14] S. Jošilo and G. Dán, "A game theoretic analysis of selfish mobile computation offloading," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[15] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2015.

[16] S. Stidham, "Optimal control of admission to a queueing system," *IEEE Transactions on Automatic Control*, vol. 30, no. 8, pp. 705–713, 1985.

[17] B. Xia, S. Shakkottai, and V. Subramanian, "Small-scale markets for bilateral resource trading in the sharing economy," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 2447–2455.

[18] F. Alotaibi, S. Hosny, H. El Gamal, and A. Eryilmaz, "A game theoretic approach to content trading in proactive wireless networks," in *2015 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2015, pp. 2216–2220.

[19] Z. Chen, Y. Liu, B. Zhou, and M. Tao, "Caching incentive design in wireless d2d networks: A stackelberg game approach," in *2016 IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–6.

[20] J. Li, R. Bhattacharyya, S. Paul, S. Shakkottai, and V. Subramanian, "Incentivizing sharing in realtime d2d streaming networks: A mean field game perspective," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 3–17, 2016.

[21] D. Narasimha, S. Shakkottai, and L. Ying, "A mean field game analysis of distributed mac in ultra-dense multichannel wireless networks," in *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*. ACM, 2019, pp. 1–10.

[22] M. A. Abd, S. F. M. Al-Rubeaai, B. K. Singh, K. E. Tepe, and R. Benlamri, "Extending wireless sensor network lifetime with global energy balance," *IEEE Sensors Journal*, vol. 15, no. 9, pp. 5053–5063, 2015.

[23] O. Agmon Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "Deconstructing amazon ec2 spot instance pricing," *ACM Transactions on Economics and Computation*, vol. 1, no. 3, p. 16, 2013.

[24] M. Harchol-Balter, *Performance modeling and design of computer systems: queueing theory in action*. Cambridge University Press, 2013.