Learning-Based Priority Pricing for Job Offloading in Mobile Edge Computing

Lingxiang Li, Marie Siew, Tony Q.S. Quek, *Fellow, IEEE,* Ju Ren, *Member, IEEE,* Zhi Chen, *Senior Member, IEEE,* and Yaoxue Zhang, *Senior Member, IEEE*

***Abstract*—Mobile edge computing (MEC) is an emerging paradigm where users offload computationally intensive jobs to the Access Point (AP). Given that the AP's resources are shared by selfish mobile users, pricing is a useful tool for incentivising users to internalize the negative externality of delay they cause to other users. Different users have different negative valuations towards delay as some are more delay sensitive. To serve heterogeneous users, we propose a priority pricing scheme where users can get served first for a higher price. Our goal is to find the prices such that in decision making, users will choose the class and the offloading frequency that jointly maximize social welfare. With the assumption that the AP knows users' profit functions, we derive in semi-closed form the optimal prices. However in practice, the reporting of users's profit information incurs a large signalling overhead. Besides, in reality users might falsely report their private profit information. To overcome this, we further propose learning-based pricing mechanisms where no knowledge of individual user profit functions is required. At equilibrium, the optimal prices and average edge delays are learnt, and users have chosen the correct priority class and offload at the socially optimal frequency.**

***Index Terms*—Mobile edge computing, multi-user offloading, differentiated services, priority pricing, decentralized mechanism.**

I. Introduction

With the recent advancements of technology, mobile devices now run novel and diverse applications such as mobile games, facial recognition, augmented and virtual reality. These applications are computationally intensive and energy consuming. With respect to meeting these requirements, the computational capabilities and battery power of mobile devices, while improving over the past years, are still relatively insufficient [1]. To deal with this insufficiency, computationally intensive jobs are typically offloaded to cloud servers via access points (e.g., base station or Wi-Fi), helping to prolong battery life and improve user experience. However, jobs involving augmented reality, human computer interaction and autonomous driving also carry stringent low latency requirements. With cloud computing, it is difficult to meet these requirements, as the huge distance between the cloud and the users results in users experiencing wide area network (WAN) delay [2].

Lingxiang Li, Ju Ren and Yaoxue Zhang are with School of Information Science and Engineering, Central South University, Changsha 410083. E-mail: {lingxiang.li, renju, zyx}@csu.edu.cn

Marie Siew and Tony Q.S. Quek are with Information Systems Technology and Design Pillar, Singapore University of Technology and Design, Singapore 487372. E-mail: {tonyquek, marie_siew}@sutd.edu.sg

Zhi Chen is with the National Key Laboratory of Science and Technology on Communications, UESTC, Chengdu 611731. E-mail: chenzhi@uestc.edu.cn

Part of this work has been published in ICASSP 2019.

In order to meet stringent low latency requirements, the trend of edge computing is occurring [3]–[5], wherein job computation or data storage will be shifted to the network edge, instead of the faraway cloud. In this way, WAN delay is avoided. At the same time, the edge has better computational capabilities in comparison to mobile devices in handling the computationally intensive jobs. In cellular networks, this kind of computing refers to Mobile Edge Computing (MEC), and the network edge refers to the access point (AP), i.e., the base stations. As the AP is also in control of the wireless channels, edge computing enables a joint scheduling optimization of edge computing and radio resources. This enhances the system wide resource allocation efficiency.

Currently, there is substantial interest in investigating the joint optimization of radio and computing resource in MEC. For example, [6]–[9] study the single user scenario. The works [10]–[13] investigate the multi user scenario, while [14] investigated the multiple AP scenario. The D2D case with mutual sharing of resources among users is considered in [15], and the scenario with energy harvesting is investigated in [16], [17]. In these works, resource allocation is given a centralized optimization formulation, in particular as a Mixed Integer Nonlinear Programming (MINP) problem. However, for centralized optimization to be implemented, the AP must be able to control users' offloading amount or frequency. Secondly, the AP must have knowledge of users' offloading profit functions. In reality, the AP is not able to control the amount offloaded by users and the users' offloading profit functions are unobservable to the AP. Besides, users are unwilling to report information like profit functions, because these encapsulate private information such as distance from the AP and battery states.

Another line of research takes queueing delay into consideration [18]–[23], since as compared to the cloud, the network edge usually has limited computational resource, and tasks might need to wait in a queue before they are executed. Specifically, the works [18]–[21] proposed dynamic offloading algorithms based on Lyapunov optimization, effectively incorporating the long-term average cost minimization problem into real-time optimizations. The first two works study the single user scenario with a user running various types of tasks; the works [20], [21] further investigate the multiple users scenario with users moving erratically or under a probabilistic constraint on the queue length violations. However, these Lyapunov-based algorithms require knowledge of queueing states, channel conditions, etc., which the AP has to periodically measure. As such, implementing these algorithms will incur a large signalling overhead. On the other hand, the works [22], [23] take average delay of queueing as the performance metric. In specific, the work [22] applies the theories of stochastic geometry and parallel computing to derive the scaling laws of communication/computation latency with respect to network-load parameters. The work [23] considers concurrent traditional uplink/downlink transmissions and MEC-based tasks, differentiating tasks with quality of service (QoS) requirements and assigning higher priorities to those tasks with lower latency requirements. To the best of the authors' knowledge, few works use priority (in a queue) as a mechanism to optimally allocate radio and compute resources in MEC.

Priority is an important consideration in optimizing resource scheduling, since users and jobs are not homogeneous, and different users have different costs or negative valuations towards waiting time [24]. For example, an application involving human computer interaction or virtual reality is more delay sensitive compared to a text messaging or even an internet browsing application, as the first two involve interaction and real time feedback [18], [21]. Providing a priority option allows users and jobs with higher waiting costs to get served first, after paying an additional fee. In contrast, providing only one service class will incur welfare loss for users with higher waiting costs. Many service industries for example telcos, the postal service and theme parks all offer a priority option. The priority mechanism increases the service provider or operator's revenue, while allocating resources in a way to maximize the overall welfare of heterogeneous users. With this in mind, we aim to approach resource optimization in MEC via a priority mechanism, in which priority is treated as a valuable resource.

A. Related works and main contributions of this paper

In this work, we aim to optimize resource allocation for MEC in a distributed manner, given that centralized optimization requires control and perfect information which the AP does not have. Other works taking this distributed approach include: in [25] the initial centralized MINP problem is broken down into sub-problems and dealt with semi-distributively. [26]–[30] reformulate the centralized problem into a game and derive the Nash Equilibrium. This equilibrium is inefficient, with the welfare loss arising because in decision making, selfish users did not internalize the negative externalities of congestion and delay they cause to the system. In light of this and because the AP cannot directly control users' decisions, we propose using pricing to indirectly enforce users to internalize their negative externality in decision making.

Pricing is a key tool to allocate resources to the users who value them the most, thus controlling congestion in resource competition cases [31], [32]. However, only a small body of works have studied MEC from an economics viewpoint. Existing works from the economics viewpoint such as [33]–[35] are based on abstract utility functions (e.g., a simple logarithm function of the amount of product), and do not capture various user-side factors, such as the processing capacity of mobile device users and the amount of remaining computing tasks. The work [36] takes those user-side factors into account, and proposes Lyapunov-based algorithms for energy/monetary cost minimization while ensuring finite processing delay. Similar to other Lyapunov-based algorithms proposed by [18]–[21], the implementation of this work requires heavy signalling overhead to share the queueing states of users and the AP at each time slot. Besides, since to perform centralized optimization, the AP also requires knowledge of users' private information, such as their sensitiveness towards money and energy. Generally, users are unwilling to share and might cheat on such private information, to get more resources. In this paper, we aim to provide users with the appropriate incentives to control their flows and improve the overall system performance in a distributed way.

To that end, we introduce an economics model encapsulating physical layer factors, such as the wireless channel condition, the computational energy and delay of both local and edge computing. Queuing delay and job scheduling is an important factor to consider in MEC resource optimization because relative to the cloud, the edge has less computational resources and jobs need to be queued and scheduled. We consider average delay (via statistical measurements over time) since the prices should not fluctuate too often. Besides optimizing over each user's frequency of offloading, we also optimize over the job scheduling sequence. This is because as mentioned earlier, due to the varying delay sensitivity of jobs, different users will have different negative valuations towards delay experienced. Therefore, providing one class of service will incur further welfare loss, on the part of users who have high delay sensitivity. To deal with heterogeneous users, our mechanism provides the option of priority, where it serves the high priority jobs first for a higher service fee. Distinct prices for the two priority classes will ensure truthfulness, preventing all users from declaring priority. The prices will be set such that in decision making, when users weigh their payoffs from the two classes, they will choose the class and the offloading frequency that jointly maximize system-wide welfare.

Our main contributions are summarized below.

1) We introduce a novel utility function (see Eq. (13)), which, after subtracting the delay cost required by edge computing, equals the profit a user achieves by offloading (see Eq. (12)). This utility function measures the difference between the local computing cost and the offloading cost (including the offloading delay and the offloading energy consumption). Besides, this utility function satisfies the basic nature of the utility function in economics, thus bridging the gap between the economic and physical layer parameter optimizations.
2) *Complete users' individual demand information:* Based on the given utility function we propose an incentive-aware job offloading framework, in which the user decides the amount to offload by comparing its utility function with the delay cost of edge computing. We characterize the socially optimal point, as a function of the users' profit functions as well as the average edge computing delays (see Theorem 1). Furthermore, we derive in semi-closed form the prices which can incentivise users to choose the *correct* priority class and offload at a socially optimal frequency.
3) *Without users' individual demand information:* We

propose learning-based pricing mechanisms where no knowledge of individual user profit functions is required (see Algorithm 1 and 2). Instead, based on the measured congestion levels, the AP broadcasts prices and expected edge delays of the distinct priority classes. At equilibrium, the optimal prices and expected edge delays are learnt, with which the AP induces self-interested users to choose the *correct* priority class and make socially optimal offloading decisions, thus maximizing the system-wide welfare in a decentralized way (see Theorem 2 and 3).

II. Communication and Edge Computing System Model

We consider an MEC system consisting of an access point (AP) and N mobile users. The wireless AP could be a base station, or a Wi-Fi access point. Other than being a conventional access point to the core network, it is installed with an additional edge computing server. The mobile devices might be running computation-intensive and delay-sensitive jobs, and may have insufficient computing power or limited battery energy to complete those jobs. As such, they may offload part/all of their jobs to the AP. In this section, we will introduce the offloading policy, the wireless channel model, followed by the models for computing in detail (see Fig. 1).

A. Job generation model and offloading policy

Jobs arrive at the mobile users following a Poisson process of rate λ_a. The service time of a job is identically distributed (i.i.d.) and exponentially distributed, with an average of L_a bytes input data to offload. Denote the processing density by B_a. Then the average CPU cycles to run for a job equals $\mu_a = L_a B_a$.

This paper considers the scenario with flat-fading channels, and assume that the user can finish offloading in a channel block. Hence, we consider the following *offloading policy*. When a job arrives, if the achievable rate is higher than a threshold β_k, the mobile user would offload its job to the AP; otherwise, the mobile user will choose local computing.

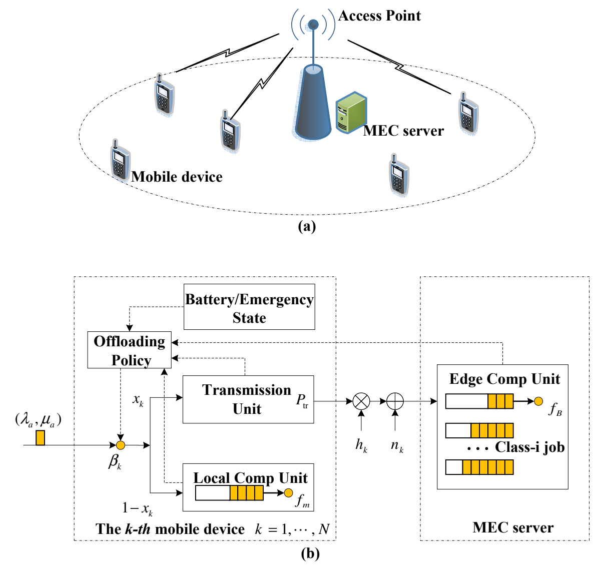

Fig. 1: An MEC system illustration with job arrival, offloading and computation.

B. Radio access model

Users *access the AP in an FDMA mode*, suffering no multi-user interference. Let h_k denote the small-scale channel gain from user k to the AP. The achievable uplink data rate is thus,

$$R_k = \log(1 + d_k^{-\alpha}|h_k|^2 P_{\text{tr}}/\sigma^2), \; k = 1, 2, \cdots, N, \qquad (1)$$

where d_k denotes user k's distance to the AP and α represents the path loss exponent. P_{tr} is the transmission power and σ^2 denotes the received noise power at the AP. In addition, by comparing the achievable data rate R_k with the expected data rate β_k and by Shannon's theorem [37]: when $R_k > \beta_k$ the mobile device can transmit its job to the AP successfully. Hence, the user k's offloading frequency (probability) is

$$x_k = \Pr(|h_k|^2 > (e^{\beta_k} - 1)\rho_k^{-1}), \; k = 1, 2, \cdots, N, \qquad (2)$$

where $\rho_k \triangleq d_k^{-\alpha} P_{\text{tr}}/\sigma^2$. A mobile user could control its offloading frequency by adjusting the threshold β_k.

C. Computation model

Based on the above radio access model and offloading policy, we discuss the total overhead/cost of local computing and edge computing. Both the processing delay and buffer delay are taken into consideration.

1) Local computing: By the Poisson arrival process and exponential job service time assumptions, we have an M/M/1 queue for local computing. Let f_m be the mobile device's computing capability (CPU cycles per second), then the expected time spent per job (including both the job execution time and the time spent awaiting in a local buffer) is

$$D_k^{\text{LC}}(x_k) = 1/(\mu_m - \lambda_a \bar{x}_k), \quad k = 1, 2, \cdots, N. \qquad (3)$$

TABLE I: Summary of key notations

Notation	Description
λ_a	Job arrival rate at the mobile device (jobs per second)
μ_a	Average CPU cycles needed by the computation job
B_a	Processing density (in cycles/bit)
L_a	The input data size (in Bytes)
κ_m	The energy coefficient of the mobile device
x_k	User k's offloading frequency
d_k	User k's distance to the access point
P_{tr}	Each user's transmit power
σ^2	Received noise power at the access point
c_k^e	User k's weight of the computational energy
c_k^d	User k's weight of the computational time
h_k	User k's instantaneous small-scale channel gain
β_k	User k's transmission rate
f_m	CPU-cycle frequency of the mobile device
μ_m	Computing service rate of User k (jobs per second)
f_B	CPU-cycle frequency of the MEC server
μ_B	Computing service rate of the MEC server (jobs per second)
g_k	User k's demand function
N	The total number of mobile device users

where the local computing probability $\bar{x}_k \triangleq 1 - x_k$, and the service rate $\mu_m \triangleq f_m/\mu_a$. Next, the computational energy of local computing is

$$E_k^{\mathrm{LC}} = \kappa_m f_m^2 \mu_a,\ k = 1, 2, \cdots, N, \tag{4}$$

where $\kappa_m f_m^2$ is the power consumption the mobile device user runs one CPU cycle, and κ_m is an energy consumption coefficient that depends on the chip architecture [38].

The total weighted cost of local computing is

$$Z_k^{\mathrm{LC}}(x_k) = c_k^e E_k^{\mathrm{LC}} + c_k^d D_k^{\mathrm{LC}}(x_k),\ k = 1, 2, \cdots, N, \tag{5}$$

where $0 < c_k^e < 1$ (in units 1/Joule) and $0 < c_k^d < 1$ (in units 1/Second) are the weights of computational energy and delay. The weights allow different users to place different emphasis in decision making. For example, if the mobile device is at a low battery state, it would give energy consumption more emphasis, choosing a bigger value of c_k^e. If the user is running urgent jobs, it would give the delay cost more emphasis. In this paper, we consider two classes of resource-hungry mobile users with different service requirements, i.e., one class of users setting $c_k^d = c_{\mathrm{H}}^d$, $k = 1, \cdots, N_{\mathrm{H}}$, and the other setting $c_k^d = c_{\mathrm{L}}^d$, $k = N_{\mathrm{H}} + 1, \cdots, N_{\mathrm{H}} + N_{\mathrm{L}}$, where $N_{\mathrm{H}} + N_{\mathrm{L}} = N$.

2) Edge computing: First, the time taken to offload to the AP is

$$D_{k,1}^{\mathrm{EC}}(x_k) = l_a \mu_a / \beta_k(x_k),\ k = 1, 2, \cdots, N. \tag{6}$$

This indicates that the energy required by offloading is

$$E_k^{\mathrm{EC}}(x_k) = P_{\mathrm{tr}} l_a \mu_a / \beta_k(x_k),\ k = 1, 2, \cdots, N. \tag{7}$$

Subsequently, the offloaded job will stay at the AP's buffer until it leaves after execution. For the purpose of increasing the net welfare, the AP applies priority queueing, where it provides different qualities of service to different classes of traffic. Suppose that traffic classes are partitioned into two sets, H and L. The job execution is FCFS, except that the jobs from the class H are always given priority over those from the class L, and the execution of an ordinary user's job is preempted if a priority user's job arrives.

Based on the aforementioned offloading policy, the mobile device will choose to offload its jobs with probability x_k. This, combined with the *splitting* property of the queueing theory [39], indicates that the job offloading from a mobile device user also follows a Poisson process, with parameter $\lambda_a x_k$. Hence, the job arrival at the AP is a superposition of multiple Poisson processes from multiple mobile device users, which, according to the *superposition* property of queueing theory [39], is another Poisson process with arrival rate as the sum arrival rate of the superposed processes. Therefore, the arrival rate at the AP is $\sum_{\mathrm{i}_j=\mathrm{H}} \lambda_a x_j$ for the higher priority class of users and $\sum_{\mathrm{i}_j=\mathrm{L}} \lambda_a x_j$ for the lower priority class of users. On the other hand, the service times are i.i.d. and exponentially distributed with parameter μ_a. As such, we get an M/M/1 queue with two priority classes for computing at the AP.

Let f_B be the AP's computing capability (CPU cycles per second). The service rate of the mobile device is then $\mu_B = f_B/\mu_a$ (jobs per second). Thus, for the user choosing the first priority, the expected time a job spending at the AP (including both the computation execution time and the time spent awaiting in the edge buffer) can be expressed as,

$$D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x}) = \frac{1}{\mu_B - \sum_{\mathrm{i}_j=\mathrm{H}} \lambda_a x_j}, \tag{8}$$

where $\mathbf{x} \triangleq (x_1, x_2, \cdots, x_N)$. For the user choosing second priority, the expected time a job spends at the AP can be expressed as,

$$D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x}) = \frac{\mu_B D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x})}{\mu_B - \sum_{\mathrm{i}_j=\mathrm{L}} \lambda_a x_j - \sum_{\mathrm{i}_j=\mathrm{H}} \lambda_a x_j}. \tag{9}$$

We neglect the energy overhead of edge computing as [16], [27], [29], since normally the AP can access to wired charing and it has no lack-of-energy issues. Combining (6) to (9) yields the total weighted cost of edge computing by user k, i.e.,

$$Z_k^{\mathrm{EC}}(\mathrm{i}_k, \mathbf{x}) = c_k^e E_k^{\mathrm{EC}}(x_k) + c_k^d (D_{k,1}^{\mathrm{EC}}(x_k) + D_{\mathrm{i}_k}^{\mathrm{EC}}(\mathbf{x})). \tag{10}$$

Note that the total cost by offloading, as well as the edge computing buffering delay $D_{\mathrm{i}_k}^{\mathrm{EC}}(\mathbf{x})$, depends on both the local variable (i.e., x_k for the k-th user) and also the offloading frequency of other users. As we will see later, due to this coupled nature of delay we have coupled objective functions.

III. Proposed Economics Model for Mobile Edge Computing and Problem Formulations

By the aforementioned offloading policy, when a job arrives the mobile user will offload with probability x_k, and locally compute with probability $\bar{x}_k$. Hence, the expected total cost of offloading is

$$Z_k(\mathrm{i}_k, \mathbf{x}) = \bar{x}_k Z_k^{\mathrm{LC}}(x_k) + x_k Z_k^{\mathrm{EC}}(\mathrm{i}_k, \mathbf{x}). \tag{11}$$

On the other hand, when there is no such edge server providing computing power, users have to run jobs locally with average cost $Z_k^{\mathrm{LC}}(0)$. Therefore the gross profit of offloading by the k-th user under a given offloading strategy $\mathbf{x}$ is,

$$V_k(\mathrm{i}_k, \mathbf{x}) = Z_k^{\mathrm{LC}}(0) - Z_k(\mathrm{i}_k, \mathbf{x}),\ k = 1, \cdots, N. \tag{12}$$

The key idea of the economics model is to introduce the utility function and the cost function. Some observations are in order. Firstly, the profit each user obtains equals the cost savings from offloading, and it is a linear combination of the energy costs and the delay costs. Secondly, the coupled delay cost $D_{\mathrm{i}_k}^{\mathrm{EC}}(\mathbf{x})$ reflects the harm/congestion each user causes to the other users, with a bigger value indicating that the AP provides worse service to the users. Thirdly, except for the expected time spent at the AP, i.e., $D_{\mathrm{i}_k}^{\mathrm{EC}}(\mathbf{x})$, which depends on all users' offloading decisions, the other items in the profit function only depend on each user's own offloading frequency x_k. Motivated by these observations, we introduce a utility function $U_k(x_k)$ which includes the items in the profit function that only depend on the local variable x_k, i.e.,

$$\begin{aligned} U_k(x_k) =& Z_k^{\mathrm{LC}}(0) - \bar{x}_k Z_k^{\mathrm{LC}}(x_k) \\ & - x_k (c_k^e E_k^{\mathrm{EC}}(x_k) + c_k^d D_{k,1}^{\mathrm{EC}}(x_k)). \end{aligned} \tag{13}$$

This, combined with (12), indicates that

$$V_k(\mathrm{i}_k, \mathbf{x}) = U_k(x_k) - C_k(\mathrm{i}_k, \mathbf{x}),\ k = 1, \cdots, N, \tag{14}$$

where $C_k(\mathrm{i}_k, \mathbf{x}) \triangleq c_k^d x_k D_{\mathrm{i}_k}^{\mathrm{EC}}(\mathbf{x})$ can be regarded as the delay cost caused by sharing an MEC server at the AP.

Taking the derivative of $U_k(x_k)$ with respect to x_k, we arrive at the demand function of user k, i.e.,

$$g_k(x_k) \triangleq \frac{\partial U_k(x_k)}{\partial x_k}.$$

For each mobile device user, it holds true that the demand function $g_k(x_k)$ is a monotonically decreasing function of the offloading frequency x_k. Moreover, there exists a unique solution of the equation $g_k(x_k) = 0$, denoted as x_k^{up}.

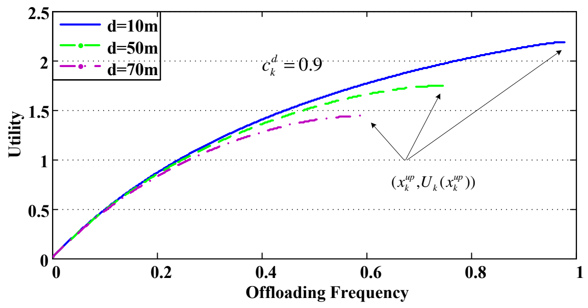

Fig. 2: The achievavble utility for varying distances to AP.

The utility function in (13) measures the welfare of offloading, while containing physical layer meaning. It is illustrated in Fig. 2, for a system with parameters set as in the numerical results section. Different distances, i.e., $d = 10$m, $d = 50$m and $d = 70$m, from the user to the AP are considered. One can see that the utility function is strictly increasing while the rate of increase, i.e., the demand, decreases as x_k increases; this is consistent with the law of diminishing marginal returns, a typical property of utility functions in economics. Moreover, the user closer to the AP has more offloading demand and can achieve a higher utility for the same offloading amount. This agrees with intuition that a user nearer the AP experiences a better wireless channel.

A. Problem formulations

We first consider the social-welfare offloading decision problem. Assume that the AP acts as a social planner. It would like users to choose their offloading frequency such that the net social welfare $\sum_{k=1}^N V_k(\mathbf{x})$ is maximized as follows.

Problem 1 (Social Problem):

$$\{\bar{\mathrm{i}}_k^\star, \bar{x}_k^\star\} \triangleq \arg \max_{\{\mathrm{i}_k, x_k\}} \quad \sum\nolimits_j V_j(\mathrm{i}_j, \mathbf{x}),$$
$$\text{s.t.} \quad \mathrm{i}_k \in \{\mathrm{H}, \mathrm{L}\}, 0 \le x_k \le 1, \forall k. \qquad (15)$$

Intuitively, in Problem 1 each user should also be concerned with the congestion it causes to other users and should keep its offloading under an appropriate amount for other users' welfare; the difficulty lies in how to incentivise users to do so when users are selfish and will choose their offloading decisions such that their individual profit $V_k(x_k)$ is maximized.

Pricing is a useful tool in incentivising users to choose the socially optimal levels of demand. The key idea is to enforce users to pay for the congestion they cause to the other users. In the following Problem 2, we study the pricing-based scheme, which charges users an additional edge computing service fee to regulate users' behavior.

Problem 2 (Regulated Selfish Problem):

$$\{\mathrm{i}_k^\star, x_k^\star\} \triangleq \arg \max_{\{\mathrm{i}_k, x_k\}} \quad U_k(x_k) - (c_k^d D_{\mathrm{i}_k}^{\mathrm{EC}} + p_{\mathrm{i}_k}) x_k,$$
$$\text{s.t.} \quad \mathrm{i}_k \in \{\mathrm{H}, \mathrm{L}\}, 0 \le x_k \le 1, \forall k. \qquad (16)$$

where p_{H} and p_{L} respectively denote the unit service fee for the MEC service with high priority and low priority.

Does there exist price pair $(p_{\mathrm{H}}, p_{\mathrm{L}})$ such that the individual objectives of self-interested users will be aligned with the social welfare objective? If it exists, how to design simple pricing schemes to achieve the highest net welfare? In the following sections, we answer the above two questions.

IV. The Socially Optimal Offloading Mechanism with One Priority Class of Users

In this section, we first study a special case with only one priority class of users, where each user sets the same weights of computational delay and energy, i.e., $c_k^d = c_{\mathrm{H}}^d$ and $N_{\mathrm{H}} = N$. Since we only have one priority class, we denote by the edge computing service fee as p.

In what follows, we will first analyze the structural property of the formulated social and regulated selfish problems, and admit in semi-closed form the optimal price as a function of users' profit functions. Based on these analyses, we then propose an evolutional pricing algorithm, which requires no individual utility functions and learns the *correct* price such that the social welfare is maximized.

A. Optimal price for offloading under complete information

For the Social Problem, by the first order condition, at the maximum is holds that

$$\partial U_k(x_k)/\partial x_k - \sum\nolimits_{j=1}^N \partial c_{\mathrm{H}}^d x_j D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x})/\partial x_k = 0$$
$$\Leftrightarrow g_k(x_k) = c_{\mathrm{H}}^d D_{\mathrm{H}}^{\mathrm{EC}} + \frac{c_{\mathrm{H}}^d \sum_{j=1}^N \lambda_a x_j}{(\mu_B - \sum_{j=1}^N \lambda_a x_j)^2}, \ \forall k. \qquad (17)$$

where $g_k(x_k)$ is the demand function of offloading.

By contrast, for the Regulated Selfish Problem, at the maximum it holds that

$$g_k(x_k) = c_{\mathrm{H}}^d D_{\mathrm{H}}^{\mathrm{EC}} + p, \ \forall k. \qquad (18)$$

Comparing (17) with (18) and for the purpose aligning the individual objectives with the social welfare objective, the price shall be set as

$$p(\mathbf{x}) = c_{\mathrm{H}}^d \sum\nolimits_{j=1}^N \lambda_a x_j / (\mu_B - \sum\nolimits_{j=1}^N \lambda_a x_j)^2. \qquad (19)$$

Here $p(\mathbf{x})$ is also referred to as the congestion level, since in our mechanism we propose using pricing to indirectly enforce users to internalize their negative externality in decision making.

Solving the equations in (17) and substituting the optimal offloading decisions, denoted by $\bar{x}_k^\star$, $\forall k$, into (19), we arrive at the optimal price $p^\star$ that induces self-interested users to make socially optimal offloading decisions.

B. Learning-based pricing which induces social-optimal offloading

That derivation of the optimal price $p^\star$ requires solving for $\bar{x}_k^\star$ based on users' report of their individual utility functions. These worthy functions include some private information such as their locations and battery states. Generally, users are not willing to share these private information. To that end, we propose an evolutional pricing algorithm, which requires no knowledge of individual utility functions and learns the *correct* price based on the observed congestion level.

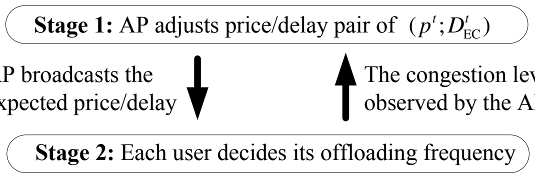
Stage 1: AP adjusts price/delay pair of $(p^t; D^t_{\rm EC})$
AP broadcasts the expected price/delay
The congestion level observed by the AP
Stage 2: Each user decides its offloading frequency

Fig. 3: Schematic view of the learning-based pricing scheme for one priority class of users.

As shown in Fig. 3, at each time slot t, the AP observes $p^{t-1}_{\rm true}$, the congestion level caused by users' offloading decisions. Based on the congestion level, the AP computes and broadcasts the unit service fee p^t and delay $D^t_{\rm EC}$ signals to users. Note that by comparing (8) and (19), for a given p^t we have a corresponding $D^t_{\rm EC}$ as follows

$$D^t_{\rm EC} = 1/(2\mu_B) + \sqrt{p^t/(c^d_{\rm H}\mu_B) + 1/4\mu_B^2}. \tag{20}$$

Viewing the signals, users decide on their offloading frequency x^t_k, which maximizes their individual welfare according to (18). The AP will observe this new congestion level $p^t_{\rm true}$. The above process iterates until convergence when the resulting congestion level, $p^t_{\rm true}$, equals the price p^t set, meaning that we have learnt the optimal price and arrived at the socially optimal equilibrium.

The key point in Fig. 3 is to provide a method to adjust the price/delay pair based on the observed congestion level, such that the iterative process converges at the socially optimal price. By definition, it can be verified that the demand function $g(x_k)$ is monotonically deceasing with respect to x_k. Besides, both the average edge delay and the congestion level increase as more jobs are offloaded to the AP. As such, if $p^t < p^\star$, the resulting $x^t_k > x^\star_k$, which results in a higher congestion level than p^t, i.e., $p^t_{\rm true} > p^t$. Therefore, by contraposition $p^t > p^t_{\rm true}$ indicates that $p^t > p^\star$. As such, the AP shall decrease the price if $p^t > p^t_{\rm true}$. Otherwise, the AP shall increase the price. Please refer to Algorithm 1 for more details on the learning process.

Convergence of Algorithm 1: According to Algorithm 1, a price of p^t, with $p^t < p^\star$, results in a congestion level higher than p^t, while the inverse case results in a congestion level lower than p^t. Therefore, with a bisection search over the price p^t, we can always find the price inducing the correct offloading policy such that the resulting congestion level equals the price. This indicates the convergence of Algorithm 1.

Algorithm 1 Learning-based pricing algorithm for one priority class of users

```
1:  Initialize: $t \leftarrow 0$, $p^t \leftarrow \mathrm{rand}$, $D^t_{\rm EC} \leftarrow$ by (20), $p_{\rm LB} \leftarrow 0$
2:  $x^t_k \leftarrow g_k^{-1}(c^d_{\rm H} D^t_{\rm EC} + p^t)$, $k = 1, 2, \cdots, N$        ▷ by (18)
3:  $p^t_{\rm true} \leftarrow$ by (19)
4:  while $p^t < p^t_{\rm true}$ do        ▷ Learn lower/upper bounds
5:      $p_{\rm LB} \leftarrow p^t$
6:      $t \leftarrow t+1$, $p^t \leftarrow 2p_{\rm LB}$, $D^t_{\rm EC} \leftarrow$ by (20)
7:      Repeat steps 2-3
8:  end
9:  $p_{\rm UB} \leftarrow p^t$
10: while $p_{\rm UB} - p_{\rm LB} > \epsilon$ do        ▷ Bisection search
11:     $t \leftarrow t+1$, $p^t \leftarrow (p_{\rm LB} + p_{\rm UB})/2$, $D^t_{\rm EC} \leftarrow$ by (20)
12:     Repeat steps 2-3
13:     if $p^t < p^t_{\rm true}$ then
14:         $p_{\rm LB} \leftarrow p^t$
15:     else
16:         $p_{\rm UB} \leftarrow p^t$
17:     end
18: end        ▷ The stop threshold $\epsilon = 0.01$
19: return $p \leftarrow p^t$ and $D^{\rm EC} \leftarrow D^t_{\rm EC}$
```

V. The Socially Optimal Offloading Mechanism with Two Priority Classes of Users

In this section, we extend our results to the more complex scenario with two priority classes of users. The priority queue is offered, wherein other than optimizing over each user's frequency of offloading, we also need to optimize over the job scheduling sequence. This is because as mentioned earlier, different users might have different negative valuations towards delay experienced. Therefore, providing one class of service will incur further welfare loss, on the part of users have high delay sensitivity. To deal with heterogeneous users, our mechanism provides the option of priority, where it serves the high priority users first for a higher service fee. Distinct prices for the two priority classes will ensure truthfulness, preventing all users from declaring priority. The prices will be set such that in decision making, when users weigh their payoffs from the two classes, they will choose the class that maximizes social welfare.

In what follows, we first analyze the structural property of the formulated social and regulated selfish problems and admit in semi-closed form the optimal prices that induce socially optimal offloading decisions, which, however, requires users' individual profit functions. As such, in the second subsection, we propose an evolutional pricing algorithm, which requires no individual utility functions and learns the *correct* prices based on the congestion level.

A. Optimal price for offloading under complete information

Before proceeding, we first introduce a lemma as follows.

Lemma 1: For the purpose of minimizing the total delay cost, the mobile device users putting more consideration to the delay cost, i.e., $c^d_k = c^d_{\rm H}$, *shall be given the higher priority service, i.e.,* $\mathrm{i}_k = \mathrm{H}$.

$$g_k(x_k) = \begin{cases} c_k^d D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x}) + \sum_{j=1}^{N_{\mathrm{H}}} \frac{\lambda_a c_{\mathrm{H}}^d x_j}{\Psi_{\mathrm{H}}^2} + \sum_{j=1}^{N_{\mathrm{L}}} \frac{\lambda_a \mu_B (\Psi_{\mathrm{H}} + \Psi) c_{\mathrm{L}}^d x_j}{\Psi^2 \Psi_{\mathrm{H}}^2}, & \text{if } c_k^d = c_{\mathrm{H}}^d, \\ c_k^d D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x}) + \sum_{j=1}^{N_{\mathrm{L}}} \frac{\lambda_a \mu_B c_{\mathrm{L}}^d x_j}{\Psi^2 \Psi_{\mathrm{H}}}, & \text{if } c_k^d = c_{\mathrm{L}}^d. \end{cases} \tag{24}$$

Proof: According to the famous *$c\mu$ rule* (or, here, the *$c_k^d \mu_a$ rule*) [40], providing a higher priority to those users with a higher value of delay cost, is optimal in terms of minimizing the system-wide delay cost. This completes the proof. ■

Based on Lemma 1, the mobile users with higher delay costs shall be given a higher priority service, while those with lower delay costs shall wait in the queue for the AP finish executing higher priority jobs. Therefore, the expression of delays in (8) and (9) can be respectively rewritten as,

$$D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x}) = 1/\Psi_{\mathrm{H}}, \tag{21a}$$
$$D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x}) = \mu_B D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x})/\Psi. \tag{21b}$$

where $\Psi_{\mathrm{H}} \triangleq \mu_B - \sum_{c_j^d = c_{\mathrm{H}}^d} \lambda_a x_j$ and $\Psi \triangleq \mu_B - \sum_j \lambda_a x_j$.

For the Social Problem, by the first order condition, at the maximum is holds that

$$\frac{\partial U_k(x_k)}{\partial x_k} - \sum_{j=1}^{N} \frac{\partial c_j^d x_j D_{\mathrm{i}_j}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} = 0$$
$$\Leftrightarrow g_k(x_k) = c_k^d D_{\mathrm{i}_k}^{\mathrm{EC}}(\mathbf{x}) + \sum_{j=1}^{N} \frac{\partial c_j^d D_{\mathrm{i}_j}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} x_j. \tag{22}$$

Substituting (21a) and (21b) into (22) and by further derivations, we arrive at the following theorem.

Theorem 1: Considering an MEC network consisting of two priority classes of users, the optimal offloading frequencies of the mobile device users are given by solving the following equations,

$$g_k(x_k) = c_k^d D_{\mathrm{i}_k}^{\mathrm{EC}}(\mathbf{x}) + \sum_{j=1}^{N_{\mathrm{H}}} \frac{\partial c_{\mathrm{H}}^d D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} x_j + \sum_{j=1}^{N_{\mathrm{L}}} \frac{\partial c_{\mathrm{L}}^d D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} x_j, \; k = 1, 2, \cdots, N, \tag{23}$$

which equals solving the equations (24) at the top of this page.

Proof: See Appendix A. ■

Remark 1: By solving the N equations in (23), we obtain the offloading solutions of the Social Problem. Furthermore, this solution is unique.

In contrast, for the Regulated Selfish Problem, at the maximum it holds that

$$g_k(x_k) = c_k^d D_{\mathrm{i}_k}^{\mathrm{EC}} + p_{\mathrm{i}_k}, \; \forall k. \tag{25}$$

Comparing (25) with (23) and for the purpose of aligning the individual objectives with the social welfare objective, the price shall be set as,

$$p_{\mathrm{i}_k} = \sum_{j=1}^{N_{\mathrm{H}}} \frac{\partial c_{\mathrm{H}}^d D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} x_j + \sum_{j=1}^{N_{\mathrm{L}}} \frac{\partial c_{\mathrm{L}}^d D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} x_j.$$

In addition, according to Lemma 1, the mobile users with lower delay costs shall be given lower priority service, in which case the derivative $\partial D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x})/\partial x_k = 0$. Hence, for the users with higher delay cost and those with lower delay cost, the AP shall respectively charge them with a service fee as follows,

$$p_{\mathrm{H}} = \sum_{j=1}^{N_{\mathrm{H}}} \frac{\partial c_{\mathrm{H}}^d D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} x_j + \sum_{j=1}^{N_{\mathrm{L}}} \frac{\partial c_{\mathrm{L}}^d D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} x_j, \tag{26a}$$
$$p_{\mathrm{L}} = \sum_{j=1}^{N_{\mathrm{L}}} \frac{\partial c_{\mathrm{L}}^d D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} x_j. \tag{26b}$$

Theorem 2: For any given price pair (p_H, p_L) satisfying (26a)(26b) and the expected delay pair $(D_{\mathrm{H}}^{\mathrm{EC}}, D_{\mathrm{L}}^{\mathrm{EC}})$ satisfying (21a)(21b), it is in all the mobile device users' self-interest to classify their jobs in their correct priority class, i.e.,

$$p_{\mathrm{i}_k} + c_k^d D_{\mathrm{i}_k}^{\mathrm{EC}} < p_{\bar{\mathrm{i}}_k} + c_k^d D_{\bar{\mathrm{i}}_k}^{\mathrm{EC}}, \mathrm{i}_k \in \{\mathrm{H}, \mathrm{L}\}.$$

where $\bar{\mathrm{i}}_k = \mathrm{L}$ if $\mathrm{i}_k = \mathrm{H}$, and $\bar{\mathrm{i}}_k = \mathrm{H}$ if $\mathrm{i}_k = \mathrm{L}$. Moreover, the resulting offloading frequencies maximize the net welfare of all the mobile device users.

Proof: See Appendix B. ■

Remark 2: Based on Theorem 2, one can see that the given price pair (p_H, p_L) is incentive-compatible. That is, under this pricing mechanism, rational users will choose to pay the right class service fee and join the right priority class in terms of social welfare maximization.

Substituting the optimal solution derived by Theorem 1 into (26a)(26b), we arrive at the right prices that the AP shall set for the purpose of inducing users to choose the right priority class. However, the derivation of the optimal offloading frequency solution in Theorem 1 requires users' individual profit functions. In the following subsection, we derive an evolutional pricing algorithm, which requires no knowledge of individual utility functions and learns the *correct* prices, such that social welfare is maximized.

B. Learning-based pricing which induces social-optimal offloading

Similar to the information flow in Fig. 3, at each time slot t, the AP observes $D_{\mathrm{H,EC}}^{\mathrm{true,t}}$ and $D_{\mathrm{L,EC}}^{\mathrm{true,t}}$, the congestion level caused by users' offloading decisions. Based on the congestion level, the AP computes and broadcasts the unit service fees p_{H}^t and p_{L}^t, and delay $D_{\mathrm{H,EC}}^t$ and $D_{\mathrm{L,EC}}^t$ signals to users. Viewing those prices and expected delays information, users then decide their priority class and offloading frequency to maximize their individual welfare. Please see Fig. 4 for the schematic view.

However, it is not easy to extend the evolutional pricing algorithm for the case with one priority class of users here, since other than inducing users to offload at the social optimal

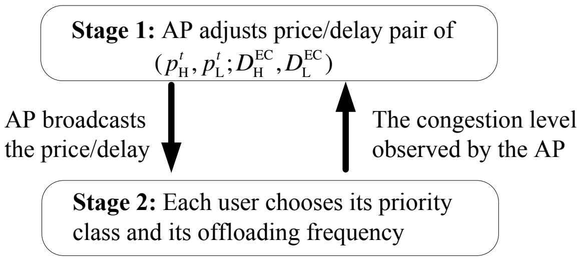
Stage 1: AP adjusts price/delay pair of $(p_{\mathrm{H}}^t, p_{\mathrm{L}}^t; D_{\mathrm{H}}^{\mathrm{EC}}, D_{\mathrm{L}}^{\mathrm{EC}})$
AP broadcasts the price/delay
The congestion level observed by the AP
Stage 2: Each user chooses its priority class and its offloading frequency

Fig. 4: Schematic view of the learning-based pricing scheme for two priority classes of users.

offloading frequency, the *correct* prices also need to induce users to choose the *correct* priority class.

The key idea is to adjust prices according to (26a)(26b). Some observations are in order. First, given a pair of average delays, i.e., $D_{\mathrm{H}}^{\mathrm{EC}}$ and $D_{\mathrm{L}}^{\mathrm{EC}}$, we can respectively determine the sum offloading rate of the two distinct priority classes of users according to (21a) and (21b), which further decides the corresponding prices according to (26a)(26b). Substituting the average delays and prices into (25), users can compute the offloading frequency that maximizes their' individual profit. Second, according to Theorem 2, given any delay pairs and price pairs respectively satisfying (21a)(21b) and (26a)(26b), it is in all the mobile device users' self-interest to classify their jobs in their correct priority class. That is, those price pairs are incentive-compatible candidates. Third, the demand function $g(x_k)$ is monotonically deceasing with respect to x_k. As such, if the right hand side of (25), i.e., the total cost, is less than that at the social optimal equilibrium, the resulting offloading frequency $x_k^t > x_k^\star$, which results in a high congestion level. Based on these observations, we propose Algorithm 2 as follows.

Specifically, the whole algorithm consists of two embedded loops, where in the outer loop the AP updates the expected average delay of $D_{\mathrm{H,EC}}^t$ for the high priority queue, while in the inner loop the AP updates the expected average delay of $D_{\mathrm{L,EC}}^t$ for the low priority queue.

In the inner loop and at each time slot, the AP observes $D_{\mathrm{L,EC}}^{\mathrm{true,t-1}}$, the congestion level of the low priority queue, caused by users' offloading decisions. Based on the observed congestion level, the AP computes the expected average delay $D_{\mathrm{L,EC}}^t$ for the low priority queue, which, combined with $D_{\mathrm{H,EC}}^t$, decides the unit service fee p_{H}^t and p_{L}^t. Following which, the AP broadcasts the service fee and expected delay signals to users. Viewing the signals, users decide which priority class to join and determine their offloading frequency x_k^t based on (25) to maximize their individual welfare. The above process iterates until convergence when the resulting congestion level $D_{\mathrm{L,EC}}^{\mathrm{true,t}}$ equals the expected average delay set, i.e., $D_{\mathrm{L,EC}}^t$.

In the outer loop and at each iteration, the AP observes $D_{\mathrm{H,EC}}^{\mathrm{true,t-1}}$, the congestion level of the high priority queue. Based on the observed congestion level, the AP computes the expected average delay $D_{\mathrm{H,EC}}^t$ for the high priority queue. The inner loop (steps 5-26) is then executed until the congestion level of the low priority queue converges to its expected

Algorithm 2 Learning-based pricing algorithm for two priority classes of users

```
1:  **Initialize**: $t \leftarrow 0$, $D_{\mathrm{H,EC}}^t \leftarrow \mu_B^{-1} + \varsigma$, $D_{\mathrm{H,EC}}^{\mathrm{true},t} \leftarrow 0$
2:  **while** $D_{\mathrm{H,EC}}^t > D_{\mathrm{H,EC}}^{\mathrm{true},t}$ **do**   $\triangleright$ Learn lower/upper bounds
3:      $D_{\mathrm{H,EC}}^{\mathrm{UB}} \leftarrow D_{\mathrm{H,EC}}^t$
4:      $t \leftarrow t+1$, $D_{\mathrm{H,EC}}^t \leftarrow \max\{D_{\mathrm{H,EC}}^t/2, \mu_B^{-1}\}$
5:      $D_{\mathrm{L,EC}}^t \leftarrow \mu_B D_{\mathrm{H,EC}}^t D_{\mathrm{H,EC}}^t$
6:      $(p_{\mathrm{H}}^t, p_{\mathrm{L}}^t) \leftarrow$ by (21a)(21b) and (26a)(26b)
7:      $\mathrm{i}_k^\star \leftarrow \arg\min_{\mathrm{i}_k}(p_{\mathrm{i}_k}^t + c_k^d D_{\mathrm{i}_k,\mathrm{EC}}^t)$
8:      $x_k^t \leftarrow g_k^{-1}(p_{\mathrm{i}_k^\star}^t + c_k^d D_{\mathrm{i}_k^\star,\mathrm{EC}}^t)$, $k = 1, 2, \cdots, N$
9:      $D_{\mathrm{L,EC}}^{\mathrm{true},t} \leftarrow$ by substituting $x_k^t$ into (21b)
10:     **while** $D_{\mathrm{L,EC}}^t < D_{\mathrm{L,EC}}^{\mathrm{true},t}$ **do**
11:         $D_{\mathrm{L,EC}}^{\mathrm{LB}} \leftarrow D_{\mathrm{L,EC}}^t$
12:         $t \leftarrow t+1$, $D_{\mathrm{L,EC}}^t \leftarrow 2D_{\mathrm{L,EC}}^t$, $D_{\mathrm{H,EC}}^t \leftarrow D_{\mathrm{H,EC}}^t$
13:         Execute steps 6-9
14:     **end**
15:     $D_{\mathrm{L,EC}}^{\mathrm{UB}} \leftarrow D_{\mathrm{L,EC}}^t$
16:     **while** $D_{\mathrm{L,EC}}^{\mathrm{UB}} - D_{\mathrm{L,EC}}^{\mathrm{LB}} > \epsilon$ **do**   $\triangleright$ Bisection search
17:         $t \leftarrow t+1$
18:         $D_{\mathrm{L,EC}}^t \leftarrow (D_{\mathrm{L,EC}}^{\mathrm{UB}} + D_{\mathrm{L,EC}}^{\mathrm{LB}})/2$, $D_{\mathrm{H,EC}}^t \leftarrow D_{\mathrm{H,EC}}^t$
19:         Execute steps 6-9
20:         **if** $D_{\mathrm{L,EC}}^t < D_{\mathrm{L,EC}}^{\mathrm{true},t}$ **then**
21:             $D_{\mathrm{L,EC}}^{\mathrm{LB}} \leftarrow D_{\mathrm{L,EC}}^t$
22:         **else**
23:             $D_{\mathrm{L,EC}}^{\mathrm{UB}} \leftarrow D_{\mathrm{L,EC}}^t$
24:         **end**
25:     **end**   $\triangleright$ The stop threshold $\epsilon = 0.01$
26:     $D_{\mathrm{H,EC}}^{\mathrm{true},t} \leftarrow$ by substituting $x_k^t$ into (21a)
27: **end**
28: $D_{\mathrm{H,EC}}^{\mathrm{LB}} \leftarrow D_{\mathrm{H,EC}}^t$
29: **while** $D_{\mathrm{H,EC}}^{\mathrm{UB}} - D_{\mathrm{H,EC}}^{\mathrm{LB}} > \epsilon$ **do**   $\triangleright$ Bisection search
30:     $t \leftarrow t+1$, $D_{\mathrm{H,EC}}^t \leftarrow (D_{\mathrm{H,EC}}^{\mathrm{UB}} + D_{\mathrm{H,EC}}^{\mathrm{LB}})/2$
31:     Execute steps 5-26
32:     **if** $D_{\mathrm{H,EC}}^t < D_{\mathrm{H,EC}}^{\mathrm{true},t}$ **then**
33:         $D_{\mathrm{H,EC}}^{\mathrm{LB}} \leftarrow D_{\mathrm{H,EC}}^t$
34:     **else**
35:         $D_{\mathrm{H,EC}}^{\mathrm{UB}} \leftarrow D_{\mathrm{H,EC}}^t$
36:     **end**
37: **end**
38: **return** $p_{\mathrm{H}} \leftarrow p_{\mathrm{H}}^t$, $p_{\mathrm{L}} \leftarrow p_{\mathrm{L}}^t$, $D_{\mathrm{H}}^{\mathrm{EC}} \leftarrow D_{\mathrm{H,EC}}^t$, $D_{\mathrm{L}}^{\mathrm{EC}} \leftarrow$
    $D_{\mathrm{L,EC}}^t$
```

average delay. Back to outer loop, the AP will check if $D_{\mathrm{H,EC}}^{\mathrm{true,t}}$, the observed congestion level of the high priority queue has converged to $D_{\mathrm{H,EC}}^t$, its expected average delay. If not, the outer loop will change its expected average delay again, and continue with the inner loop (steps 5-26). Otherwise, the entire process stops and we have arrived at the optimal prices and expected delays of the two priority classes that maximize the system-wide welfare.

The following theorem proves that Algorithm 2 converges.

Theorem 3: Algorithm 2 finally converges to a socially optimal point where users classify their jobs in their correct priority class and offload at the offloading frequencies which *jointly maximize the system-wide welfare.*

Proof: See Appendix C. ■

VI. Numerical Results

In this section, we provide numerical simulations which substantiate our theoretical results. The simulations show the convergence and optimality of our proposed pricing mechanisms, and give us some insight into the system performance at the optimal allocation.

A number of 100 mobile users are uniformly placed at random on a ring of radius $10 \leq d_k \leq 75$ (unit: meters) whose center is located at the AP. The path loss exponent is $\alpha = 3.5$. The channels are assumed to be identically distributed (i.i.d.) and $|h_k|^2 \sim \exp(1)$. The other parameters are summarized in Table II.

TABLE II: Simulation Parameters [18], [21], [41]

Parameter	Value	Parameter	Value
W	360KHz	κ_m	10^{-27}Watt $\cdot$ s^3/cycles^3
P_{tr}	100mW	B_a	8250 cycles/bit
σ^2	-40dBm	L_a	100K Bytes
α	3.5	λ_a	0.01 jobs/s
f_B	3GHz	c_k^d	{0.9,0.1}
f_m	0.5GHz	c_k^e	{0.1,0.9}

A. The case with one priority class of users

In this subsection we consider the scenario where users have the same computational energy and delay requirements ($c_k^d = c_{\text{H}}^d$) and one priority class is offered. To learn the price and delay pair that achieve optimal social welfare, the AP implements our learning based pricing mechanism (Algorithm 1, Section 4.2).

Fig. 5 illustrates the convergence and optimality of our proposed pricing mechanisms for the one priority class with users setting $c_k^d = 0.1$. Specifically, Fig. 5(a) and 6(b) illustrate the convergence of our proposed algorithm. In particular, the average delay D_{EC}^t and price p^t both converge after only a few iterations, learning the price and delay which achieves optimal social welfare. Convergence to optimality occurs despite the AP not having knowledge of the users' utility functions. The only information required on the part of the AP is the congestion level, which it can measure. As seen, the algorithm converges quickly, after only several iterations, making it suitable for future extension and implementation in dynamic and online scenarios.

Fig. 5(c) plots $\dfrac{\text{Cost by the proposed scheme}}{\text{Cost by local computing only}}(\%)$ for each user. It can be seen that at the equilibrium point of our proposed pricing scheme Algorithm 1, all users obtain the optimal cost savings. Fig. 5(c) also shows that the closer the user is to the AP, the higher the cost savings achieved. Fig. 5(d) illustrates the offloading frequency of different users, who have different distances from the AP. As expected, the users closer to the AP offload jobs at higher frequencies. This is because, the users closer to the AP experience better channels. As such, they can offload with higher rates and less energy.

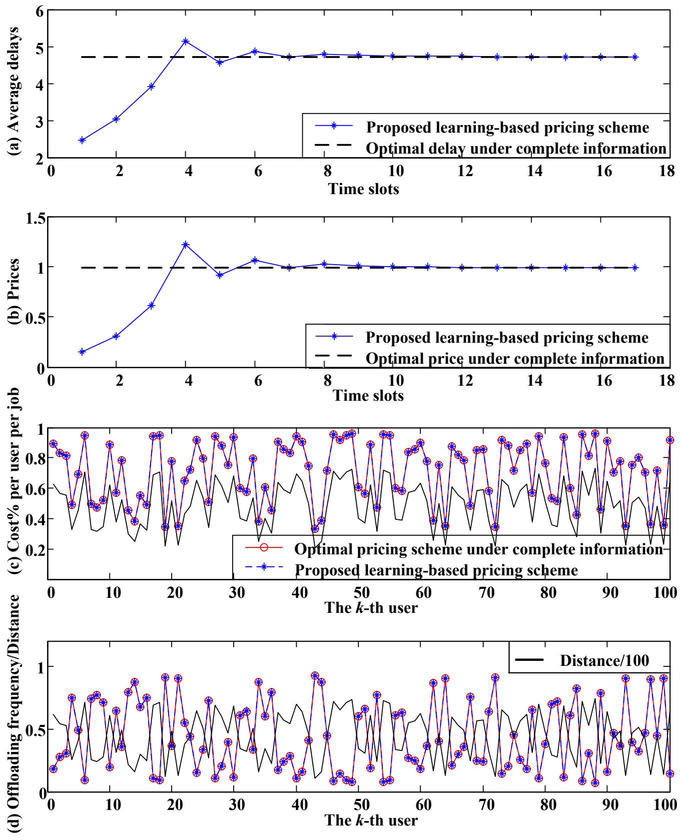

Fig. 5: (a) Posted delay v.s. time; (b) Posted price v.s. time; (c) Cost percentage as compared with that by local computing only; (d) Offloading frequency v.s. users' distance to the AP.

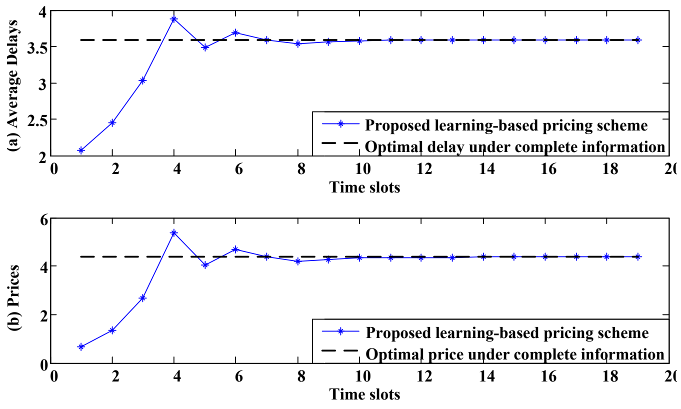

Fig. 6: (a) Posted delay v.s. time; (b) Posted price v.s. time.

For comparison, in Fig. 6 we plot the result for the scenario with computational weight of $c_k^d = 0.9$. One can see that for the former case, the posted delay converges to a bigger value while the price converges to a smaller value (see Fig. 5). This is consistent with intuition that for users with less emphasis on delay, they are more tolerant of a larger computing delay and are less willing to pay for the service.

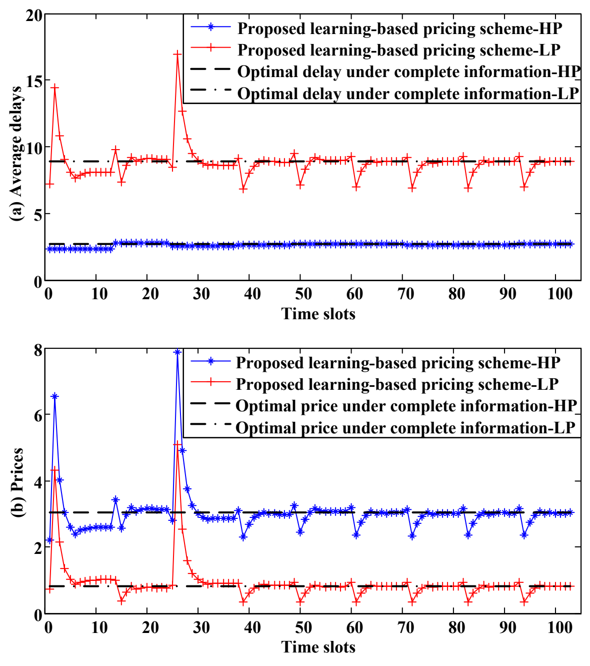

Fig. 7: (a) Posted delay v.s. time; (b) Posted price v.s. time

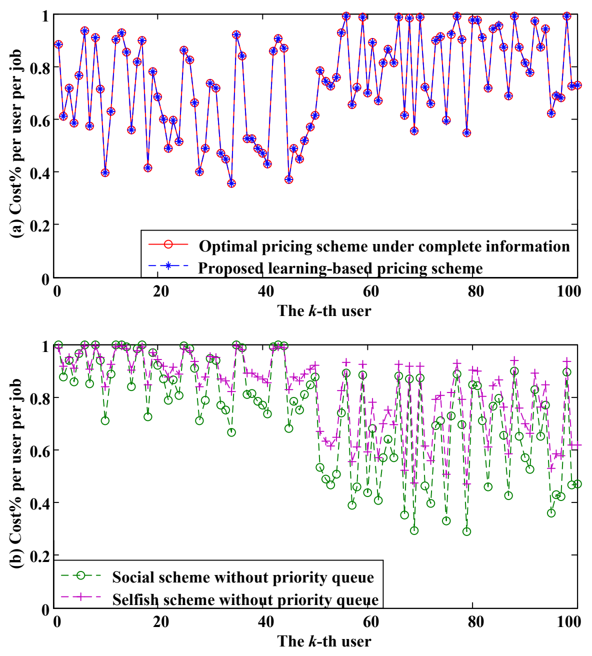

Fig. 8: Job execution costs (%) via different schemes.

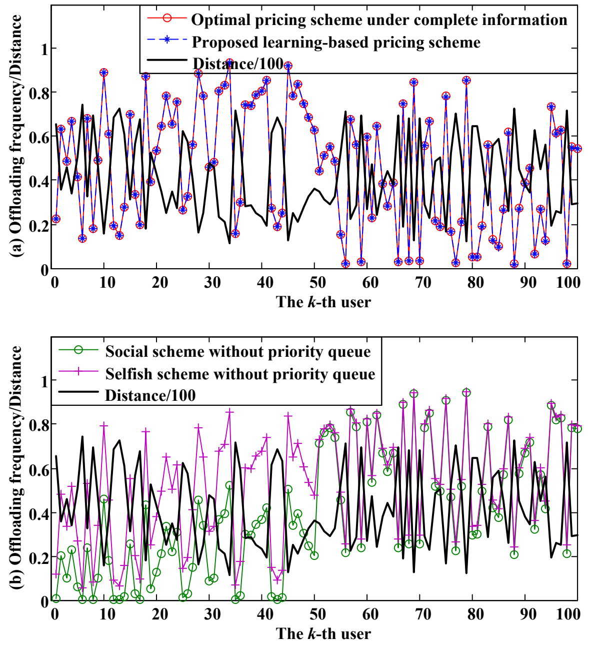

Fig. 9: Offloading frequency v.s. distance to the AP

B. The case with two priority classes of users

In this subsection, we consider the heterogeneous scenario where users have varying computational delay and energy requirements, and a priority based pricing mechanism offering two priority classes (Algorithm 2 in Section 5.2) is proposed to maximize the overall welfare of heterogeneous users. Without loss of generality, we assume that the first 50 mobile users set their computational weights as $c_k^d = 0.9$ and the other 50 mobile users set their weights as $c_k^d = 0.1$.

In Fig. 7 we plot how the average delays and prices for both the low and high priority classes varying over time. Recall that Algorithm 2 consists of two embedded loops. In the inner loop the expected average delay of the low priority class $D^t_{\mathrm{L,EC}}$ is updated until the resulting congestion level $D^{\mathrm{true},t}_{\mathrm{L,EC}}$ equals $D^t_{\mathrm{L,EC}}$. While in the outer loop, the expected average delay of the high priority class $D^t_{\mathrm{H,EC}}$ is updated, until the resulting congestion level $D^{\mathrm{true},t}_{\mathrm{H,EC}}$ equals $D^t_{\mathrm{H,EC}}$. This embedded loop structure can be visualised by the algorithm's trajectory in the figures. From Fig. 7, we can see that our mechanism learns the optimal delays and prices for both classes. The AP learns these optimal values without knowledge of users' utility functions (which encapsulate private information like battery states). It can be seen that the optimal price for the high priority class is around 3 times higher, because i) they are paying for the higher delay they cause to the low priority users (see eq. (26a) and (26b)), ii) they have low delay tolerance and would be willing to pay a higher price for a lower average delay.

Fig. 8 illustrates the individual users' cost per job as a percentage of that by local computing only. The subplot Fig. 8(a) illustrates that at the equilibrium point of our mechanism, the users achieve the optimal cost savings. As comparison, in Fig. 8(b) we also plot the individual users' cost by other schemes without priority queue, the social scheme and the selfish scheme. Specifically, in the social scheme we maximize the net welfare of users, while in the selfish problem users maximize their individual interest. Similar to Subsection 4.1, optimal solutions of these two schemes could be respectively obtained by solving the equations arising from the derivatives of the net welfare and users' individual interest. From Fig. 8, we can see that as compared with the proposed scheme, under the schemes without priority queue, users with stricter delay requirement (i.e., bigger computational weight) suffers from profit loss. This can be explained as follows. Given a same expected value of average delay, users with a bigger computational weight pay more due to the higher delay cost. As such, their demand in offloading decreases. In contrast,

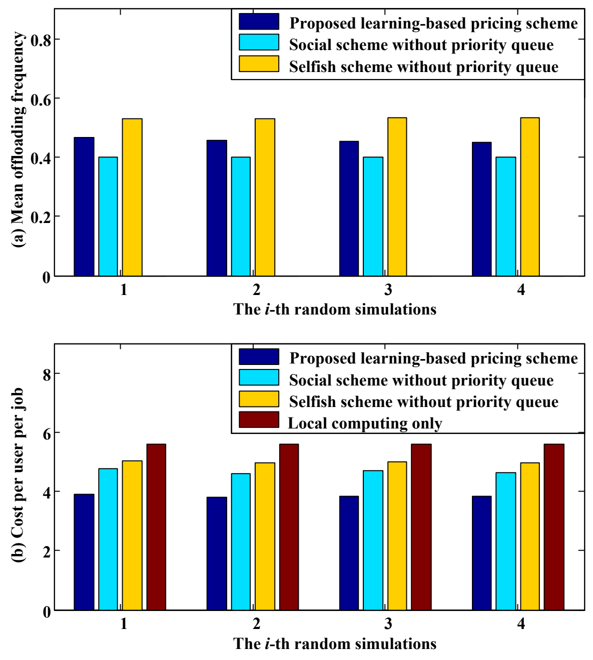

Fig. 10: (a) Mean offloading frequency; (b) Average cost.

users with a smaller computational weight are more willing to offload. Therefore, the former group of users cannot enjoy well the cost saving brought by edge computing.

Fig. 9 illustrates the offloading frequency of different users, and compares the results with that by the schemes without priority queue. As expected, the users closer to the AP offload jobs at higher frequencies. In addition, by comparison it can be seen that the users with stricter delay requirements benefit from the option of priority.

We conclude this section with Fig. 10, in which we run the simulation for the four different schemes. In each simulation, we place mobile users uniformly, and respectively plot the resulting mean offloading frequency of users as well as the job execution cost of the different schemes. From Fig. 10, we can see that by providing the option of priority, users offload more often, which results in bigger cost savings (almost reduced 30% of the cost, as compared to local computing only). In contrast, users offload more under the selfish scheme, as compared to the social scheme, however, they suffer from an improper high congestion level. As such, they suffer welfare loss on average.

VII. Conclusion

In this paper, we have proposed an incentive-aware offloading control mechanism for an MEC system, which consists of an access point (AP) of finite computing power, and serves two class of resource-hungry and selfish mobile users via charging users a edge computing service fee. Different users might have different service requirements. To deal with heterogeneous users, our mechanism provides the option of priority, in which it serves the high priority users first for a higher service fee. We started with a special case with only one priority class of users, for which we have characterized in semi-closed form the optimal prices to incentivise socially optimal offloading under complete profit functions of users. Besides, we have considered the practical scenario where the AP has no knowledge of user profit functions, and proposed a learning based pricing mechanism. Then, we have extended these results to the case with two priority classes of users, under both the complete and the unknown profit information of users. We have proved that the proposed mechanisms converge to the optimal prices and expected edge delays. At equilibrium, the AP induces self-interested users to choose the correct priority class and make socially optimal offloading decisions, thus maximizing the system-wide welfare in a decentralized way.

This work assumed that the MEC server executes jobs one after another. A more powerful MEC server may apply parallel edge computing, which can handle more complex systems running higher processing density jobs. Thus, further work is needed to take parallel computing into account. Also, for the purpose of gaining thorough insight into the computing resource allocation, in this paper we have considered the scenario in which users access to the AP via an FDMA mode. As such, users suffer no wireless interference from each other when offloading data to the AP. This alleviates the additional stress in radio resource allocation. An interesting alternative is to combine with the radio resource allocation and investigate the scaling laws of communication/computation latency with respect to network-load parameters.

Appendix A
Proof of *Theorem 1*

According to Lemma 1, for the purpose of maximizing the net welfare of users, a user with stricter delay requirement shall choose the higher priority service; otherwise, it shall choose the lower priority service. As such, the edge computing delay can be rewritten as follows,

$$D_{\mathrm{i}_j}^{\mathrm{EC}}(\mathbf{x}) = \begin{cases} D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x}), \text{ if } c_j^d = c_{\mathrm{H}}^d, \\ D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x}), \text{ if } c_j^d = c_{\mathrm{L}}^d, \end{cases} \tag{27}$$

where $D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x})$ and $D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x})$ are given by (21a) and (21b), respectively. Substituting (27) into (22) yields (23).

Besides, taking the derivative of the formula in (21a) with respect to x_k yields

$$\frac{\partial D_{\mathrm{H}}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} = \begin{cases} \lambda_a/\Psi_{\mathrm{H}}^2, & \text{if } c_k^d = c_{\mathrm{H}}^d, \\ 0, & \text{if } c_k^d = c_{\mathrm{L}}^d. \end{cases} \tag{28}$$

Similarly, taking the derivative of the formula in (21b) yields

$$\frac{\partial D_{\mathrm{L}}^{\mathrm{EC}}(\mathbf{x})}{\partial x_k} = \begin{cases} \dfrac{\lambda_a \mu_B (\Psi_{\mathrm{H}} + \Psi)}{\Psi^2 \Psi_{\mathrm{H}}^2}, & \text{if } c_k^d = c_{\mathrm{H}}^d, \\ \dfrac{\lambda_a \mu_B}{\Psi^2 \Psi_{\mathrm{H}}}, & \text{if } c_k^d = c_{\mathrm{L}}^d. \end{cases} \tag{29}$$

Substituting (28) and (29) into (23), we have (24). This completes the proof.

Appendix B
Proof of *Theorem 2*

In this section, we will prove the following two equations, thus completing the proof of Theorem 2.

$$p_{\rm H}+c^d_{\rm H}D^{\rm EC}_{\rm H}<p_{\rm L}+c^d_{\rm H}D^{\rm EC}_{\rm L}, \tag{30a}$$
$$p_{\rm L}+c^d_{\rm L}D^{\rm EC}_{\rm L}<p_{\rm H}+c^d_{\rm L}D^{\rm EC}_{\rm H}. \tag{30b}$$

Combining (26a)(26b) and (28)(29), we arrive at

$$\begin{aligned}p_{\rm H}-p_{\rm L}=&\sum\nolimits_{j=1}^{N_{\rm H}}\frac{\lambda_a c^d_{\rm H}x_j}{\Psi^2_{\rm H}}+\sum\nolimits_{j=1}^{N_{\rm L}}\frac{\lambda_a\mu_B c^d_{\rm L}x_j(\Psi_{\rm H}+\Psi)}{\Psi^2\Psi^2_{\rm H}}\\&-\sum\nolimits_{j=1}^{N_{\rm L}}\frac{\lambda_a\mu_B c^d_{\rm L}x_j}{\Psi^2\Psi_{\rm H}}\\=&c^d_{\rm H}\frac{\sum_{j=1}^{N_{\rm H}}\lambda_a x_j}{\Psi^2_{\rm H}}+\frac{\mu_B c^d_{\rm L}}{\Psi^2\Psi_{\rm H}}\frac{\Psi(\Psi_{\rm H}-\Psi)}{\Psi_{\rm H}} \qquad (31a)\\<&\frac{\mu_B c^d_{\rm H}}{\Psi\Psi_{\rm H}}-\frac{c^d_{\rm H}}{\Psi_{\rm H}}=c^d_{\rm H}(D^{\rm EC}_{\rm L}-D^{\rm EC}_{\rm H}), \qquad (31b)\end{aligned}$$

where (31a) comes from the fact that $\Psi_{\rm H}-\Psi=\sum_{j=1}^{N_{\rm L}}\lambda_a x_j$; the inequality in (31b) comes from the fact that $c^d_{\rm H}>c^d_{\rm L}$. Similarly, it can be verified that the last inequality in (31b) can be replaced by $p_{\rm H}-p_{\rm L}>c^d_{\rm L}(D^{\rm EC}_{\rm L}-D^{\rm EC}_{\rm H})$. This completes the proof.

Appendix C
Proof of *Theorem 3*

Let $(D^\star_{\rm L,EC},D^\star_{\rm H,EC})$ and $(p^\star_{\rm L},p^\star_{\rm H})$ respectively represent the optimal delays and prices arising from the system-wide welfare offloading, denoted by $x^\star_k$. In the following, we will respectively show the convergence of the inner and the outer loop, followed by the proof that Algorithm 2 converges to the offloading decisions that jointly maximzie the system-wide welfare.

(1) For any given fixed average delay of the high priority queue, i.e., $D^t_{\rm H,EC}=D_{\rm H,EC}$, the inner loop converges to a point where $D^{{\rm true},t}_{\rm L,EC}=D^t_{\rm L,EC}$. Some observations are in order. First, when setting $D^t_{\rm L,EC}=D_{\rm H,EC}+\varsigma$ we get $D^{{\rm true},t}_{\rm L,EC}>D^t_{\rm L,EC}$, and when setting $D^t_{\rm L,EC}=\infty-\varsigma$ we get $D^{{\rm true},t}_{\rm L,EC}<D^t_{\rm L,EC}$. Here ς is an almost zero but positive value. Second, $D^{{\rm true},t}_{\rm L,EC}$ is a decreasing function of $D^t_{\rm L,EC}\in(D_{\rm H,EC},\infty)$. This can be proved as follows. Assume that $D^{t_1}_{\rm L,EC}<D^{t_2}_{\rm L,EC}$. Then, $p^{t_1}_{\rm H}<p^{t_2}_{\rm H}$ and $p^{t_1}_{\rm L}<p^{t_2}_{\rm L}$. This, combined with the fact that the demand function $g_k(x_k)$ in (25) is monotonically decreasing, indicates that $x^{t_1}_k>x^{t_2}_k$. Hence, it holds that $D^{{\rm true},t_1}_{\rm L,EC}>D^{{\rm true},t_2}_{\rm L,EC}$. Concluding the above observations, the inner loop converges to a point where $D^{{\rm true},t}_{\rm L,EC}=D^t_{\rm L,EC}$, where, for the ease of exposition, we respectively use $D^e_{\rm L,EC}$, $p^e_{\rm L}$ and $p^e_{\rm H}$ to represent the equilibrium delay and prices.

(2) The outer loop converges to the optimal point where $D^{{\rm true},t}_{\rm H,EC}=D^t_{\rm H,EC}=D^\star_{\rm H,EC}$. In the following, we will prove by contradiction that $D^{{\rm true},t}_{\rm H,EC}<D^t_{\rm H,EC}$ holds true only if $D^t_{\rm H,EC}>D^\star_{\rm H,EC}$; the argument for the inverse case is similar. Concluding, the outer loop converges to a point where $D^{{\rm true},t}_{\rm H,EC}=D^t_{\rm H,EC}=D^\star_{\rm H,EC}$.

Assume $D^t_{\rm H,EC}\le D^\star_{\rm H,EC}$. Then, we have the following inference.

i) Its embedded inner loop converges to a point where

$$p^e_{\rm L}+c^d_{\rm L}D^e_{\rm L,EC}\le p^\star_{\rm L}+c^d_{\rm L}D^\star_{\rm L,EC}; \tag{32}$$

otherwise, assume that $p^e_{\rm L}+c^d_{\rm L}D^e_{\rm L,EC}>p^\star_{\rm L}+c^d_{\rm L}D^\star_{\rm L,EC}$, and since the demand function $g_k(x_k)$ in (25) is monotonically decreasing with respect to x_k, the resulting offloading frequency of users associated with the low priority queue satisfies $x_k<x^\star_k$. This, combined with the assumption $D^t_{\rm H,EC}\le D^\star_{\rm H,EC}$, indicates that $p^e_{\rm L}+c^d_{\rm L}D^e_{\rm L,EC}<p^\star_{\rm L}+c^d_{\rm L}D^\star_{\rm L,EC}$, which, however, contradicts with the assumption.

ii) It holds that $D^{{\rm true},t}_{\rm H,EC}\ge D^t_{\rm H,EC}$. This can be explained as follows. By Algorithm 2, the posted average edge delay and price candidates always respectively satisfying (21a)(21b) and (26a)(26b), based on which it can be verified that

$$\begin{aligned}&p^e_{\rm H}+c^d_{\rm H}D^t_{\rm H,EC}\\&=(c^d_{\rm H}-c^d_{\rm L})\mu_B {D^t_{\rm H,EC}}^2+p^e_{\rm L}+c^d_{\rm L}D^e_{\rm L,EC} \qquad (33a)\\&\le(c^d_{\rm H}-c^d_{\rm L})\mu_B {D^\star_{\rm H,EC}}^2+p^\star_{\rm L}+c^d_{\rm L}D^\star_{\rm L,EC} \qquad (33b)\\&=p^\star_{\rm H}+c^d_{\rm H}D^\star_{\rm H,EC}, \qquad (33c)\end{aligned}$$

where (33b) comes from the inference (32) and the assumption $D^t_{\rm H,EC}\le D^\star_{\rm H,EC}$. In addition, since the demand function $g_k(x_k)$ is monotonically decreasing with respect to x_k, the resulting offloading frequency of users associated with the high priority queue satisfies $x_k\ge x^\star_k$. As such, it holds that $D^{{\rm true},t}_{\rm H,EC}\ge D^\star_{\rm H,EC}\ge D^t_{\rm H,EC}$.

The second inference contradicts with the fact that $D^{{\rm true},t}_{\rm H,EC}<D^t_{\rm H,EC}$. This completes the proof that $D^{{\rm true},t}_{\rm H,EC}<D^t_{\rm H,EC}$ holds true only if $D^t_{\rm H,EC}>D^\star_{\rm H,EC}$.

(3) Combining the above analysis (1) and (2), one can see that Algorithm 2 converges to a point where $D^{{\rm true},t}_{\rm L,EC}=D^t_{\rm L,EC}$ and $D^{{\rm true},t}_{\rm H,EC}=D^t_{\rm H,EC}$. This, combined with Theorem 1, indicates that Algorithm 2 converges to offloading decisions that jointly maximize the system-wide welfare. This completes the proof.

References

[1] A. B. Ericsson, "Ericsson mobility report: On the pulse of the networked society," in *Tech. Rep. EAB-14 61078*, Ericsson, Sweden, 2015.

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

[3] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture," *IEEE Commun. Surveys & Tuts.*, vol. 19, no. 3, pp. 1628–1656, Third quarter 2017.

[4] Y. Mao, C. You, J. Zhang, and et al., "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys & Tuts.*, vol. 19, no. 4, pp. 2322–2358, Fourth quarter 2017.

[5] W. Yu, F. Liang, X. He, and et al., "A survey on the edge computing for the internet of things," *IEEE Access*, vol. 6, 2018.

[6] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.

[7] C. You, K. Huang, and H. Chae, "Energy efficient mobile cloud computing powered by wireless energy transfer," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1757–1771, May 2016.

[8] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proc. IEEE ISIT*, Barcelona, Spain, Jul. 2016, pp. 1451–1455.

[9] Y. Tao, C. You, P. Zhang, and K. Huang, "Stochastic control of computation offloading to a dynamic helper," in *ICC Workshops*, Kansas City, MO, USA, May 2018.

[10] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Power-delay tradeoff in multi-user mobile-edge computing systems," in *GLOBECOM*, Washington, DC, USA, Dec. 2016, pp. 1–6.

[11] ——, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sep. 2017.

[12] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[13] M.-H. Chen, B. Liang, and D. Ming, "Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point," in *INFOCOM*, Atlanta, GA, USA, Apr. 2017, pp. 1863–1871.

[14] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Adaptive computation scaling and task offloading in mobile edge computing," in *WCNC*, San Francisco, CA, USA, Mar. 2017, pp. 1–6.

[15] L. Pu, X. Chen, J. Xu, and X. Fu, "D2D fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted D2D collaboration," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3887–3901, Dec. 2016.

[16] F. Wang and X. Zhang, "Dynamic interface-selection and resource allocation over heterogeneous mobile edge-computing wireless networks with energy harvesting," in *INFOCOM WKSHPS*, Honolulu, HI, USA, Apr. 2018, pp. 190–195.

[17] F. Guo, L. Ma, H. Zhang, H. Ji, and X. Li, "Joint load management and resource allocation in the energy harvesting powered small cell networks with mobile edge computing," in *INFOCOM WKSHPS*, Honolulu, HI, USA, Apr. 2018, pp. 754–759.

[18] J. Kwak, Y. Kim, J. Lee, and S. Chong, "DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2510–2523, Dec. 2015.

[19] G. Zhang, W. Zhang, Y. Chao, and L. Wang, "Energy-delay tradeoff for dynamic offloading in mobile-edge computing system with energy harvesting devices," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4642–4655, Oct. 2018.

[20] T. Ouyang, Z. Zhou, and X. Chen, "Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing," *IEEE J. Sel. Areas Commun.*, vol. 99, no. 99, pp. 1–1, Sep. 2018.

[21] C.-F. Liu, M. Bennis, and H. V. Poor, "Latency and reliability-aware task offloading and resource allocation for mobile edge computing," in *GLOBECOM WKSHPS*, Singapore, Dec. 2017, pp. 1–7.

[22] S.-W. Ko, K. Han, and K. Huang, "Wireless networks for mobile edge computing: Spatial modeling and latency analysis," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5225–5240, Aug. 2018.

[23] S. Guo, D. Wu, H. Zhang, and D. Yuan, "Queueing network model and average delay analysis for mobile edge computing," in *ICNC WKSHPS*, Maui, Hawaii, USA, Mar. 2018, pp. 172–176.

[24] S. Rao and E. R. Petersen, "Optimal pricing of priority services," *Operations Research*, vol. 46, no. 1, pp. 46–56, Jan. 1998.

[25] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3435–3447, Apr. 2017.

[26] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

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

[28] S. Guo, B. Xiao, Y. Yang, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *INFOCOM*, San Francisco, CA, USA, Apr. 2016, pp. 1–9.

[29] J. Zheng, Y. Cai, Y. Wu, and X. S. Shen, "Dynamic computation offloading for mobile cloud computing: A stochastic game-theoretic approach," *IEEE Trans. Mobile Comput.*, vol. 99, no. 99, pp. 1–1, 2018.

[30] L. Tang and X. Chen, "An efficient social-aware computation offloading algorithm in cloudlet system," in *GLOBECOM*, Washington, DC, USA, Dec. 2016, pp. 1–6.

[31] C. Courcoubetis and R. Weber, *Pricing Communication Networks: Economics, Technology and Modelling*. Hoboken, NJ, USA: Wiley, 2003.

[32] J. Huang and L. Gao, *Wireless Network Pricing*. San Rafael, CA, USA: Morgan & Claypool, 2013.

[33] W. Fang, X. Yao, X. Zhao, J. Yin, and N. Xiong, "A stochastic control approach to maximize profit on service provisioning for mobile cloudlet platforms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 4, pp. 522–534, Apr. 2018.

[34] A.-L. Jin, W. Song, P. Wang, D. Niyato, and P. Ju, "Auction mechanisms toward efficient resource sharing for cloudlets in mobile cloud computing," *IEEE Trans. Services Comput.*, vol. 9, no. 6, pp. 895–909, Nov. 2016.

[35] A. Kiani and N. Ansari, "Toward hierarchical mobile edge computing: An auction-based profit maximization approach," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 2082–2091, Dec. 2017.

[36] Y. Kim, J. Kwak, and S. Chong, "Dual-side optimization for cost-delay tradeoff in mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1765–1781, Feb. 2018.

[37] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[38] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *HotCloud'10 Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, Boston, MA, USA, 2010, pp. 1–7.

[39] R. Hassin and M. Haviv, *To Queue Or Not to Queue: Equilibrium Behavior in Queueing Systems*. Kluwer Academic Publishers, 2003.

[40] B. A. and E. Sid, "Optimal priority assignment with heterogeneous waiting costs," *Operations Research*, vol. 23, no. 1, pp. 107–117, Feb. 1975.

[41] S. Kosta, A. Aucinas, P. Hui, and et al., "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *INFOCOM*, Orlando, Florida, Mar. 2012, pp. 945–953.