Price-Based Distributed Offloading for Mobile-Edge Computing with Computation Capacity Constraints

Mengyu Liu and Yuan Liu, *Member, IEEE*

Abstract—Mobile-edge computing (MEC) is a promising technology to enable real-time information transmission and computing by offloading computation tasks from wireless devices to network edge. In this study, we propose a price-based distributed method to manage the offloaded computation tasks from users. A Stackelberg game is formulated to model the interaction between the edge cloud and users, where the edge cloud sets prices to maximize its revenue subject to its finite computation capacity, and for given prices, each user locally makes offloading decision to minimize its own cost which is defined as latency plus payment. Depending on the edge cloud's knowledge of the network information, we develop the uniform and differentiated pricing algorithms, which can both be implemented in distributed manners. Simulation results validate the effectiveness of the proposed schemes.

Index Terms—Mobile-edge computing (MEC), computation offloading, pricing, Stackelberg game.

I. INTRODUCTION

With the increasing popularity of new mobile applications of Internet of Things (IoT), future wireless networks with billions of IoT devices are required to support ultra low-latency communication and computing. However, the IoT devices are usually with small physical sizes and limited batteries, thus always suffer from intensive computation and high resource-consumption for real-time information processing [1]–[3].

Mobile-edge computing (MEC) is a promising technology to address this problem. Unlike conventional cloud computing integrated with remote central clouds results in long latency and fragile wireless connections, MEC migrates intensive computation tasks from IoT devices to the physically proximal network edge, and provides low-latency as well as flexible computing and communication services for IoT devices. As a result, MEC is commonly agreed as a key technology to realize next-generation wireless networks [4].

Joint radio and computation resource allocation for MEC has been recently investigated in the literature, e.g., [5]–[12]. In general, the MEC paradigm can be divided into two categories: binary offloading [5]–[9] and partial offloading [10]–[12]. With binary offloading, the computation tasks at users can not be partitioned but must be executed as a whole

The authors are with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: liu.mengyu@mail.scut.edu.cn, eeyliu@scut.edu.cn).

either at users or at edge cloud. With partial offloading, the computation tasks at users can be partitioned into different parts for local computing and offloading at the same time.

Among the mentioned literature, a handful of works [8], [9] adopted game theory to design distributed mechanisms for efficient resource allocation in MEC systems. For example, multiuser binary offloading was considered in [8] using a Nash game to maximize the offloaded tasks subject to the total time and energy constraints. The work [9] considered competition among multiple heterogeneous clouds. In view of prior works, most of them assumed that the computation capacity of the edge cloud is infinite. However, as MEC server is located at network edge, its computation capacity should be finite, especially in the networks with intensive workloads. In this case, a mechanism is needed to control users' offloaded tasks to make the network feasible. Moreover, the mechanism should provide reasonable incentives for both edge cloud and users to efficiently allocate network resources in a distributed fashion.

Motivated by the above issues, we consider an edge cloud with finite computation capacity, which is treated as a divisible resource to be sold among the users. The interaction between the edge cloud and users is modeled as a Stackelberg game, where the edge cloud sets prices to maximize its revenue and each user designs the offloading decision individually to minimize its cost that is defined as latency plus payment. The main contributions are two-fold: 1) We propose a new game-based distributed MEC scheme, where the users compete for the edge cloud's finite computation resources via a pricing approach, which is modeled as a Stackelberg game. 2) The optimal uniform and differentiated pricing algorithms are proposed, which can be implemented with a distributed manner.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. System Model

We consider a MEC system with K users and one base station (BS) that is integrated with a MEC server to execute the offloaded data of the users. All nodes have a single antenna. The users' computation data can be arbitrarily divided in bit-wise for partial local computing and partial offloading. We assume that the total bandwidth B is equally divided for K users such that each user can occupy a non-overlapping frequency to offload its data to the edge cloud simultaneously. The quasi-static channel model is considered, where channels remain unchanged during each offloading period, but can vary in different offloading periods. We also assume that the computation offloading can be completed in a period.

Let C_k denote the number of CPU cycles for computing 1-bit of input data at user k. It is assumed that user k has to

execute R_k bits of input data in total, where $0 \leq \ell_k \leq R_k$ bits are offloaded to the edge cloud while the rest $(R_k - \ell_k)$ bits are computed by its local CPU. The local CPU frequency of user k is denoted as F_k that is measured by the number of CPU cycles per second. Then the time for local computing at user k is $t_{\text{loc},k} = (R_k - \ell_k)C_k/F_k$. The offloading time $t_{\text{off},k}$ of user k comprises three parts: the uplink transmission time $t_{\text{u},k}$, the execution time at the cloud $t_{\text{c},k}$, and the downlink feedback time $t_{\text{d},k}$. Thus, the offloading time $t_{\text{off},k}$ is

$$t_{\text{off},k} = t_{\text{u},k} + t_{\text{c},k} + t_{\text{d},k}. \tag{1}$$

Since the local computing and offloading can be performed concurrently, the required time of user k for executing the total R_k bits data can be expressed as $t_k = \max\{t_{\text{loc},k}, t_{\text{off},k}\}$.

More specially, the data size of the computed result fed back to user k is $\alpha_k \ell_k$, where α_k $(\alpha_k > 0)$ accounts for the ratio of output to input bits offloaded to the cloud [13], which depends on the applications of the users. Then we have $t_{\text{u},k} = \ell_k/r_k$ and $t_{\text{d},k} = \alpha_k \ell_k/r_{\text{B},k}$, where $r_k = \frac{B}{K}\log_2\left(1 + \frac{p_k h_k}{B/KN_0}\right)$ and $r_{\text{B},k} = \frac{B}{K}\log_2\left(1 + \frac{P_{\text{B},k} h_k}{B/KN_0}\right)$ denote the uplink and downlink transmission rates for user k, respectively. Here N_0 is the noise power spectrum density, h_k is the channel gain between the BS and user k, and $P_{\text{B},k}$ and p_k are the downlink and uplink power for user k, respectively. Moreover, let $f_{\text{c},k}$ denote the computational speed of the edge cloud assigned to user k, then we have $t_{\text{c},k} = \ell_k C_k/f_{\text{c},k}$. Here we consider equal $f_{\text{c},k}$ allocation for simplicity, i.e., $f_{\text{c},k} = f_C/K$, where f_C is the total computational speed of the could.

We consider a practical constraint that the edge cloud has finite computation capacity so that its CPU cycles for computing the sum received data in each offloading period are upper bounded by $\overline{F}$, the constraint can be expressed as

$$\sum_{k=1}^{K} \ell_k C_k \leq \overline{F}. \tag{2}$$

Note that $\overline{F}$ and f_C represent the MEC server's computational quantity and speed for the offloaded CPU cycles, respectively.

B. Stackelberg Game Formulation

In this paper, the users consume the edge cloud's resources to execute the computation tasks while the edge cloud has to ensure its available CPU cycles for computing the total offloaded data to be below the computation capacity. Hence, to adjust the demand and supply of the computation resources, it is considered that the edge cloud prices the CPU cycles $\ell_k C_k$ of the offloaded data for each user k. Thus the Stackelberg game can be applied to model the interaction between the edge cloud and users, where the edge cloud is the leader and the users are the followers. The edge cloud (leader) first imposes the prices for CPU cycles of users. Then, the users (followers) divide their input data for local computing and offloading individually based on the prices announced from the edge cloud.

Denote the CPU cycle prices for users as a set $\boldsymbol{\mu} = \{\mu_1, \cdots, \mu_K\}$. The objective of the edge cloud is to maximize its revenue obtained from selling the finite computation resources to users. Mathematically, the optimization problem at the edge cloud's side can be expressed as (leader problem)

$$\textbf{P1}: \quad \max_{\boldsymbol{\mu} \succeq 0} \quad U_B(\boldsymbol{\mu}) = \sum_{k=1}^{K} \mu_k \ell_k C_k \quad \text{s.t.} \quad (2).$$

Note that the offloaded data ℓ_k for user k is actually a function of μ_k, since the size of data that each user is willing to offload is dependent on its assigned price.

At the users' side, each user's cost is defined as its latency plus the payment charged by the edge cloud, i.e.,

$$U_k(\ell_k, \mu_k) = t_k + \mu_k \ell_k C_k, \tag{3}$$

which is equivalent to

$$U_k(\ell_k, \mu_k) = \begin{cases} \left(\mu_k - \frac{1}{F_k}\right)\ell_k C_k + \frac{R_k C_k}{F_k}, & 0 \leq \ell_k \leq m_k, \\ \beta_k \ell_k + \mu_k \ell_k C_k, & m_k < \ell_k \leq R_k, \end{cases} \tag{4}$$

where $\beta_k = \frac{1}{r_k} + \frac{C_k}{f_{\text{c},k}} + \frac{\alpha_k}{r_{\text{B},k}}$ and $0 < m_k < R_k$ is defined as $m_k = \frac{C_k R_k}{\beta_k F_k + C_k}$.

The goal of each user k is to minimize its own cost by choosing the optimal offloaded data size ℓ_k for given price μ_k set by the edge cloud. Mathematically, this problem can be expressed as (follower problem)

$$\textbf{P2}: \quad \min_{\ell_k} \quad U_k(\ell_k, \mu_k) \quad \text{s.t.} \quad 0 \leq \ell_k \leq R_k.$$

It is worth noting that the payment term in Problem **P1** and Problem **P2** can be cancelled each other from the net utility perspective. Problem **P1** and Problem **P2** in the Stackelberg game are coupled in a complicated way, i.e., the pricing strategies of the edge cloud have an influence on the offloaded data sizes of the users which also impact the edge cloud's revenue in turn.

III. OPTIMAL ALGORITHM

To analyze the considered Stackelberg game, each user k independently decides its offloading strategy ℓ_k^* by solving Problem **P2** with given price $\boldsymbol{\mu}$. Knowing each user's offloading decision $\ell_k^*(\mu_k)$, the edge cloud sets its optimal price $\boldsymbol{\mu}^*$ by solving Problem **P1**. The above process is known as the backward induction. In this paper, two optimal pricing strategies are considered, which are termed as uniform pricing and differentiated pricing [14]. In the following, we will investigate the two pricing schemes respectively.

A. Uniform Pricing

For the uniform pricing scheme, the edge cloud sets and broadcasts a uniform price to all users, i.e., $\mu = \mu_1 \cdots = \mu_K$. For given uniform price μ, the objective function U_k is a piecewise function of ℓ_k, which is linear in each interval from (4). Then, by exploiting the structure of U_k, we can obtain the optimal solution for Problem **P2** in the following proposition.

Proposition 1: The optimal offloading decision of each user in Problem **P2** follows the threshold-based policy, i.e.,

$$\ell_k^*(\mu) = m_k x_k, \quad \forall k, \tag{5}$$

where the binary variable x_k is defined as

$$x_k = \begin{cases} 1, & \mu \leq \frac{1}{F_k}, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

Proof: Please refer to Appendix A. ∎

From Proposition 1, we obtain that the offloading threshold is $1/\mu$, i.e., user k prefers to offload m_k bits to the edge cloud if its CPU frequency F_k is smaller than or equal to the threshold, and leaves all bits for local computing otherwise. In other words, the computation offloading is beneficial if the user has small computational speed F_k and it is likely to compute locally otherwise.

Then we turn our attention to Problem **P1**. By substituting (5) into Problem **P1**, the optimization problem at the edge cloud for the uniform pricing scheme can be rewritten as

$$\textbf{P3}: \quad \max_{\mu \geq 0} \quad U_B(\mu) = \mu \sum_{k=1}^{K} m_k x_k C_k \quad (7)$$

$$\text{s.t.} \quad \sum_{k=1}^{K} m_k x_k C_k \leq \overline{F}. \quad (8)$$

Proposition 2: Without loss of generality, we sort $1/F_1 < \cdots < 1/F_{K-1} < 1/F_K$, the optimal uniform price μ^* must belong to the set $\{1/F_1, \cdots, 1/F_{K-1}, 1/F_K\}$.

Proof: Please refer to Appendix B. ∎

According to Proposition 2, the revenue maximization problem in **P3** reduces to the one-dimensional search problem over K values in $\{1/F_k\}_{k=1}^{K}$ and we summarize the whole method in Algorithm 1 formally. Specially, the edge cloud bargains with the users by announcing price in the decreasing order of $\{1/F_k\}_{k=1}^{K}$. Since the required sum CPU cycles $\sum_{k=1}^{K} \ell_k C_k$ decreases with the price μ, the price bargaining ends as long as the computation capacity constraint (8) is active and there is no need for bargaining the rest of price candidates.

It is obvious that the total complexity of Algorithm 1 to search μ^* is $\mathcal{O}(\log K)$. For the uniform pricing scheme, the edge cloud needs the limited network information, i.e., F_k and C_k, which are collected by the cloud before the algorithm. In each iteration, after knowing the price μ broadcasted by the edge cloud, each user independently makes its offloading decision ℓ_k and reports it to the edge cloud for updating the price. Therefore, the cloud broadcasts the price μ and each user reports its offloading decision ℓ_k, which are the information exchanged between the edge cloud and the users in each iteration. Hence, Algorithm 1 is a fully distributed algorithm.

B. Differentiated Pricing

Here, we consider the general case where the edge cloud charges the different users with different prices. Similar to the uniform pricing case, the optimal solution for Problem **P2** is also (5), except that μ in x_k is replaced by μ_k. And, Problem **P1** can be rewritten as

$$\textbf{P4}: \quad \max_{\boldsymbol{\mu} \succeq 0} \quad U_B(\boldsymbol{\mu}) = \sum_{k=1}^{K} \mu_k m_k x_k C_k \quad \text{s.t.} \quad (8).$$

It is worth noting that the price μ_k is actually a function of x_k for user k. Specifically, $x_k = 1$, i.e., $\mu_k \leq 1/F_k$ and the

---

Algorithm 1 Optimal Uniform Pricing Policy for Problem **P3**

1: The edge cloud initializes $\tau = K$ and $\mu^\tau = 1/F_\tau$.
2: **repeat**
3:      Every user decides its optimal offloaded data size $\ell_k^*(\mu^\tau)$ according to (5).
4:      The edge cloud computes its revenue $U_B(\mu^\tau)$ from (7).
5:      **if** $\sum_{k=1}^{K} \ell_k^*(\mu^\tau) C_k \leq \overline{F}$ **then**
6:          Update the price $\mu^{\tau-1} = 1/F_{\tau-1}$, and $\tau \leftarrow \tau - 1$;
7:      **else**
8:          Set $U_B(\mu^\tau) = 0$; **beak;**
9:      **end if**
10: **until** $\tau \leq 0$.
11: Output $\mu^* \leftarrow \arg\max_{\mu^\tau} U_B(\mu^\tau)$.

---

optimal price for user k is thus given by $\mu_k^* = 1/F_k$ as the objective function of Problem **P4** is an increasing function of μ_k. When $x_k = 0$, the edge cloud sets the price for user k as $\mu_k^* = \infty$ and earns no revenue. Based on the above analysis, Problem **P4** is thus equivalent to

$$\textbf{P4}': \quad \max_{x_k \in \{0,1\}} \quad U_B(x_k) = \sum_{k=1}^{K} \frac{m_k x_k C_k}{F_k} \quad \text{s.t.} \quad (8).$$

Problem **P4$'$** is actually a *binary knapsack problem* with the weight $m_k C_k$ and the value $m_k C_k / F_k$ for user k. Since the problem is NP-complete, there is no efficient algorithm solving it optimally. However, we can apply dynamic programming [15] to solve the above binary knapsack problem in pseudo-polynomial time.[1]

Each user k needs to report m_k, C_k, and F_k to the edge cloud for solving Problem **P4$'$** and there is no need for iteration between the edge cloud and users. Obtaining the optimal price μ_k^*, user k decides its optimal strategy based on (5). Therefore, the differentiated pricing scheme is also a distributed algorithm, but it needs more information and higher complexity than that of the uniform pricing scheme.

IV. NUMERICAL RESULTS

In the simulation setup, we assume that the total channel bandwidth B is 1 MHz and the noise power spectrum density N_0 is -174 dBm/Hz. Each h_k is assumed to be uniformly distributed in $[-50, -30]$ dBm. The local CPU frequency F_k for each user k is uniformly selected from the set $\{0.1, 0.2, \cdots, 1\}$ GHz, and the required number of CPU cycles per bit and the data size for user k are uniformly distributed with $C_k \in [500, 1500]$ cycles/bit and $R_k \in [100, 500]$ KB, respectively. Unless otherwise noted, the remaining parameters are set as follows: $p_k = 0.1$ W, $P_{B,k} = 1$ W, $\overline{F} = 6 \times 10^9$ cycles/slot, $\alpha_k = 0.2$, and $f_C = 100$ GHz.

The average performance of the two proposed pricing schemes is evaluated and compared in terms of average latency and revenue. Besides, we consider the scheme where all input data is computed locally at users for comparison. In Fig. 1(a), both latency and revenue performance become better as the computation capacity increases, while the scheme of only local

---

[1]We adopt kp01($\cdot$) software package in MATLAB.

(a) $K = 30$

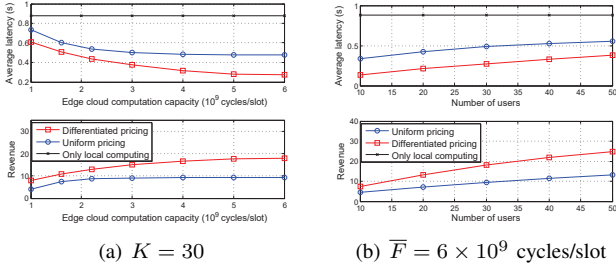(b) $\overline{F} = 6 \times 10^9$ cycles/slot

Fig. 1: Performance comparison.

computing has the worst latency performance and is not related with the computation capacity. In addition, the differentiated pricing scheme has better performance in both latency and revenue, which shows that it is more accurate to allocate resource. Thus there exists a tradeoff between performance and complexity for the two pricing schemes.

Fig. 1(b) illustrates the effect of the number of users on the average latency and revenue, and we have the similar observations as Fig. 1(a) for the three schemes. Besides, with the increasing number of users, the allocated spectrum for each user decreases, resulting in lower transmission rate and thus higher latency. Moreover, it is expected that competition with more users forces up the prices and revenue of the edge cloud.

V. CONCLUSION

In this work, we investigated the price-based computation offloading for a multiuser MEC system. The finite computation capacity of the edge cloud was considered to manage the offloaded tasks from users. The Stackelberg game was applied to model the interaction between the edge cloud and users. Based on the edge cloud's knowledge of the network information, we proposed uniform and differentiated pricing schemes, which can both be implemented with distributed manners.

APPENDIX A
THE PROOF OF PROPOSITION 1

For given μ, the optimal solution for Problem **P2** is

$$\ell_k^*(\mu) \begin{cases} = m_k, & \mu < \frac{1}{F_k}, \\ \in [0, m_k], & \mu = \frac{1}{F_k}, \\ = 0, & \text{otherwise.} \end{cases} \tag{9}$$

The case $\mu = 1/F_k$ for all k occurs with probability 0 and we let $\ell_k = m_k$ in this case, which completes the proof.

APPENDIX B
THE PROOF OF PROPOSITION 2

It can be proved by contradiction as follows. Suppose that the optimal price μ^* can exist in the interval $(1/F_i, 1/F_{i+1})$, $\forall i \in \{1, \cdots, K-1\}$. Then, we consider the case $\widetilde{\mu} = 1/F_{i+1}$. From (6), we can obtain that $x_k = 0$ with $k = 1, \cdots, i$ and $x_k = 1$ with $k = i+1, \cdots, K$ for both $\widetilde{\mu}$ and μ^*. Therefore, the CPU cycles of the sum offloaded data $\sum_{k=1}^{K} m_k x_k C_k$ for $\widetilde{\mu}$ are equivalent to that for μ^*. As the objective function $U_B(\mu)$ given in (7) is an increasing linear function of the price

μ, we can always have that the case $\widetilde{\mu} = 1/F_{i+1}$ achieves a higher revenue than the case μ^*. Thus, this contradicts with the assumption that μ^* is optimal for Problem **P3** with $1/F_i < \mu^* < 1/F_{i+1}$. Therefore, the optimal price μ^* must exist in the set $\{1/F_1, ..., 1/F_{K-1}, 1/F_K\}$.

REFERENCES

[1] A. u. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 393–413, February 2014.

[2] K. Kumar and Y. H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, April 2010.

[3] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," Available: https://arxiv.org/abs/1701.01090.

[4] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, October 2016.

[5] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 1, no. 2, pp. 89–103, June 2015.

[6] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks," *IEEE Access*, vol. 4, pp. 5896–5907, 2016.

[7] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, December 2016.

[8] 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, October 2016.

[9] T. Zhao, S. Zhou, X. Guo, Y. Zhao, and Z. Niu, "Pricing policy and computational resource provisioning for delay-aware mobile edge computing," in *2016 IEEE/CIC International Conference on Communications in China (ICCC)*, July 2016, pp. 1–6.

[10] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, September 2017.

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

[12] J. Xu, L. Chen, and S. Ren, "Online learning for offloading and autoscaling in energy harvesting mobile edge computing," *IEEE Transactions on Cognitive Communications and Networking*, vol. PP, no. 99, pp. 1–1, 2017.

[13] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Transactions on Communications*, vol. 64, no. 10, pp. 4268–4282, October 2016.

[14] Y. Liu, R. Wang, and Z. Han, "Interference-constrained pricing for D2D networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 1, pp. 475–486, January 2017.

[15] S. Martello, D. Pisinger, and P. Toch, "Dynamic programming and strong bounds for the 0-1 knapsack problem," *Management Science*, vol. 45, no. 3, pp. 414–424, 1999.