This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LWC.2018.2810292, IEEE Wireless Communications Letters



# Computation Offloading and Activation of Mobile Edge Computing Servers: A Minority Game

Shermila Ranadheera, Setareh Maghsudi, and Ekram Hossain, *IEEE Fellow*

***Abstract*—With the ever-increasing popularity of resource-intensive mobile applications, Mobile Edge Computing (MEC), e.g., offloading computationally expensive tasks to the cellular edge, has become a prominent technology for the next generation wireless networks. Despite its great performance in terms of delay and energy, MEC suffers from restricted power allowance and computational capability of the edge nodes. Therefore, it is imperative to develop distributed mechanisms for computation offloading, so that not only the computational servers are utilized at their best capacity, but also the users' latency constraints are fulfilled. In this letter, by using the theory of Minority Games, we develop a novel distributed server activation mechanism for computation offloading. Our scheme guarantees energy-efficient activation of servers as well as satisfaction of users' quality-of-experience (QoE) requirements in terms of latency.**

*Keywords*: Computation offloading, mobile edge computing, server mode selection, minority game.

## I. Introduction

Due to the ever-increasing popularity of computationally intensive applications, computation offloading capability has become a prerequisite for next generation wireless networks. Since the energy, storage, and computing capacity of small mobile devices are limited, mobile users need to transfer computationally expensive tasks to powerful computing servers. Despite its higher computational capability, remote cloud may not be the ideal option, as the long distance between the cloud and the user device yields substantial latency and energy cost. In contrast, small scale computing servers located in the network edge might provide services at reduced latency and energy cost, compared to the remote cloud. This is referred to as Mobile Edge Computing (MEC) [1]. Naturally, efficient utilization of MEC servers is vital, since they have limited computational resources and power. To this end, one solution is to activate only a specific number of servers, while keeping the rest in the energy saving mode. At the same time, users' latency requirements should be taken into account, as overloading the servers with computational tasks can result in unacceptable delay. Therefore, addressing this trade-off is a major issue in developing efficient MEC systems. This becomes challenging in the presence of uncertainty in task arrival and/or in the absence of any central controller. Other challenges include minimizing users' energy consumption and efficient radio resource management.

In [1], the authors develop a distributed algorithm in a game-theoretic framework to address the decision making problem for computation offloading by the users. In [2], the authors investigate a computational offloading problem, where mobile users offload to a variety of edge nodes such as macro base stations and access points. Reference [3] provides centralized resource allocation algorithms for mobile edge computation offloading system, which minimize the weighted sum energy consumption under delay constraints. A multi-objective offloading problem is formulated and analyzed using queuing theory in [4]. In [5], the authors address optimization of offloading decision making for mobile users in the presence of fading. A comprehensive survey can be found in [6].

The majority of the existing literature focus on user-centric objectives such as meeting users' delay constraints and minimizing users' energy consumption. On the contrary, our work presents a two-sided view, where both servers' and users' standpoints are considered. In doing so, we address the uncertainty caused by the randomness in channel quality and users' requests. We first analyze the statistical characteristics of the offloading delay. Based on this, we model the computational offloading problem as a planned market, where the price of computational services is determined by an authority. Afterward, by using the theory of minority games [7], we develop a novel approach for efficient mode selection (or activation) at the servers' side. The designed mode selection mechanism guarantees a minimal server activation to ensure energy efficiency, while meeting the users' delay constraints. Moreover, it is distributed, and does not require any prior information at the servers' side. We numerically investigate the performance of the proposed method.

In Section II, we present the system model and formulate the server activation problem. In Section III, we cast the servers' mode selection problem in a minority game-theoretic framework and provide an algorithmic solution. Numerical results and discussions are presented in Section IV.

## II. System model and problem formulation

We consider an MEC system consisting of a virtual pool of $M$ computational servers (e.g., small base stations), denoted by a set $\mathcal{M}$, and a set of users (e.g., mobile devices). Each user has some delay sensitive computational tasks to be completed in consecutive offloading periods. Each offloading period is referred to as one *time slot*. In every time slot $t$, computational jobs are offloaded by the users to the pool following a Poisson distribution, with mean job arrival rate of $\lambda$. Prior to task arrival, every server independently decides whether to

- accept computation jobs (*active* mode); or
- not to accept any computation job (*inactive* mode).

On one hand, to optimize the servers' energy consumption, selective activation of a limited number of servers is beneficial. On the other hand, from the users' point of view, the number of active servers should be large enough for the users to

S. Ranadheera and E. Hossain are with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, Canada (e-mail: ranadhes@myumanitoba.ca, Ekram.Hossain@umanitoba.ca). S. Maghsudi is with the Department of Electrical Engineering and Computer Science, Technical University of Berlin, Berlin, Germany (e-mail:maghsudi@tu-berlin.de).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LWC.2018.2810292, IEEE Wireless Communications Letters



meet their required QoE. Therefore, for the offloading system to perform efficiently, the number of active servers at any offloading round $t$, denoted by $c(t)$, should be determined in a way that both servers and users are satisfied. We denote this value by $c_{\text{th}}$. During each offloading round $t$, incoming jobs are equally divided among the $c(t)$ active servers, implying that the computational jobs arrive at every active server at a mean rate of $\lambda/c(t)$, following a Poisson distribution. Therefore, any active server can be modeled as an M/M/1 queuing system, where each job is processed in the order of arrival [8]. In what follows, we use some results from queuing theory [9] to derive the parameters of the described offloading system.

Let $\mu$ denote the service rate (the number of tasks processed per unit time) of any active server. Naturally, to avoid the infinite queue lengths, the arrival rate should be less than the service rate; thus, for servers to operate with finite queue length, we need to have $\lambda/c(t) < \mu$, or

$$c(t) > \frac{\lambda}{\mu}. \tag{1}$$

Hence $c_{\text{th}}$ must satisfy the following condition:

$$\text{Condition I:} \qquad c_{\text{th}} > c_{\min(1)}, \tag{2}$$

where $c_{\min(1)} = \lambda/\mu$. Clearly, the total time a job stays at any server (*response time*), denoted by $t_c$, is the sum of the waiting time in the queue, $t_w$, and the service time $t_s$. Since we consider an M/M/1 queue model, $t_s$ and $t_c$ are exponentially distributed with parameters $\mu$ and $\mu - \frac{\lambda}{c(t)}$, respectively [9].

Considering Rayleigh fading, the channel gain ($h$) is exponentially distributed with parameter $\nu$. We model the round trip transmission delay (from the user to the servers pool) as a linear function of the channel gain. The channel gains in both directions are assumed to be equal. Formally,[1]

$$t_0 = 2(ah + b), \tag{3}$$

where $a < 0$ and $b > 0$ are constants such that $t_0 \geq 0$. The total offloading delay $\theta$, is the sum of total delay at the server $t_c$, and the round trip transmission delay $t_0$. Thus,

$$\theta = t_c + t_0. \tag{4}$$

The following proposition characterizes $\theta$ statistically.[2]

**Proposition 1.** *The cumulative distribution function (cdf) of offloading delay $\theta$ can be calculated as*

$$F_\Theta(\theta) = 1 - \frac{\nu e^{(2b-\theta)(\mu-\lambda/c(t))}}{\nu - 2a(\mu - \lambda/c(t))}, \quad 2b - \theta < 0. \tag{5}$$

*The expected value and variance of $\theta$ are respectively given by*

$$\mathrm{E}(\theta) = \frac{1}{\mu - \frac{\lambda}{c(t)}} + \frac{2(a + b\nu)}{\nu}, \tag{6}$$

$$\mathrm{Var}(\theta) = \frac{1}{\left(\mu - \frac{\lambda}{c(t)}\right)^2} + \frac{4a^2}{\nu^2}. \tag{7}$$

[1]Assuming a linear model of the transmission delay for the transmission delay does not limit the applicability of the proposed model, and similar analysis can be performed with any other model.

[2]The proof follows by simple probability rules given the independence of $t_c$ and $t_0$. We omit the proof due to space limitation. Note that, for $2b - \theta \geq 0$, the cdf is slightly different; however, similar analysis can be done to determine the optimal number of active servers.

Every user requires its offloaded job(s) to be completed by some deadline $T$. Moreover, due to the uncertainty caused by the randomness, deterministic performance guarantee in terms of delay is not feasible. Therefore, we assume a probabilistic guarantee of users' QoE requirement. Formally, let $\Pr[\theta > T]$ be the probability that $\theta$ exceeds $T$, i.e., the likelihood that the delay requirement of some offloading user(s) is not satisfied. We require that $\Pr[\theta > T]$ remains below a predefined threshold $\beta$. That is,

$$\Pr[\theta > T] \leq \beta. \tag{8}$$

### A. Condition for Users

Recall that the users' QoE requirement is given by (8). Then, by (5) and (8), we require

$$\frac{\nu e^{(2b-T)(\mu-\lambda/c(t))}}{\nu - 2a(\mu - \lambda/c(t))} \leq \beta. \tag{9}$$

For simplicity of notation, let $y = 1 - \frac{2a}{\nu}\left(\mu - \frac{\lambda}{c(t)}\right)$. Then

$$e^{(2b-T)\frac{\nu(1-y)}{2a}} \leq \beta y. \tag{10}$$

Since $\ln(z)$ is an increasing function, we obtain

$$\frac{(2b - T)\nu(1 - y)}{2a} \leq \ln(\beta) + \ln(y). \tag{11}$$

Simple rearrangement of (11) yields

$$y + \frac{2a\ln(y)}{\nu(2b - T)} + \frac{2a\ln(\beta)}{\nu(2b - T)} - 1 \geq 0. \tag{12}$$

Define $p = 1, q = \frac{2a}{\nu(2b-T)}$, and $r = \frac{2a\ln(\beta)}{\nu(2b-T)} - 1$. Then, (12) boils down to $py + q\ln(y) + r \geq 0$. For the equality case,

$$\begin{aligned}
&\frac{p}{q}y + \ln(\frac{p}{q}y) + \frac{r}{q} - \ln\left(\frac{p}{q}\right) = 0 \\
&\implies e^{\frac{p}{q}y} e^{\ln\left(\frac{p}{q}y\right)} = e^{\ln\left(\frac{p}{q}\right)} e^{\frac{-r}{q}} \\
&\implies \frac{p}{q} y e^{\frac{p}{q}y} = \frac{p}{q} e^{\frac{-r}{q}} \\
&\implies y = \frac{q}{p} W\left(\frac{p}{q} e^{\frac{-r}{q}}\right)
\end{aligned}$$

with $W(z)$ being the *Lambert W function*. By substitution,

$$y = \frac{2a}{\nu(2b - T)} W\left(\frac{\nu(2b - T)}{2a} e^{-(\frac{2a\ln(\beta)}{\nu(2b-T)} - 1)/(\frac{2a}{\nu(2b-T)})}\right). \tag{13}$$

Therefore, for the equality case in (9), we obtain

$$c(t) = \frac{\lambda}{\mu - \frac{\nu(1-y)}{2a}}. \tag{14}$$

Obviously, with larger number of active servers, the users achieve better QoE. Hence, the value of $c(t)$ in (14) is the minimum number of active servers that guarantees the users' QoE requirement with high probability; that is,

$$c_{\min(2)} = \frac{\lambda}{\mu - \frac{\nu(1-y)}{2a}}. \tag{15}$$

Therefore, the following condition should be satisfied when selecting the threshold $c_{\text{th}}$:

$$\text{Condition II:} \qquad c_{\text{th}} \geq c_{\min(2)}. \tag{16}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LWC.2018.2810292, IEEE Wireless Communications Letters



### B. Condition for Servers

To become active, each server incurs a fixed energy cost represented by $e_{\mathrm{f}}$ (dimensionless value). In addition, doing *each task* yields an extra $e_{\mathrm{j}}$ units of energy cost. Naturally, the required energy to perform a job is directly proportional to the required service time $t_s$. Formally, $e_{\mathrm{j}} = \delta t_s$, where $\delta > 0$ is a server-specific parameter. Consequently, since $t_s$ is exponentially distributed with parameter $\mu$ (see Section II), $e_{\mathrm{j}}$ is also exponentially distributed with parameter $\mu/\delta$. Thus, the mean energy cost per job is given by

$$\bar{e_{\mathrm{j}}} = \delta/\mu. \tag{17}$$

Moreover, by processing each job, a server receives a reimbursement (benefit) equal to $e_{\mathrm{p}} = \gamma e_{\mathrm{j}}$, where $\gamma > 1$. Thus, $e_{\mathrm{p}}$ is exponentially distributed with parameter $\frac{\mu}{\gamma\delta}$, and the mean energy price per job is given by

$$\bar{e_{\mathrm{p}}} = \gamma\delta/\mu. \tag{18}$$

Let $k(t)$ be the mean number of jobs in any active server at $t$. By using the M/M/1 queue results [9], the mean number of jobs in each active server is

$$k(t) = \frac{\lambda/c(t)}{\mu - \lambda/c(t)}. \tag{19}$$

Thus, each active server processes $k(t)$ jobs on average, and thus earns a mean reward given by

$$R(t) = (\bar{e_{\mathrm{p}}} - \bar{e_{\mathrm{j}}})k(t) - e_{\mathrm{f}}. \tag{20}$$

For each server, being in active mode is attractive only if a minimum desired reward, denoted by $R_{\mathrm{th}} > 0$, is obtained. Therefore, using (17), (18), and (20), the mean number of jobs in any active server $k(t)$ has to be at least

$$k_{\min} = \frac{\mu(R_{\mathrm{th}} + e_{\mathrm{f}})}{\delta(\gamma - 1)}, \tag{21}$$

in order to achieve the minimum desired reward. Thus, by (19) and (21), at most

$$c_{\max} = \frac{\lambda}{\mu} + \frac{\lambda\delta(\gamma - 1)}{\mu^2(R_{\mathrm{th}} + e_{\mathrm{f}})} \tag{22}$$

servers can be in the active mode so that every active server receives a mean reward of $R_{th}$, while inactive servers receive no reward. Accordingly, the following condition must be satisfied when selecting the cut-off $c_{\mathrm{th}}$:

$$\text{Condition III:} \qquad c_{\mathrm{th}} \leq c_{\max}. \tag{23}$$

By (2), (16), and (23), the optimal number of active servers, $c_{\mathrm{th}}$, is determined as

$$c_{\min} \leq c_{\mathrm{th}} \leq c_{\max}, \tag{24}$$

where $c_{\min} = \mathrm{Max}\{c_{\min(1)}, c_{\min(2)}\}$. Hence, the system performs optimally in terms of servers' energy and users' delay when $c_{\mathrm{th}}$ servers are active. If we define the threshold $c_{\mathrm{th}}$ as

$$c_{\mathrm{th}} = c_{\min} = c_{\max}, \tag{25}$$

then in order to ensure that the entire system works efficiently, also the price of receiving computing services (i.e., $e_{\mathrm{p}}$) must be set by an authority (for instance, macro base station or network planner). This is determined by solving (25) for $\gamma$. Thus we obtain

$$\gamma = \frac{\mu^2}{\lambda\delta}(c_{\mathrm{th}} - \frac{\lambda}{\mu})(R_{\mathrm{th}} + e_{\mathrm{f}}) + 1. \tag{26}$$

In fact, in a distributed system, if a price larger than (26) is charged, more servers than $c_{\mathrm{th}}$ would become active, since every server achieves mean reward of $R_{\mathrm{th}}$ with lower number of tasks than $k_{\min}$ according to (21). In contrast, for $\gamma$ lower than (26), achieving $R_{\mathrm{th}}$ requires more tasks per server than $k_{\min}$, thus resulting in longer queue lengths which negatively affect the users' QoE.

Now the challenge is to activate $c_{\mathrm{th}}$ servers in a self-organized manner, which is addressed in the next section.

## III. Modeling the problem as a Minority Game

A Minority game (MG) can model the interaction among a large number of players competing for limited shared resources. In a basic MG, the players select between two alternatives and the players belonging to the *minority* group win. The minority is typically defined by using a cut-off value. The collective sum of the selected actions by all players is referred to as the *attendance*. The advantages of MG include simple implementation, low overhead, and scalability to large set of players, which are of vital importance in a dense wireless network. Details can be found in [7], [10].

We model the formulated server mode selection problem as an MG, where the $M$ servers represent the players, with a cut-off value $c_{\mathrm{th}}$ for the number of active servers. In each offloading period, the servers decide between the two actions, i.e., being *active* or *inactive*, denoted by 1 and 0, respectively. We denote the action of a given player $i$ in the time slot $t$ by $a_i(t)$. The number of active servers $c(t)$ maps to the attendance. Each player has $S$ strategies. According to our formulated mode selection problem for servers and the analysis in Section II,

- If $c(t) \leq c_{\mathrm{th}}$, each of the $c(t)$ active servers (the minority) earns a reward higher than or equal to the minimum desired reward, $R_{\mathrm{th}}$.
- If $c(t) > c_{\mathrm{th}}$, $c(t)$ active servers cannot achieve $R_{\mathrm{th}}$. In this case, inactivity (i.e., the action of the minority) is considered as the winning choice, since inactive servers spend no cost without being properly reimbursed.

### A. Control Information

After each round of play, a central unit (e.g., a macro base station) broadcasts the winning choice to all servers by sending a one-bit control information:

$$w(t) = \begin{cases} 1, & \text{if } c(t) \leq c_{\mathrm{th}} \\ 0, & \text{otherwise.} \end{cases} \tag{27}$$

As neither the attendance value $c(t)$ nor the cut-off $c_{\mathrm{th}}$ is known by the players, the overhead remains very low.

### B. Utility

Let $U_{i,\mathrm{a}}(t)$ and $U_{i,\mathrm{p}}(t)$ denote the utility that server $i$ receives for being active and being inactive, respectively. Based on the discussion above, we define

$$U_{i,\mathrm{a}}(t) = \begin{cases} 1, & \text{if } c(t) \leq c_{\mathrm{th}} \\ 0, & \text{otherwise} \end{cases} \tag{28}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LWC.2018.2810292, IEEE Wireless Communications Letters



and

$$U_{i,\mathrm{p}}(t) = \begin{cases} 1, & \text{if } c(t) > c_{\mathrm{th}} \\ 0, & \text{otherwise.} \end{cases} \tag{29}$$

### C. Distributed Learning Algorithm

Every player applies a basic strategy reinforcement technique to solve the formulated MG, summarized in **Algorithm 1** for some player $i$. Details can be found in [7].

**Algorithm 1** Distributed learning algorithm to solve server mode selection MG [7]

1: **Initialization**: Randomly draw $S$ strategies from the universal strategy pool, gathered in a set $\mathcal{S}$. Moreover, For every $s \in \mathcal{S}$, set the score $V_{i,s}(0) = 0$.
2: **for** $t = 1, 2, \ldots$ **do**
3: If $t = 1$, select the current strategy, $s_i(1)$, uniformly at random from the set $\mathcal{S}$. Otherwise, select the best strategy so far, defined as

$$s_i(t) = \underset{s \in \mathcal{S}}{\operatorname{argmax}}\, V_{i,s}(t). \tag{30}$$

4: Select the action $a_i(t)$, predicted by $s_i(t)$ as the winning choice.
5: The central unit broadcasts the control information (winning choice), $w(t)$.
6: Update the score of the strategy $s_i(t)$ as

$$V_{i,s}(t+1) = \begin{cases} V_{i,s}(t) + 1, & \text{if } a_i(t) = w(t) \\ V_{i,s}(t), & \text{otherwise} \end{cases} \tag{31}$$

7: **end for**

## IV. Numerical Results

For numerical analysis, we choose $M = 20$, $\lambda = 105$ tasks per second, $\mu = 15$ tasks per second, $R_{\mathrm{th}} = 100$, $e_{\mathrm{f}} = 50$, $\delta = 75/\text{seconds}$, $\beta = 0.05$, $T = 0.5$ seconds, $\nu = 1$, $a = -0.05$, and $b = 0.1$. Simulation is carried out for 32 runs and in each run, the servers randomly draw a set of strategies ($S = 2$) and repeatedly execute the MG for 1000 offloading periods. For the given parameters, the cut-off value is obtained as: $c_{\mathrm{th}} = 16$. The optimal (central activation) and random choice game (each server selects its action uniformly at random) are also simulated for comparison.

Fig. 1 shows the changes in users' probability measure, i.e., $\Pr[\theta \leq T]$. The users meet their QoE certainty requirement whenever $c(t) \geq c_{\mathrm{th}}$. As the attendance fluctuates near $c_{\mathrm{th}}$, the probability value also remains near the desired certainty.

The average utility per user is depicted in Fig. 2. It can be seen that the utility of MG-based strategy is higher than that of random selection. Yet, it is below the average utility of the optimal scenario. This is due to the fact that in MG-based method, servers make decisions under minimal external information and without any coordination with other servers.

## V. Conclusion

We have investigated the edge server activation problem in an MEC offloading system. We have analyzed the conditions for optimizing the servers' energy consumption while satisfying the users' QoE requirement, using queuing theory. Moreover, we have presented an MG-based distributed algorithm for server activation. The performance of the proposed method has been analyzed both theoretically and numerically.

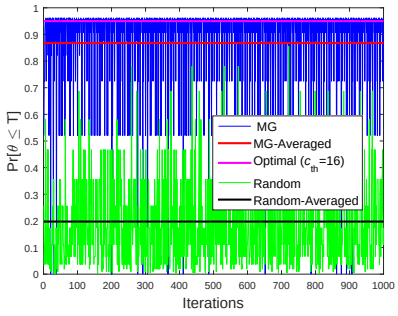


Fig. 1. Users' QoE measure.

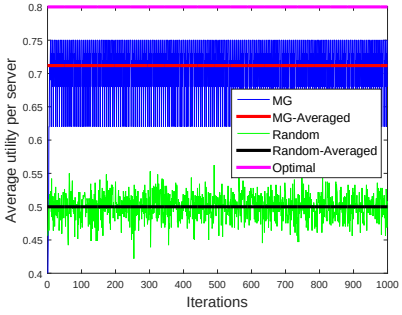


Fig. 2. Average utility of a server.

## References

[1] S. Josilo and G. Dan, "A game theoretic analysis of selfish mobile computation offloading," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, May 2017, pp. 1–9.
[2] W. Wang and W. Zhou, "Computational offloading with delay and capacity constraints in mobile edge," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.
[3] 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.
[4] L. Liu, Z. Chang, X. Guo, and T. Ristaniemi, "Multi-objective optimization for computation offloading in mobile-edge computing," in *2017 IEEE Symposium on Computers and Communications (ISCC)*, July 2017, pp. 832–837.
[5] S. M. Azimi, O. Simeone, O. Sahin, and P. Popovski, "Ultra-reliable cloud mobile computing with service composition and superposition coding," in *2016 Annual Conference on Information Science and Systems (CISS)*, March 2016, pp. 442–447.
[6] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1628–1656, thirdquarter 2017.
[7] D. Challet, M. Marsili, and Y. C. Zhang, *Minority Games: Interacting Agents in Financial Markets*. Oxford, UK: Oxford University Press, 2014.
[8] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," *IEEE/ACM Transactions on Networking*, vol. 21, no. 5, pp. 1378–1391, Oct 2013.
[9] J. Sztrik, J. Bíró, and Z. Heszberger, "Basic queueing theory," 2012.
[10] S. Ranadheera, S. Maghsudi, and E. Hossain, "Minority games with applications to distributed decision making and control in wireless networks," *IEEE Wireless Communications*, vol. 24, no. 5, pp. 184–192, October 2017.