Flat-Rate Pricing for Green Edge Computing With Latency Guarantee: A Stackelberg Game Approach

Zhan-Lun Chang*, Hung-Yu Wei†
Graduate Institute of Electrical Engineering, National Taiwan University
*r07921045@ntu.edu.tw, †hywei@ntu.edu.tw,

Abstract*—Mobile Edge Computing (MEC) is a promising paradigm to ease the computation burden of mobile devices by leveraging computing capabilities at the network edge. With the yearning needs for resource provision from the comparatively limited edge servers, an admission control charging a flat-rate price and resistant from egoistic manipulations is proposed to both utilize the scarce resources to the fullest and guarantee end-to-end latency for served mobile devices. Besides, to achieve energy sustainability and maximize profit, edge servers can lessen the energy consumption, and thus the energy expenditure by partial offloading certain requests to the cloud with a payment. Nonetheless, the cloud can reduce its energy consumption and enlarge the revenue through raising the service price to maximize its profit as well. The inherent hierarchy and interdependence between edge servers and the cloud are captured by Stackelberg game where the unique Stackelberg equilibrium is reached. The utility maximization problems for determining the optimal offloading ratio of edge servers and setting the optimal pricing of the cloud are investigated by exploiting the strict convexity and KKT conditions. Simulation results confirm the effectiveness of our scheme and several insights are illustrated.

I. Introduction

The conflict between the resource hungry applications and resource-constrained mobile devices brings in the unprecedented challenges to satisfactory quality of experience (QoE). To tackle this issue, a newly emerged fog/edge computing paradigm [1] is envisioned as a promising approach to provide cloud computing capabilities within the radio access networks in close vicinity of mobile devices. MEC has the potential to significantly reduce latency and prolong the battery lifetime of mobile devices by migrating their computation tasks to physically proximate edge servers.

Several studies have been conducted in this aspect. The authors in [2] minimized the energy consumption by jointly optimizing the offloading and radio resource allocation under MEC in 5G heterogeneous network. In [3], to improve the energy efficiency in MEC, a multi-user offloading game in a multi-channel interference environment was formulated and a distributed algorithm was designed to achieve Nash Equilibrium. The authors in [4] aimed to minimize the energy consumption of mobile device for a multi-user MEC system in a centralized framework. The authors in [5] considered multi-node partial green task offloading under MEC scenario.

However, when requests of mobile devices flood into the edge servers simultaneously, the response time may become unacceptably high, partly because of the fierce competition for relatively limited resources of edge servers and partly because of the contention of wireless access [6]. This necessitates the need for an admission control mechanism to not only allocate the resources of edge servers but also guarantee end-to-end latency. Moreover, if the information on which the admission rule relies is susceptible to untruthful reporting of self-interested mobile devices, this mechanism can not even work, let alone be efficacious. Designing an admission control mechanism free from this manipulations and with latency guarantee is our first concern.

Furthermore, with the proliferation of energy consumption of the information and communication technology [7], green computing is much-needed. Different from the aforementioned works which focused on mobile devices side, we put more emphasis on the edge servers and the cloud because, from the systematic view, the combined energy consumption of these two entities accounts for most of the overall energy consumption.

If the energy consumption of edge servers is too costly, whether in terms of the energy expenditure or of the carbon emission, edge servers are able to reduce energy consumption by turning off some virtual machines and outsourcing requests to the cloud or other edge servers. The authors [8] proposed a cooperative offloading policy between two fog data centers for load balancing. The authors in [9] investigated the workload allocation problem and the trade-off between power consumption and transmission delay in the fog-cloud computing system. However, the operator of the cloud may not provide abundant computing resources to service providers of different edge servers without monetary compensation for the incurred computation overhead.

With high-volume and ever-growing service requests, the energy consumption of the cloud on powering up and cooling cloud servers is soaring as well. In our work, the cloud can lift the service price to curb request demands to attenuate energy consumption. Nevertheless, setting a prohibitive price may be detrimental to the overall profit of the cloud in that there might be not offloaded requests at all. The offloading ratios of edge servers and the service price of the cloud depend on each other. How to unwrap the mutual dependency to solve the profit maximization problems, where each entity aims at cutting down the energy consumption to be green, is our second concern. To the best of our knowledge, we are the first in the literature to guarantee end-to-end latency in green edge computing. Specific technical contributions of this paper can be summarized as follows:

978-1-7281-0962-6/19/$31.00 ©2019 IEEE

1) We proposed an admission control mechanism with a flat price for edge servers. It is immune to the manipulations of mobile devices and guarantees end-to-end latency.
2) The coupled relationship between the offloading ratios of edge servers and service price of the cloud is unraveled by the Stackelberg game approach where the uniqueness of Stackelberg equilibrium is proved.
3) The utility maximization problem incorporating energy reduction for edge servers and the cloud are both formulated as strict convex optimization problems and proved to have a unique solution.

II. System Architecture

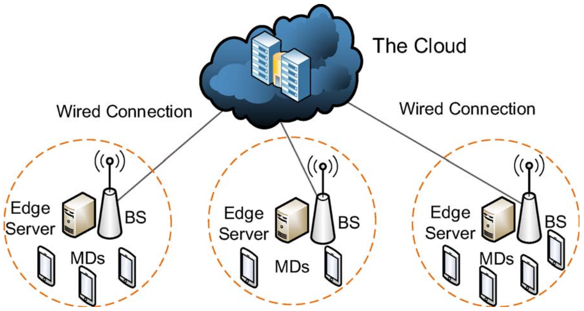


Fig. 1. System Architecture

As shown in Fig 1, the system is composed of three entities: the cloud which comprises a collection of servers, base stations (BSs) associated with an edge server, mobile devices (MDs). The BSs and the edge servers are responsible for communication and computation respectively. Hereafter, for simplicity, we regard the BS and its associated edge server as one entity, which is called the edge server instead. The set of edge servers is denoted by $\mathcal{F} = \{1, \cdots, |\mathcal{F}|\}$ and indexed by i. The set of MDs within the wireless coverage of edge server $i \in \mathcal{F}$ is represented as $\mathcal{M}_i = \{1, \cdots, |\mathcal{M}_i|\}$ and indexed by j. It is assumed that each MD runs a delay-sensitive and computationally intensive task. Executing such applications can be deleterious to the restricted batter lifetime of MDs. Therefore, MDs in $\mathcal{M}_i$ have strong inclination to offload all the tasks to edge server $\mathcal{F}_i$ via a wireless channel. Edge server $\mathcal{F}_i$ can judiciously determine the ratio $\rho_i^{\mathcal{F}}$ of all received service requests from MDs to be offloaded to the cloud with the aim of reducing its energy consumption.

A. Transmission and Execution Time Model

For the sake of quantifying average response time, computation tasks at MD j in $\mathcal{M}_i$ follow a Poisson process with mean arrival rate λ_i, which is a prevailing assumption in edge system [10]. The heterogeneity of tasks between different MDs can be reflected by the various delay requirements and thus is not compromised despite our assumption that all MDs in $\mathcal{M}_i$ have the same arrival rate λ_i. The data transmission rate r_{ij} between MD j and $\mathcal{F}_i$ by Shannon's formula is

$$r_{ij} = BWlog_2(1 + \frac{p_j g_{ij}}{\sigma^2 + \sum_{k \in M_i, k \neq j} p_k g_{ik}}) \tag{1}$$

where BW is channel bandwidth, p_j is the transmission power of MD j, and g_{ij} denotes the channel gain between $\mathcal{F}_i$ and MD j, $\sum_{k \in \mathcal{M}_i, k \neq j} p_{ik} g_{ik}$ is the interference induced by the uplink transmission of MDs other than j on the same channel, σ^2 is the additional white Gaussian noise power.

From (1), we are able to calculate the transmission time of data with size z_j generated at MD j to $\mathcal{F}_i$ as

$$T_{ij}^t = \frac{\lambda_i \cdot z_j}{r_{ij}} \tag{2}$$

Due to the limited processing abilities of edge servers, we devise an admission control mechanism to determine which and how many MDs will be served from the perspective of $\mathcal{F}_i$. The details of the mechanism will be specified in section III. Here, the set of MDs served by $\mathcal{F}_i$ as an output of the admission control mechanism is denoted by A_i with cardinality $|A_i|$. The overall arriving rate of $\mathcal{F}_i$ can be expressed as $\lambda_i^{\mathcal{F}} = \lambda_i \cdot |A_i|$ followed by the properties of the Poisson process. We make use of $M/M/1$ queue [11] [12] to model the behaviors of edge servers. The service rate of $\mathcal{F}_i$ and the sending rate of $\mathcal{F}_i$ are noted as $\mu_i^{\mathcal{F}}$, μ_i^S respectively. If $\mathcal{F}_i$ decides to offload received requests with ratio $\rho_i^{\mathcal{F}}$, the average processing time is:

$$T_i^P(\rho_i^{\mathcal{F}}) = \frac{1}{\mu_i^{\mathcal{F}} - (1 - \rho_i^{\mathcal{F}}) \cdot \lambda_i^{\mathcal{F}}} \tag{3}$$

The overall requests delivered from all edge servers to the cloud follow the Poisson process with a total mean arrival rate

$$\lambda_c(\boldsymbol{\rho}^{\mathcal{F}}) = \sum_{i \in \mathcal{F}} \rho_i^{\mathcal{F}} \cdot \lambda_i^{\mathcal{F}} \tag{4}$$

where $\boldsymbol{\rho}^{\mathcal{F}} = (\rho_1^{\mathcal{F}}, \rho_2^{\mathcal{F}}, \cdots, \rho_{|\mathcal{F}|}^{\mathcal{F}})$ is the profile of offloading ratios of all edge servers. In contrast to the constrained computing capabilities of edge servers, the cloud posses a myriad of resources. Accordingly, we model the cloud as a collection of K homogeneous servers with service rate μ_c^p, which is usually much larger than that of edge servers. The queueing delay at the cloud can be captured by parallel $M/G/1$ Processor Sharing queue [13], [14] with $\lambda_c(\boldsymbol{\rho}^{\boldsymbol{B}})$ partitioned equally

$$T_c^P(\boldsymbol{\rho}^{\mathcal{F}}) = \frac{1}{\mu_c^p - \frac{\lambda_c(\boldsymbol{\rho}^{\mathcal{F}})}{K}} \tag{5}$$

We ignore the delay to send back the computed outcomes to the edge servers due to the fact that for numerous applications, the size of the computed outcomes is generally much smaller than the input data [3], [11].

B. Energy Consumption Model

The energy consumption of edge server $\mathcal{F}_i$ consists of two components: processing part and sending part. The processing part is consumed when the edge server is active for processing. Let α_i symbolize the energy coefficient for the processing power of $\mathcal{F}_i$. The processing energy consumption is

$$E_i^P(\rho_i^{\mathcal{F}}) = \alpha_i \cdot T_i^P(\rho_i^{\mathcal{F}}) \tag{6}$$

On the other hand, the sending part is proportional to the duration of sending.

$$E_i^S(\rho_i^{\mathcal{F}}) = \beta_i \frac{\rho_i^{\mathcal{F}} \cdot \lambda_i^{\mathcal{F}}}{\mu_i^S} \tag{7}$$

where β_i is the energy coefficient for sending. Combining these two results yields the total energy consumption of $\mathcal{F}_i$:

$$E_i^{all}(\rho_i^{\mathcal{F}}) = E_i^P(\rho_i^{\mathcal{F}}) + E_i^S(\rho_i^{\mathcal{F}}) \tag{8}$$

Let ξ_c stand for the energy coefficient for the processing power of servers in the cloud. The overall processing energy consumption of the cloud is the sum of processing energy consumption of K servers constituting it:

$$E_c^P(\boldsymbol{\rho}^{\mathcal{F}}) = K \cdot \xi_c \cdot T_c^P(\boldsymbol{\rho}^{\mathcal{F}}) = \frac{K\xi_c}{\mu_c^p - \frac{\lambda_c(\boldsymbol{\rho}^{\mathcal{F}})}{K}} \tag{9}$$

The energy consumption here merely contains the processing part. The energy needed for sending the results back to edge servers is negligible since the time required for sending can be neglected as stated in subsection II-A.

III. Game Problem formulation

A. Interactions Between Mobile Devices And Edge Servers

As each $\mathcal{F}_i$ has limited computational resources, the MDs inevitably have to compete with one another to fulfill their stringent delay requirements. As processing requests consumes energy, each MD will be charged a fixed price p to reimburse the increased energy expenditure of $\mathcal{F}_i$. To be stable, the maximum workload that edge server $\mathcal{F}_i$ can afford is capped at a fixed value λ_i^{max} [11] which satisfies the relation $\lambda_i^{max} < \mu_i^{\mathcal{F}}$. The edge server $\mathcal{F}_i$ first announces a fixed price p. Next, MDs in $\mathcal{M}_i$ report its end-to-end delay constraints d_j. It's critical to ensure truthfulness so that the outcome of the admission control mechanism is immune to selfish behaviors of MDs trying to alleviate their benefits. Whether or not truthfulness in reporting will be attained heavily depends on the rule of admission control as well as the design of utility function of MDs. The rule will be clarified in section IV.

The utility of MD j is

$$U_j(d) = \begin{cases} v_j - p & \text{if MD } j \text{ is served and } d \le d_j \\ -p & \text{if MD } j \text{ is served and } d > d_j \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where d is the actual end-to-end delay that MD j perceives, d_j is the end-to-end delay constraint of MD j, and v_j is the valuation of service of MD j if $d \le d_j$. On the contrary, if $d > d_j$ or MD j is not served, the valuation equals zero. Only when the valuation of MD j is higher than p will the MD j report its delay constraint. By focusing on those MDs that reported, there exists a natural constraint on $v_j : v_j - p \ge 0$ which is an indispensable condition for guaranteeing truthfulness in our proof. We defer the case that price p is adjustable by $\mathcal{F}_i$ to future work.

B. Interactions Between Edge Servers And The Cloud

The winner set A_i of MDs is established after the execution of our admission control mechanism, which further determines $\lambda_i^{\mathcal{F}}$. The choice of offloading ratio $\rho_i^{\mathcal{F}}$ is motivated by the service price w set by the cloud. If the benefit generated from the reduction of energy expenditure is more than the extra payment to the cloud, the edge server would rather offload some portion of $\lambda_i^{\mathcal{F}}$. If the price w is comparatively higher, $\mathcal{F}_i$ would reserve more service requests for local processing to curtail the payment to the cloud. Therefore, it's essential to strike the balance to maximize the utility of $\mathcal{F}_i$.

We make use of Stackelberg game to characterize the intrinsic hierarchy structure. Stackelberg game, also known as a leader-follower game, is an effective mathematical tool to study sequential decision-making process. The leader moves first and the followers move subsequently in response to the decision made by the leader. In our context, the cloud is the leader declaring w first. Based on w, the followers, namely edge servers, select their offloading ratios to maximize their utilities. Formally, the Stackelberg game Γ can be constructed as

$$\Gamma = \{(C, \mathcal{F}_i), (w, \rho_i^{\mathcal{F}}), \big(U_C(w), U_i(\rho_i^{\mathcal{F}})\big)\} \tag{11}$$

Player	The cloud C (leader)	Edge servers $\mathcal{F}_i$ (followers)
Action	service price w	offloading ratio $\rho_i^{\mathcal{F}}$
Utility	$U_C(w)$	$U_i(\rho_i^{\mathcal{F}})$

C. Utility Functions of Edge Servers And The Cloud

Specifically, the utility function of $\mathcal{F}_i$ is defined as the revenue accrued from MDs minus the sum of energy expenditure and the payment to the cloud:

$$U_i(\rho_i^{\mathcal{F}}|w) = p \cdot |A_i| - \tau \cdot E_i^{all}(\rho_i^{\mathcal{F}}|w) - w \cdot \rho_i^{\mathcal{F}} \cdot \lambda_i^{\mathcal{F}} \tag{12}$$

where $U_i(\rho_i^{\mathcal{F}}|w)$ is the utility of $\mathcal{F}_i$ given w, $p|A_i|$ is the revenue earned by providing service to MDs in A_i, τ is the unit energy price and finally $w \cdot \rho_i^{\mathcal{F}} \cdot \lambda_i^{\mathcal{F}}$ is the payment to the cloud.

As for the cloud, depending on the offloading profile $\boldsymbol{\rho}^{\mathcal{F}}$, it chooses the pricing strategy prudently to maximize its utility function given as follows:

$$U_C(w|\boldsymbol{\rho}^{\mathcal{F}}) = w \cdot \lambda_c(\boldsymbol{\rho}^{\mathcal{F}}) - \psi_c E_c^P(\boldsymbol{\rho}^{\mathcal{F}}) \tag{13}$$

The first term is the profit by helping edge servers process their offloaded requests. ψ_c is the unit energy price of the cloud that may be different from τ. It's noteworthy that $\boldsymbol{\rho}^{\mathcal{F}}$ is a function of w.

The Stackelberg equilibrium from which nor the edge servers and the cloud has incentives to deviate in our one-leader-multi-followers Stackelberg game can be defined as

Definition 1 (Stackelberg equilibrium). *Let $(\rho_i^{\mathcal{F}}(w))^*$ be the set of the optimal offloading ratios of edge server $\mathcal{F}_i$ given w. That is, for any $(\rho_i^{\mathcal{F}}(w))'$ that is not $\rho_i^{\mathcal{F}}(w)$, this set is*

$$\{\rho_i^{\mathcal{F}}(w) \in [0,1] : U_i\big(\rho_i^{\mathcal{F}}(w)|w\big) \ge U_i\big((\rho_i^{\mathcal{F}}(w))'|w\big)\}$$

A set of points $\big(w^*, (\rho_i^{\mathcal{F}}(w^*))^*\big)$ are Stackelberg equilibriums if

$$U_C(w^*|(\rho_i^{\mathcal{F}}(w^*))^*) \ge U_C(w|(\rho_i^{\mathcal{F}}(w))^*) \tag{14}$$

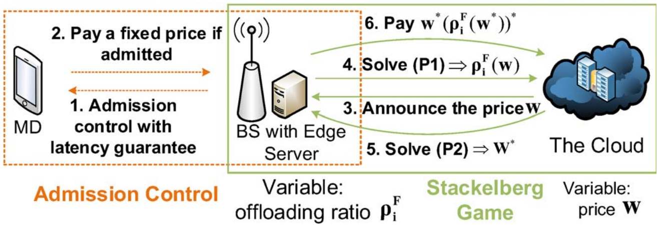

2. Pay a fixed price if admitted
1. Admission control with latency guarantee
MD
BS with Edge Server
6. Pay $\mathbf{w}^*(\rho_i^F(\mathbf{w}^*))^*$
4. Solve (P1) $\Rightarrow \rho_i^F(\mathbf{w})$
3. Announce the price $\mathbf{w}$
5. Solve (P2) $\Rightarrow \mathbf{W}^*$
The Cloud
Admission Control
Variable: offloading ratio ρ_i^F
Stackelberg Game
Variable: price $\mathbf{w}$

Fig. 2. Overall Optimization Procedure

IV. Game Theoretic Optimization

The overall optimization procedure is explicated as Fig. 2. The admission control is performed by edge servers before entering the Stackelberg game. We would like to stress that the offloading behaviors are different between MDs and edge servers. If MD j is served, it offloads all requests. However, edge servers consider more generally the partial offloading problem and determine the optimal offloading ratios.

A. Admission Control Mechanism

We first analyze end-to-end latency in the worst case. Denote the arrival rate of $\mathcal{F}_i$ after MD j is served by λ. From the perspective of $\mathcal{F}_i$, the longest end-to-end latency occurs when other edge servers offload the maximum possible arrival rates to the cloud, in other words, under the offloading profile $(\lambda_1^{max}, \cdots, \lambda_{i-1}^{max}, \lambda_{i+1}^{max}, \cdots, \lambda_{|\mathcal{F}|}^{max})$. The overall arrival rate at the cloud in this situation is

$$\lambda_c^{max}(\lambda) = \lambda + \sum_{r \in \mathcal{F}, r \neq i} \lambda_r^{max} \tag{15}$$

The longest end-to-end latency is given by

$$D_i(\lambda) = T_{ij}^t + \frac{\lambda}{\mu_i^S} + (\mu_c^p - \frac{\lambda_c^{max}(\lambda)}{K})^{-1} \tag{16}$$

We now explain algorithm 1 (referred to as LGA algorithm) in details. In the first place, all MDs in $\mathcal{M}_i$ report their end-to-end delay constraints. After that, $\mathcal{F}_i$ sorts these values in descending order. The motivation behind the arrangement is that with the same arrival rate, those applications requiring stringent end-to-end delay consume more resources. Assigning higher priority to MDs with less critical constraints maximizes the revenue of $\mathcal{F}_i$ because in this manner $\mathcal{F}_i$ can serve more MDs. If $\mathcal{F}_i$ can still afford and D_i after serving MD j is still not larger than d_j, the MD j will be included into current A_i. Otherwise, the algorithm is terminated because if a more relaxed delay constraint cannot be satisfied, much less the more stringent one. The salient feature of LGA algorithm lies in two aspects. For one, the running time complexity is polynomial, in fact, $O(|\mathcal{M}_i|)$ for $\mathcal{F}_i$. For another, the LGA algorithm is distributed. Consequently, the information of the cloud other than the service rate and the number of requests in other edges servers is not needed to implement the LGA algorithm. Under the design of utility (10), truthfulness can be guaranteed by the following theorem.

Theorem 1. *Reporting the true end-to-end delay constraint is the weakly dominant strategy for MD j. In other words, we need to show that $U_j(d_j) \geq U_j(d_j')$ where d_j represents the true end-to-end delay constraint and $d_j' \neq d_j$.*

Algorithm 1 Latency-guarantee Allocation Algorithm of $\mathcal{F}_i$

INPUT: λ_i, λ_i^{max}, μ_c^p, r_{ij}
OUTPUT: the winning set A_i

```
$\mathcal{A}_i = \emptyset$
for all $j \in \mathcal{M}_i$ do
    report end-to-end delay constraint $d_j$ to $\mathcal{F}_i$
end for
Store all $d_j$ in descending order in array $\mathbf{D}$
$\lambda_{\text{win}} = \lambda_i$
$t = 1$
while $t \leq \lfloor \lambda_i^{max} / \lambda_i \rfloor$ AND $D_i(\lambda_{\text{win}}) \leq \mathbf{D}[t]$ do
    $t \leftarrow t + 1$
    $\lambda_{\text{win}} \leftarrow \lambda_{\text{win}} + \lambda_i$
    $A_i \leftarrow A_i \cup \{$MD $j$ that submit $\mathbf{D}[t]\}$
end while
```

Proof. We prove by discussing all possible cases.

(i) MD j is served by $\mathcal{F}_i$ by truthfully bidding its end-to-end delay constraint d_j: Suppose now if MD j bid d_j' untruthfully,

a) $d_j' \geq d_j$, MD will be served by $\mathcal{F}_i$ but with utility $U_j(d_j')$ at most $v_j - p$, not larger than $U_j(d_j) = v_j - p$

b) $d_j' < d_j$, If MD is still served by $\mathcal{F}_i$, we have $U_j(d_i') = U_j(d_i) = v_j - p$. If MD is not served by $\mathcal{F}_i$, then $U_j(d_j') = 0 \leq U_j(d_j) = v_j - p$.

(ii) MD j is not served by $\mathcal{F}_i$ by truthfully bidding its end-to-end delay constraint d_j: Suppose now if MD j bid d_j' untruthfully,

a) $d_j' \geq d_j$, If MD j is served, $U_j(d_j') = -p$, which is smaller than $U_j(d_j) = 0$. If MD j is not served, then $U_j(d_j') = U_j(d_j) = 0$

b) $d_j' < d_j$, MD j will not be served by $\mathcal{F}_i$. Thus, $U_j(d_j') = U_j(d_j) = 0$.

B. Optimization of edge servers

Given the service price w and the winning set A_i generated as the output of LGA algorithm, each edge server chooses the offloading ratio $\rho_i^{\mathcal{F}}$ cautiously to maximize (12). The optimization problem **(P1)** for $\mathcal{F}_i$ is formulated as

$$\begin{aligned} \underset{\rho_i^{\mathcal{F}}}{\text{maximize}} \quad & p \cdot |A_i| - \tau \cdot E_i^{all}(\rho_i^{\mathcal{F}}|w) - w \cdot \rho_i^{\mathcal{F}} \cdot \lambda_i^{\mathcal{F}} \\ \text{subject to} \quad & 0 \leq \rho_i^{\mathcal{F}} \leq 1 \end{aligned} \tag{17}$$

For simplicity, we replace $U_i(\rho_i^{\mathcal{F}}|w)$ with $U_i(\rho_i^{\mathcal{F}})$ henceforth.

Theorem 2. *The optimal solution $(\rho_i^{\mathcal{F}})^*$ to **(P1)** exists and is unique if $U_i(\rho_i^{\mathcal{F}})$ is a strictly concave function of $\rho_i^{\mathcal{F}}$ for all i. When the strict inequality of (17) holds, $(\rho_i^{\mathcal{F}})^*$ is given by $1 - \frac{\mu_i^{\mathcal{F}}}{\lambda_i^{\mathcal{F}}} + \frac{1}{\lambda_i^{\mathcal{F}}}\sqrt{\tau\alpha_i(w + \frac{\tau\beta_i}{\mu_i^S})^{-1}}$.*

Proof. The feasible region of **(P1)** is evidently a convex set. If $U_i(\rho_i^{\mathcal{F}})$ is a strictly concave function of $\rho_i^{\mathcal{F}}$ for all i, **(P1)** is a strictly convex optimization, which has at most one optimal point. In addition to strictly convexity, $U_i(\rho_i^{\mathcal{F}})$ is continuous in $\rho_i^{\mathcal{F}}$. Also, the feasible region of **(P1)** is a close and bounded interval in $\mathbb{R}$. The existence of the optimal solution then comes from the Extreme Value Theorem. What remains to check is the strict concavity of $U_i(\rho_i^{\mathcal{F}})$. As $U_i(\rho_i^{\mathcal{F}})$ is a twice-differentiable function with respect to $\rho_i^{\mathcal{F}}$, the strict concavity can be characterized by the second-order condition:

$$\frac{\partial^2 U_i(\rho_i^{\mathcal{F}})}{\partial(\rho_i^{\mathcal{F}})^2} = (-2)\tau\alpha_i \frac{(\lambda_i^{\mathcal{F}})^2}{(\mu_i^{\mathcal{F}} - (1-\rho_i^{\mathcal{F}})\lambda_i^{\mathcal{F}})^3} < 0 \qquad (18)$$

We are able to prove that when $0 < (\rho_i^{\mathcal{F}})^* < 1$, $(\rho_i^{\mathcal{F}})^*$ has the desired form stated in the theorem by solving KKT conditions,

$$\begin{cases} \rho_i^{\mathcal{F}} - 1 \leq 0 \\ -\rho_i^{\mathcal{F}} \leq 0 \\ h_1(\rho_i^{\mathcal{F}} - 1) = h_2(-\rho_i^{\mathcal{F}}) = 0 \\ h_1, h_2 \geq 0 \\ \tau\alpha_i \frac{\lambda_i^{\mathcal{F}}}{\left[(\mu_i^{\mathcal{F}} - (1-\rho_i^{\mathcal{F}})\lambda_i^{\mathcal{F}})\right]^2} - w\lambda_i^{\mathcal{F}} - \frac{\lambda_i\tau\beta_i}{\mu_i^S} + h_1 - h_2 = 0 \end{cases}$$

where h_1, h_2 are Lagrange multipliers associated with inequality constraints. Multiplying the last equation by $\rho_i^{\mathcal{F}}$ and using the third equation give

$$(\tau\alpha_i \frac{\lambda_i^{\mathcal{F}}}{\left[(\mu_i^{\mathcal{F}} - (1-\rho_i^{\mathcal{F}})\lambda_i^{\mathcal{F}}\right]^2} - w\lambda_i^{\mathcal{F}} - \frac{\lambda_i\tau\beta_i}{\mu_i^S})(\rho_i^{\mathcal{F}} - 1) = 0$$

When $\rho_i^{\mathcal{F}} < 1$, the first term must equal zero. Solving this equation gives the desired result. For the other direction ($\rho_i^{\mathcal{F}} > 0$), just begin the process by multiplying $(\rho_i^{\mathcal{F}} - 1)$. □

C. Optimization of The Cloud

Rewriting $\lambda_c(\boldsymbol{\rho}^{\mathcal{F}})$ as a function solely of w by the result of Theorem 2 yields

$$\lambda_c(w) = \sum_{i\in\mathcal{F}} \left(\lambda_i^{\mathcal{F}} - \mu_i^{\mathcal{F}} + \sqrt{\tau\alpha_i(w + \frac{\tau\beta_i}{\mu_i^S})^{-1}}\right) \qquad (19)$$

With this representation, the optimization problem **(P2)** of the cloud is

$$\begin{aligned} \underset{w}{\text{maximize}} \quad & w \cdot \lambda_c(w) - \frac{K\psi_c\xi_c}{\mu_c^p - \frac{\lambda_c(w)}{K}} \\ \text{subject to} \quad & 0 \leq \frac{\lambda_c(w)}{K} < \mu_c^p \end{aligned} \qquad (20)$$

Combining second order condition and changing the variable proves the following theorem. Details are omitted due to space limitations.

Theorem 3. *$U_C(w)$ is a strictly concave function of w.*

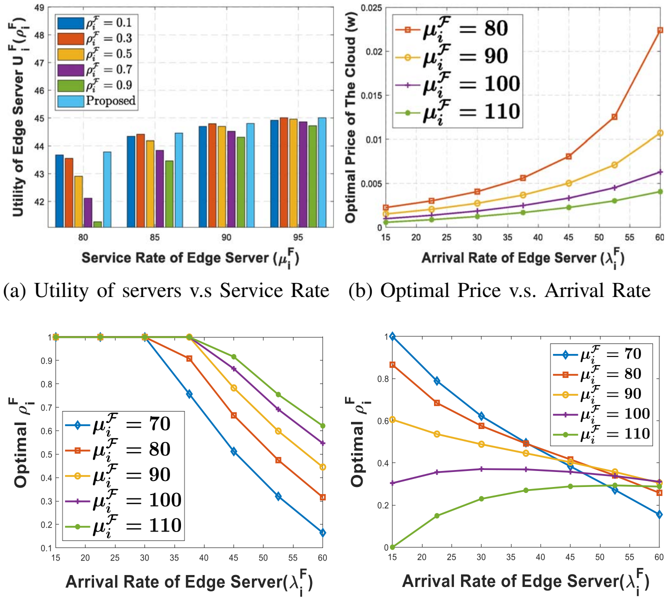
Applying KKT conditions to **(P2)**, we are able to derive the following theorem. The proof is similar to the techniques in Theorem 2 and is omitted due to the limited space.

Theorem 4. *When the strict inequality of (20) holds, the optimal solution w^* can be given by solving*

$$\sum_{i\in\mathcal{F}}(\lambda_i^{\mathcal{F}} - \mu_i^{\mathcal{F}} + \frac{1}{2}\sqrt{\frac{\tau\alpha_i}{w + \frac{\tau\beta_i}{\mu_i^S}}}) + \frac{\psi_c\xi_c}{2}\frac{\sum_{i\in\mathcal{F}}\sqrt{\frac{\tau\alpha_i}{(w+\frac{\tau\beta_i}{\mu_i^S})^3}}}{\left(\mu_c^p - \frac{\lambda_c(w)}{K}\right)^2} = 0$$

TABLE I

Parameter	$\lvert\mathcal{F}\rvert$	p_j	σ	$\mu_i^{\mathcal{F}}$	α_i	p	ψ_c
Value	10	23 dBm	-100 dBm	80	8	2	2
Parameter	K	BW	z_j	μ_c^p	β_i	τ	λ_i
Value	20	10 MHz	0.5 MB	10^3	.06	4	1.5



(a) Utility of servers v.s Service Rate

(b) Optimal Price v.s. Arrival Rate

(c) Optimal Ratio v.s. Arrival Rate

(d) Optimal Ratio v.s. Arrival Rate

Fig. 3. Simulation results : (c) with β_i=0.01, (d) with β_i=0.06

From Definition 1, Theorem 2, 3 and 4, we conclude that Γ admits a unique Stackelberg equilibrium. For clarity, the optimization flow of Stackelberg game is delineated in Fig 2.

V. Simulation Results

A. Simulation Setup

For the network setup, we consider a circle area with radius 1 km where the base stations and mobile devices are uniformly distributed. For radio access, the channel gain between MD j to $\mathcal{F}_i$ is d^{-4} where d is the distance in meters between them. The sending rate μ_i^S is fixed as 80. Other key parameters are summarized in Table I if not specified particularly.

B. Comparison to fixed offloading ratio scheme

We compare the performance of the proposed scheme to fixed offloading ratio with $\rho_i^{\mathcal{F}} = [0.1, 0.3, 0.5, 0.7, 0.9]$ without loss of generality. The service rate $\mu_i^{\mathcal{F}}$ varies in $[80, 85, 90, 95]$. To obliterate the influence of heterogeneity, $\mu_i^{\mathcal{F}}$ and μ_i^S are the same across all edge servers $\mathcal{F}$. The distribution of end-to-end delay constraint and the value λ_i^{max} are designed to generate any random number between 40 and 50 mobile devices on each edge server $\mathcal{F}_i$.

As seen in Fig 3(a), when $\mu_i^{\mathcal{F}}$ increases, the energy consumption decreases because the processing time decreases. Thus, the utilities of all scheme increase as $\mu_i^{\mathcal{F}}$ increases. No matter what $\mu_i^{\mathcal{F}}$ is, our proposed scheme outperforms all other fixed ratio schemes in terms of the utilities of edge servers.

C. Impact of arrival rate on the optimal price of the cloud

From Fig 3(b), as the arrival rate $\lambda_i^{\mathcal{F}}$ increases, the amount of offloaded requests from edge servers to the cloud increases. The cloud would lift the service price to suppress offloaded requests. By doing so, the cloud can reduce its processing energy consumption. When edge servers have the same $\lambda_i^{\mathcal{F}}$, the service prices w for the edge servers with higher service rates $\mu_i^{\mathcal{F}}$ are lower. Raising the price too high will render more powerful edge servers less inclined to offload the requests. The decrease in revenue from edge servers would overshadow the saved energy expenditure for the cloud. Hence, the prices that these edge servers are charged are smaller than edge servers with inferior service rate.

D. Impact of arrival rate on optimal offloading ratio

In this subsection, we observe the optimal offloading ratios $\rho_i^{\mathcal{F}}$ with different β_i. In Fig 3(c), when the arrival rate $\lambda_i^{\mathcal{F}}$ is small, offloading all requests to the cloud is advantageous for all edge servers. As the service rate μ_i^F surpasses some threshold (e.g. when $\lambda_i^{\mathcal{F}}$= 30 or 38 approximately), edge servers start processing requests on their own. The threshold is higher when the edge server has a higher $\mu_i^{\mathcal{F}}$.

It's noteworthy to mention that this phenomenon is kind of counter-intuitive. The instinct tells that the more powerful the edge servers are, the earlier for them to start processing some part of the requests. However, this is mainly from the perspective of reducing the energy expenditure without taking the payment to the cloud into account. As seen from Fig 3(b), the higher the μ_i^F, the lower the prices w. For more powerful edge servers, offloading all of the requests when their arrival rates go beyond the thresholds of less powerful edge servers is still instrumental in gaining more utility.

There is a strikingly different trend between Fig 3(c) and Fig 3(d). In Fig 3(c), as the arrival rate $\lambda_i^{\mathcal{F}}$ increases, the offloading ratios of all edge servers decrease. On the other hand, in Fig 3(d), the behaviors of offloading ratios are disparate between edge servers. We can derive the approximated value of the service rate above which offloading more has benefits when the arrival rate increases. By Theorem 2, the value is denoted by $\hat{\mu}_i^B(\lambda_i^{\mathcal{F}})$ and is given by

$$\hat{\mu}_i^B(\lambda_i^{\mathcal{F}}) = \sqrt{\tau\alpha_i(w + \frac{\tau\beta_i}{\mu_i^S})^{-1} + \frac{\lambda_i^{\mathcal{F}}\sqrt{\tau\alpha_i}}{2}(w + \frac{\tau\beta_i}{\mu_i^S})^{-1.5}\frac{\partial w}{\partial\lambda_i^{\mathcal{F}}}}$$

From the numerical analysis, for edge servers with service rates 100 and 110 in Fig 3(d), their service rates are higher than $\hat{\mu}_i^B(\lambda_i^{\mathcal{F}})$ for any $\lambda_i^{\mathcal{F}}$. All other edge servers in Fig 3(d) and all edge servers in Fig 3(c) have service rates lower than $\hat{\mu}_i^B(\lambda_i^{\mathcal{F}})$ for any $\lambda_i^{\mathcal{F}}$. Ostensibly, when the $\beta_i = 0.01$ in Fig 3(c) is changed to 0.06 in Fig 3(d), the offloading ratios of edge servers with service rate 100 or 110 should drop more intensely owing to the increased cost of sending energy consumption. In fact, in the light of decreased w induced by higher β_i (validated by simulation), the reduction in payment offsets the increased sending energy expenditure. As a result, these two edge servers can achieve higher utilities by offloading more requests when the arrival rate increases.

VI. Conclusions

In this paper, we first propose an admission control mechanism with end-to-end latency guarantee that is resistant to selfish manipulations of mobile devices. Afterward, both the utility maximization problems for selecting the optimal offloading ratios of edge servers and setting the optimal service price of the cloud are proved to be strict convex and solved by KKT conditions. The hierarchy nature between the edge servers and the cloud is indicated and captured by the Stackelberg game which is shown to admit a unique Stackelberg equilibrium. The effectiveness over any fixed offloading strategy is demonstrated and the counter-intuitive phenomena are elucidated from the holistic point of view.

VII. Acknowledgment

The authors are grateful for the funding support from MOST of Taiwan under grant 108-2218-E-002-060- and 107-2923-E-002-006-MY3.

References

[1] Y.-J. Ku, D.-Y. Lin, C.-F. Lee, P.-J. Hsieh, H.-Y. Wei, C.-T. Chou, and A.-C. Pang, "5g radio access network design with the fog paradigm: Confluence of communications and computing," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 46–52, 2017.

[2] 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.

[3] 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, 2016.

[4] 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, 2017.

[5] Y.-L. Hsu, H.-Y. Wei, and M. Bennis, "Green fog offloading strategy for heterogeneous wireless edge networks," in *2018 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, IEEE, 2018.

[6] M. V. Barbera, S. Kosta, A. Mei, and J. Stefa, "To offload or not to offload? the bandwidth and energy costs of mobile cloud computing," in *2013 Proceedings Ieee Infocom*, pp. 1285–1293, IEEE, 2013.

[7] S. Lambert, W. Van Heddeghem, W. Vereecken, B. Lannoo, D. Colle, and M. Pickavet, "Worldwide electricity consumption of communication networks," *Optics express*, vol. 20, no. 26, pp. B513–B524, 2012.

[8] R. Beraldi, A. Mtibaa, and H. Alnuweiri, "Cooperative load balancing scheme for edge computing resources," in *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC)*, pp. 94–100, IEEE, 2017.

[9] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet of Things Journal*, vol. 3, pp. 1171–1181, Dec 2016.

[10] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[11] L. Liu, Z. Chang, X. Guo, S. Mao, and T. Ristaniemi, "Multiobjective optimization for computation offloading in fog computing," *IEEE Internet of Things Journal*, vol. 5, pp. 283–294, Feb 2018.

[12] Y. Wang, X. Lin, and M. Pedram, "A nested two stage game-based optimization framework in mobile cloud computing system," in *2013 IEEE Seventh International Symposium on Service-Oriented System Engineering*, pp. 494–502, March 2013.

[13] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy, "Optimal power allocation in server farms," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 37, pp. 157–168, ACM, 2009.

[14] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Greening geographical load balancing," in *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, pp. 233–244, ACM, 2011.