QoS-Aware Matching of Edge Computing Services to Internet of Things

Nafiseh Sharghivand[1][2], Farnaz Derakhshan[1], Lena Mashayekhy[2]
[1]Department of Computer Engineering, University of Tabriz, Tabriz, Iran
{n.sharghivand, derakhshan}@tabrizu.ac.ir
[2]Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716
mlena@udel.edu

***Abstract*—Edge computing is a new paradigm of computing, which aims at enhancing user experience by bringing computing resources closer to where data is produced by Internet of Things (IoT). Cloudlets, additional infrastructure components nearby users, facilitate edge services to decrease latency and network traffic. IoT users require edge services for their applications meeting a strict quality of service (QoS). A key challenge is how to efficiently match cloudlets to IoT applications to enable a convenient any-time access to edge computing services. In this paper, we address this problem by proposing novel two-sided matching solutions for edge services considering QoS requirements in terms of service response time. The matching mechanisms enhance the quality of experience of the users. In addition, we determine dynamic pricing of the edge services based on preferences and incentives of cloudlets, IoT users, and the system. The proposed matchings are Pareto-efficient, incentive compatible, weakly budget balanced, and computationally efficient. We perform a comprehensive assessment through extensive performance analysis experiments to evaluate our proposed matching and pricing solutions.**

***Index Terms*—edge computing, cloudlet, Internet of Things, two-sided matching**

I. Introduction

The number of Internet-connected devices (such as wearable devices, connected vehicles, and drones) is estimated to reach 50 billion by 2020. These smart devices, so-called Internet of Things (IoT), are restricted by weight, size, battery life, and heat dissipation imposing limitations on their computing capabilities to execute applications. To empower the resource shortage of IoT devices, cloud computing offers many services, which allows these devices to offload their tasks to a more powerful computing infrastructure. However, these devices require heterogeneous quality of service (QoS) depending on their applications [1], and offloading to a conventional centralized cloud is infeasible for applications mandating low-latency communications and real-time responses due to physical distance between the cloud and IoT users.

Edge computing is a new computing paradigm that enhances cloud computing for IoT applications by extending cloud services closer to the users through an additional infrastructure component, called *cloudlet* (near or colocated with the wireless access point). Cloudlets are small-sized edge clouds, which reside between the IoT device and the centralized data center, and they can provision their resources in the form of Virtual Machines (VMs) [2], [3]. However, to enable a convenient any-time access to edge computing resources, a key challenge is how to efficiently match cloudlets to IoT applications for providing edge services.

978-1-5386-6808-5/18/$31.00 ©2018 IEEE

A cloudlet has limited amount of available resources, and it can guarantee a service response time for the provided edge services. On the other hand, an IoT user has specific resource requirements for its application and may need a strict guarantee for the service response time. IoT requests are distributed among cloudlets, and they are matched to a set of cloudlets to cooperatively fulfill them leading to significant decrease in data communications over WANs and in response time. In this paper, we model the interactions of IoT users and cloudlets and design QoS-aware matching algorithms to give incentives to them to participate. We formulate this problem as a two-sided matching game between the IoT devices and the cloudlets.

A two-sided matching game is an assignment problem between the sets of devices and cloudlets (players), where the players of each set have preferences over the players of the other set. The preference relations allow every IoT device/cloudlet to be best matched for the edge services while maximizing its own benefit in the system. Finding the best matching of a two-sided matching game is an NP-hard problem. In this paper, we propose two novel matching solutions that are computationally efficient. We also demonstrate how our solutions improve quality of experience and user satisfaction by considering QoS metrics in the matching mechanisms.

Our Contribution. We model the cloudlet-IoT user matching as an Integer Programming model, and propose a novel two-sided matching, QMECS, for edge services that is Pareto-efficient, incentive compatible, weakly budget balanced, and computationally efficient. Our proposed two-sided matching is multi attributes that considers QoS requirements of IoT users and QoS guarantees of cloudlets in addition to their pricing preferences. We extend our proposed two-sided matching by proposing Modified QMECS (M-QMECS) to augment social welfare and avoid zero budget surplus. QMECS and M-QMECS improve users' satisfaction and quality of experience by considering QoS metrics in the matchings. In addition, our mechanisms avoid user service rejection after their assignment due to unacceptable quality of experience. We provide a comprehensive assessment through extensive performance analysis experiments and compare the solutions obtained with the optimal solutions.

Related Work. Traditional cloud computing solutions rely on centralized or semi-centralized (i.e., in-site distributed) VM provisioning, allocation, and placement approaches [4]–[6]. However, they require global information and often centralized controllers, yielding significant overhead and complexity especially when dealing with combinatorial integer programming problems to be solved. In addition, they do not consider the possibility of interactions among clouds and/or users, and thus they employ optimization techniques without considering users and clouds as decision makers. These limitations of optimization have led to an interesting body of literature dealing with the use of game theory in cloud resource management [7], [8].

Most of game-theoretical studies in cloud computing focus on only one side of the market [7], [9], where users interact with only one provider. Main stream cloud provider powerhouses such as Amazon have been offering cloud services in a one-sided auction market (Amazon Spot Market) for several years [10]. In our previous studies [11]–[14], we proposed truthful offline and online one-sided mechanisms for allocation and pricing of VMs with heterogeneous resources in clouds. Several researchers have studied resource management in cloud federations to facilitate big data processing [15], [16]. However, these studies only focus on interactions among a group of cloud providers to provide a single big data processing service, and they are not suitable to be adapted in edge computing environment. Efficient resource management mechanisms need to consider both supply and demand sides together. Several studies focused on designing double-auction mechanisms for cloud computing [17]–[19]. Nallur and Bahsoon [20] proposed market-based heuristic algorithms using a continuous double-auction to allow applications to decide which services to choose. Garg et al. [21] identified the various technical and market requirements and challenges in designing such an exchange market for cloud computing environment.

Most existing cloud-based solutions rely on conventional Internet for connectivity to the cloud. These solutions do not address the challenges of "being at the edge", and are not appropriate for edge computing [22]. Edge computing leverages cloud computing infrastructure and provides fully distributed services, while the proximity of cloudlets to end users is a crucial property of edge computing [2], [23]. The new emerging challenges in IoT and how edge computing provides effective ways to address these challenges are discussed in [24], [25]. Also, Jutila et al. [26] proposed adaptive edge computing solutions for IoT networking to optimize and control traffic flows and network resources. Most studies in edge computing focus on decisions on efficient caching [27], [28] and computation offloading (fully or partially) to cloudlets considering one user application [29] or multiple applications [30]. However, the main goal of edge computing is to satisfy the service time requirements of users by bringing cloud resources close to them. To the best of our knowledge, this is the first work that models QoS in the matchings of edge services to guarantee service time requirements. Moreover, the current level of understanding of analysis of interactions among IoT users and cloudlets is very limited. We model preferences and interactions of these decision makers using game theory and matching theory, and design novel QoS-aware matchings.

II. System Model

In this section, we describe the system model, where J is the set of cloudlets and I is the set of IoT users. Each user requests an indivisible bundle of VMs (one or several VMs) for her IoT application requiring some QoS metrics to be satisfied. Each cloudlet offers a set of VMs that can be allocated to different users, and it guarantees several QoS metrics for the offered services. The vector L defines the QoS metrics including Average Response Time (ART), Maximum Response Time (MRT), and Response Time Failure (RTF). All these QoS metrics are defined as quantified metrics to quantify service response time [31], where they can be simply declared in numeric values. Vector L can be expanded to include other quantified QoS metrics.

More specifically, each cloudlet $j \in J$ declares its number of available VMs, M_j, the reserve price p_j^c for providing a VM instance, and the service response time it guarantees denoted by $(RT_{jl}^c)_{l\in L}$. Hence, the specification of each cloudlet is denoted by $B_j^c = (M_j, p_j^c, (RT_{jl}^c)_{l\in L})$. For example, for a typical cloudlet j with twenty VMs available, the reserve price of \$2 for each instance, and guarantees of $ART = 1.5$ ms, $MRT = 2$ ms, and $RTF = 0.01\%$, its specification is $B_j^c = (20, \$2, [1.5ms, 2ms, 0.01\%])$. The superscript c refers to cloudlets in all the notations.

Each IoT device requests several VM instances of the same type, sets a suggested price for the requested bundle, and specifies QoS requirements for the requested edge service. Hence, the specification of each IoT request of a user consists of three parts. First, N_i represents the number of requested VMs for device i. Second, p_i^d represents bided price for the whole requested bundle, that is the maximum price user i is willing to pay for the requested bundle of VMs N_i. Finally, $(RT_{il}^d)_{l\in L}$ denotes the QoS requirements of device i, where RT_{il}^d for any $l \in L$ is an upper bound for the QoS metric l. These QoS values suggest the least acceptable service response time qualities for the edge services requested by IoT device i. As a result, each IoT request is denoted by $B_i^d = (N_i, p_i^d, (RT_{il}^d)_{l\in L})$. The superscript d refers to users in all the notations. For instance, the bid of user i, who is willing to pay upto \$8 for two VMs with $ART = 2$ ms, $MRT = 2.5$ ms, and $RTF = 0.05\%$ is denoted by $B_i^d = (2, \$8, [2ms, 2.5ms, 0.05\%])$. Figure 1 shows the supply and demand sides of the edge computing system.

To find the best matching of the services between IoT users and cloudlets, we formulate the problem as an integer program (IP). First, we define the decision variables as follows: α_{ij} is the number of allocated VMs by cloudlet j to IoT user i;

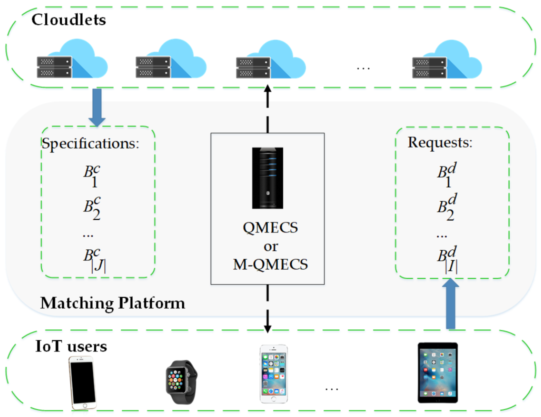


Fig. 1: System Model.

$x_i \in \{0, 1\}$ shows whether IoT user i has received his/her requested bundle or not; and y_j is the number of VMs allocated by cloudlet j. In addition, we define $Z(i, j) = \vec{RT}^d_{iL} \succeq \vec{RT}^c_{jL}$ as the indicator function for the match between cloudlet j and user i based on the offered and requested QoS, which returns 1 if $RT^d_{il} \geq RT^c_{jl}, \forall l \in L$, that is the offered QoS by cloudlet j meets the requirements of user i, and 0 otherwise. The value of $Z(i, j)$ indicates the feasibility of matching VMs provided by cloudlet j to IoT user i. In the above example, cloudlet j with $B^c_j = (20, \$2, [1.5ms, 2ms, 0.01\%])$ is qualified to provide the requested service by user i with $B^d_i = (2, \$8, [2ms, 2.5ms, 0.05\%])$, since it guarantees a better service response time; in this case $Z(\vec{RT}^d_{iL} \succeq \vec{RT}^c_{jL}) = 1$. We now formulate the problem as a mixed IP as follows:

$$\texttt{Matching}(I,J): \text{ Max } V(I, J) = \sum_{i\in I} p^d_i x_i - \sum_{j\in J} p^c_j y_j \quad (1)$$

Subject to:

$$\sum_{j\in J} \alpha_{ij} = N_i x_i, \forall i \in I, \quad (1a)$$

$$\sum_{i\in I} \alpha_{ij} = y_j, \forall j \in J, \quad (1b)$$

$$0 \leq \alpha_{ij} \leq M_j\, Z(i, j), \forall i \in I, \forall j \in J, \quad (1c)$$

$$x_i \in \{0, 1\}, \forall i \in I, \quad (1d)$$

$$y_j \in \{0, \dots, M_j\}, \forall j \in J. \quad (1e)$$

The objective function is to maximize the social function $V(I, J)$ defined by (1). Constraints (1a) ensure that each IoT user receives his/her whole requested bundle or nothing. Constraints (1b) guarantee that each cloudlet supplies VMs based on the availability of them. Constraints (1c) ensure that each cloudlet can serve an IoT user if only it can provide the requested VMs and meet the QoS requirements of that user. Constraints (1d) and (1e) restrict the allocated VMs under required and available VMs for users and cloudlets, respectively. Once a cloudlet is matched to an edge application(s), it allocated VMs according to the requested VMs of the IoT application(s).

III. Proposed Two-Sided Matching Mechanisms

Cloudlets and IoT users are self-interested and rational, meaning that their objectives are to maximize their own utilities. To promote the transactions and attract both users and cloudlets, we design QoS-aware matching and pricing solutions (QMECS and M-QMECS) that maximize the social welfare, i.e., the summation of the broker's payoff and each participant's utility.

To find the best matching of cloudlets and IoT users, our proposed mechanisms first add a phantom IoT user with a specific request and unlimited budget to the system, and then determine a selected set of users. The introduction of the phantom user creates an imbalance between VM availability and demand, and it provides an efficient way to improve market price equilibrium, which leads to a budget surplus. Then, our mechanisms find a mapping between the selected set of users and the original set of cloudlets. Finally, the transaction prices for both sides (users and cloudlets) are determined. The main difference between QMECS and M-QMECS is the introduction of VCG prices in M-QMECS. This way, we modify the matching and pricing steps to augment social welfare and avoid zero budget surplus.

In order to have a computationally efficient mechanism, the matching and pricing decisions are determined based on the obtained linear program solutions described in the following subsections.

A. QoS-aware Matching of Edge Computing Services (QMECS)

Our proposed QMECS mechanism is defined as follows:

Step 1: Preference submission. Collect one sealed bid/ask preference from each IoT user/cloudlet.

Step 2: User set reduction. Solve the following linear program $\texttt{SELECT}(I, J)$, with IoT set I and cloudlet set J:

$$\texttt{SELECT}(I, J): \text{Max } \tilde{V}(I, J) = \sum_{i\in I} p^d_i x_i - \sum_{j\in J} p^c_j y_j$$

Subject to:

$$\sum_{j\in J} \alpha_{ij} = N_i x_i, \forall i \in I,$$

$$\sum_{i\in I} \alpha_{ij} = y_j, \forall j \in J,$$

$$0 \leq \alpha_{ij} \leq M_j\, Z(i, j), \forall i \in I, \forall j \in J,$$

$$0 \leq x_i \leq 1, \forall i \in I,$$

$$0 \leq y_j \leq M_j, \forall j \in J.$$

For any IoT user $i \in I$, if $x_i = 1$, then i remains for the matching. All other users which could not acquire their whole requested bundles are eliminated. Let $\tilde{I}$ denote the set of remaining IoT users at this step.

Step 3: Setting padding. We define a padding q equal to the highest amount of supply, that means $q = \max_{j\in J}(M_j)$. Then, for each selected user $u \in \tilde{I}$, we define a padding vector $\boldsymbol{Q}_u$ of size $|\tilde{I}|$ such that $Q^i_u = q$ if $i = u$; otherwise $Q^i_u = 0$. The padding for IoT user u can be viewed as a phantom user with unlimited budget, same constraints as user u, and q units demand.

Step 4: Matching. For selected users in $\tilde{I}$, we solve the following linear program $\texttt{PADDING}(\tilde{I}, J, \boldsymbol{Q}_u)$ to determine the final set of IoT users to be matched for the edge services:

$\texttt{PADDING}(\tilde{I}, J, \boldsymbol{Q}_u)$:

$$\text{Max } \hat{V}(\tilde{I}, J, \boldsymbol{Q}_u) = \sum_{i\in\tilde{I}} p_i^d x_i - \sum_{j\in J} p_j^c y_j$$

Subject to:

$$\sum_{j\in J} \alpha_{ij} = N_i x_i + Q_u^i, \forall i \in \tilde{I},$$

$$\sum_{i\in\tilde{I}} \alpha_{ij} = y_j, \forall j \in J,$$

$$0 \le \alpha_{ij} \le M_j Z, \forall i \in \tilde{I}, \forall j \in J,$$

$$0 \le x_i \le 1, \forall i \in \tilde{I},$$

$$0 \le y_j \le M_j, \forall j \in J.$$

For any IoT user $u \in \tilde{I}$, if $x_u = 1$, then u is considered as one of final matched users. Let $\hat{I}$ denote the set of users selected in this step. Then, we solve $\texttt{SELECT}(\hat{I}, J)$ to determine the set of matching cloudlets and their allocation of edge services considering the trading set of IoT users $\hat{I}$.

Step 5: Determining payment. The buying price of edge services for a matched user $u \in \hat{I}$ is equal to π_u^d, that is the minimum bid price such that $x_u = 1$ in the optimal solution to $\texttt{PADDING}(\tilde{I}, J, \boldsymbol{Q}_u)$. This price can be viewed as a shadow price and can be calculated through sensitivity analysis. The revenue for each cloudlet is the VCG payment, $\pi_j^c = p_j^c y_j + \tilde{V}(\hat{I}, J) - \tilde{V}(\hat{I}, J/\{j\})$.

The QMECS mechanism is given in Algorithm 1. It has two inputs: the preferences of IoT users and cloudlets, and it returns the matching solution and payments as outputs. The matching solution α shows the number of allocated VMs by each cloudlet to each IoT user. The set $\Pi^d = \{\pi_i^d | i \in \hat{I}\}$ determines the buying prices for the matched IoT users. Finally, the last output is the selling prices of the matched cloudlets, i.e., $\Pi^c = \{\pi_j^c | j \in J\}$.

1) Example: We provide an example to demonstrate how our proposed mechanism works. Consider six cloudets and three IoT users with the preferences shown in Tables I and II, respectively. All cloudlets are qualified to provide the services to the users since the QoS requirements are satisfied.

After collecting the preferences, $\texttt{SELECT}(I, J)$ is solved, where user 1 acquires his/her whole requested bundle, user 2 acquires 3/4 of his/her bundle, and user 3 acquires nothing. Therefore, $\tilde{I} = \{i_1\}$. In the next step, for each user $u \in \tilde{I}$, we solve $\texttt{PADDING}(I, J, \boldsymbol{Q_u})$, where $Q_u^i = 2$, i.e., the maximum supply, if $i = u$, and $Q_u^i = 0$ if $i \neq u$.

User 1 is in the optimal solution to $\texttt{PADDING}(I, J, \boldsymbol{Q_1})$. Therefore, user 1 is determined as a winning user, and it is matched to the most efficient cloudlets based on the optimal solution to $\texttt{SELECT}(\hat{I}, J)$. The results show that user 1 receives his/her requested VMs from cloudlets 1 to 4. The

Algorithm 1 QMECS Mechanism

```
/*Step 1: Preference submission*/
Input: $B_i^d = (N_i, p_i^d, (RT_{il}^d)_{l\in L}); \forall i \in I$
Input: $B_j^c = (M_j, p_j^c, (RT_{jl}^c)_{l\in L}); \forall j \in J$
/*Step 2: User set reduction*/
$\tilde{I} = \{i | i \in I, x_i = 1$ in the optimal solution to $\texttt{SELECT}(I, J)\}$
/*Step 3: Setting padding*/
for all $u \in \tilde{I}$ do
    if $i = u$ then
        $Q_u^i = q \quad \{q = \max\{M_j | j \in J\}\}$
    else if $i \neq u$ then
        $Q_u^i = 0$
/*Step 4: Matching*/
for all $u \in \tilde{I}$ do
    $\hat{I} = \{u | u \in \tilde{I}, x_u = 1$ in the optimal solution to $\texttt{PADDING}(I, J, \boldsymbol{Q_u})\}$ $\{\hat{I}$ is the set of matched IoT users$\}$
Solve $\texttt{SELECT}(\hat{I}, J)$ to determine $\alpha$, where
$\alpha = \{\alpha_{ij} | \forall i \in \hat{I}, \forall j \in J\}$
/*Step 5: Determining payment*/
$\Pi^d = \{\pi_u^d$ : min price in $\texttt{PADDING}(\tilde{I}, J, \boldsymbol{Q}_u)$, where $u$ is selected$| \forall u \in \hat{I}\}$
$\Pi^c = \{\pi_j^c = p_j^c y_j + \tilde{V}(\hat{I}, J) - \tilde{V}(\hat{I}, J/j) | \forall j \in J\}$
Output: $\alpha; \Pi^d; \Pi^c$
```

TABLE I: Ask Preferences of 6 Cloudlets

Cloudlets	No. of VMs	Unit price ($)	Guaranteed QoS		
			ART	*MRT*	*RTF*
c_1	2	3	1.5	1.6	0.01
c_2	1	3	1.1	1.5	0.01
c_3	2	3	1.0	1.9	0.01
c_4	2	3	1.8	2.1	0.02
c_5	1	3	1.8	2.0	0.02
c_6	2	3	1.5	2.1	0.03

TABLE II: Bid Preferences of 3 IoT Users

	Asked bundle		Asked QoS		
IoT users	No. of VMs	price	*ART*	*MRT*	*RTF*
u_1	7	77	2.2	2.5	0.05
u_2	4	38	2.0	2.3	0.05
u_3	4	36	1.9	2.2	0.04

buying price for user 1 is equal to \$3 and the selling price for each cloudlet 1 to 4 is \$3. This means \$21 is received from user 1 and it is distributed as follows: \$6 to cloudlet 1, 3, and 4, and \$3 to cloudlet 2.

2) Discussion: The proposed QMECS mechanism demonstrates the properties of Incentive Compatibility (IC), Individual Rationality (IR), and weakly Budged Balance (BB), if $q + 1 > \max\{M_j | j \in J\}$. These properties are critical for the mechanism in order to be practical in real world. Furthermore, the QMECS mechanism is Computationally Efficient (CE), as it only requires to solve linear programs to determine matchings, allocations, and payments.

In the QMECS mechanism, allocations of VMs are based on the lexicographic order of users and cloudlets. That is, a user has a higher chance to be matched if it submits a higher maximum price. Conversely, the lower prices cloudlets offer, the higher chance for them to be matched.

However, the QMECS mechanism may result in a non-optimal social welfare and zero budget surplus. Frequent occurrence of zero budget for the system may lead to losing incentives for the broker to continue running the mechanism. This is due to the fact that the cost of running the QMECS mechanism is not compensated with the zero payoff in the long-term. In the following, we propose M-QMECS mechanism to augment social welfare and avoid zero budget surplus.

B. Modeifed QoS-aware Matching of Edge Computing Services (M-QMECS)

In the M-QMECS mechanism, we define the VCG price for each IoT user i as $\pi_i^d = p_i^d - (\tilde{V}(I,J) - \tilde{V}(I/\{i\},J))$, where $\tilde{V}(I/\{i\},J)$ is the social welfare when user i is absent (does not participate). The procedure of M-QMECS mechanism is as follows:

Step 1: Preference submission. Collect one sealed bid/ask preference from each IoT user/cloudlet.

Step 2: VCG price calculation. For each IoT user $i \in I$, we calculate its VCG-based payment π_i^d. We then eliminate all users with $p_i^d < \pi_i^d$. Let I' denote the updated user set.

Step 3: User set reduction. Solve the linear program $\texttt{SELECT}(I',J)$ with user set I' and cloudlet set J. Let $\tilde{I}'$ denote the set of remaining users $i \in I'$, where $x_i = 1$ in the optimal solution to $\texttt{SELECT}(I',J)$.

Step 4: Setting padding. For each user $u \in \tilde{I}'$, we set the padding vector $\boldsymbol{Q}_u$ the same as Step 3 in QMECS mechanism.

Step 5: Matching. For each user $u \in \tilde{I}'$, we solve the $\texttt{PADDING}(\tilde{I}',J,\boldsymbol{Q}_u)$ to obtain the final IoT user set $\hat{I}'$ that will receive the edge services. Then, we solve $\texttt{SELECT}(\hat{I}',J)$ to determine the matching and allocations. Next, we compare $\tilde{V}(\hat{I}',J)$ with $\tilde{V}(\hat{I},J)$ in QMECS mechanism. If $\tilde{V}(\hat{I}',J) < \tilde{V}(\hat{I},J)$, we follow the matching in QMECS mechanism (Steps 2-4 in QMECS), and then the payment determination in M-QMECS (Step 6). Otherwise, if $\tilde{V}(\hat{I}',J) > \tilde{V}(\hat{I},J)$, we follow the steps in M-QMECS.

Step 6: Determining payment. The buying price for each matched user $u \in \hat{I}$ obtained from $\texttt{PADDING}(\tilde{I}',J,\boldsymbol{Q}_u)$ is denoted by $\hat{\pi}_u^{d*}$. If $p_i^d \geq \pi_i^d$, then $\hat{\pi}_u^{d*} = \max\{\hat{\pi}_c^d, \pi_i^d\}$, otherwise $\hat{\pi}_u^{d*} = \hat{\pi}_u^d$. Similar to QMECS, the revenue for each cloudlet j is the VCG payment, $\pi_j^c = p_j^c y_j + \tilde{V}(\hat{I}',J) - \tilde{V}(\hat{I}',J/\{j\})$.

The M-QMECS mechanism is given in Algorithm 2. The mechanism receives the bids of users and the asks of cloudlets as inputs, and it returns α, Π^d, Π^c as outputs.

1) Example: To demonstrate the procedure of M-QMECS mechanism, consider the example in Section III-A1. After collecting the preferences, the VCG payments for IoT users are calculated. The VCG payment for users 1 to 3 is equal to \$71, \$38, and \$36, respectively. Therefore, $I' = \{2,3\}$. Then, $\texttt{SELECT}(I',J)$ is solved, and users $\{2,3\}$ remain in $\tilde{I}'$. The padding is set to $q = 2$. Based on the matching step, all users in $\tilde{I}'$ receive their requested VMs such that, user 2 acquires his/her VMs from cloudlets 1 to 3 and user 3 acquires his/her VMs from cloudlets 3 to 5.

Algorithm 2 M-QMECS Mechanism

```
/*Step 1: Preference submission*/
Input: $B_i^d = (N_i, p_i^d, (RT_{il}^d)_{l \in L})$; $\forall i \in I$
Input: $B_j^c = (M_j, p_j^c, (RT_{jl}^c)_{l \in L})$; $\forall j \in J$
/*Step 2: VCG-based payment calculation*/
for all $i \in I$ do
    $\pi_i^d = p_i^d - (\tilde{V}(I,J) - \tilde{V}(I/\{i\},J))$
$I' = \{i | i \in I, \pi_i^d \geq p_i^d\}$
/*Step 3: User set reduction*/
$\tilde{I}' = \{i | i \in I', x_i = 1$ in the optimal solution to $\texttt{SELECT}(I',J)\}$
/*Step 4: Setting padding*/
for all $u \in \tilde{I}'$ do
    if $i = u$ then
        $Q_u^i = q \quad \{q = \max\{M_j | j \in J\}\}$
    else if $i \neq u$ then
        $Q_u^i = 0$
/*Step 5: Matching*/
for all $u \in \tilde{I}'$ do
    $\hat{I}' = \{u | u \in \tilde{I}', x_u = 1$ in the optimal solution to $\texttt{PADDING}(I',J,\boldsymbol{Q_u})\}$ $\{\hat{I}'$ is the set of matched IoT users$\}$
Solve $\texttt{SELECT}(\hat{I}',J)$ to determine $\alpha$, where
$\alpha = \{\alpha_{ij} | \forall i \in \hat{I}', \forall j \in J\}$
if $\tilde{V}(\hat{I}',J) \leq \tilde{V}(\hat{I},J)$ then
    Follow steps 2-4 in QMECS, then step 6 in M-QMECS
else if $\tilde{V}(\hat{I}',J) > \tilde{V}(\hat{I},J)$ then
    Follow the steps in M-QMECS
/*Step 6: Determining payment*/
for all $i \in I$ do
    if $p_i^d \geq \pi_i^d$ then
        $\hat{\pi}_c^{d*} = \max\{\hat{\pi}_c^d, \pi_i^d\}$
    else if $p_i^d < \pi_i^d$ then
        $\hat{\pi}_u^{d*} = \hat{\pi}_u^d$
    $\Pi^d \leftarrow \Pi^d \bigcup \hat{\pi}_c^{d*}$
$\Pi^c = \{\pi_j^c = p_j^c y_j + \tilde{V}(\hat{I}',J) - \tilde{V}(\hat{I}',J/j) | \forall j \in J\}$
Output: $\alpha; \Pi^d; \Pi^c$
```

Since $\tilde{V}(\hat{I}',J) = \$50$ and is higher than $\tilde{V}(\hat{I},J) = \$49$, the final payments are calculated based on Step 6 of M-QMECS. The payments of users 2 and 3 are equal to \$9.5 and \$9, respectively. Each cloudlet 1 to 4 receives \$3. The system receives a total of \$74 from the selected users and pays \$24 to the matched cloudlets (\$6 to each cloudlet 1, 3, and 4, and \$3 to cloudlets 2 and 5). Therefore, the system's payoff is \$50, which is equal to the optimal social welfare.

2) Discussion: Our proposed M-QMECS mechanism is IC, IR, and BB, if $q + 1 > \max\{M_j | j \in J\}$. The mechanism is also CE. Unlike QMECS mechanism, the matching in M-QMECS mechanism is not based on the lexicographic order of users and cloudlets. However, compared to QMECS mechanism, the M-QMECS mechanism augments social welfare and avoid zero budget surplus. In the above example, the maximum social welfare and a higher budget surplus are realized by adopting M-QMECS mechanism.

TABLE III: Statistics of Datasets with different workloads.

Dataset	# of users	# of cloudlets	# of asked VMs by each user	# of offered VMs by each cloudlet	Description
Dataset-0	[5,10]	[40,45]	[1,10]	[1,10]	Very high supply
Dataset-1	[15,20]	[35,40]	[1,10]	[1,10]	High supply
Dataset-2	[40,45]	[5,10]	[1,10]	[1,10]	Very high demand
Dataset-3	[35,40]	[15,20]	[1,10]	[1,10]	High demand
Dataset-4	25	50	[2,10]	[1,5]	Moderate supply-demand ratio with a small padding size
Dataset-5	25	10	[2,10]	[5,25]	Moderate supply-demand ratio with a medium padding size
Dataset-6	25	5	[2,10]	[10,50]	Moderate supply-demand ratio with a high padding size

TABLE IV: Statistics of Datasets with different QoS requirements and guarantees.

	asked by each user			guaranteed by each cloudlet			
Dataset	ART	MRT	RTF	ART	MRT	RTF	Description
Dataset-7	[2,7]	[4,9]	[0.2%,0.4%]	[3,8]	[5,10]	[0.3%,0.5%]	Lower QoS guarantees than requirements
Dataset-8	[3,8]	[5,10]	[0.3%,0.5%]	[3,8]	[5,10]	[0.3%,0.5%]	Similar QoS requirements and guarantees
Dataset-9	[3,8]	[5,10]	[0.3%,0.5%]	[2,7]	[4,9]	[0.2%,0.4%]	Higher QoS guarantees than requirements

C. Properties

We now prove how M-QMECS mechanism improves social welfare and budget surplus compared to the results of QMECS.

Theorem 1. *M-QMECS mechanism can augment the social welfare compared to that of QMECS mechanism.*

Proof. We consider the following two possible cases between p_i^d and π_i^d for user i:

Case 1: Assume $p_i^d \geq \pi_i^d$ and $i \in I'$. According to $\pi_i^d = p_i^d - (\tilde{V}(I,J) - \tilde{V}(I/\{i\},J))$, and given that $p_i^d \geq \pi_i^d$, we can conclude $\tilde{V}(I,J) \geq \tilde{V}(I/\{i\},J)$. Therefore, if user i receives the edge services, the obtained social welfare is the highest compared with all other schemes without user i.

Case 2: Assume $p_i^d < \pi_i^d$ and $i \in I'$. According to $\pi_i^d = p_i^d - (\tilde{V}(I,J) - \tilde{V}(I/\{i\},J))$, and given that $p_i^d < \pi_i^d$, we can conclude $\tilde{V}(I,J) < \tilde{V}(I/\{i\},J)$. Therefore, if user i receives the edge services, the obtained social welfare is not the highest compared with all other schemes without user i. However, by eliminating all users with $p_i^d < \pi_i^d$ from the matching in Step 2 of M-QMECS mechanism, we can avoid this case compared to QMECS mechanism. $\square$

Lemma 1. *The social welfare obtained by M-QMECS mechanism is not less than that of QMECS mechanism.*

Proof. We consider the following two cases:

Case 1: If all users are constrained by $p_i^d \geq \pi_i^d$, then the final matching results will be the same for both mechanisms leading to the same social welfare.

Case 2: If more than one user is constrained by $p_i^d < \pi_i^d$, then let $I'_{-i}(I'_{-i} \subseteq I')$ denote the set of users who can be selected because of the absence of user i. Then, according to $\pi_i^d = p_i^d - (\tilde{V}(I,J) - \tilde{V}(I/\{i\},J))$ and given $p_i^d < \pi_i^d$, we can conclude $\tilde{V}(I,J) < \tilde{V}(I/\{i\},J)$ and subsequently $p_i^d x_i < \sum_{i' \in I'_{-i}} p_{i'}^d x_{i'}$. Since the matching sequence of users follows the lexicographic order from high to low, the users in I'_{-i} must have the lowest bid prices among the user set $\tilde{I}'$. If all users in I'_{-i} are in the optimal solution to $\texttt{PADDING}(\tilde{I}', J, \boldsymbol{Q}_{i'})(i' \in I'_{-i})$, then the obtained social welfare by M-QMECS mechanism is higher than that of QMECS mechanism. Since either $\tilde{V}(\hat{I}', J) < \tilde{V}(\hat{I}, J)$ or $\tilde{V}(\hat{I}', J) > \tilde{V}(\hat{I}, J)$, M-QMECS mechanism selects the matching that leads to a higher social welfare.

To sum up, the obtained social welfare by M-QMECS mechanism is higher than or equal to the social welfare obtained by QMECS mechanism. $\square$

Theorem 2. *The obtained budget surplus by M-QMECS mechanism is not less than that of QMECS mechanism.*

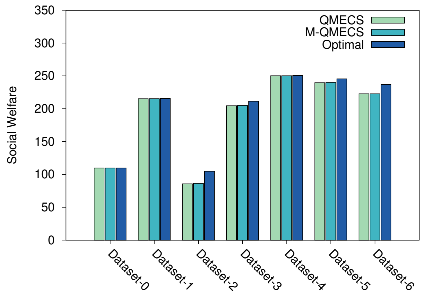
Proof. In the M-QMECS mechanism, the payment of each user u is $\hat{p}_u^{d*} = \max\{\hat{p}_u^d(\tilde{I}', J, \boldsymbol{Q}_u), \pi_i^d\}$. As a result, the user payments in M-QMECS mechanism are not less than those induced by the QMECS mechanism. Moreover, in M-QMECS mechanism each cloudlet owns the same utility as QMECS mechanism since both mechanisms use the same payment function for the cloudlets. Therefore, using Lemma 1, we can conclude that the budget surplus of M-QMECS mechanism is higher than or equal to the surplus obtained by QMECS mechanism. $\square$

IV. Experimental Results

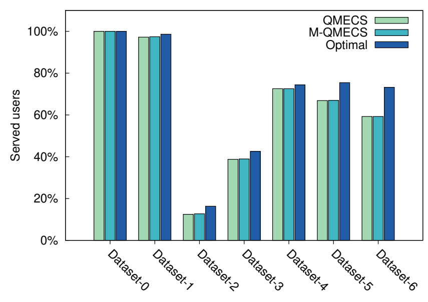
Our proposed QMECS and M-QMECS mechanisms are IC, IR, BB, and CE. In addition, we perform extensive experiments to evaluate the performance of the two mechanisms.

Experimental setup. We used Java and ILOG Concert Technology to simulate the mechanisms, and we created several datasets, each with specific features, to evaluate the mechanisms under different workloads (Table 3) and QoS requirements and guarantees (Table 4). We generated the bid and ask prices of each VM instance based on Amazon EC2 pricing from the uniform distributions on [3.5, 5.5] and [2, 4], respectively.

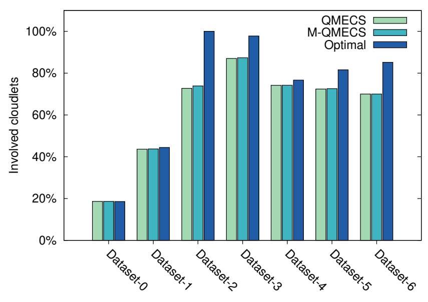
Analysis of Results. Figure 2a shows the realized social welfare by both mechanisms and also the optimal social welfare. The results show when supply is higher than demand, both mechanisms can obtain the optimal social welfare. Conversely, for the datasets with lower supply to demand ratio, the difference between obtained social welfare by the mechanisms and the optimum welfare increases. For the datasets with moderate supply-demand ratio, as the padding size increases a higher



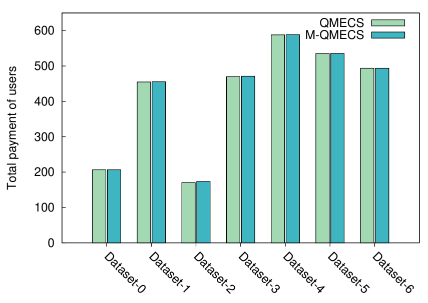
(a) Social welfare



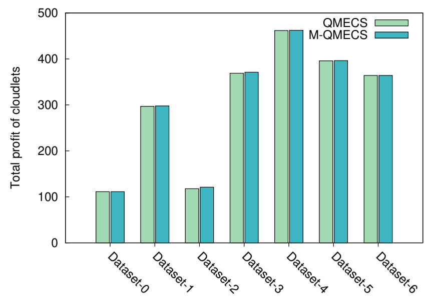
(b) Percentage of matched users



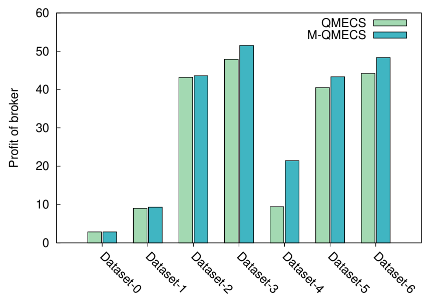
(c) Percentage of matched cloudlets



(d) Users' payments



(e) Cloudlets' profit



(f) Broker's profit

Fig. 2: Comparison of the matching mechanisms under variant supply-demand ratios

proportion of users are eliminated in the matching step. This in turn results in a lower social welfare. For all datasets, the social welfare of M-QMECS mechanism is slightly higher than or equal to that of QMECS mechanism, which is consistent with Lemma 1.

According to Figure 2b, as supply to demand ratio increases, users have higher chances to acquire their requested services. However, for the datasets with the same supply-demand ratio, the percentage of successful users strongly depends on the padding size. This is quite obvious in the results for Datasets 4-6, where higher padding size has resulted in lower success rate for users. As mentioned above, this is due to the fact that a high number of users are eliminated in the matching phase. Here again the results for M-QMECS are slightly higher compared to QMECS.

Figure 2c demonstrates the involvement percentage of cloudlets in serving user requests. The results show that in Datasets 1-2 the involvement rate is the lowest since the demand is really low such that a small number of cloudlets are sufficient to fulfill their needs. As the demand increases, more coudlets succeed to allocate their resources in the matching. When the padding size increases, lower number of users remain for the matching phase and as a result the chances for cloudlets in order to allocate their resources decrease. This in turn causes a reduction in the percentage of cloudlets' involvement in the matching. The cloudlets involvement rate for M-QMECS is slightly higher than or equal to QMECS.

According to Figure 2d, the total payments from users in M-QMECS is higher than that of QMECS. Note that according to the payment step of M-QMECS, each user's payment is either higher than or equal to its payment in QMECS. Moreover, a slightly higher success rate in M-QMECS leads to a higher total payments in M-QMECS. Similarly, as it is illustrated in Figure 2e, the higher involvement rate of cloudlets in the matching step of M-QMECS leads to a higher total profit for the cloudlets.

Figure 2f illustrates the broker's utility under various workloads. The results show that the broker's utility is always higher in M-QMECS, which is expected based on Theorem 2.

To sum up, both mechanisms show a better performance as the number of cloudlets increases and the padding size decreases. Nonetheless, M-QMECS outperforms QMECS in all the cases, and more specifically, it brings a higher budget surplus leading to a higher profit for the broker of the system. This makes M-QMECS a more attractive choice for the broker. However, M-QMECS does not follow the lexicographic order of users in the matchings and requires more computations.

Figure 3 shows the efficiency of our proposed mechanisms in edge computing due to considering QoS metrics in the matchings. For comparison, we use MECS and M-MECS based on QMECS and M-QMECS mechanisms, respectively. MECS and M-MECS apply the matchings without considering QoS requirements and guarantees, while maximizing the social welfare. By the end of matchings, there may be users whose QoS requirements are not satisfied by the matched cloudlets. The results show that the consideration of QoS metrics in the matchings of our proposed mechanisms improves user satisfaction and quality of experience.

We conclude that QMECS and M-QMECS are suitable two-sided matching mechanisms for edge computing environment

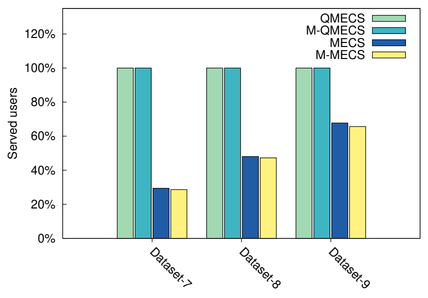


Fig. 3: User served after matching.

that consider incentives, preferences, and QoS requirements of the IoT users, the cloudlets, and the system broker.

V. Conclusion

With rapid changes in the quality-of-service requirements of Internet-of-Things and emerging new paradigms such as edge computing, efficient QoS-aware matching algorithms are needed to match cloudlets to IoT applications when providing edge computing services. In this paper, we addressed this challenge by proposing novel two-sided matching solutions, QMECS and M-QMECS, for edge services considering QoS requirements in terms of service response time. In addition, we proposed payment determination solutions considering preferences and incentives of cloudlets, IoT users, and the system. The proposed matchings are Pareto-efficient, incentive compatible, weakly budget balanced, and computationally efficient. The experimental results showed that the mechanisms are suitable for edge computing services by providing close to optimal matching, efficient pricing, and guaranteed QoS. For future work, we plan to extend the mechanisms considering mobility of the IoT devices and cloudlets.

Acknowledgment. This research was supported in part by NSF grant CNS-1755913.

References

[1] E. Balevi and R. D. Gitlin, "Unsupervised machine learning in 5g networks for low latency communications," in *Proc. of the 36th IEEE International Performance Computing and Communications Conference*, 2017, pp. 1–2.

[2] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[3] W. Shi and S. Dustdar, "The promise of edge computing," *Computer*, vol. 49, no. 5, pp. 78–81, 2016.

[4] S. Singh and I. Chana, "QoS-aware autonomic resource management in cloud computing: a systematic review," *ACM Computing Surveys*, vol. 48, no. 3, p. 42, 2016.

[5] G. Portaluri, D. Adami, A. Gabbrielli, S. Giordano, and M. Pagano, "Power consumption-aware virtual machine placement in cloud data center," *IEEE Transactions on Green Communications and Networking*, vol. 1, no. 4, pp. 541–550, 2017.

[6] M. Li, Y.-E. Sun, H. Huang, J. Yuan, Y. Du, Y. Bao, and Y. Luo, "Profit maximization resource allocation in cloud computing with performance guarantee," in *Proc. of the 36th IEEE International Performance Computing and Communications Conference*, 2017, pp. 1–2.

[7] H. Zhang, H. Jiang, B. Li, F. Liu, A. V. Vasilakos, and J. Liu, "A framework for truthful online auctions in cloud computing with heterogeneous user demands," *IEEE Transactions on Computers*, vol. 65, no. 3, pp. 805–818, 2016.

[8] W. Shi, L. Zhang, C. Wu, Z. Li, F. Lau, W. Shi, L. Zhang, C. Wu, Z. Li, and F. Lau, "An online auction framework for dynamic resource provisioning in cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2060–2073, 2016.

[9] L. Zhang, Z. Li, and C. Wu, "Dynamic resource provisioning in cloud computing: A randomized auction approach," in *Proc. of IEEE INFOCOM*, 2014, pp. 433–441.

[10] Amazon EC2 Spot Instances. [Online]. Available: https://aws.amazon.com/ec2/spot/pricing/

[11] L. Mashayekhy, M. Nejad, and D. Grosu, "A PTAS mechanism for provisioning and allocation of heterogeneous cloud resources," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 9, pp. 2386–2399, 2015.

[12] ——, "Physical machine resource management in clouds: A mechanism design approach," *IEEE Transactions on Cloud Computing*, vol. 3, no. 3, pp. 247–260, 2015.

[13] M. Nejad, L. Mashayekhy, and D. Grosu, "Truthful greedy mechanisms for dynamic virtual machine provisioning and allocation in clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 2, pp. 594 – 603, 2015.

[14] L. Mashayekhy, M. M. Nejad, D. Grosu, and A. V. Vasilakos, "An online mechanism for resource allocation and pricing in clouds," *IEEE transactions on computers*, vol. 65, no. 4, pp. 1172–1184, 2016.

[15] Z. Ma, Z. Sheng, and L. Gu, "DVM: A big virtual machine for cloud computing," *IEEE Transactions on Computers*, 2013.

[16] L. Mashayekhy, M. M. Nejad, and D. Grosu, "Cloud federations in the sky: Formation game and mechanism," *IEEE Transactions on Cloud Computing*, vol. 3, no. 1, pp. 14–27, 2015.

[17] D. Kumar, G. Baranwal, Z. Raza, and D. P. Vidyarthi, "A systematic study of double auction mechanisms in cloud computing," *Journal of Systems and Software*, vol. 125, pp. 234–255, 2017.

[18] B. Shi, J. Wang, Z. Wang, and Y. Huang, "Trading web services in a double auction-based cloud platform: A game theoretic analysis," in *Proc. of the IEEE International Conference on Services Computing*, 2017, pp. 76–83.

[19] L. Mashayekhy, M. Nejad, and D. Grosu, "A two-sided market mechanism for trading big data computing commodities," in *Proc. of the IEEE International Conference on Big Data*, 2014, pp. 153–158.

[20] V. Nallur and R. Bahsoon, "A decentralized self-adaptation mechanism for service-based applications in the cloud," *IEEE Transactions on Software Engineering*, vol. 39, no. 5, pp. 591–612, 2013.

[21] S. K. Garg, C. Vecchiola, and R. Buyya, "Mandi: a market exchange for trading utility and cloud computing services," *The Journal of Supercomputing*, vol. 64, no. 3, pp. 1153–1174, 2013.

[22] 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, 2017.

[23] M. N. Islam, V. C. Patil, and S. Kundu, "Determining proximal geolocation of IoT edge devices via covert channel," in *Proc. of the 18th Intl. Symposium on Quality Electronic Design*, 2017, pp. 196–202.

[24] M. Chiang and T. Zhang, "Fog and IoT: An overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.

[25] 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, 2016.

[26] M. Jutila, "An adaptive edge router enabling internet of things," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1061–1069, 2016.

[27] W. Li, Y. Li, W. Wang, Y. Xin, and T. Lin, "A dominating-set-based and popularity-driven caching scheme in edge ccn," in *Proc. of the 34th IEEE International Performance Computing and Communications Conference*, 2015, pp. 1–2.

[28] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Cooperative content caching in 5g networks with mobile edge computing," *IEEE Wireless Communications*, vol. 25, no. 3, 2018.

[29] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proc. of the IEEE Intl. Symposium on Information Theory*, 2016, pp. 1451–1455.

[30] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, no. 5, pp. 2795–2808, 2016.

[31] S. K. Garg, S. Versteeg, and R. Buyya, "A framework for ranking of cloud computing services," *Future Generation Computer Systems*, vol. 29, no. 4, pp. 1012–1023, 2013.