IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 22, 2018, accepted July 23, 2018, date of publication July 31, 2018, date of current version August 28, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2861561

Non-Payment Incentive Mechanism Design for Resource Allocation in a Private Cloud System

WEIWEI WU[1], (Member, IEEE), MINMING LI[2], (Senior Member, IEEE), JIANPING WANG[2], (Member, IEEE), AND XIUMIN WANG[3], (Member, IEEE)

[1]School of Computer Science and Engineering, Southeast University, Nanjing 211189, China
[2]Department of Computer Science, City University of Hong Kong, Hong Kong
[3]School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

Corresponding author: Xiumin Wang (xmwang@scut.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1003000, in part by the National Natural Science Foundation of China under Grants 61672154 and 11771365, in part by the Hong Kong General Research Fund under Project 11213114, and in part by the Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University.

ABSTRACT Truthful resource request from users is the premise to achieve the maximum social welfare in an enterprise private cloud. To stimulate the truthfulness of users, most previous works mainly rely on introducing the payment, which however, might not be applicable in enterprise private clouds, where there is a lack of money transfer. To address this issue, this paper proposes non-payment but efficient mechanisms in private clouds to stimulate the truthfulness of the users and meanwhile maximize the social welfare. Moreover, different from previous works that allow only one job request from one user, this paper studies a more general model, where multiple jobs can be submitted by each user. Specifically, we consider two task models: the migration-admissible model and non-migration model. In the former model, jobs can be executed at different servers, and may undergo migration if necessary. Alternatively, in the latter model, jobs can only be executed at one server without migration. For both models, we design incentive resource allocation mechanisms to maximize the social welfare. Theoretically analysis shows that the proposed mechanisms are truthful for general monotonic profit functions and the worst-case performance on the social welfare are well-bounded within a constant factor of the optimal solution for linear profit functions. Simulation results also demonstrate that the performances of the proposed mechanisms are very close to the optimal solution, in terms of maximizing the social welfare.

INDEX TERMS Cloud computing, resource allocation, mechanism design non-payment, incentives, approximation.

I. INTRODUCTION

Private clouds, with lower cost and better security control, have attracted more and more enterprises in recent years [1], [6]. However, it is hard to ensure the user truthfulness because there is no payment from the users for resource consumption in a private cloud. Without payment for resources consumption, users may behave selfishly in order to get more resources or earlier task completion time, which would lead to a waste of system resources and reduce overall efficiency.

We use the following example, possibly a counter-intuitive one, to demonstrate that an incentive mechanism is necessary to stimulate users' truthfulness. Suppose that a private cloud accepts resource requirements from users periodically and allocates resource to users in the coming round according to its resource scheduler strategy. In each round, a user can submit her resource requirements for multiple jobs. The utility of each user is the total completion time of her jobs. As shown in Fig. 1, a provider collects the requests/bids at the beginning, allocates the resource in period $[1, T]$, and then starts a new round. Suppose that the total resource is 10 units in the system. The three jobs of the first user are ready to be executed in the first round and need $2, 4, 3$ units of resource respectively. The three jobs of the second user require $5, 6, 7$ units of resource respectively. Suppose that the provider allocates the resource according to RR-rule (Round Robin) or WSPT-rule (weighted shortest processing time first). When the first user truthfully reveals all her jobs

2169-3536 © 2018 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

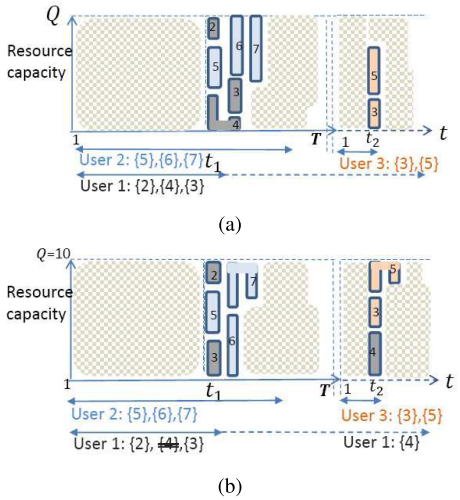

FIGURE 1. An example of the untruthful bidding of multi-job request. (a) When user 1 truthfully reveals all her jobs (grey), only the job that needs 2 units of resource is finished by time t_1 due to resource capacity $Q = 10$. (b) When user 1 unreveals/delays the request of the job that needs 4 units of resource, the jobs that need 2 and 3 units of resource are finished by time t_1.

at the beginning, one of her jobs is finished within $[1, t_1]$ while other jobs are finished later. If the first user strategically unreveals the job with 3 required units of resource at the first round of bidding and postpones the revealing in later rounds, then the first user finishes one more job in interval $[1, t_1]$ and gains more utility from the selfish bidding by under-reporting her job. The intuition behind such a phenomena lies in the fact that, through hiding/delaying behaviors, the users successfully manipulate the system and change the processing priority (determined by the system) of her jobs, and hence improves her own utility. Moreover, the untruthful behavior of the first user may hinder the profit of other users. For example, when one job of the first user is postponed to be revealed, the third user finishes less jobs in $[1, t_2]$ in the coming round. To avoid the untruthful reporting of job list, an incentive mechanism, which adopts the concept of Nash Equilibrium (NE) [14], [4], [24] to regulate the untruthful behaviors of the users, is strongly necessary.

In the literature, lots of research efforts have been conducted on designing truthful mechanisms in cloud systems [13], [18], [20]–[23]. Nevertheless, most of them rely on introducing the payment function or money transfer between users and the cloud provider. Specifically, the payment function effectively helps in stimulating the truthfulness, since technically it is equivalent to introduce powerful and flexible complementary functions for each user to compensate for the user's utility function. However, the implementation of payment function resorts to the support of money transfer between the users and the provider, and a virtual payment scheme is not that effective to attract the users in practice. Thus, a natural and practical concern arises that, how should the cloud provider design mechanisms to incentivize the truthful participation of users without resorting to payment, especially in scenarios without money transfer, *e.g.* in the private clouds inside the enterprises.

Different from the existing work, this paper conducts the first theoretical work on the non-payment truthful mechanism design for resource allocation in cloud computing. Actually, without payment, it is quite challenging to achieve the local objectives of truthfulness for all users and the system objective of good social welfare in general domains, as stated in the impossibility results of Gibbard-Satterthwaite theorem [14]. Existing works on non-payment mechanism design are restricted in limited fields, like facility location [17], social choices [5], assignment problem on bipartite graphs [7]. To the best of our knowledge, no prior works have addressed the non-payment mechanism design resource allocation in cloud computing. What is more, existing works for the truthful mechanism design are mainly restricted to the scenario that users request their resource for only one job [12], [21], [25], [26]. However, as we mentioned earlier, a user who has multiple jobs can under-report his jobs in one round/bid in order to get more utility, which could offer a new way for users to manipulate the resource allocation scheduler and calls for truthful mechanism design against such untruthful behaviors.

Our main contributions are summarized as follows.

- This paper considers the non-payment incentive mechanism design on resource allocation in cloud computing. We propose randomized mechanisms to stimulate users' truthfulness (i.e, to enforce the users to report truthfully while optimizing their utilities) and maximize the social welfare (the overall profit gained from the users). The proposed mechanisms generally follow a two-step framework, consisting of a virtual allocation and correspondingly a randomized rounding procedure, where the allocation strategies and rounding methods can be adaptive to the allocation constraints in different models. In general, the virtual allocation aims to achieve a good social welfare at the first step, and the rounding procedure aims to incentivize the truthfulness for every user while preserving the performance on social welfare. Our work moves a step forward towards the non-payment incentive mechanism design in cloud resource allocation.
- We first consider the migration-admissible model where jobs can be executed on different servers simultaneously and migrated to different servers over time. We greedily allocate the resource over time to the jobs in the virtual allocation while in the rounding procedure we match the profit of the jobs to the profit of the partitioned jobs to enforce the optimality of the users. Our proposed mechanism achieves the truthfulness for general monotonic profit functions, i.e. the users maximize their utilities when truthfully bidding their job lists. Moreover, theoretical analysis shows that the mechanism guarantees a constant approximation to the optimal social welfare for linear profit functions, i.e. the social welfare achieved by

the mechanism is always within a constant ratio of the optimal solution.
- We further consider the non-migration model where jobs can only be executed at one server at a time and over time. To deal with the non-migration constraints, we adopt the traditional list scheduling as the virtual allocation and develop a novel matching scheme and a charging scheme to analyze the performance of the mechanism. Our proposed mechanism achieves the truthfulness for general monotonic profit functions and meanwhile a constant approximation on the social welfare for linear profit functions.
- We further perform simulations on our proposed mechanisms and validate that the average performances are closer to the optimal solution than the constant ratios in our theoretical analysis.

The organization of the paper proceeds as follows. Section II presents the problem formulation. Section III proposes a mechanism for the migration-admissible model. The non-migration model is investigated in Section IV. The simulation results are presented in Section V. Finally, we discuss the related work in Section VI and conclude the paper in Section VII.

II. PRELIMINARIES

In this section, we introduce the system model first, and then define user utility and social welfare. We consider one cloud service provider (e.g. a large data center) with Q units of resource. The cloud service provider accepts the resource requirements from the users periodically and allocates the resource in the coming round according to its scheduler strategy to the users.

A. INFORMATION REPORTED FROM THE USERS

Each cloud user holds multiple jobs and requests the resource from the cloud service provider by reporting her request. Let S_j be the set of jobs held by user j. Suppose that a job $i \in S_j$ needs s_i units of resource and has value/weight w_i.

As demonstrated in the example in introduction, a user with multiple jobs can misreport her job list in one round to gain unfair advantages. The job list/membership information S_j, held by user j, is almost impossible to be detected by the cloud service provider since unreported jobs cannot be identified. While for a specific job, the actual resource requirement is easier to be detected since the job would be uploaded and executed in the cloud. Thus, we assume that the job list of each user is private information while the size of resource needed for a job s_i and the value/weight w_i are public information.

A user bids her jobs, denoted by B_j, to the cloud service provider and may strategically bid $B_j \neq S_j$ to change the allocation of the cloud service provider. The bidding strategy of users is a combinatorial structure and each user has an exponential number of bidding strategies.

B. RESOURCE ALLOCATION OF THE CLOUD PROVIDER

The cloud service provider periodically collects the bids of job requests from the users at the beginning of each round and decides the allocation $A(B_j, B_{-j})$ in time interval $[1, T]$ of the coming round according to the reported requests (B_j, B_{-j}), where B_{-j} represents the vector $(B_1, \ldots, B_{j-1}, B_{j+1}, \ldots, B_n)$ of job sets reported from the users excluding j.

We consider two models, namely, the migration-admissible model and the non-migration model, respectively, to formulate the constraints of resource allocation.

In the migration-admissible model, jobs can be executed at different servers simultaneously and migrated between different servers over time (all Q units of resource at the same time are indistinguishable and allowed to be assigned to the same job over time with migration), as formulated in [23], [25] for centralized resource pool. Assume that the allocation $A(B_j, B_{-j})$ assigns $q(t, i)$ units of resource at time t to job i. A feasible schedule should satisfy the *capacity constraints* that the total units of resource assigned to the jobs at any time t is at most Q. That is,

$$\sum_i q(t, i) \leq Q, \quad t \in [1, T].$$

A job i is *completed* at time $c_i(A(B_j, B_{-j}))$ (which will be written as c_i if there is no ambiguity) if it is assigned its required s_i units of resource at time c_i. That is,

$$\sum_{1 \leq t \leq c_i - 1} q(t, i) < s_i \ \& \ \sum_{1 \leq t \leq c_i} q(t, i) \geq s_i.$$

In the non-migration model, jobs can only be executed at one server at a time and over time. We consider m identical servers, each of which has q units of resource and $Q = mq$ in total. Assume that the allocation assigns $q(t, i, u)$ units of the resource at time t from the same server u to job i. Correspondingly, a feasible schedule should satisfy the *capacity constraints* that

$$\sum_i q(t, i, u) \leq q, \quad \forall u \in [1, m], \forall t \in [1, T]$$

and the *non-migration constraints* that any job i is not assigned to more than two servers with $q(t, i, u) > 0$. A job is completed at time c_i if

$$\sum_{1 \leq t \leq c_i - 1} q(t, i, u) < s_i \ \& \ \sum_{1 \leq t \leq c_i} q(t, i, u) \geq s_i.$$

C. UTILITY FUNCTIONS AND SOCIAL WELFARE

The profit of a job depends on how much time it is finished by time T and the weight of the job. Thus, we define the profit of job i to be $F(c_i, w_i) = w_i f(T - c_i)$. $F(c_i, w_i)$ is called *monotonic profit function* if $w_i f(T - c_i)$ is monotonically decreasing with c_i and *linear profit function* if $F(c_i, w_i) = w_i(T - c_i)$.

The utility $u_j(B_j, B_{-j})$ of a user j is the sum of the profit obtained from her jobs,

$$u_j(B_j, B_{-j}) = \sum_{i \in S_j} w_i f(T - c_i(A(B_j, B_{-j}))). \quad (1)$$

The social welfare is the total profit gained from the users,

$$\sum_j u_j(B_j, B_{-j}) = \sum_j \sum_{i \in S_j} w_i f(T - c_i(A(B_j, B_{-j}))). \quad (2)$$

D. TRUTHFULNESS AND APPROXIMATIONS ON THE SOCIAL WELFARE

We aim at designing a non-payment mechanism that is polynomial time tractable to stimulate the users to behave truthfully and maximize the social welfare. The *truthfulness* of a non-payment mechanism is defined as follows, which is an outcome satisfying the equilibrium and economic incentives.

Definition 1 (Truthfulness Without Money Transfer): A non-payment mechanism is truthful if every user maximizes her utility by truthfully bidding her set, regardless of the bids of other users. That is,

$$u_j(S_j, B_{-j}) \geq u_j(B_j, B_{-j}), \quad \forall j, \ \forall B_j \neq S_j, \ \forall B_{-j}. \quad (3)$$

Besides the truthful requirement for the users, the other critical requirement is the performance on the social welfare which is measured by the worst-case approximation ratio defined as follows.

Definition 2 (Approximation): A mechanism A is γ-approximation if the social welfare is at least $\frac{1}{\gamma}$ of the optimal solution for all possible inputs. That is,

$$\sum_j u_j(B_j, B_{-j}) \geq \frac{1}{\gamma} OPT, \ \forall (B_j, B_{-j}) \quad (4)$$

where OPT is the profit of the optimal solution.

III. INCENTIVE MECHANISM FOR MIGRATION-ADMISSIBLE MODEL

In this section, we consider the migration-admissible model and design a non-payment truthful mechanism with a constant approximation. We will first prepare a basic mechanism, named Basic-Mech, for a simplified case in Section III-A and then develop a mechanism for the migration-admissible model in Section III-B by adopting Basic-Mech as a building block.

A. A BASIC PROCEDURE

In this section, we consider the special case of $Q = 1$ where only one unit of resource at each time can be allocated to a job. We propose a basic procedure, referred to as Basic-Mech, which would be adopted as a building block later.

The high level idea is as follows. On one hand, to ensure the good approximation to the optimal solution, we design a virtual allocation to assign the jobs by a greedy rule with high weight-per-size first. However, such a greedy rule itself would fail in eliciting the truthful behaviors. The reason mainly lies in two facts: 1) a user gains her profit from her jobs only when the resource requirements are integrally satisfied, and 2) a user untruthfully reporting her jobs may advance the jobs' processing priority to make some jobs contribute more profit. Observing this, on the other hand, we then perform a randomized rounding/allocating procedure to stimulate the truthfulness, while guaranteeing the feasibility and preserving the approximation performance on the social welfare. To enforce the truthfulness, our key idea is to ensure that, even when one unit or a part of the resource requirement of a job is granted in the virtual allocation, the user gains an expected profit from that job, which thus ensures the maximum utility of each user when she bids her true information.

Note that the proposed mechanisms in later sections will also follow such a two-step framework (consisting of a virtual allocation and a rounding procedure). However, different allocation/rounding methods should be derived to address different allocation constraints in different models.

Mechanism 1 presents the detailed design of Basic-Mech. The jobs reported from the users, denoted by S, are partitioned into two sets $X_1 = \{i : s_i \leq \tau\}$ and $X_2 = S \backslash X_1$. Here, τ is a parameter to guide the allocation, which will be carefully chosen to ensure the feasibility of the allocation, the truthfulness, and also a good social welfare. Jobs in X_1 are then tackled by a virtual allocation and a rounding procedure. In the virtual allocation, the resource in period $[1, \tau]$ is greedily allocated to the jobs according to the order of non-increasing weight-per-size $\frac{w_i}{s_i}$. Note that the virtual allocation is profit-maximum for the unit-jobs in the partitioned instance where each job i is partitioned into s_i unit-jobs with size one and weight $\frac{w_i}{s_i}$. The virtual allocation gains a good profit but fails to be truthful. Thus, we perform a rounding procedure to induce the truthfulness. When a unit of resource is assigned to a job, we say that one *piece* of a job is satisfied. Let $J(i, k)$ be the time slot in which job i's k-th piece is satisfied in the virtual allocation. In the rounding procedure, each job is finished with probability $\frac{1}{s_i}$ at time slot $J(i, k) + \tau$ where $1 \leq k \leq s_i, J(i, k) \leq \tau$. That is,

$$Pr(c_i = J(i, k) + \tau) = \frac{1}{s_i}.$$

This is achieved by granting the available resource in time period $[1, J(i, k) + \tau]$ to job i to satisfy the residual $s_i - k$ pieces of job i when the event $c_i = J(i, k) + \tau$ happens. By doing this, when job i is granted a unit of resource in the virtual allocation (e.g. at time $J(i, k)$), the user can gain an expected profit (e.g. $Pr(c_i = J(i, k)+\tau) \cdot F(J(i, k)+\tau, w_i) = \frac{1}{s_i} w_i f(T - J(i, k) + \tau)$).

To ensure the feasibility and also a good approximation on the social welfare, we set $\tau = \lfloor \frac{T}{3} \rfloor$ (which implies $3\tau \leq T \leq 3\tau + 2$). With the choice of τ, even when all the first $i - 1$ jobs together with job i are moved earlier to be finished before $J(i, k) + \tau$, the total size $\sum_{j<i} s_j + s_i$ is at most $J(i, k) + \tau$ for any job i since $\sum_{j<i} s_j \leq J(i, k)$ and $s_i \leq \tau$. This verifies the feasibility of the algorithm. Note that choosing a small τ may affect the feasibility of the rounding procedure, while choosing a large τ may reduce the social welfare since the jobs are postponed and completed later in expectation.

Since the mechanism outputs a randomized allocation, the utility of a user in Eq. 1 will be measured by the expected

Algorithm 1 Basic-Mech

1: Partition the jobs into two sets, $X_1 = \{i : s_i \leq \tau\}$ and $X_2 = S \backslash X_1$ where $\tau = \lfloor \frac{T}{3} \rfloor$. Jobs X_1 are sorted so that the ratio w_i/s_i is non-increasing as the index i increases. Let i^* be the job with maximum value $w_i f(T - s_i)$ for the jobs in X_2.
2: With probability 1/2, run the following procedure on set X_1.
3: (Virtual Allocation) While there is resource remaining in time period $[1, \tau]$, greedily allocate the resource and satisfy the requirement of jobs according to the order of non-increasing w_i/s_i till all resource/jobs are processed to get a virtual allocation GA. Let $J(i, k)$ be the time in which job i's k-th piece is satisfied where $1 \leq k \leq s_i$, $1 \leq J(i, k) \leq \tau$.
4: (Rounding Procedure) Perform the rounding so that each job i is finished with probability $\frac{1}{s_i}$ at time $J(i, k) + \tau$ where $1 \leq k \leq s_i$, $J(i, k) + \tau \leq T$, by satisfying/moving the remaining $s_i - k$ pieces of job i in/to the available time before $J(i, k) + \tau$ as early as possible.
5: With probability 1/2, run the following procedure on set X_2.
6: Finish job i^* by assigning the resource in time period $[1, s_{i^*}]$.

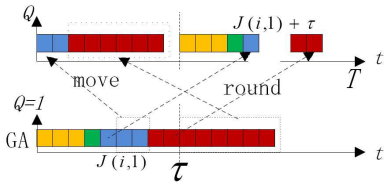

FIGURE 2. An example that shows the virtual allocation GA and rounding procedure in the mechanism Basic-Mech. $J(i, k)$ $(1 \leq k \leq s_i)$ is the time in which the k-th piece of job i is satisfied in virtual allocation GA. Job i is rounded to complete at time $J(i, 1) + \tau$ with probability $\frac{1}{s_i}$ by satisfying/moving the residual $s_i - 1$ pieces of job i in advance using the available resource in time period $[1, J(i, 1) + \tau]$.

profit obtained from her jobs, which equals

$$\sum_{i \in S_j} \mathbf{E}(w_i f(T - c_i)) = \sum_{i \in S_j} \sum_{k:1 \leq k \leq s_i, J(i,k) \leq \tau} \frac{w_i}{s_i} f(T - J(i, k) + \tau).$$

We prove the truthfulness of the proposed mechanism in the following theorem. The detailed proof can be referred to in Appendix.

Theorem 1: Mechanism Basic-Mech *is truthful for monotonic profit functions.*

Note that the proposed mechanism is truthful for general monotonic functions. Now we further verify its efficiency on the performance of social welfare. We will focus on the linear profit function, which is a more concrete but representative profit function. We show that the proposed mechanism guarantees a constant approximation. The proof is moved to Appendix.

Theorem 2: Mechanism Basic-Mech *guarantees an* $O(1)$*-approximation for linear profit functions.*

B. MECHANISM FOR MIGRATION-ADMISSIBLE MODEL

Now we are ready to design an incentive mechanism for the migration-admissible model where jobs could be executed at different servers simultaneously and migrated to different servers over time.

Even regardless of the truthfulness, computing the optimal allocation to maximize the social welfare in this model is NP-hard. This can be verified by setting $T = 2$ with linear profit function where the social welfare in such a setting equals the total weight of jobs that are satisfied in the first time slot with resource capacity Q, which corresponds to the well-known knapsack problem that is NP-hard [27]. Therefore, we should derive a polynomial-time tractable algorithm to simultaneously achieve a good social welfare and the truthfulness.

One natural idea is to apply the following Greedy-Mech: while there is resource remaining, greedily assign Q units of resource at each time slot to the jobs according to the order of non-increasing weight-per-size w_i/s_i. However, the greedy strategy itself is not truthful, which can be easily seen from an example similar to the one in the introduction.

Although the greedy strategy fails to be truthful, by developing a rounding procedure, we propose a mechanism MGT-Mech that will be proved to be truthful. The high level idea is as follows. We partition the jobs submitted to the system into two sets, the ones with larger size and the ones with smaller size. For the large-size jobs, we directly apply the basic procedure Basic-Mech to enforce the truthfulness, while for the small-size jobs, we follow the two-step framework of Basic-Mech and design a new virtual allocation and a corresponding rounding procedure to ensure that a user could gain a (maximum) profit from each unit of resource required by her jobs, which would guarantee the optimality of truthful bidding and thus stimulates the truthfulness.

In detail, MGT-Mech works as follows. The set of jobs are partitioned into two sets $X_1 = \{i : s_i \leq Q'\}$ and $X_2 = \{i : s_i > Q'\}$ where we set $Q' = \max\{\lfloor \frac{Q}{3} \rfloor, 1\}$. The choice of Q' will ensure the feasibility in the rounding procedure. First, for the jobs in X_1, as shown in Step 2-4, we apply Greedy-Mech based on a modified capacity Q' to obtain a virtual allocation GA: while there is resource remaining in time period $[1, T]$, we greedily assign Q' units of resource at each time to the jobs according to the order of non-increasing $\frac{w_i}{s_i}$ utill all resource/jobs are processed. Let job i gets s_i^1 and s_i^2 units of its required resource respectively at the two adjacent time slots c_i^1 and $c_i^2 = c_i^1 + 1$. Then, we design a rounding procedure to ensure that a user can gain a (maximum) profit from each

Algorithm 2 MGT-Mech

1: Partition the jobs into two sets $X_1 = \{i : s_i \le Q'\}$ and $X_2 = \{i : s_i > Q'\}$ where $Q' = \max\{\lfloor \frac{Q}{3} \rfloor, 1\}$. Jobs in set X_1 are sorted so that the ratio w_i/s_i is non-increasing as the index i increases.
2: With probability 1/2, run the following procedure on set X_1.
3: (Virtual Allocation) While there is resource remaining in time period $[1, T]$, greedily assign Q' units of resource at each time to the jobs according to the order of non-increasing w_i/s_i till all resource/jobs are processed. Denote the resulting virtual allocation to be GA. Let job i get its s_i^1 units of required resource at time c_i^1 and s_i^2 units at time $c_i^2 = c_i^1 + 1$.
4: (Rounding Procedure) Perform the rounding so that job i is completed at time c_i^1 with probability $\frac{s_i^1}{s_i}$ and completed at time c_i^2 with probability $\frac{s_i^2}{s_i}$ by using the remaining $Q - Q'$ units of resource at each time.
5: With probability 1/2, run the following procedure on X_2.
6: Refine the size and weight of job i to be $\bar{s}_i = \lceil \frac{s_i}{Q} \rceil$, $\bar{w}_i = w_i$. Reorder the index of the jobs in X_2 to be $i' = \{1, 2, 3...\}$ according to the order of non-increasing value $\frac{\bar{w}_i}{\bar{s}_i}$.
7: Run Basic-Mech on jobs in X_2 with refined sizes and weights.

unit of its source required by her jobs and achieve the optimal profit when bidding truthfully. In the rounding procedure, each job i is completed at time slot c_i^1 with probability $\frac{s_i^1}{s_i}$ and completed at time slot c_i^2 with probability $1 - \frac{s_i^1}{s_i}$. The expected profit gained from job i after rounding would be $f(T - c_i^1) \cdot w_i \frac{s_i^1}{s_i} + f(T - c_i^2) \cdot w_i(1 - \frac{s_i^1}{s_i})$. For the remaining jobs in X_2, we refine the size (and accordingly the weight) of job i to be an integer $\lceil \frac{s_i}{Q} \rceil$ and apply Basic-Mech with the input of refined sizes and weights.

The design of the mechanism has the following advantages. With size $s_i \le Q'$, every job in X_1 gets all its required resource from at most two time slots. This implies that in the rounding procedure in Step 4, at most two extra jobs' requirement has been fulfilled after rounding in each time slot. Thus, by the choice of Q' where $2Q' + \lfloor \frac{Q}{3} \rfloor \le Q$, the rounding procedure always returns a feasible solution. Moreover, the profit obtained from jobs in X_2 after rounding is close to the maximum achievable profit from that set with capacity Q as the input.

For the remaining jobs in X_2, since each job has a large size $s_i > Q'$ and can be satisfied Q units of resource each time slot, it can be refined to be $\lceil \frac{s_i}{Q} \rceil$ without losing much accuracy. Then, we can apply Basic-Mech to the refined instance to induce the truthfulness and gain a good profit. The feasibility of Step 3 follows from the correctness of Basic-Mech.

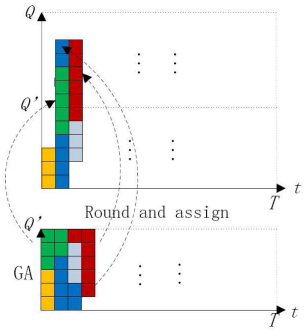

FIGURE 3. An example that shows the rounding procedure in the migration-admissible model. The red (and blue) job is completed with probability $\frac{1}{6}$ (and $\frac{2}{3}$) to be completed at time 3 (and time 2).

Now we prove that the proposed mechanism is truthful for general monotonic profit functions.

Theorem 3: Mechanism MGT-Mech *is truthful for the migration-admissible model with monotonic profit functions.*

Proof: We need to examine the allocations for jobs in X_1 and X_2 in MGT-Mech, respectively.

Consider the allocation for jobs in X_1 first. The size of any job in X_1 is at most Q' and job i gets its s_i^1 units of resource at time c_i^1 and s_i^2 units of resource at time $c_i^2 = c_i^1 + 1$.

We examine the following partitioned instance. We partition job i in GA into exactly s_i unit-jobs and each unit-job has a weight $\frac{w_i}{s_i}$ and size one. The profit from the unit-jobs partitioned from job i in the allocation GA is $f(T - c_i^1) \cdot s_i^1 \frac{w_i}{s_i} + f(T - c_i^2) \cdot (s_i - s_i^1) \frac{w_i}{s_i}$, which exactly equals the expected profit in the rounding procedure. Hence, to analyze the truthfulness in the rounding procedure, it is sufficient to verify the truthfulness of the users who hold unit-jobs in the partitioned case.

Consider first that user j bids a subset $\hat{S}_j$ with $\hat{S}_j \subset S_j$. We examine two cases. In the first case, assume that all the jobs are held by one user j. The total profit of the unit-jobs is exactly $\sum_{i \in S_j, i \in X_1} f(T - c_i^1) \cdot w_i \frac{s_i^1}{s_i} + f(T - c_i^1) \cdot w_i(1 - \frac{s_i^1}{s_i})$ where a unit-job with a larger weight ratio is assigned earlier. Such a profit in GA is the maximum profit to assign all unit-jobs over time with capacity Q' among all possible reordering of the unit-jobs. Some jobs are not revealed in such a case. When unrevealing the unit-jobs that are not assigned before time T in GA, the allocation GA for both the true bid S_j and the false bid $\hat{S}_j$ are the same and incurs the same profit. When the unit-jobs completed before time T in GA are unrevealed, the time assigned to the unit-job with weight ratio $\frac{w_i}{s_i}$ in the true bidding is occupied by another unit-job with a smaller weight ratio in the false bidding. The total profit generated in the true bidding is thus at least that of the false bidding. This verifies the truthfulness in the first case. In the second case, when the jobs are held by multiple users, the profit would be further decreased, since the time slot may be occupied by another unit-job which is not held by the user, making the

user lose more profit. Therefore, the users have no incentives to unreveal her jobs.

Then, consider that the user reports a job i that does not belong to her. Since the virtual allocation assigns the resource according to the order of non-increasing value $\frac{w_i}{s_i}$, unit-jobs partitioned from other jobs either have the same completion time or are postponed to be finished. Thus, the user gains no more profit from unit-jobs that belong to her. She also gains no profit from job i since the utility of user j is the total profit of jobs listed in her true membership. This implies that the users also have no incentives to report jobs that do not belong to them.

Therefore, user j has no incentive to misreport for the input of partitioned instance. Furthermore, the expected profit of the user in the allocation matches the profit of the unit-jobs in the partitioned instance. User j maximizes her expected profit when bidding truthfully, and hence the allocation for jobs X_1 is truthful.

For jobs in X_2, we apply Mechanism Basic-Mech with the the input virtual sizes and weights. Recall that the users have no incentive to lie by the truthfulness of Basic-Mech. Accordingly, it is also truthful for the allocation for the jobs in X_2. Therefore, MGT-Mech guarantees the truthfulness. ■

Remark: The truthfulness relies on the virtual allocations and rounding procedures respectively performed on jobs with large sizes (X_2) and small sizes (X_1). The high level idea to induce the truthfulness is that, 1) the virtual allocation is the optimal allocation for the unit-jobs in the partitioned instance, and 2) the rounding procedure preserves the fact that users who behave truthfully would maximize their utility gained from the unit-jobs in the partitioned instance after rounding.

Furthermore, we prove that the combination of the two steps in the proposed mechanism can guarantee good social welfare.

Theorem 4: Mechanism MGT-Mech *guarantees an $O(1)$-approximation for the migration-admissible model with linear profit functions.*

Proof: Let $ALG(X_1)$ and $ALG(X_2)$ be the expected profit obtained by the allocation of jobs X_1 and X_2. Let $OPT(X_1, Q, T)$ (and $OPT(X_2)$) be the optimal profit to allocate the jobs in X_1 (and X_2) in time period $[1, T]$ with capacity Q each time.

Consider jobs in X_1 first where all $1 \leq s_i \leq Q'$. We take as if job i is divided into s_i virtual unit-jobs with size one and weight $\frac{w_i}{s_i}$. Denoted by $w_t^{Q'}(G)$ the total weight of unit-jobs that are assigned to time slot t with resource capacity Q' on an allocation G. Obviously, the expected profit $ALG(X_1) = \sum_t w_t^{Q'}(GA) \cdot (T - t)$. Let O be the optimal allocation for the unit-jobs. Then, $\sum_t w_t^{Q}(O) \cdot (T - t)$ is the optimal profit for the partitioned unit-jobs, which is an upper bound of the optimal profit $OPT(X_1, Q, T)$. Since the unit-jobs are assigned according to the greedy rule in order of non-increasing ratio $\frac{w_i}{s_i}$, we have $w_t^{Q'}(GA) \geq \frac{1}{6} w_t^{Q}(O)$ by the fact that the capacity Q' at each time slot in GA is at lease $\frac{1}{6}Q$ (since $2\lfloor \frac{Q}{3} \rfloor \geq \frac{Q}{3}$ when $Q \geq 3$ and $Q' \geq 1$). Therefore, $ALG(X_1) \geq \frac{1}{6} OPT(X_1, Q, T)$.

Then, consider the jobs in X_2. The jobs in X_2 have size at least $\lceil \frac{Q}{3} \rceil$ and job i has its size s_i rounded to $\bar{s}_i = \lceil \frac{s_i}{Q} \rceil$. Job i occupies $\bar{s}_i$ time slots when applying Basic-Mech in Step 7. Let X_2' be another set of jobs where each job i' is generated with size $s_i' = \lceil \frac{s_i}{Q} \rceil Q$ and weight $w_i' = \frac{w_i}{s_i} \lceil s_i' \rceil$. We have $3s_i \geq s_i'$ (this can be verified by writing $s_i = Qp + q$ where $q \geq \lceil \frac{Q}{3} \rceil$ when $p = 0$ and $q \geq 1$ when $p \geq 1$). If we had applied Basic-Mech to X_2' with the same refinement process as in Step 7, the rounded size $\bar{s}_i' = \lceil \frac{s_i'}{Q} \rceil$ would also equal $\bar{s}_i$, thus job i' would be assigned to occupy $\bar{s}_i' = \bar{s}_i$ time slots. Denoted by $A(S)$ the profit returned by applying Basic-Mech to set S. Clearly, $ALG(X_2) = A(X_2)$. When applying Basic-Mech, the expected profit generated in Step 2-4 of Basic-Mech corresponds to the total profit of the unit-jobs in the partitioned instance in the algorithm; moreover, the jobs in Step 5-6 have the same rounded size ($\bar{s}_i' = \bar{s}_i$). In addition, each job i in X_2 is expanded to size $s_i' \leq 3s_i$ while keeping the ratio unchanged ($\frac{w_i'}{s_i'} = \frac{w_i}{s_i}$). Thus, we have $A(X_2) \geq \frac{1}{3} A(X_2')$. Let $FOPT(X_2')$ and $FOPT(X_2)$ respectively be the optimal profit of the unit-jobs generated by partitioning the jobs in X_2' and X_2. We have $A(X_2') \geq O(1) FOPT(X_2')$ by the approximation of Basic-Mech. Clearly, $FOPT(X_2') \geq FOPT(X_2) \geq OPT(X_2)$ where the last inequality holds because the profit gained from the partitioned unit-jobs is an upper bound of the profit achievable for the original jobs. Thus, $ALG(X_2)) \geq \frac{1}{3} A(X_2') \geq O(1) OPT(X_2)$.

Finally, by combining the results, we have $ALG = \frac{1}{2}(ALG(X_1) + ALG(X_2)) \geq O(1) \cdot (OPT(X_1, Q, T) + OPT(X_2)) \geq O(1) \cdot OPT$. This shows the constant approximation of the mechanism. ■

IV. INCENTIVE MECHANISM FOR NON-MIGRATION MODEL

We design a truthful mechanism for the non-migration model in this section. The jobs have to be assigned without migration to m servers where each server has q units of resource and $Q = mq$ in total. This new constraint brings us the difficulty in maintaining the truthfulness.

A. MECHANISM DESIGN

Interestingly, we can adopt the traditional list scheduling [10] in the scheduling literature as the virtual allocation and design a rounding scheme to enforce the truthfulness. The original list scheduling greedily allocates the jobs one by one to the server that incurs the smallest completion time in the current allocation. It was proposed to fairly allocate the jobs with respective lengths of execution time and minimize the maximum completion time of machines.

The general idea of our design is as follows. We observe that the list scheduling can be applied to our setting to develop a virtual allocation that guarantees a high total profit gained from the jobs. Then, based on this, we propose a corresponding rounding scheme to elicit the truthfulness.

Algorithm 3 NMGT-Mech

1: Partition the jobs into two sets, $X_1 = \{i : s_i \le \tau q\}, X_2 = S \backslash X_1$. Jobs X_1 are sorted so that the ratio $\frac{w_i}{s_i}$ is non-increasing.
2: With probability $\frac{1}{2}$, run the following allocation and rounding procedure on set X_1.
3: (Virtual Allocation) Run list scheduling (by assigning job i according to the order of non-increasing ratio $\frac{w_i}{s_i}$ to the server that incurs the smallest completion time for job i) to get a virtual allocation $LS(\tau)$ with total τ available time. Let $J(i, u, k)$ be the time slot to which job i's k-th piece is assigned on server u and be 0 if it is not assigned to server u.
4: (Rounding Procedure) Perform the following rounding scheme to the allocation LS. For every job i, if $J(i, u, k) > 0$, then with probability $\frac{1}{s_i}$ make job i be executed on the server u and finished at time slot $J(i, u, k) + \tau$).
5: With probability $\frac{1}{2}$, run the following allocation on set X_2.
6: Pick out the jobs that have the first m largest value $w_i f(T - s_i)$. Finish each of the m jobs at a server.

The detailed design of our proposed mechanism is presented in Mechanism NMGT-Mech. The jobs are partitioned into two sets $X_1 = \{i : s_i \le \tau q\}, X_2 = S \backslash X_1$ where τ will be discussed later. We apply list scheduling for jobs in X_1. Let LS be the non-migration allocation returned by the list scheduling algorithm in time period $[1, \tau]$. Let $J(i, u, k)$ be the time to which job i's k-th piece is assigned on server u and be 0 if it is not assigned to server u. For every job i, if $J(i, u, k) > 0$, then the rounding procedure makes job i be executed on server u and finish at time slot $J(i, u, k) + \tau$ with probability $\frac{1}{s_i}$.

B. PERFORMANCE GUARANTEE

Now we show the theoretical performance guarantee of the proposed mechanism.

To simplify the presentation, the following analysis will focus on the case $q = 1$, and it can be directly generalized to any value q. In the allocation LS, assume that job i is allocated to the lowest available time slot a_i (also called the starting time of job i) on server u_i and hence finishes at time b_i. Denoted by $p(u_i, a_i, b_i)$ the space/time slot that is occupied by job i in LS. Fig. 5 shows an example for the allocation LS and the rounding procedure.

We first prove the truthfulness of the proposed mechanism. The rounding procedure achieves the following expected profit function $g_i(LS)$ for the jobs in LS. Let $LS(i)$ be the allocation of jobs $1, 2, \ldots, i$ in X_1 in LS. In the rounding procedure, job i is completed at time $t + \tau$ where $t \in [a_i, \min\{b_i, \tau\}]$ with probability $\frac{1}{s_i}$. Thus the expected profit is

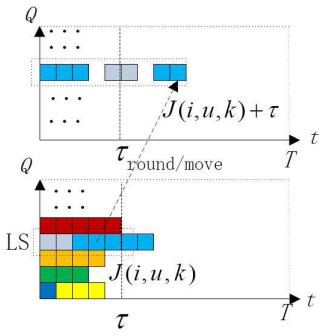

FIGURE 4. An example to show the virtual list scheduling LS and the corresponding rounding procedure. The job assigned on server u is rounded to get the resource on the same server.

$g_i(LS) = \sum_{a_i \le t \le \min\{b_i, \tau\}} \frac{1}{s_i} w_i f(T - t - \tau) = \sum_{a_i \le t \le \min\{b_i, \tau\}} \frac{w_i}{s_i} f(T - t - \tau)$. Let LS, LS' respectively be the allocation when user j bids her true set S_j and false set $\hat{S}_j$. To show the truthfulness, we need to prove $\sum_{i \in S_j} g_i(LS) \ge \sum_{i \in S_j} g_i(LS')$ for any j and any set $\hat{S}_j \ne S_j$. Our method is to develop a matching scheme to characterize the movement of the jobs in the truthful biding and false bidding so that we can examine the profit change of the bidding. The result is stated in the following theorem and the detailed proof can be referred to in Appendix.

Theorem 5: Mechanism NMGT-Mech *is truthful for the non-migration model with monotonic profit functions.*

Remark: The intuition of the truthfulness is that some of the unit-jobs in the partitioned instance are moved in the allocations LS, LS' before and after misreporting. We characterize the movement from LS to LS' in terms of the unit-jobs partitioned from a job (instead of that job itself) and average the profit of the job over all the carefully chosen unit-jobs in the movement. Based on these operations, we can find the movements that form a chain and compare the profit change of different bidding strategies on that chain.

Now we show the approximation of the mechanism.

Our method is to develop a marking/charging scheme between our allocation and the optimal solution to compare their outputs. In order to obtain a lower bound of the profit obtained from the mechanism in the charging scheme, we set $\tau = \frac{1}{4}T$ and make the jobs after rounding be finished before time $\frac{T}{2}$ by the fact $J(i, u, k) + \tau \le 2\tau \le \frac{T}{2}$. For the ease of presentation, we only consider the setting that τ is an integer without examining the floor operation as introduced in the previous sections. Note that with the choice of τ, the rounding procedure always returns a feasible solution since at most τq pieces of the jobs are moved to time period $[1, \tau]$. The result is stated in the following theorem and the detailed proof is presented in Appendix.

Theorem 6: Mechanism NMGT-Mech *guarantees an $O(1)$-approximation for the non-migration model with linear profit functions.*

Remark: The high level idea of the proof is to examine the unit-jobs in the partitioned instance and observe the fact

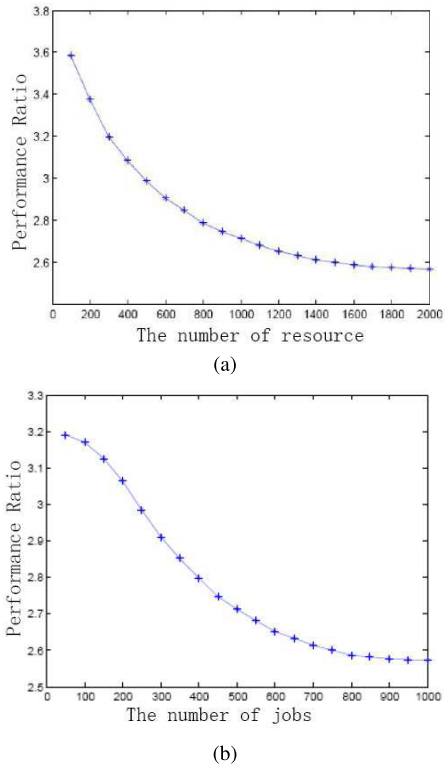

FIGURE 5. **Performances of Mechanism** MGT-Mech**, which is measured by the ratio between the total profit achievable in the upper bound of the optimal solution and that of our mechanism. (a) Influence of resource number in the migration-admissible model. (b) Influence of job number in the non-migration model.**

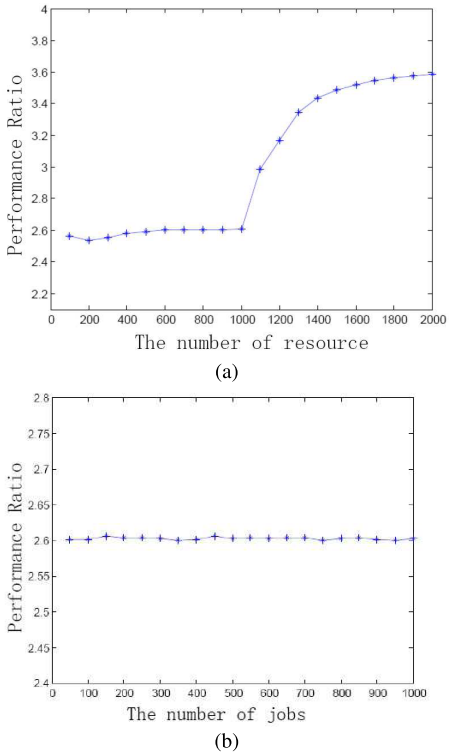

FIGURE 6. **Performances of Mechanism** NMGT-Mech**. (a) Influence of resource number in the migration-admissible model. (b) Influence of job number in the non-migration model.**

that unit-jobs not assigned in time period $[0, \tau]$ *have smaller weights. These allow us to develop a charging scheme, which properly matches the unit-jobs that are assigned in the optimal solution but not in the proposed mechanism to the ones in the proposed mechanism, and accordingly bound the profits between the optimal solution and our proposed mechanism.*

V. SIMULATION RESULTS

We have theoretically proved the performance guarantee of the proposed mechanisms. The goal of this section is to further validate the average performance of the mechanisms on maximizing the social welfare. Considering the fairness, we have not compared with other allocation algorithms since no prior works have provided truthful mechanisms the same as our non-payment setting. So we compare our proposed mechanisms to the upper bounds of the optimal solutions, which derived in our approximation analysis, considering that computing the optimal solution is NP-hard.

We first examine the performance of Mechanism MGT-Mech for the migration-admissible model. The performance will be measured by the ratio between the total profit achievable in the upper bound of the optimal solution and that of our mechanism. Each job size is uniformly sampled from 100 to 1500, and correspondingly its weight is sampled from 1 to 2000. The number of time slots T is set to be 100. Each point in the figures is a mean value of 1500 random instances. We start by examining the influence of the number of resource Q to the performance. The number of jobs is set to be 500. It can be seen in Fig. 5(a) that the performance ratio varies between 2.6 and 3.6, indicating that the total profit achieved by our mechanism is close to the optimal soltion. Then we examine the influence of the number of jobs to the performance ratio. The number of resource is set to be $Q = 1000$. It can be seen in Fig. 5(b) that the performance ratio is decreasing and then stabilizes at around 2.57 as the number of jobs increases.

The average performance of the mechanism is even better than the theoretical guarantee derived in analysis.

Then we examine the performance of Mechanism NMGT-Mech for the non-migration model. Each job size is uniformly sampled from 1 to 1500, and correspondingly its weight is uniformly distributed from 1 to 2000. The number of time slots T is set to be 100 and the number of servers is $m = 20$. We first examine the influence of the number of resource to the performance ratio. The number of jobs is set to be 1000. It can be seen in Fig. 6(a) that the performance is slowly increasing at the beginning and then stabilizes at around 3.6. The intuitive explanation is that the profit contribution of the jobs in the algorithm increases slower than the optimal solution with non-migration constraints at the beginning. Then we examine the influence of the number of jobs to the performance ratio. The number of resource is set to be $Q = 1000$. It can be seen that in Fig. 6(b) the performance

ratio stably varies between 2.59 to 2.6, which is even better than the theoretical performance guarantee derived in our analysis.

Summarizing our results, the average performances of the mechanisms are close to the optimal solution, which validates the efficiency of our proposed mechanisms.

VI. RELATED WORK

Much research attention has been invested on optimizing the resource allocation in cloud computing. Since it is impossible to summarize all the work, we just review the related ones from the perspective of mechanism design that regulates the strategic behaviours of cloud requests.

The first research direction is to study the pricing mechanisms by fixing the price of the resources and making the cloud users compete for the resource [3], [13]. For example, Abhishek *et al.* [2] investigate the pricing policies to maximize the revenue. Zhao *et al.* [28] develop an efficient online pricing algorithm of VM resource across data centers in a geo-distributed cloud to maximize the profit.

The second research direction is to adopt the equilibrium concepts in the game theory to study the competition behaviors between users and the provider [19]. For example, Pal and Hui [15] examines what price and QoS should be set for multiple cloud providers, showing that a unique pure strategy Nash equilibrium (NE) exists in a cloud market. Tsakalozos *et al.* [19] Guo *et al.* [9] model the fair bandwidth allocation in cloud computing as a Nash bargaining game and design a distributed algorithm to achieve the fairness.

The third research direction is to adopt the paradigm of incentive mechanism design to stimulate the truthful behaviours of the cloud users in competing for the resources [23], [25], [26], which has received more attention. For example, [8], [18], [26] study the truthful combinatorial auction of heterogeneous VMs. Prasad *et al.* [16] address the problem of selecting different bundles of resources from cloud vendors by developing a combinatorial auction mechanism for multiple resource procurement. Jain *et al.* [12] study the truthful mechinism for deadline-sensitive jobs and a user pays according to the value function associated to the completion time of her job. The works in [11] and [21] consider the resource in the centralized pool where the resource is indistinguishable and allowed to be assigned to the same job over time with migration. Zhang *et al.* [25] address the truthful online auctions when users have heterogeneous demands. Wang *et al.* [20], [22] address the task assignment in mobile device clouds by designing truthful auction mechanisms to match the demands and supplies of mobile devices.

Despite the research efforts above, all these mechanisms rely on money transfer/payment function to induce the truthfulness of user behaviours. Such mechanisms, however, fail to be applied in scenarios where money transfer is not realistic, e.g, private clouds with internal enterprises users. In this paper, we examine the non-payment truthful mechanism design problem in cloud computing. What is more challenging, we consider requests from users with multiple jobs. To the best of our knowledge, no prior works have addressed similar problems, and we attempt to shed some light on the non-payment incentive mechanism design in cloud computing.

VII. CONCLUSION

In this paper we conduct the first theoretical work of non-payment truthful mechanism design on the cloud resource allocation. We study both the migration-admissible resource allocation and the non-migration resource allocation. To escape the impossibility results known for non-payment mechanism design in game theory, we develop a two-step framework and propose different virtual allocations and rounding methods to deal with the allocation constraints in different models. The theoretical analysis verifies that our proposed mechanisms achieve the truthfulness of the users for general monotonic profit functions and constant approximation ratios on the social welfare for linear profit functions. Simulation results further validate that the average performances of the mechanisms are close to the optimal solution. Our work has shed some light on the design of non-payment incentive mechanisms in cloud computing.

APPENDIX

A. PROOF OF THEOREM 1

Proof: Consider the allocation for jobs in X_1 first. The expected profit generated from rounding job $1 \leq i \leq i_l$ is $\sum_{k:1\leq k\leq s_i, J(i,k)\leq\tau} w_i f(T - J(i,k) - \tau)\frac{1}{s_i}$. To prove the truthfulness, we consider the case that job i is partitioned into s_i virtual unit-jobs that have size one and weight $\frac{w_i}{s_i}$. The unit-jobs are assigned according to the order of non-increasing value $\frac{w_i}{s_i}$ in GA. When the unit-job is assigned to time slot $J(i,k)+\tau$, the profit gained from the unit-job is $\frac{w_i}{s_i} f(T - J(i,k) - \tau)$, and hence the total profit of the unit-jobs partitioned from job i is exactly $\sum_{k:1\leq k\leq s_i, J(i,k)\leq\tau} f(T - J(i,k) - \tau)\frac{w_i}{s_i}$. Therefore, to show the truthfulness for the users, it is sufficient to show the truthfulness for the case that the users hold the unit-jobs in the partitioned instance.

When a user j bids her true type S_j, the profit gained from the unit-jobs equals $\sum_{i\leq i_l, i\in S_j} \sum_{k:1\leq k\leq s_i, J(i,k)\leq\tau} f(T - J(i,k) - \tau)\frac{w_i}{s_i}$, where the unit-job with larger weight ratio $\frac{w_i}{s_i}$ is finished earlier.

Consider that the user bids a subset $\hat{S}_j$ with $\hat{S}_j \subset S_j$ first. Some jobs, and correspondingly their unit-jobs, are not revealed in such a case. When the unit-job with $J(i,k) > \tau$ in GA is not revealed by the user, the profit gained by the user would keep the same since the allocation GA keeps the same which makes the rounding procedure the same. When the unit-job with $J(i,k) \leq \tau$ in GA is unrevealed by the user, the time slot $J(i,k)$ would be released; another unit-job, that has a weight ratio at most $\frac{w_i}{s_i}$ and originally occupies a time slot after $J(i,k)$, say $J(i',k')$, would occupy the time slot $J(i,k)$. If the user holds i', the change of profit gained from i' is positive and at most $\frac{w_i}{s_i}(f(T - J(i,k) - \tau) - f(T - J(i',k') - \tau))$. Note that $J(i',k')$ then would be occupied by

another unit-job with larger time, say $J(i'', k'')$, which incurs a change of profit at most $\frac{w_i}{s_i}(f(T - J(i', k') - \tau) - f(T - J(i'', k'') - \tau))$. Similar induction would produce a series of change of the profit. The sum of these values would be an upper bound of the total change gained from the user, which is at most $\frac{w_i}{s_i}f(T - J(i, k) - \tau)$. However, the user loses a profit $\frac{w_i}{s_i}f(T-J(i, k)-\tau)$ from the unrevealed job. Therefore, under-reporting the jobs gains no more profit than revealing all her unit-jobs.

On the other hand, for the case of bidding a superset $\hat{S}_j$ with $S_j \subset \hat{S}_j$, the allocation of unit-jobs of user j may be proposed when reporting the jobs that do not belong to her. She gains no profit from those jobs since her utility is the total profit $\sum_{i\in S_j} u_j(A(\hat{S}_j, -S_j))$ of the jobs listed in her true membership.[1] This implies that reporting jobs that do not belong to the user can only decrease the profit for the user herself. Thus, users also have no incentives to report jobs that do not belong to them.

Therefore, the untruthful behavior cannot increase the profit for any user and the allocation for jobs X_1 is truthful. Finally, for the allocation of jobs X_2 that selects the job with maximum value $w_i f(T - s_i)$, we can use similar discussions to verify that untruthfully reporting the jobs cannot increase the profit. This completes the proof. ■

B. PROOF OF THEOREM 2

Proof: Let $ALG(X_1), ALG(X_2)$ respectively be the profit returned on set X_1 and X_2. Let $OPT(X_1, T)$ and $OPT(X_2, T)$ be the optimal profit to allocate the jobs in X_1 and X_2 in T time slots. Consider the allocation of X_2 first. Job i in X_2 has size at least $\lfloor \frac{T}{3} \rfloor$. $OPT(X_2, T)$ can pack at most three jobs in time period $[1, \tau]$. Since the mechanism packs the job i^* with maximum value $w_{i^*}(T - s_{i^*})$ to the first s_i^* time slots, we have $ALG(X_2) = w_{i^*}(T - s_{i^*}) \geq \frac{1}{3}OPT(X_2, T)$.

It remains to prove $ALG(X_1) \geq O(1)OPT(X_1, T)$ for jobs in X_1 where $s_i \leq \tau$. The expected profit of $ALG(X_1)$ could be taken as if every job i is partitioned into s_i unit-jobs and each unit-job has size one and weight $\frac{w_i}{s_i}$. Denoted by weight $\bar{w}_t$ the ratio $\frac{w_i}{s_i}$ if job i is assigned to the time slot t in GA. For the partitioned unit-jobs, the value $\bar{w}_t$ is non-increasing when index t increases, since these unit-jobs are greedily allocated in the virtual allocation. The expected profit gained from the rounding is $ALG(X_1) = \sum_{i\leq i_l}\sum_{k:1\leq k\leq s_i, J(i,k)\leq\tau}(T - J(i, k) - \tau)\frac{w_i}{s_i} = \sum_{t\in[1,\tau]}(T - t - \tau)\bar{w}_t$. On the other hand, the maximum profit among all possible reordering of virtual unit-jobs in time period $[1, T]$, denoted as $FOPT(X_1, T)$, is an upper bound of the optimal solution $OPT(X_1, T)$. For the unit-jobs, we have $FOPT(X_1, T) = \sum_{t\in[1,T]}(T - t)\bar{w}_t$ since $\bar{w}_t$ is non-increasing as t increases. Note that $\sum_{t\in[1,\tau]}(T - t - \tau)\bar{w}_t \geq \sum_{t\in[\tau+1,2\tau]}(T - t)\bar{w}_t$ and $2\sum_{t\in[1,\tau]}(T - t - \tau)\bar{w}_t \geq \sum_{t\in[1,\tau]}(T - t)\bar{w}_t$. Moreover, since $T \leq 3\tau + 2$, $\sum_{t\in[1,\tau]}(T - t - \tau)\bar{w}_t \geq \sum_{t\in[2\tau+1,T]}(T - t)\bar{w}_t$. Hence we have $4ALG(X_1) \geq FOPT(X_1, T) \geq OPT(X_1, T)$.

Combining the results, we have $ALG = \frac{1}{2}(ALG(X_1) + ALG(X_2)) \geq \frac{1}{8}(OPT(X_1, T) + OPT(X_2, T)) \geq O(1) \cdot OPT$. ■

C. PROOF OF THEOREM 5

Proof: We consider the non-trivial procedure for jobs in X_1 and tackle the case with all $b_i \leq \tau$ first. For any two rounding procedures that are applied on the virtual allocations LS and LS' respectively, we first derive the following two basic properties.

- **P1**: If only one job i is moved from time slot t in LS to t' with $t' < t$ in LS' then the expected profit $\sum_i g_i(\cdot)$ increases by $w_i(f(T - t') - f(T - t))$.
- **P2**: If only one job is moved from its current server u to another server u', which makes the lowest available server change from u' to u but keeps the lowest available time still the same, then the allocation for the remaining jobs would incur the same expected profit since the algorithm only cares about the lowest available time instead of the index of the server.

We start by analyzing a simple case that there is only one user in the input. We first focus on the case that the user bids a subset $\hat{S}_j \subset S_j$ and only one job i is unreported. The order of the jobs being assigned in the algorithm excluding i is still the same. For jobs $1, 2, \ldots, i - 1$ that are assigned before i, clearly their allocation are the same in LS and LS' and hence $LS(i-1) = LS'(i-1)$. For the true bidding, all jobs $i+1, \ldots, n$ are assigned one by one and would have starting time at least a_i. When job i is unreported in LS', the space $p(u_i, a_i, a_i + s_i - 1)$ is released and a_i is the lowest available time for job $i + 1$.

Let us examine the effect of the release of space $p(u_i, a_i, a_i+s_i-1)$ and investigate the interval $[a_i, a_i+s_i-1]$. Job $i + 1$ is moved from $p(u_{i+1}, a_{i+1}, a_{i+1} + s_{i+1} - 1)$ in LS to the empty space $p(u_i, a_i, a_i + s_{i+1} - 1)$ in LS'. According the basic properties (P1) and (P2), the profit is increased by $w_{i+1}(f(T-a_i)-f(T-a_{i+1}))$ by the movement of job $i + 1$. We make a marking from each piece of the resource requirement of job $i + 1$, denoted by $p(i + 1)$, in $p(u_{i+1}, a_{i+1} + k, a_{i+1} + k)$ in LS to $p(u_i, a_i + k, a_i + k)$ in LS', and each marking is associated with an average profit $r_{p(i+1)} = \frac{w_{i+1}(f(T-a_i)-f(T-a_{i+1}))}{s_{i+1}}$ in the movement for all $0 \leq k \leq s_{i+1}-1$. Note that in $LS(i+1)'$, the lowest available time on server u_{i+1} is a_{i+1} and the space $p(u_{i+1}, a_{i+1}+k, a_{i+1}+k)$ in LS is released for all $0 \leq k \leq s_{i+1} - 1$ which may be occupied by one of the remaining job j with $j > i + 1$.

Assume that i' is the next job with $i' > i + 1$ in LS' that occupies the released space $p(u_{i+1}, a_{i+1}, a_{i+1})$ in $LS(i + 1)'$. Note that $\frac{w_{i'}}{s_{i'}} \leq \frac{w_{i+1}}{s_{i+1}} \leq \frac{w_i}{s_i}$. Apply the same marking scheme to mark from the earliest piece $p(i')$ of job i' with space $p(u_{i'}, a_{i'}, a_{i'})$ in LS to $p(u_{i+1}, a_{i+1}, a_{i+1})$ in LS' and associate a profit $r_{p(i')} = \frac{w_{i'}(f(T-a_{i+1})-f(T-a_{i'}))}{s_{i'}}$ to the marking. Together with the previous marking from $p(u_{i+1}, a_{i+1}, a_{t+1})$

[1] On the other hand, bidding a superset with jobs that are not held by the user is also risky since the user cannot provide those jobs to be consistent with her request and may be further punished.

to $p(u_i, a_i, a_i)$, they compose a marking chain with length two. In general, every marking starts from the space on which the piece is in LS to a lowest available space it is moved to in LS'. Therefore, by applying the marking rule to all the remaining pieces, we have a chain that ends in $p(u_{i+1}, a_i, a_i)$. Denoted by C_0 the pieces on the chain. We have $\frac{w_{\hat{i}}}{s_{\hat{i}}} \leq \frac{w_i}{s_i}$ for all $p(\hat{i}) \in C_0$. For every two adjacent markings on the same chain, the former one starts from a released space while the latter one points to the released space of the former one. Thus the total profit marked on the same chain would be at most $\sum_{p(\hat{i})\in C_0} r_{p(\hat{i})} \leq \frac{w_i}{s_i} f(T - a_i)$. There are at most s_i such chains where chain k ($0 \leq k \leq s_i - 1$) ends in $p(u_{i+1}, a_i + k, a_i + k)$. We could denote by C_k the pieces on chain k. Then in general chain k has a total profit at most $\frac{w_i}{s_i} f(T - a_i + k)$. Hence, LS' gains a profit at most $\sum_{0\leq k\leq s_i-1} \frac{w_i}{s_i} f(T - a_i + k)$ more than LS from jobs $i + 1, i + 2, \ldots, n$. However, LS' loses a profit at least $\sum_{0\leq k\leq s_i-1} \frac{w_i}{s_i} f(T - a_i + k)$ due to unrevealing job i. Therefore, the user gains no benefit by unrevealing a job.

Now consider the general case that there are multiple users. Excluding the job reported by user j, all other jobs are considered in the same order in LS and LS'. The marking scheme stated above does not rely on to which user the job belongs. With more users as the input, the benefit gained by j could only be reduced or keep the same compared to the case that all jobs are held by j, since if otherwise some job is held by another user, then the profit counted in the marking chain would be reduced. Hence, any user has no incentive to lie in a multi-user system. For the case that multiple jobs are unreported by a user, all the chains in the marking scheme just terminate in the released space of the unreported jobs. The total profit is the cumulated profit of all the chains that end in the space released by the unreported jobs. Therefore, there is no benefit for each user to bid any subset $\hat{S}_j$ with $\hat{S}_j \subset S_j$ of her jobs. On the other hand, by biding a superset $\hat{S}_j$ with $\hat{S}_j \supset S_j$ or a job that does not belong to her, the allocation can only postpone the allocation of other jobs due to the occupation of space. Thus the user cannot gain more profit from the jobs that do not belong to her since the utility is the total profit of her jobs or jobs that are in her true job list but postponed in the allocation. Thus she also has no incentive to report any job that does not belong to her. This shows the truthfulness of the users and completes the proof.

To tackle the case that allows $b_i > \tau$ in LS, the analysis can be extended by modifying the jobs and values a_i, b_i as follows. Take time $\tau + 1$ as a virtual time that can be placed multiple pieces instead of one. For any job with $b_i \leq \tau$, keep a_i, b_i the same. For jobs with $a_i > \tau$, partition job i into s_i unit-jobs with size one where every unit-job p has $a_p = b_p = \tau + 1$. For any job with $a_i \leq \tau$ and $b_i > \tau$, partition job i into $b_i - \tau$ unit-jobs with size one and another sub-job with size $\tau - a_i + 1$ where we assume that each unit-job p has weight $\frac{w_i}{s_i}$ and $a_p = b_p = \tau + 1$ and the sub-job q with size $\tau - a_i + 1$ has weight $\frac{\tau - a_i + 1}{s_i} w_i$ and $a_q = a_i, b_q = \tau + 1$. Applying the above analysis to the modified jobs could show the truthfulness of the allocation for X_1. The allocation for X_2 is truthful by simply picking the first Q jobs with the largest value $w_i f(T - s_i)$ and assigning them to each server. This completes the proof. ■

D. PROOF OF THEOREM 6

Proof: Consider the allocation for jobs X_1 first. Let $ALG(X_1)$ be the expected profit obtained from jobs X_1. Denoted by LS the allocation of jobs in X_1 by the list scheduling algorithm. Take as if each job i is partitioned into s_i virtual unit-jobs with size one and weight $\frac{w_i}{s_i}$. Write the set of all the partitioned unit-jobs to be $\hat{X}$ and let $\hat{w}_{\hat{k}}$ be $\frac{w_i}{s_i}$ if unit-job $\hat{k}$ is partitioned from job i. Denoted by $OPT(\hat{X})$ the optimal profit to assign the partitioned unit-jobs $\hat{X}$ with the constraint that the non-migration property of the original jobs X still holds. Clearly we have $OPT(X_1) \leq OPT(\hat{X})$. Let $LS(\tau)$ be the allocation of the unit-jobs in time period $[1, \tau]$ of LS and correspondingly $\hat{X}(LS, \tau)$ be the partitioned unit-jobs allocated in $LS(\tau)$. Let $\hat{k} \in \hat{X}(LS, \tau)$ be assigned to time $c_{\hat{k}}$ in $LS(\tau)$. Then the expected profit $ALG(X_1)$ equals $\sum_{\hat{k}\in\hat{X}(LS,\tau)} \hat{w}_{\hat{k}}(T - c_{\hat{k}} - \tau)$, since job i in $LS(\tau)$ incurs an expected profit $\frac{1}{s_i} w_i (T - c_{\hat{k}} - \tau)$ when the piece $\hat{k}$ is finished at time $\hat{c}_k + \tau$. For $\hat{k} \in \hat{X}(LS, \tau)$, we have $c_{\hat{k}} \leq \tau$ and $ALG(X_1) \geq \sum_{\hat{k}\in\hat{X}(LS,\tau)} \hat{w}_{\hat{k}}(T - c_{\hat{k}} - \tau) \geq \sum_{\hat{k}\in\hat{X}(LS,\tau)} 2\tau\hat{w}_{\hat{k}}$.

Denoted by $O(\tau)$ the optimal allocation to assign the partitioned unit-jobs $\hat{X}$ in interval $[1, \tau]$ in order to maximize the profit in that interval with the constraint that the non-migration property of the original jobs X_1 still holds, and correspondingly let $\hat{X}(O, \tau)$ be the unit-jobs assigned in the interval. Obviously, we have $\sum_{\hat{k}\in\hat{X}(O,T)} \hat{w}_{\hat{k}} \leq 4 \sum_{\hat{k}\in\hat{X}(O,\tau)} \hat{w}_{\hat{k}}$ since the input jobs X_1 are the same while the time is enlarged 4 times when $T = 4\tau$. $OPT(\hat{X}) \leq \sum_{\hat{k}\in\hat{X}(O,T)} \hat{w}_{\hat{k}} T \leq 4 \sum_{\hat{k}\in\hat{X}(O,\tau)} \hat{w}_{\hat{k}} T$. Hence, to show the desrired approximation that $OPT(X_1) \leq 16 ALG(X_1)$, we only need to prove that $\sum_{\hat{k}\in\hat{X}(LS,\tau)} w_{\hat{k}} \geq \frac{1}{2} \sum_{\hat{k}\in\hat{X}(O,\tau)} \hat{w}_{\hat{k}}$ which bounds the profit of unit-jobs in interval $[1, \tau]$ between the allocations $LS(\tau)$ and $O(\tau)$.

To show $\sum_{\hat{k}\in\hat{X}(LS,\tau)} w_{\hat{k}} \geq \frac{1}{2} \sum_{\hat{k}\in\hat{X}(O,\tau)} \hat{w}_{\hat{k}}$, we will examine the unit-jobs in $\hat{X}(O, \tau)\backslash\hat{X}(LS, \tau)$ by introducing a one-to-one matching from $\hat{X}(O, \tau)\backslash\hat{X}(LS, \tau)$ to $\hat{X}(LS, \tau)$. These unit-jobs are executed at time later than τ in $\hat{X}(LS, \tau)$ and will be divided into two sets. Denoted by set $A(u)$ the unit-jobs in $\hat{X}(O, \tau)\backslash\hat{X}(LS, \tau)$ that are partitioned from the jobs that start at time earlier than time τ and hence finished later than time τ on the same server u. The first set is then $A = \cup_u A(u)$. The second set $B = \hat{X}(O, \tau)\backslash\hat{X}(LS, \tau)\backslash A$ contains the unit-jobs partitioned from the jobs that start later than time τ in $\hat{X}(LS, \tau)$. For the unit-jobs in $A(u)$, we match each of them to the unit-job that is the earliest executed on server u in $\hat{X}(LS, \tau)$ and is not matched yet. The one-to-one matching always exists since such server u is full-assigned with T unit-jobs in $\hat{X}(LS, \tau)$. When $|B| = 0$, we already have $\sum_{\hat{k}\in\hat{X}(O,\tau)\backslash\hat{X}(LS,\tau)} \hat{w}_{\hat{k}} \leq \sum_{\hat{k}\in\hat{X}(LS,\tau)} \hat{w}_{\hat{k}}$, since each matching starts from a unit-job with a smaller weight to some unit-job with a larger weight. For the case that $|B| > 0$, any

unit-job in B is partitioned from a job that is assigned to the earliest available time larger than τ. According to the strategy of list scheduling, all unit-jobs in B have smaller weight than any unit-job in $\hat{X}(LS,\tau)$. Thus it is sufficient to match the unit-job in B to an arbitrary unit-job in $\hat{X}(LS,\tau)$ that is not matched yet. Such a one-to-one matching exists since $|A \cup B| \leq |\hat{X}(LS,\tau)| = Q\tau$ when $|B| > 0$. Each matching is from a unit-job with a smaller weight to some unit-job with a larger weight, hence we have $\sum_{\hat{k}\in\hat{X}(O,\tau)\setminus\hat{X}(LS,\tau)} \hat{w}_{\hat{k}} \leq \sum_{\hat{k}\in\hat{X}(LS,\tau)} \hat{w}_{\hat{k}}$. Therefore, $2\sum_{\hat{k}\in\hat{X}(LS,\tau)} w_{\hat{k}} \geq \sum_{\hat{k}\in\hat{X}(O,\tau)} \hat{w}_{\hat{k}}$ and this completes the approximation analysis for jobs X_1.

It remains to consider the allocation for X_2. Let $ALG(X_2)$ be the expected profit obtained from jobs X_2 in the mechanism and $OPT(X_2)$ be the optimal profit when assigning jobs in X_2. All jobs in X_2 have size at least τ and each server is assigned at most four jobs in X_2 in the optimal allocation. The allocation just assigns each server one such job among the jobs that have the first m largest value $w_i(T - s_i)$. Therefore, this procedure ensures $ALG(X_2) \geq \frac{1}{4}OPT(X_2)$.

Combining both the analysis for jobs X_1 and X_2, the algorithm is $O(1)$-approximation. ■

REFERENCES

[1] Amazon.com. (2009). *Amazon Elastic Compute Cloud*. [Online]. Available: http://aws.amazon.com/ec2/

[2] V. Abhishek, I. A. Kash, and P. Key, "Fixed and market pricing for cloud services," in *Proc. IEEE INFOCOM Workshops*, Orlando, FL, USA, 2012, pp. 157–162.

[3] M. Al-Roomi, S. Al-Ebrahim, S. Buqrais, and I. Ahmad, "Cloud computing pricing models: A survey," *Int. J. Grid Distrib. Comput.*, vol. 6, no. 5, pp. 93–106, 2013.

[4] A. Archer and E. Tardos, "Truthful mechanisms for one-parameter agents," in *Proc. FOCS*, Oct. 2001, pp. 482–491.

[5] I. Ashlagi, F. Fischer, I. Kash, and A. Procaccia, "Mix and match," in *Proc. ACM Conf. Electron. Commerce*, 2010, pp. 305–314.

[6] R. Birke, A. Podzimek, L. Y. Chen, and E. Smirni, "Virtualization in the private cloud: State of the practice," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 3, pp. 608–621, Sep. 2016.

[7] S. Dughmi and A. Ghosh, "Truthful assignment without money," in *Proc. 11th ACM Conf. Electron. Commerce*, 2010, pp. 325–334.

[8] H. Fu, Z. Li, C. Wu, and X. Chu, "Core-selecting auctions for dynamically allocating heterogeneous VMs in cloud computing," in *Proc. IEEE Cloud*, Anchorage, AK, USA, Jun./Jul. 2014, pp. 152–159.

[9] J. Guo, F. Liu, D. Zeng, J. C. S. Lui, and H. Jin, "A cooperative game based allocation for sharing data center networks," in *Proc. INFOCOM*, Apr. 2013, pp. 2139–2147.

[10] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM J. Appl. Math.*, vol. 17, no. 2, pp. 416–429, 1969.

[11] N. Jain, I. Menache, J. Naor, and J. Yaniv, "Near-optimal scheduling mechanisms for deadline-sensitive jobs in large computing clusters," in *Proc. 24th Annu. ACM Symp. Parallelism Algorithms Archit.*, 2012, pp. 255–266.

[12] N. Jain, I. Menache, J. Naor, and J. Yaniv, "A truthful mechanism for value-based scheduling in cloud computing," *Theory Comput. Syst.*, vol. 54, no. 3, pp. 388–406, 2014.

[13] A. Mazrekaj, I. Shabani, and B. Sejdiu, "Pricing schemes in cloud computing: An overview," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 2, pp. 80–86, 2016.

[14] N. Noam, R. Tim, T. Eva, and V. Vijay, *Algorithmic Game Theory*. Cambridge, U.K.: Cambridge Univ. Press, 2007.

[15] R. Pal and P. Hui, "Economic models for cloud service markets," in *Proc. 13th Int. Conf. Distrib. Comput. Netw.*, Hong Kong, China: Springer-Verlag, 2012, pp. 382–396.

[16] G. V. Prasad, A. S. Prasad, and S. Rao, "A combinatorial auction mechanism for multiple resource procurement in cloud computing," *IEEE Trans. Cloud Comput.*, to be published.

[17] A. D. Procaccia and M. Tennenholtz, "Approximate mechanism design without money," in *Proc. 10th ACM Conf. Electron. Commerce*, 2009, pp. 177–186.

[18] W. Shi, L. Zhang, C. Wu, Z. Li, and F. C. M. Lau, "An online auction framework for dynamic resource provisioning in cloud computing," in *Proc. ACM SIGMETRICS*, Austin, TX, USA, Jun. 2014, pp. 71–83.

[19] K. Tsakalozos, H. Kllapi, E. Sitaridi, M. Roussopoulos, D. Paparas, and A. Delis, "Flexible use of cloud resources through profit maximization and price discrimination," in *Proc. IEEE 27th Int. Conf. Data Eng.*, Apr. 2011, pp. 75–86, 2013.

[20] X. Wang, X. Chen, and W. Wu, "Towards truthful auction mechanisms for task assignment in mobile device clouds," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Atlanta, GA, USA, May 2017, pp. 1–9.

[21] W. Wang, B. Li, and B. Liang, "Towards optimal capacity segmentation with hybrid cloud pricing," in *Proc. 32nd Int. Conf. Distrib. Comput. Syst.*, Jun. 2012, pp. 425–434.

[22] X. Wang, Y. Sui, J. Wang, C. Yuen, and W. Wu, "A distributed truthful auction mechanism for task allocation in mobile cloud computing," *IEEE Trans. Services Comput.*, to be published.

[23] Q. Wang, K. Ren, and X. Meng, "When cloud meets eBay: Towards effective pricing for cloud computing," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 936–944.

[24] W. Wu *et al.*, "Incentive mechanism design to meet task criteria in crowdsourcing: How to determine your budget," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 2, pp. 502–516, Feb. 2017.

[25] 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 Trans. Comput.*, vol. 65, no. 3, pp. 805–818, Mar. 2016.

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

[27] Q. Zhang, W. Wu, and M. Li, "Minimizing the total weighted completion time of fully parallel jobs with integer parallel units," *Theor. Comput. Sci.*, vol. 507, pp. 34–40, Oct. 2013.

[28] J. Zhao, H. Li, C. Wu, Z. Li, Z. Zhang, and F. C. M. Lau, "Dynamic pricing and profit maximization for the cloud with geo-distributed data centers," in *Proc. IEEE INFOCOM*, Toronto, ON, Canada, Apr./May 2014, pp. 118–126.

WEIWEI WU (M'14) received the B.Sc. degree from the South China University of Technology and the Ph.D. degree from the Department of Computer Science, City University of Hong Kong, and the University of Science and Technology of China in 2011. He was a Post-Doctoral Researcher with the Mathematical Division, Nanyang Technological University, Singapore, in 2012. He is currently an Associate Professor with Southeast University, China. His research interests include optimizations and algorithm analysis, game theory, cloud computing, crowdsourcing, and wireless communications.

MINMING LI (M'13–SM'15) received the B.Eng., M.Eng., and Ph.D. degrees from Tsinghua University, Beijing, China. He is currently an Associate Professor with the Department of Computer Science, City University of Hong Kong, Hong Kong. His research interests include wireless networks, algorithms design and analysis, combinatorial optimization, and scheduling and algorithmic game theory.

JIANPING WANG (M'03) received the B.S. and M.S. degrees in computer science from Nankai University, Tianjin, China, in 1996 and 1999, respectively, and the Ph.D. degree in computer science from The University of Texas at Dallas in 2003. She is currently an Associate Professor with the Department of Computer Science, City University of Hong Kong. Her research interests include dependable networking, optical networks, cloud computing, service-oriented networking, and data center networks.

XIUMIN WANG received the B.S. degree from the Department of Computer Science, Anhui Normal University, China, in 2006, and the Ph.D. degree from the Department of Computer Science, University of Science and Technology of China, Hefei, China, and the Department of Computer Science, City University of Hong Kong, under a joint Ph.D. Program. She is currently an Associate Professor with the School of Computer Science and Engineering, South China University of Technology, China. Her research interests include wireless networks and mobile computing.