Efficient Three-stage Auction Schemes for Cloudlets Deployment in Wireless Access Network

Gangqiang Zhou*, Jigang Wu*, Long Chen*, Guiyuan Jiang†, Siew-Kei Lam†
*School of Computer Science and Technology
Guangdong University of Technology, Guangzhou, Guangdong 510006
†School of Computer Science and Engineering, Nanyang Technological University, Singapore
Email:gq_zhou@outlook.com , asjgwucn@outlook.com, lonchen@mail.ustc.edu.cn,
gyjiang@ntu.edu.sg, assklam@ntu.edu.sg

Abstract

Cloudlet deployment and resource allocation for mobile users (MUs) have been extensively studied in existing works for computation resource scarcity. However, most of them failed to jointly consider the two techniques together, and the selfishness of cloudlet and access point (AP) are ignored. Inspired by the group-buying mechanism, this paper proposes three-stage auction schemes by combining cloudlet placement and resource assignment, to improve the social welfare subject to the economic properties. We first divide all MUs into some small groups according to the associated APs. Then the MUs in same group can trade with cloudlets in a group-buying way through the APs. Finally, the MUs pay for the cloudlets if they are the winners in the auction scheme. We prove that our auction schemes can work in polynomial time. We also provide the proofs for economic properties in theory. For the purpose of performance comparison, we compare the proposed schemes with HAF, which is a centralized cloudlet placement scheme without auction. Numerical results confirm the correctness and efficiency of the proposed schemes.

Keywords

loudlet; Auction Mobile cloud computing Incentive mechanism Resource allocationloudlet; Auction Mobile cloud computing Incentive mechanism Resource allocationC

1 Introduction

In recent years, portable devices such as smartphones and tablet PCs have evolved to reach a significant performance enhancement. However, applications running on those mobile devices also consume many

This article has been accepted by Wireless Networks, contents differ from the final version. This is not the final version. This article is used only for quick dissemination of research findings and only for education purpose.

resources, e.g. computing, storage, et al. Particularly, multiple applications are often run on the same devices of mobile users (MUs).

Therefore, the resource-limited mobile devices still require a lot more resources for better performance, to tackle real-time and delay-sensitive tasks, such as Virtual Reality games and Automatic driving.

A cloudlet is formed by a group of internet-well-connected, resource-rich, and trusted computers When the centralized cloud is too far away from MUs. Cloudlet can be utilized by neighboring MUs [1], and it also can bring us a good solution for the resource requirement problem as described above. MUs can achieve much better performance by offloading their delay-sensitive or computation-intensive tasks to the cloudlet nearby [2], because the cloudlets can provide them with low-latency and rich computing resource access [3].

The resource allocation has been investigated in the work [4], and the cloudlet deployment for task offloading has been discussed in [5], [6]. Many efficient algorithms have been proposed in [7], [8], to balance the workload among the cloudlets for reducing the MUs' delay. But access points (APs) and cloudlets may be reluctant to provide those services without any rewards, due to selfishness. To inspire cloudlets sharing their resources with MUs, incentive mechanisms have been introduced [9], [10]. However, one cloudlet only serve one MU in those works. Moreover, the resource in a cloudlet is always too expensive to be employed by a single MU.

To solve the above problems, there are several challenges: 1) How to place the cloudlets at APs efficiently. 2) How to assign cloudlet resources to the MUs when each MU has limited budget. 3) How to provide incentive for the three kinds entities (MUs, APs, Cloudlets).

Therefore, motivated by the group-buying scheme for spectrum allocation [11], we propose three efficient auction schemes to solve the problems of cloudlet placement and resource assignment jointly, which consists of three stages in each scheme. In the first stage, we divide all MUs into several small groups of MUs according to the AP they connected to, and then we figure out the total budget for each group of MUs. In the second stage, we assign cloudlets to APs. Finally, we charge MUs in the third stage according to the matching results.

The main contributions of this work can be summarized as follows.

1) We propose three auction schemes for joint cloudlet placement and resource assignment. The first scheme randomly generates a number m according to the capacity of each given cloudlet, followed by selecting the first m MUs according to the performance price ratio, calculating the budget for the given cloudlet.
2) Based on the first scheme, the second scheme calculates several profitable cases and then randomly selects one from them. It can improve the revenue of the small MU groups significantly. In the third scheme, we match cloudlets with APs in a global way based on the second scheme.
3) We prove that all three schemes can work in polynomial time. We also provide proofs for individual

rationality, budget balance and truthfulness. Both theoretical analysis and simulation results show that the proposed schemes outperform the existing work in this paper.

The rest of the paper is organized as follows. Section 2 describes related works about incentive mechanisms for resource allocation in mobile cloud computing. Section 3 formulates the resource allocation problem and describe the three-stage auction model. Section 4 introduces our algorithms in the auction model, together with some examples. In section 5, we prove the economic properties for the proposed auction model. Simulation results are given in section 6. Finally, section 7 concludes this paper.

2 Related Work

Resource allocation in mobile cloud computing is one of the fresh and meaningful topics in recent years [12], [13]. Mobile users offload their heavy tasks to the neighboring cloudlet, this has been an appealing way to relieve their demand for resources [14], [15]. For cloudlet deployment, many existing works such as [3], [7], [8] care about the cloudlet placement in a given network, and most of them focus on allocation cloudlet resource in a centralized manner. Mike and his partners [3] [7] discuss the challenge of cloudlet load balancing, and they proposed a useful algorithm which is fast and scalable to balance the workload for each cloudlet in the wireless metropolitan area networks. In [8], how to place cloudlets is first considered to reduce the processing delay for tasks while the resource of the cloudlet is limited. Authors propose a heuristic algorithm and an approximation algorithm to place cloudlets. However, those works [3] [7] [8] do not take the cost of cloudlets and APs into consideration. Cloudlets and APs in this system may feel reluctant to share their resource to the mobile users without any reward.

Incentive mechanisms which take those costs into consideration have been discussed in [16]. Resource allocation schemes in those works are more flexible and intelligent. Also, the resource holder and relay nodes are willing to serve users. The auction schemes are wildly used in the study of computer science, the details can be seen in [17], [18]. In [16], a cooperative bargaining game-theoretic algorithm is addressed for resource allocation in cognitive small cell networks. However, one cloudlet can only serve one MU in those works. The group-buying idea is introduced in [11] and [19]. In [11], a group-buying auction model is proposed to manage the spectrum redistribution, and the problem of that a single buyer cannot afford the whole spectrum is fixed.

In this paper, we introduce group-buying model into cloudlet deployment, to divide independent MUs into small groups based on the associated APs. Therefore, MUs of each group can afford those expensive cloudlets, and the cloudlet may share its resources with MUs in a flexible and efficient way. Different from our conference version [20], we have added one more auction scheme in this work and we have extended the conference work to better present the main idea of the three stage auction scheme.

3 System Model and Problem Formulation

3.1 Problem Formulation

The MU can be regarded as the buyer in our auction schemes. The cloudlet is constituted by resource-abundant devices, it is also the seller in our auction schemes. The AP is the access point of the wireless network for MUs, and it also can be placed with a cloudlet to improve mobile devices' performance, so it is the auctioneer between MUs and cloudlets.

Assume that the number of cloudlets is K. C_k indicates the kth cloudlet. Cap^k indicates the resource capacity of C_k. As defined as in [21], the cost function of cloudlet is

$$Cos(k) = c(k) \cdot w(k), \tag{1}$$

where $c(k)$ is the cost factor of C_k, and $w(k)$ is the workload brought by MUs' offloaded tasks. In this paper, we try to make the cloudlet share its resources to a suitable small group of MUs rather than just one MU. To inspire cloudlets sharing their resources, we define the reserve price of C_k, denoted as r_k,

$$r_k = c(k) \cdot Cap^k + \delta. \tag{2}$$

Where C_k must be paid at least r_k, no matter which group of MUs finally wins C_k. Cloudlets in this paper may be heterogeneous, we assume that their capacity and cost factor may be different with each other, so their reserve price will also be different. While cloudlet C_k joins in the auction scheme, its total resource capacity Cap^k and cost factor $c(k)$ are fixed. C_k cannot change them during the whole auction. Then, r_k is also fixed. By the way, C_k can adjust its reserve price r_k by changing its parameter δ after a whole auction, such as increasing the value of δ if its resource is over competitive in the market, and decreasing the value of δ while the resource is oversupplied, which will make C_k benefit more from the auction, but this feedback mechanism is out of the scope this paper. Therefore, we assume $\delta = 0$ in this paper.

Assume that the number of AP is n in the given network. a_i indicates the ith AP, and it is connected with n_i MUs. In this paper, MUs connect to the wireless network through AP. Therefore, we can easily divide MUs into some groups base on the connected AP by the MUs. Each group of MUs can be assigned at most one cloudlet, and if the group of MUs which connects with a_i is assigned with cloudlet C_k, the MUs in the group cannot request other cloudlet resource, and the cloudlet C_k can only serve for the MUs in the group of a_i. It is noteworthy that this is different with [7], where MU can request service from other cloudlets if it's local AP do not have cloudlet or the assigned cloudlet is out of service. In our auction schemes, APs are the auctioneer who deals with the transaction between MUs and cloudlets.

For MUs that connected with the wireless network through the ith AP, we call them the ith group of MUs. Different groups have different amount of tasks to offload. Let m_i^j be the jth MU of the ith AP.

Its valuation for each cloudlet may be different. The mobile user m_i^j may give a higher valuation for the cloudlet it preferred (such as the cloudlets which have a good quality of service to it). Then it will submit a much higher bid on those cloudlets based on their valuation. Instead, m_i^j will submit a much lower bid on the cloudlets which m_i^j do not like. Then, the valuations of m_i^j on the kth cloudlet C_k is $v_i^j(k)$, which is private information of m_i^j. The budget of m_i^j for C_k is $b_i^j(k)$, which is public information, as this budget is the bid that MU submits for cloudlets. Namely, MUs' valuation for each cloudlet depends on their preference of those cloudlets, and is known only by themselves. Different MUs may produce different valuations on the same cloudlet, according to their different preferences. Usually, in an auction schemes, the buyer bid truthfully only if its budget equals its valuation. For instance, MU m_i^j bid truthfully on cloudlet C_k only if $b_i^j(k) = v_i^j(k)$. But MUs' valuation for each cloudlet is unknown to others, so the auction scheme must be truthful enough to prevent MU benefit more by bidding untruthfully, or the auction will bankrupt soon.

When the transactions are done after our three-stage auctions, the winner MUs will pay for the winner cloudlets and the connected APs, the winner cloudlets will be placed on its matching AP and serve for the small group of MUs connected by this AP. For instance, if MUs in a_i wins C_k, C_k will be placed on a_i, and then C_k provides services to MUs in a_i. Let w_i be the winner set, which consists of the winner MUs in the group of MUs in a_i. Let p_i^j be clearing price of the MU m_i^j.

If m_i^j is a winner, then m_i^j will be charged at p_i^j after the auction. For the case of that m_i^j bid truthfully, we define its utility u_i^j as

$$u_i^j = \begin{cases} v_i^j(k) - p_i^j & if\ m_i^j \in w_i, \\ 0 & otherwise, \end{cases} \tag{3}$$

where $v_i^j(k)$ is the valuation of m_i^j on the cloudlet C_k it wins. This equation implies the m_i^j obtains the benefit from the auction. Similarly, the winner set W contains the winner APs. If a_i is a winner AP, its clearing price is P_i. When a_i bid truthfully, its utility u_i is defined as

$$u_i = \begin{cases} R_i^k - P_i & if\ a_i \in W, \\ 0 & otherwise, \end{cases} \tag{4}$$

where R_i^k is the actual revenue that a_i calculates for its winner cloudlet C_k. Let W' be the set of winner cloudlets, and P^k be the clearing price of C_k. Its utility u^k is defined as

$$u^k = \begin{cases} P^k - r_k & if\ C_k \in W', \\ 0 & otherwise. \end{cases} \tag{5}$$

The social welfare can quantify the efficiency of our auction schemes. Let SW be the social welfare, which means the total utility of all participants in the auction. It is defined as

$$SW = \sum_{i=1}^{n}\sum_{j=1}^{n_i} u_i^j + \sum_{i=1}^{n} u_i + \sum_{k=1}^{K} u^k. \tag{6}$$

TABLE 1
Symbols of Participant

Definition	C_k	a_i	m_i^j
Quantity	K	n	n_i for a_i
Capacity or Workload	Cap^k	$-$	l_i^j
Cost, Revenue or Valuation	$Cos(k)$	R_i^k	$v_i^j(k)$
Reserve price or Budget	r_k	B_i^k	$b_i^j(k)$
Clearing price	P^k	P_i	p_i^j
Utility	u^k	u_i	u_i^j
Winner set	W'	W	w_i

3.2 System Model

The Fig. 1 shows the model of our three-stage auction schemes. In the first stage, we divide MUs into n small groups according to the APs that connect the MUs and cloudlets. Then, in each group the AP calculates its total revenue for each cloudlet, e.g. the AP a_i calculates the revenue R_i^k for cloudlet C_k. R_i^k is calculated according to the budget of the MU group in a_i, and these budgets are their bids for C_k, i.e., $b_i^j(k)(j \in [1,\dots,n_i])$. The total revenue quantify the preference of the MU group on each cloudlet. In the AP a_i, the MU m_i^j which has been utilized in calculating R_i^k will be regarded as a potential winner for cloudlet C_k, and its potential price is $p_i^j(k)$. If a_i wins C_k in the next stage, C_k will share its resources with m_i^j, and m_i^j will be charged at $p_i^j(k)$, i.e., its clearing price p_i^j equals to $p_i^j(k)$. On the other hand, C_k will only share its resources with the MU who paid for it. We cannot ensure that all MUs in a_i can be served by C_k, due to the constraints of economic properties. The rest of MUs will be left to the next round of the auction, which is not within the scope of this paper.

In the second stage, APs submit their budget to each cloudlet. This budget is the total budget of the MU group in the corresponding AP, which is generated base on the revenue for each cloudlet. For instance, the budget of a_i for C_k is B_i^k which is the price that a_i bid for C_k. For each AP, its revenue R_i^k is provided by its MU group. It is a real value, and the budget B_i^k is generated by itself, we can easily find that both R_i^k and B_i^k are public information. Therefore, we can easily verify that whether a_i bid truthfully or not. After that, we try to match cloudlets with APs while subjecting to our desired properties. As a result, for the winner set of cloudlets W' and the winner set of APs W, the matching result between W' and W can be defined by the mapping function $\sigma(\dot{})$. For example, $\sigma(i) = k$ means cloudlet C_k is assigned to AP a_i, and their clearing prices P_i and P^k are same.

Then, in the third stage, the winner MUs set in a_i is w_i, winning APs will charge them according to their potential winner price generated in the first stage.

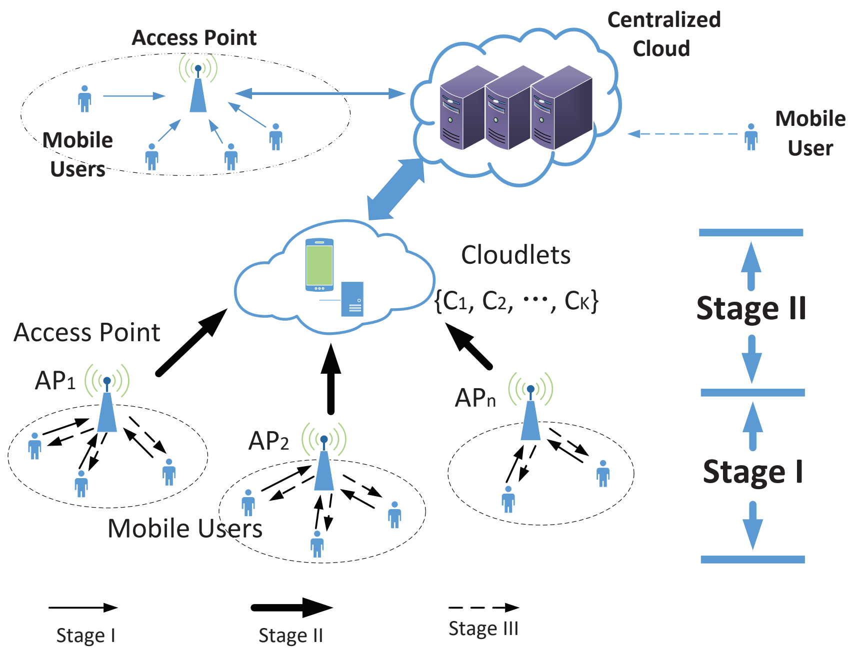

Fig. 1. Three-stage auction model.

3.3 Desirable Properties

3.3.1 Truthfulness

Let θ be a positive number. The participants may pay an extra cost θ to figure out how to bid in the auction scheme that makes them benefit more. When MU m_i^j bid untruthfully, we define the utility $\widetilde{u}_i^j$ as follows.

$$\widetilde{u}_i^j = \begin{cases} v_i^j(k) - p_i^j - \theta & if\ m_i^j \in w_i, \\ -\theta & otherwise. \end{cases} \tag{7}$$

Similarly, we now define the utility $\widetilde{u}_i$ for the case of that AP a_i bid untruthfully.

$$\widetilde{u}_i = \begin{cases} R_i^k - P_i - \theta & if\ a_i \in W, \\ -\theta & otherwise. \end{cases} \tag{8}$$

The extra cost θ varies for different MUs and different APs. The different market situation also causes different extra cost even for the same MU (or the same AP). In this paper, we define truthfulness as a weakly dominate strategy as mentioned in [9], where the player cannot improve its utility by bidding an untruthful bid in truthful auction scheme. Truthfulness is significant for an auction, we must ensure $u_i^j \geq \widetilde{u}_i^j$ and $u_i \geq \widetilde{u}_i$ for each MUs and APs to keep our auction truthful. In our auction scheme, we discuss the truthfulness in which only one player can change its bid or strategy, and the others cannot.

3.3.2 Budget balance

The total price charging for buyers is not less than the total price paid for sellers. If $\sigma(i) = k$, then

$$\sum_{i=1}^{n}\sum_{j=1}^{n_i} p_i^j \geq \sum_{k=1}^{K} P^k + (\sum_{i=1}^{n} R_i^k - \sum_{i=1}^{n} P_i).$$

3.3.3 Individual rationality

For sellers, they cannot benefit at a price smaller than it's asked, i.e., $P^k \geq r_k$. For buyers, they cannot be charged at a price bigger than it's bid, i.e., $b_i^j(k) \geq p_i^j(k) = p_i^j$ if $\sigma(i) = k$. For APs, we define their individual rationality as $R_i^k \geq B_i^k \geq P_i$ if $\sigma(i) = k$.

3.3.4 Computation efficiency

We will prove that the schemes can be performed in polynomial time.

4 AUCTION SCHEMES

In this section, we describe the proposed three auction schemes. The first is for Three-stage Auction scheme for Cloudlet Deployment, named TACD. The second, named TACDp, is an improved version of TACD by refining the first stage of the auction scheme. The third is called TACDpp, that is derived from TACDp by improving the mapping approach in its second stage.

4.1 Framework of the Schemes

All these three schemes are inspired by the idea of "group-buying". Each scheme consists of three stages. In stage I, APs calculate the revenue from their small group of MUs, and figure out the potential winner MUs for each cloudlet, the algorithm used in this stage is named ACRC. The revenue matrix is indicated as $\{R_i^k\}(i \in [1, \ldots, n], k \in [i, \ldots, K])$, which is formed by the revenues of the APs for each cloudlet. APs can bid for cloudlets according to $\{R_i^k\}$, and these bids form the budget matrix $\{B_i^k\}(i \in [1, \ldots, n], k \in [i, \ldots, K])$. In stage II, we match APs with cloudlets according to the budget matrix $\{B_i^k\}$ and the reserve price vector $\{r_k\}(k \in [1, \ldots, K])$, where the vector is formed by the reserve price of cloudlets, and the algorithm in this stage named ASC. In stage III, the winner APs, which are placed with cloudlets, allocate resources to their winner MUs and charge these MUs.

4.2 Scheme 1: TACD

4.2.1 Stage I: Calculating Revenue

The algorithm used in the first stage of TACD is named ACRC. For more details, see Algorithm 1. At first, for each AP such as a_i, we calculate its revenue R_i^k for all cloudlets. Obviously, the revenue R_i^k is calculated from the small group of MUs in a_i. Let $t_i^j(k)$ be the performance price ratio of the MU m_i^j.

In other words, $t_i^j(k)$ is the unit budget of m_i^j for the cloudlet C_k, and it is defined as follows.

TABLE 2
Symbols in Algorithms

Symbol	Definition
$t_i^j(k)$	m_i^j's performance price ratio on C_k
A	Array of MU sorted by $t_i^j(k)$
l_x	The workload of the xth MU in A
A_x, L_x	The first x MUs in A, and their total workload
s	The maximum quantity of MUs in A while $L_s \leq Cap^k$
S_x	The revenue of the first x MUs in A
m	The independent integer
w_i^k	The potential winner MUs in a_i for C_k
p	The unit price of MUs
$p_i^j(k)$	m_i^j's potential price on C_k
top_1, top_2	The top factor in ACRC, ASC
A'	The randomly sorted AP set
D	The profit matrix $\{B_i^k - r_k\}$
σ	Mapping function from a_i to C_k

Algorithm 1 ACRC: AP a_i Calculating the Revenue vector for each Cloudlet

Input: Sorted MUs array A, cloudlets' capacity set $\{Cap^k\}$
Output: a_i's revenue vector $\{R_i^k\}$, a_i's potential winner matrix $\{w_i^k\}$ and a_i's potential price matrix $\{p_i^j(k)\}$

```
for k = 1 to K do
    Maximizing the number s subject to L_s ≤ Cap^k and L_s + l_{s+1} > Cap^k.
    If L_{n_i} ≤ Cap^k, then s = n_i.
    The revenue set {S_x} = GTR(A, s), and the revenue of the first s − 1 cases is S_1, S_2, ..., S_{s−1}.
    The integer m is randomly generated in [(s + 1)/2, s − 1].
    Then the revenue R_i^k = S_m.
    a_i's potential winner set for C_k is w_i^k = A_m.
    Then the unit price p equals to the (m + 1)th MU's performance price ratio in A.
    if m_i^j ∈ w_i^k then
        p_i^j(k) = l_i^j · p
    else
        p_i^j(k) = 0
    end if
end for
return {R_i^k}, {w_i^k}, {p_i^j(k)}
```

$$t_i^j(k) = \frac{b_i^j(k)}{l_i^j}, \tag{9}$$

where l_i^j is the workload of m_i^j, and the value of l_i^j is kept unchange no matter which cloudlet receives the tasks offloaded by m_i^j. The value of $t_i^j(k)$ will increase with the increasing $b_i^j(k)$, i.e., m_i^j will get a higher performance price ratio on C_k if it has more budget on C_k.

The set A consists of the MUs in a_i, where the MUs are sorted in descending order in terms of their performance price ratio $t_i^j(k)$. Let A_x be the set of MUs which are the first x ($x \leq n_i$) members of A. Let l_x be the workload of the xth MU in A, i.e., l_1 is the workload of the first MU in A. Let L_x be the total workload of A_x, i.e., $L_x = l_1 + l_2 + l_3 + ... + l_x$. We try to find the index s in A to maximize L_s, in which $L_s \leq Cap^k$ and $L_s + l_{s+1} > Cap^k$. If the total workload of the MUs in a_i is less than or equal to Cap^k, i.e., $L_{n_i} \leq Cap^k$, then $s = n_i$.

Let S_x be the revenue which is generated by the first x MUs of A. Let $S_x = p \cdot L_x$, where p is the unit

Algorithm 2 GTR: Getting the Revenue set

Input: A, s
Output: The revenue set $\{S_x\}$
1: Let $\{S_x\}$ be the revenue set of the first $s-1$ cases in A.
2: **for** $x = 1$ to $s-1$ **do**
3: The unit price p is equal to the $(x+1)$-th MU's performance price ratio in A.
4: L_x is total workload of the first x MUs in A.
5: Then $S_x = p \cdot L_x$.
6: **end for**
7: **return** $\{S_x\}$

price which equals to the performance price ratio of the $(x+1)$th member in A. The Algorithm 2 which named GTR is to get the revenue set $\{S_x\}$, where $\{S_x\} = \{S_1, S_2, \ldots, S_{s-1}\}$. In order to keep the MUs bid truthfully, we randomly generate an integer m, where $(s+1)/2 \leq$m$\leq s-1$. The random number m is independent of the bids of MUs'. Then a_i's revenue for C_k equals to S_m, i.e., $R_i^k = S_m$. The set of potential winner of a_i for C_k consists of the first m MUs of A, i.e., $w_i^k = A_m$. The unit price p equals to the performance price ratio of the $(m+1)$th MU in A. For the MU m_i^j in a_i, its potential price on C_k is $p_i^j(k)$, and $p_i^j(k) = l_i^j \cdot p$ if $m_i^j \in w_i^k$, or $p_i^j(k) = 0$ if $m_i^j \notin w_i^k$. It means if a_i is allocated with C_k after the whole auction scheme, then the MUs which $m_i^j \in w_i^k$ are winners, and they will be charged at $p_i^j(k)$ by a_i. The sum of $\{p_i^j(k)\}$ equals to R_i^k, i.e., $R_i^k = \Sigma_{j=1}^{n_i} p_i^j(k)$, that is the preference of the MUs in a_i for the cloudlet C_k.

In TACD, we choose the random number m in $[s+1)/2, s-1]$ based on the following reasons. First, the number m must be a random number to keep our auction truthful, and we will discuss it later. Second, if the random number m is close to 1, the unit price p will be increased but the number of winner MUs will be reduced, and it will go opposite side if m close to s. The performance comparisons for different m values are mentioned in [11], the authors addressed that the APs will get more budget while the number of MUs fall in $[30\%, 70\%]$. Similarly, in this paper the APs will get more budget when the number m is randomly generated in $[s+1)/2, s-1]$. Third, for each AP, the more budget it calculates the easier it wins a more profitable cloudlet in the second stage. Finally, if AP gets the same revenue at $m_1 = 0.3s$ and $m_2 = 0.7s$, it will win the next stage at the same probability, but there is a big difference between the social welfare derived from the two settings of m. It is clear that $m = 0.7s$ is better. In summary, we generate the random number in $[s+1)/2, s-1]$, so that AP can calculate a higher budget and get more profits.

To illustrate the detail of ACRC in TACD, we provide an simple example to demonstrate how this algorithm works for AP a_i. In this example, the performance price ratios of the MUs on C_1 and C_2 are shown in Table 3(a). Their workload vector is shown in Table 3(b), and the capacity vector of cloudlet is shown in Table 3(c). For cloudlet C_1, we sort MUs in terms of their performance price ratio $t_i^j(1)$ in descending order at first. Then the order of MUs in the sorted array A is: $A = \{m_i^4, m_i^1, m_i^5, m_i^9, m_i^6, m_i^{10}, m_i^2, m_i^3, m_i^7, m_i^8\}$. Let l_s be the workload of the sth MU in A. The workloads of the MUs in A are $\{l_1 = 1.4, l_2 = 1.5, l_3 = 1.6, \ldots\}$,

TABLE 3
Example for ACRC

(a) MUs' performance price ratio on each Cloudlet

	$t_i^1(k)$	$t_i^2(k)$	$t_i^3(k)$	$t_i^4(k)$	$t_i^5(k)$
C_1	6	2.9	2.7	6.4	5.6
C_2	6	2.5	4.5	5.7	3.1
...					

	$t_i^6(k)$	$t_i^7(k)$	$t_i^8(k)$	$t_i^9(k)$	$t_i^{10}(k)$
C_1	3.6	2	1.7	3.7	3.6
C_2	1.8	3.2	4.3	3.7	2.9
...					

(b) The total workload of MUs' offloading task(s)

l_i^1	l_i^2	l_i^3	l_i^4	l_i^5	l_i^6	l_i^7	l_i^8	l_i^9	l_i^{10}
1.5	2.7	2.2	1.4	1.6	2.2	2.5	2.3	2.4	2.2

(c) Cloudlets' resource capacity

Cap^1	Cap^2	Cap^3	Cap^4	Cap^5	Cap^6	Cap^7	...
17	22	25	11	19	21	18	...

which are shown in Fig. 2. Let L_x be the total workload of the first x members of A. For instance, $L_3 = l_1 + l_2 + l_3 = 4.5$. According to ACRC, $s = 8$, MUs m_i^7 and m_i^8 which are painted in red are losers in ACRC. Then we calculate the revenue for this $s-1$ cases. The unit price p for S_x is the $(x+1)$-th performance price ratio of MU in A, and $S_x = p \cdot L_x$. For instance, the unit price p for S_5 is the 6th performance price ratio of MU in A, i.e., $p = t_i^{10}(1) = 3.6$. Then, $S_5 = p \cdot L_5 = 3.6 * 9.1 = 32.76$. We get a random number within $(4, 7)$. Assume that $m = 6$. We 'sacrifice' MUs m_i^2, m_i^3 which are painted in yellow to keep ACRC truthful. Therefore $R_i^1 = S_6 = 32.8$ and the unit price $p = t_i^2(1) = 2.9$. The first 6 MUs in this example form the potential winner set for C_1, i.e., $w_i^1 = \{m_i^4, m_i^1, m_i^5, m_i^9, m_i^6, m_i^{10}\}$. For these MUs, their potential price $p_i^j(k) = p \cdot l_i^j$. In this example, their potential price in A is $p_i^4(1) = 2.9 * 1.4 = 4.06$, $p_i^1(1) = 2.9 * 1.5 = 4.35$, $p_i^5(1) = 2.9 * 1.6 = 4.64$, and $p_i^9(1) = 6.96$, $p_i^6(1) = 6.38$, $p_i^{10}(1) = 6.38$. For the rest of MUs $m_i^j \notin w_i^1$, their potential price $p_i^j(1) = 0$. Then we can get the potential price set $\{p_i^j(1)\}$. It is similar for C_1 when AP a_i calculates revenue for other cloudlets.

After all the APs have calculated the revenue of each cloudlet, the revenue matrix $\{R_i^k\}$ is formed. Then APs will bid for each cloudlet in the next stage. These bids constitute the budget matrix $\{B_i^k\}$ which means the APs' budget for each cloudlet. All these APs have submitted their truthful bid if $\{B_i^k\} = \{R_i^k\}$, or there must be one/some cheater(s). The later case is what we need to avoid.

4.2.2 *Stage II: Matching Cloudlet for AP*

The algorithm used in this stage is named ASC, more details are shown in Algorithm 3. In this stage, APs deal with cloudlets according to the budget of APs and the reserve price of cloudlets. In TACD, we assign cloudlet to AP in a greedy manner, as mentioned in the existing work [22]. In ASC, we generate the profit

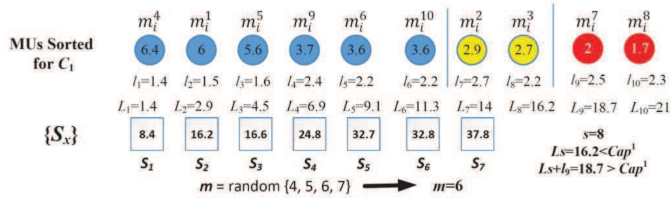

Fig. 2. Illustration of ACRC in TACD.

matrix D at first, where $D = \{B_i^k\} - \{r_k\}$, and $d_i^k = B_i^k - r_k$. Then we distribute the terms of APs randomly to A'. For each AP in A', we try to match it with an available cloudlet C_k to maximize the profit $B_i^k - r_k$ by the algorithm FRM.

The algorithm FRM is shown in Algorithm 4. For the AP a_i in A', then we try to match it with a most profitable cloudlet among the rest of available cloudlets. The profit vector of a_i is D_i which is the ith row of the matrix D. Then we select the largest element d_i^k in D_i. The cloudlet C_k is the most profitable cloudlet for a_i among the rest of available cloudlets. If ties, we choose the C_k with the smaller k. As a result, FRM matches a_i with C_k and return the matching to ASC. For this AP-cloudlet matching, its profit is d_i^k. Then, the algorithm ASC judges that if their profit is a positive value, i.e., whether $d_i^k > 0$. The budget of a_i is bigger than the reserve price of C_k if $d_i^k > 0$, i.e., if $B_i^k > r_k$. Then we try to find a bid for C_k from the other APs. The selected bid has the biggest value between B_i^k and r_r. In other words, we try to find the B_j^k where $B_i^k \geq B_j^k \geq \ldots \geq r_k$ and $i \neq j$. If there is no such B_j^k, then a_i fails to be allocated with C_k, and we set $d_i^k = 0$. Otherwise, we allocate C_k on a_i, i.e., let $\sigma(i) = k$. The clearing prices of a_i and C_k equal to the highest bid between B_i^k and r_k, i.e., $P_i = P^k = B_j^k$. Then we add a_i and C_k in their winner set, such as $W = W \cup a_i$ and $W' = W' \cup C_k$. Finally, for the matrix D we set the values of all elements in the ith row to 0. Meanwhile, we also set the values of all elements in the kth column to 0.

Algorithm ASC can ensure the utility of both APs and cloudlets if they are winners in the auction. For each winner AP-cloudlet matching, their clearing price P_i, P^k are independent with B_i^k and r_k, both a_i and C_k cannot modify the clearing price by themselves. This is helpful to keep the auction truthful.

4.2.3 Stage III: Charging for winner

In this stage, the winner APs choose the winner MUs according to their potential winner set, and then charge them at their potential winner price $p_i^j(k)$. For instance, while a_i wins C_k in stage II, the MUs in the potential winner set w_i^k is the winner MUs of a_i. For each MU m_i^j where $m_i^j \in w_i^k$, it will be charged by a_i at the clearing price p_i^j, where $p_i^j = p_i^j(k)$.

4.3 Scheme 2: TACDp

In this subsection, we propose a more efficient scheme named TACD plus (TACDp). The TACDp improves the first stage of TACD by changing the generation method of m in ACRC, so that the APs in TACDp can get more revenue. In TACD, m is randomly generated in $[(s+1)/2, s-1]$, it may sacrifice many MUs, resulting

Algorithm 3 ASC: APs' auction to Select suitable Cloudlet

Input: $\{B_i^k\}$, $\{r_k\}$, D
Output: W, W', $\{P_i\}$, $\{P^k\}$, σ

Distributing APs randomly into A'.
for $x = 1$ to n **do**
  Getting AP a_i and its matching cloudlet C_k by algorithm FRM(D, A', x).
  if $d_i^k > 0$ **then**
    if $B_i^k \geq B_j^k \geq \ldots \geq r_k$, which $j \neq i$ **then**
      $\sigma(i) = k$
      $P_i = P^k = B_j^k$
      $W = W \cup a_i$
      $W' = W' \cup C_k$
      Setting the values of elements in ith row of D to 0
      Setting the values of elements in kth column of D to 0
    else
      $d_i^k = 0$
    end if
  end if
end for
return W, W', $\{P_i\}$, $\{P^k\}$, σ

Algorithm 4 FRM: Finding a Rational Matching to a_i

Input: D, A', x
Output: AP a_i and it's matching cloudlet C_k

Let a_i denote the xth AP of A'.
Let vector D_i be the ith row of matrix D.
d_i^k is the maximum of D_i.
return a_i, C_k

in the performance decrease of TACD, although it can keep the auction scheme truthful. In TACDp, we calculate several profitable revenues and then randomly select one from them as the revenue of the target AP. In this section, we assume that the default value of top_1 is 3. Then, we select the top 3 profitable revenues $S_{x_1}, S_{x_2}, S_{x_3}$ from S, and m is randomly selected from $\{x_1, x_2, x_3\}$, denoted as $m = random\{x_1, x_2, x_3\}$. We can also change the value of top_1 to get a better result, e.g., $top_1 = 2$, then we select the top 2 profitable revenues S_{x_1}, S_{x_2} from S, and $m = random\{x_1, x_2\}$. The different value of top_1 will lead to different average revenue and different degree of truthfulness. The effect of top_1 will be discussed in the next section.

To illustrate the first stage of TACDp, we calculate the revenue of a_i on C_2, which is shown in Table 3. The ACRC in TACDp is shown in Fig. 3. In this example, $top_1 = 3$. Following TACD, we generate the number s and the revenue set S, resulting in $s = 10$ and $S = \{8.5, 13.0, 21.9, 27.3, 31.3, 38.1, 40.3, 40.2, 33.8\}$. The top 3 cases in S is S_7, S_8, S_6, then $m = random$ $\{6, 7, 8\}$, the average revenue is 39.5. It is worthwhile to point out that, the average revenue in TACD is 36.7. Thus, the revenue of the APs in TACDp is improved.

The rest steps of TACDp are the same with TACD. Note that, the value of top_1 must be larger than 1. In this case, let S_{max} be the most profitable revenue of S, we cannot fix the revenue of AP at S_{max}. This is because, we cannot keep ACRC truthful if we always choose $R_i^k = S_{max}$. For instance, $S_{max} = S_7$, i.e.,40.3 in Fig 3 and the unit price $p = t_i^{10}(2)$, i.e., 2.9 while MUs bid truthfully. For m_i^5, it's valuation on C_2 is

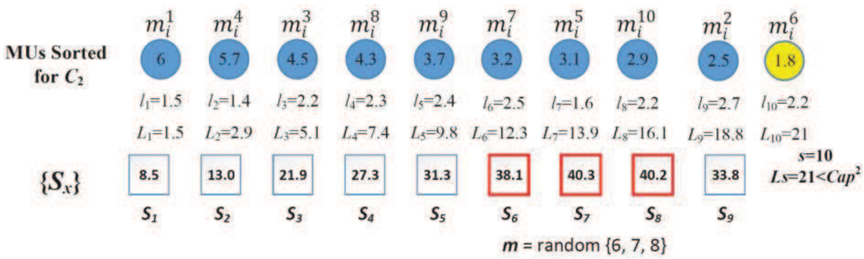

Fig. 3. Illustration of ACRC in TACDp.

$v_i^5(2)$ where $v_i^5(2) = b_i^5(2)$, i.e., 4.96, its potential price $p_i^5(2) = p \cdot l_i^5 = 2.9 * 1.6 = 4.64$. We assume that a_i wins C_2 in stage II, then m_i^5 will be charged at clearing price $p_i^5 = p_i^5(2) = 4.64$ in stage III. Therefore, the utility of m_i^5 is u_i^5 where $u_i^5 = v_i^5(2) - p_i^5 = 4.96 - 4.64 = 0.32$. However, if m_i^5 bid untruthfully, we assume that m_i^5 changes its budget on C_2 to $b_i^5(2) = 4.32$ which is less than its valuation on C_2. Then the performance price ratio of m_i^5 on C_2 is $t_i^5(2)$ where $t_i^5(2) = 2.7$ and it will be sorted behind $t_i^{10}(2)$ according to ACRC. $L_7 = L_6 + l_i^{10} = 12.3 + 2.2 = 14.5$, $L_8 = L_7 + l_i^5 = 14.5 + 1.6 = 16.1$, and $S_6 = L_6 * 2.9 = 35.67$, $S_7 = L_7 * 2.7 = 39.15$, $S_8 = L_8 * 2.5 = 40.25$, then $S_{max} = S_8$. If R_i^k always equal to the most profitable revenue, then $R_i^2 = S_8$ and its unit price $p = 2.5$. We assume that the matching result are the same in stage II, then m_i^5 will be charged at the clearing price $p_i^5 = p_i^5(2) = l_i^5 \cdot p = 1.6 * 2.5 = 4$ in stage III. Then, if m_i^5 bids untruthfully, its utility is $\widetilde{u}_i^5$ where $\widetilde{u}_i^5 = v_i^5(2) - p_i^5 - \theta = 4.96 - 4 - \theta = 0.96 - \theta$. m_i^5 can improve its utility if the value of the extra cost θ is small enough, e.g., $\theta < 0.96$, when m_i^5 bids untruthfully.

4.4 Scheme 3: TACDpp

We introduce another efficient algorithm named TACDpp in this subsection. The TACDpp is the improved version of TACDp, which refines the second stage of TACDp. The difference between TACDp and TACDpp is that, TACDpp replaces algorithm FRM with algorithm FRMG in ASC. The first stage of TACDpp is the same as that of TACDp. In the second stage, TACDpp matches cloudlets for APs in a global way, which is different with TACDp. In TACDpp, we match cloudlets with APs by algorithm FRMG, which is shown in Algorithm 5. Let top_2 be a small number, it is the top factor in FRMG, its default value is 2. For each round, FRMG gets a random integer rnd in $[1, top_2]$, then selects the rndth profitable value d_i^k from the profit matrix D, and it returns $\{a_i, C_k\}$ to ASC for further judgement. When the network is unbalanced between supply and demand, i.e., $K \neq n$, TACDpp can perform better due to the global idea. It is also worth to mention that we must ensure $top_2 > 1$ which is similar with top_1. It will be discussed later.

The performance comparison of the proposed schemes is shown in Table 4. This table lists the algorithms employed in each stage and the generation approach of the number m.

5 Desired Properties

5.1 Truthfulness

Theorem 1: The schemes TACD, TACDp and TACDpp are truthful in ACRC.

Algorithm 5 FRMG: Finding a Rational Matching in the Global scope

Input: D, top_2
Output: a_i, C_k
if $top_2 > 1$ **then**
  rnd is the random integer in $[1, top_2]$
else
  $rnd = 1$.
end if
Finding out the rnd-th profitable matching d_i^k from D.
return a_i, C_k

TABLE 4
Comparison for TACD, TACDp and TACDpp

Schemes	Stage I	The number m	Stage II
TACD	**ACRC+GTR**	$[(s+1)/2, s-1]$	**ASC+FRM**
TACDp	**ACRC+GTR**	One of top_1 cases	**ASC+FRM**
TACDpp	**ACRC+GTR**	One of top_1 cases	**ASC+FRMG**

Proof: To verify the truthfulness of ACRC, we only need to prove that MUs are truthful in our auction. In TACD, for the MU m_i^j, $b_i^j(k)$ is the truthful bid of m_i^j. Let $\widetilde{b}_i^j(k)$ be the untruthful bid. Then the utility of m_i^j is u_i^j when it bids truthfully. Let $\widetilde{u}_i^j$ be the utility when it bids untruthfully. We prove that m_i^j cannot improve its utility by submitting an untruthful bid as follows, i.e., $\widetilde{u}_i^j \le u_i^j$.

There are four cases for MU m_i^j in TACD:

1) MU m_i^j fails in the auction both in truthful bid $b_i^j(k)$ and untruthful bid $\widetilde{b}_i^j(k)$. Then, $u_i^j = 0$ and $\widetilde{u}_i^j = -\theta$.
2) MU m_i^j wins the auction while bid truthfully and fails in the auction while bid untruthfully. In this case, $u_i^j \ge 0$, and $\widetilde{u}_i^j = -\theta$.
3) MU wins the auction both in truthful bid and untruthful bid. When m_i^j wins the auction in TACD, its clearing price is c in our rules. On the other hand, if m_i^j also wins the auction in another bid, from the definition of truthfulness, the clearing price is also c while other bids of MUs are fixed. Then $\widetilde{u}_i^j = u_i^j - \theta$.
4) MU fails in the auction while bid truthfully and wins the auction while bid untruthfully. When m_i^j fails in TACD and it bids truthfully, the clearing price c is greater than or equal to its bid, i.e., $c \ge b_i^j(k)$. And if m_i^j wins the auction in another bid $\widetilde{b}_i^j(k)$, it must have $\widetilde{b}_i^j(k) \ge c$, so $\widetilde{b}_i^j(k) > b_i^j(k)$ and $\widetilde{b}_i^j(k) > v_i^j(k)$, then we have $\widetilde{u}_i^j \le u_i^j = 0$.

We have now discussed the truthfulness of MUs in TACD, while MU m_i^j bid for the kth cloudlet. And the other cloudlets do not need care about whether m_i^j cheat or not, if the kth cloudlet C_k is assigned to the AP a_i finally.

Similarly, MUs in TACDp and TACDpp are also truthful in ACRC, because these two schemes only change the way we get the random integer m. □

Theorem 2: The schemes TACD, TACDp and TACDpp are truthful in ASC.

Proof:

For TACD and TACDp, their algorithms in the second stage are similar to the algorithm *fixed price auction* as mentioned in[22]. This auction scheme has been proved to be truthful, we only change the generation manner of clearing price in TACD and TACDp while the transactions is done. Furthermore, the clearing price is independent to AP and cloudlet in the second stage of TACD and TACDp. Therefore, TACD and TACDp are also truthful for ASC.

For TACDpp in ASC, we ensure its truthfulness by the top factor top_2, which is discussed in the simulation section. □

5.2 Budget Balanced

Theorem 3: The schemes TACD, TACDp and TACDpp are budget balanced.

Proof:

In this paper, we only prove that TACD is budget balanced. The proof of TACDp and TACDpp are identical to that of TACD.

In TACD, if $\sigma(i) = k$, $a_i \in W$ and $C_k \in W'$, then the total clearing price charge for the MUs is val_1 where $val_1 = \sum_{i=1}^{n} \sum_{j=1}^{n_i} p_i^j$. Similarly, the total clearing price for cloudlets is val_2 where $val_2 = \sum_{k=1}^{K} P^k$, the total clearing price for APs is val_3 where $val_3 = \sum_{i=1}^{n} (R_i^k - P_i)$. The total budget of APs is val_4 where $val_4 = \sum_{i=1}^{n} B_i^k$, then $val_1 = val_4$ according to ACRC, and $val_4 = val_2 + val_3$ according to ASC. Then, $val_1 = val_4 = val_3 + val_2$, and $val_1 \geq val_3 + val_2$, i.e.,

$$\sum_{i=1}^{n} \sum_{j=1}^{n_i} p_i^j \geq \sum_{k=1}^{K} P^k + (\sum_{i=1}^{n} R_i^k - \sum_{i=1}^{n} P_i).$$

□

5.3 Individual Rationality

Theorem 4: The schemes TACD, TACDp and TACDpp are subject to the individual rationality.

Proof:

The individual rationality for TACD can be proved as follows. For sellers, according to the judgement in ASC, the clearing price for cloudlets cannot smaller than they asked, i.e., P^k is always bigger than r_k.

For buyers, if MU m_i^j wins the cloudlet C_k, the MU will be charged at p_i^j where $p_i^j = p \cdot l_i^j$. p is the performance price ratio of the mth MU in A, and $p \leq t_i^j(k)$. Therefore, $p_i^j \leq t_i^j(k) \cdot l_i^j = b_i^j(k)$.

For APs, we obtain $B_i^k = R_i^k$ according to the ACRC. Also, the adjustment factor f is in the scope of $(0, 1)$ in ASC, thus, the clearing price of AP $P_i = f \cdot B_i^k < B_i^k = R_i^k$. Therefore, $R_i^k \geq B_i^k \geq P_i$.

The proof of individual rationality for TACDp and TACDpp iss the same as that of TACD. □

5.4 Computational Efficiency

Theorem 5: The time complexity of TACD as well as TACDp is $O(K \cdot n \log n)$.

Proof:

For ACRC, the sorting needs $O(n \log n)$ time, finding the number s takes $O(n)$ time, and the algorithm GTR also takes $O(n)$ time. The time complexity of ACRC in TACD and TACDp is $O(K \cdot n \log n)$. For ASC, distributing APs randomly takes $O(n \log n)$ time, the algorithm FRM takes $O(K)$ time. So, the time complexity of ASC in TACD and TACDp is $O(n \cdot K)$. Therefore the total time complexity of TACD and TACDp are $O(K \cdot n \log n)$.

□

Theorem 6: The time complexity of TACDpp is $O(K \cdot n^2)$.

Proof:

The time complexity of ACRC in TACDpp is the same as that of TACDp, which is $O(K \cdot n \log n)$. For ASC, the algorithm FRMG takes $O(n \cdot K)$ time, which is different from the algorithm FRM. Thus, the time complexity of ASC in TACDpp is $O(K \cdot n^2)$. Therefore the total time complexity of TACDpp is $O(K \cdot n^2)$.

□

6 Numerical Results

6.1 Simulation Setup

In this paper, we simulate our works on MATLAB R2014a. In the simulation, the capacities of all the cloudlets follow the normal distribution $N(25, 5)$ and each capacity Cap^k satisfy the constraint $10 \leq Cap^k \leq 30$. Its cost factor $c(k)$ follows to the normal distribution $N(0.75, 0.1)$ and $0.5 \leq c(k) \leq 1$. Then, its reserve price $\{r_k\}$ can be calculated by formula 2. For each AP such as a_i, the number of MUs in a_i follows the uniform distribution $U(5, 30)$. For the MUs in a_i such as m_i^j, their workload follow the normal distribution $N(2, 1)$ and $1 \leq l_i^j \leq 3$. Their valuations for each cloudlet follow the uniform distribution $U(1, 15)$.

We compare our auction schemes with the strategy Heaviest Access Point First (HAF) [7]. HAF is an efficient scheme for cloudlet placement and resource allocation without auction. In this paper, the strategy HAF is working in the following way, at first, HAF sorts APs in terms of the total workload of MUs in descending order. Then, HAF sorts cloudlets in terms of their capacity in descending order. At last, HAF matches cloudlets for APs by turns. For instance, HAF assigns the first cloudlet whose capacity is the biggest to the first AP whose total workload of MUs is the heaviest, then HAF assigns the second cloudlet to the second AP and so on. If C_k is assigned to a_i, the budget that a_i bid for C_k is B_i^k. It is calculated using the method as in ACRC, but the number m is a fixed integer where $m = s$, and the potential winner MUs is the first m MUs in A, i.e., A_m. The unit price p charged by AP is the performance price ratio of the mth MU in A. In HAF, a_i only needs to calculate the budget on C_k. The transaction between a_i and C_k will be done if $B_i^k \geq r_k$, which is different from the algorithm ASC. It is obvious that, if HAF is an incentive mechanism, then it is untruthful. Moreover, the time complexity of HAF is $O(n \log n) + O(K \log K)$. In the first stage of HAF, the sorting of APs and cloudlets takes $O(n \log n)$ and $O(K \log K)$ time, respectively. In the second stage of HAF, the matching algorithm takes $O(n) + O(K)$ time.

6.2 Simulation Results

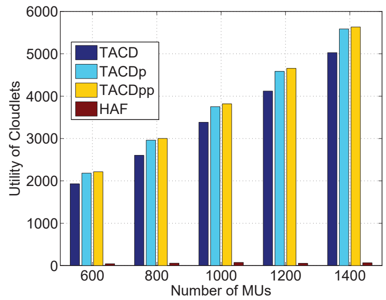

Fig. 4. Utility of Cloudlet.

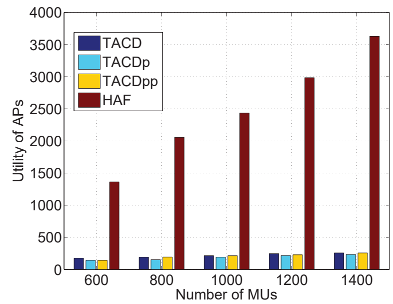

Fig. 5. Utility of APs.

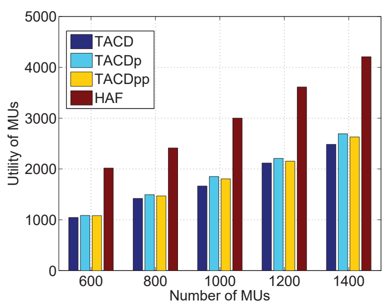

Fig. 6. Utility of MUs.

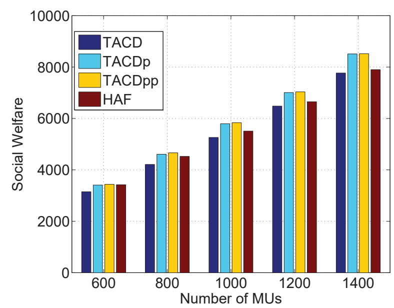

Fig. 7. Social welfare.

In the first part of our simulation, the top factors top_1 and top_2 are set to 2, and the market is balanced, i.e., $K = n$. The utility of cloudlets, APs and MUs are shown in Fig. 4, Fig. 5, and Fig. 6, respectively. The social welfare of auction schemes are shown in Fig. 7. There are big differences between our schemes and the HAF in the first three figures. Fig. 4 shows that our schemes are good for cloudlets, while Fig. 5 show that our schemes are weak for APs. The differences are caused due to the following reasons. In our schemes, we select a bid B_j^k other than B_i^k and r_k to keep ASC truthful where $B_i^k \geq B_j^k \geq r_k$. The clearing price of this transaction is B_j^k which is bigger than r_k. However, if C_k is assigned for a_i, HAF does not care about the truthfulness, the transaction is done while $B_i^k \geq r_k$, and the clearing price is equal to r_k. As a result, the utility of cloudlets is close to 0 in HAF as shown in Fig. 4. Moreover, the APs in HAF may catch many profits during the transaction as shown in Fig. 5. For the Fig. 6, it shows that HAF is more profitable for MU than our algorithms. It is because that the winner cloudlet in HAF serve for more MUs by a greedy manner and these MUs are charged with a lower unit price by AP than our schemes. In our schemes, the number of winner MUs is $m - 1$ where $m \leq s$, the unit price of these MUs is the performance price ratio of the mth MU in A. However, the number of winner MUs in HAF is m where

$m = s$, and the unit price of these MUs are the performance price ratio of the sth MU in A. Then, the number of winner MUs in HAF is more than our schemes, and these winner MUs are charged by a lower price than us. Therefore, it is more profitable for MUs as show in Fig. 5. The social welfare demonstrates that, while the number of MUs is 1000, the social welfare in TACD is 5% less than HAF, TACDp is 4.5% higher than HAF, and TACDpp is 5.6% higher than HAF. Moreover, our schemes perform better if there are more MUs in the wireless access network. For example, when the number of MUs is 1400, TACD is 1.7% less than HAF, TACDp and TACDpp are 7.6% and 7.9% higher than HAF respectively.

If the number of APs is bigger than the number of cloudlets, i.e., $n > K$, the performance of our auction schemes in "unbalanced market" is shown in Fig. 8. In this situation, the TACDpp performs better than that in the balanced market, because the global matching algorithm FRMG works better.

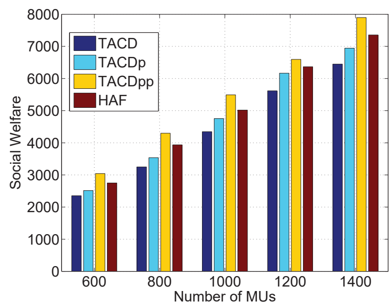

Fig. 8. Performance in unbalanced market.

Now, we evaluate the second stage of TACDpp while modifying the value of top_2 in a smaller data set, and we verify the truthfulness of TACDpp through different values of B_1^1. In this section, we fix the value of top_1 at 2 and modify the value of top_2 from 1 to 2 and then to 5. The utility of B_1^1 are shown in Fig. 9, Fig. 10 and Fig. 11, for the cases of $top_2 = 1, 2, 5$, respectively. In these figures, the solid line shows the profit of a_1 when a_1 bids truthfully in ASC, i.e. $B_1^1 = 85.5$. The dotted line shows the profit of a_1 when it bids untruthfully from $\widetilde{B}_1^1 = B_1^1 - 80$ to $\widetilde{B}_1^1 = B_1^1 + 50$ with the increase unit of 1. The result is the averaged over 100 random instances. Fig. 9 is the utility of AP for the case of $top_2 = 1$. In this case, TACDpp matches cloudlet C_k for AP a_i, while the profit of this matching is the most profitable one in the rest of cloudlets and APs. The utility of a_1 is U_1 and it is 18.7. It is stable and profitable, because TACDpp always makes the same strategy to match cloudlets with APs. In such a fixed strategy, a_1 will get the same profit if it bids truthfully, so the solid line is straight in Fig. 9. However, it is hard to check whether TACDpp is truthful in ASC while $top_2 = 1$. Because it may has some "bugs", in which APs can benefit more from their preferred cloudlet, by biding budgets that lower than their revenues. For instance, as we can see in Fig. 9, the utility of a_1 is $\widetilde{U}_1$ where $\widetilde{U}_1 = 22.7 - \theta$, while a_1 bid untruthfully among $\{64.5, 65.5, 66.5\}$. $\widetilde{U}_1$ is larger than U_1, if $\theta < 4$. It is because that, when $\widetilde{B}_1^1 = \{64.5, 65.5, 66.5\}$, the profit $\widetilde{B}_1^1 - r_1$ is so big that a_1 still wins C_1. Also, there is another AP a_x whose budget is B_x^1 where $B_x^1 \leq 64.5$, and it is the largest B_j^1 in which $B_j^1 \leq \widetilde{B}_1^1$, $j \in [1, n]$ and $j \neq 1$. Then, the clearing price will be much lower than that when it bids truthfully. Therefore, if AP a_1 pays some extra price θ to figure out these more profitable cases, it will get more profits firmly by bidding untruthfully.

Simulation results of TACDpp are shown in Fig. 10 where $top_2 = 2$. The solid line shows the utility of a_1 while it bids truthfully. This line is not a straight line as shown in Fig. 9, as the matching strategy is not a fixed pure strategy anymore.

When $top_2 = 2$, the matching strategy turns to a mixed strategy, we combine the following two strategies with equal probability, i.e. $1/2$,

1) Matching cloudlet C_k with AP a_i whose profit $B_i^k - r_k$ is the most profitable one.
2) Matching cloudlet C_k with AP a_i whose profit $B_i^k - r_k$ is the second profitable one.

So the utility of a_1 is not a stable value, even a_1 always bid truthfully. The utility varies within an interval near 18, which is shown in green solid line. In contrast, the green dotted line shows the utility of a_1 while it bids untruthfully. There are also some more profitable cases while a_1 bids untruthfully, such as $\{64.5, 65.5, 66.5\}$ as occour ed as in the case of $top_2 = 1$. But the difference is that, if $top_2 = 2$, a_1 can also benefit more in those cases with the probability of 50%. Otherwise, a_1 will be matched with other less profitable cloudlets, and it also must pay an extra cost θ to find those cases. Therefore, there is not any evident case in which a_1 can get more utility than the truthful case. It is worthless for a_1 to pay an extra cost θ to determine how to bid untruthfully. Therefore, TACDpp is truthful while $top_2 = 2$.

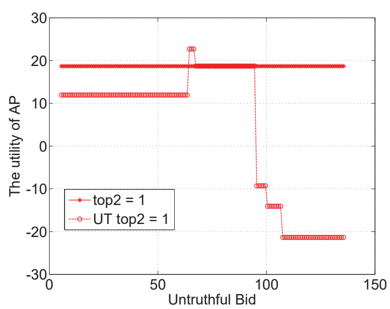

Fig. 9. $top_2 = 1$.

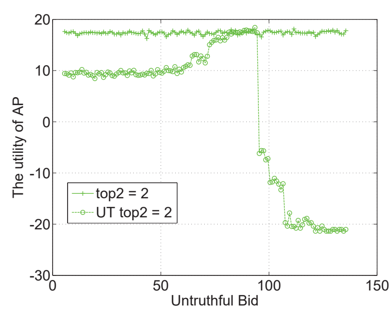

Fig. 10. $top_2 = 2$.

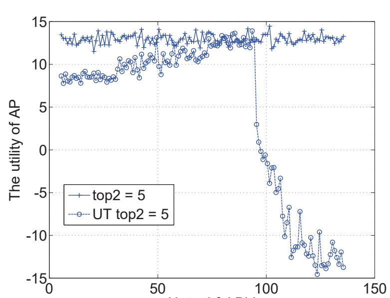

Fig. 11. $top_2 = 5$.

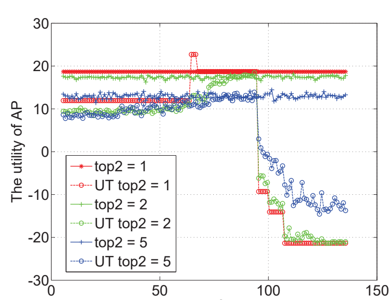

Fig. 12. Comparison.

Similarly, Fig. 11 shows the utility of a_1 while $top_2 = 5$. It is also a mixed strategy by 5 pure strategies, with the probability of $1/5$ for each strategy. These 5 pure strategies are used to match cloudlets to APs. The jth pure strategy is corresponding to the jth profitable value of $B_i^k - r_k$ for $j = 1, 2, \cdots, 5$. In the mixed strategy, the utility of a_1 varies with a larger range than that in Fig. 10 while a_1 bid truthfully. The value of its utility fall in $[12, 14.5]$, and it is less than that in Fig. 10. In other words, the strategy for $top_2 = 5$ is less profitable and less stable than the strategy for $top_2 = 2$, while APs bid truthfully. This is because the stronger randomness brings APs many solutions which are not profitable. For truthfulness, there is no evidence that a_1 can get more utility than the truthful one.

7 Conclusion

In this paper, we have proposed efficient auction schemes for cloudlets placement and resource allocation in wireless networks to improve the social welfare subject to economic properties. We have introduced the group-buying model to inspire cloudlets to serve the MUs. In our auction schemes, MUs can get access to cloudlets through APs, according to their preference and resource demands for cloudlets. The whole three entities MUs, APs, and cloudlets are motivated to participate in resource sharing. We have verified that our schemes are truthful, individual rational, budget balanced and computational efficient. Through simulations, we have shown that our schemes TACDp and TACDpp outperform HAF by about 4.5% and 5.6% respectively, in terms of social welfare, for the case that the number of MUs is 1000.

References

[1] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Computing*, 8(4):14–23, Oct 2009.

[2] Karthik Kumar, Jibang Liu, Yung-Hsiang Lu, and Bharat Bhargava. A survey of computation offloading for mobile systems. *Mobile Networks and Applications*, 18(1):129–140, Feb 2013.

[3] M. Jia, W. Liang, Z. Xu, and M. Huang. Cloudlet load balancing in wireless metropolitan area networks. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9, April 2016.

[4] F. Fang, H. Zhang, J. Cheng, and V. C. M. Leung. Energy-efficient resource scheduling for noma systems with imperfect channel state information. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–5, May 2017.

[5] Q. Xia, W. Liang, and W. Xu. Throughput maximization for online request admissions in mobile cloudlets. In *38th Annual IEEE Conference on Local Computer Networks*, pages 589–596, Oct 2013.

[6] M. Satyanarayanan. Pervasive computing: vision and challenges. *IEEE Personal Communications*, 8(4):10–17, Aug 2001.

[7] M. Jia, J. Cao, and W. Liang. Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks. *IEEE Transactions on Cloud Computing*, 5(4):725–737, Oct 2017.

[8] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo. Capacitated cloudlet placements in wireless metropolitan area networks. In *2015 IEEE 40th Conference on Local Computer Networks (LCN)*, pages 570–578, Oct 2015.

[9] A. L. Jin, W. Song, and W. Zhuang. Auction-based resource allocation for sharing cloudlets in mobile cloud computing. *IEEE Transactions on Emerging Topics in Computing*, 6(1):45–57, Jan 2018.

[10] Parnia Samimi, Youness Teimouri, and Muriati Mukhtar. A combinatorial double auction resource allocation model in cloud computing. *Information Sciences*, 357:201 – 216, 2016.

[11] P. Lin, X. Feng, Q. Zhang, and M. Hamdi. Groupon in the air: A three-stage auction framework for spectrum group-buying. In *2013 Proceedings IEEE INFOCOM*, pages 2013–2021, April 2013.

[12] Y. Liu, M. J. Lee, and Y. Zheng. Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system. *IEEE Transactions on Mobile Computing*, 15(10):2398–2410, Oct 2016.

[13] M. Reza Rahimi, Jian Ren, Chi Harold Liu, Athanasios V. Vasilakos, and Nalini Venkatasubramanian. Mobile cloud computing: A survey, state of art and future directions. *Mobile Networks and Applications*, 19(2):133–143, Apr 2014.

[14] M. Chen, Y. Hao, Y. Li, C. F. Lai, and D. Wu. On the computation offloading at ad hoc cloudlet: architecture and service modes. *IEEE Communications Magazine*, 53(6):18–24, June 2015.

[15] M. Chen, Y. Hao, C. F. Lai, D. Wu, Y. Li, and K. Hwang. Opportunistic task scheduling over co-located clouds in mobile environment. *IEEE Transactions on Services Computing*, pages 1–13, 2017. doi:10.1109/TSC.2016.2589247.

[16] H. Zhang, C. Jiang, N. C. Beaulieu, X. Chu, X. Wang, and T. Q. S. Quek. Resource allocation for cognitive small cell networks: A cooperative bargaining game theoretic approach. *IEEE Transactions on Wireless Communications*, 14(6):3481–3493, June 2015.

[17] Simon Parsons, Juan A. Rodriguez-Aguilar, and Mark Klein. Auctions and bidding: A guide for computer scientists. *ACM Comput. Surv.*, 43(2):10:1–10:59, February 2011.

[18] Y. Zhang, C. Lee, D. Niyato, and P. Wang. Auction approaches for resource allocation in wireless systems: A survey. *IEEE Communications Surveys Tutorials*, 15(3):1020–1041, Third 2013.

[19] D. Yang, G. Xue, and X. Zhang. Truthful group buying-based spectrum auction design for cognitive radio networks. In *2014 IEEE International Conference on Communications (ICC)*, pages 2295–2300, June 2014.

[20] Gangqiang Zhou, Jigang Wu, and Long Chen. Tacd: A three-stage auction scheme for cloudlet deployment in wireless access network. In Liran Ma, Abdallah Khreishah, Yan Zhang, and Mingyuan Yan, editors, *Wireless Algorithms, Systems, and Applications*, pages 877–882, Cham, 2017. Springer International Publishing.

[21] X. Kang and S. Sun. Incentive mechanism design for mobile data offloading in heterogeneous networks. In *2015 IEEE International Conference on Communications (ICC)*, pages 7731–7736, June 2015.

[22] Hartline J.D. Goldberg A.V. Competitive auctions for multiple digital goods. *Lecture Notes in Computer Science*, 2161:416–427, 2001.