Efficient three-stage auction schemes for cloudlets deployment in wireless access network

Gangqiang Zhou · Jigang Wu · Long Chen · Guiyuan Jiang · Siew-Kei Lam

Published online: 16 April 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

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 Cloudlet · Auction · Mobile cloud computing · Incentive mechanism · Resource allocation

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 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 Sect. 5, we prove the economic properties for the proposed auction model. Simulation results are given in Sect. 6. Finally, Sect. 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 that work is 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. $\text{Cap}^k$ indicates the resource capacity of $C_k$. As defined as in [21], the cost function of cloudlet is

$$C_{\text{os}}(k) = c(k) \cdot w(k),$$

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 $r_k$,

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

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 $\text{Cap}^k$ 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 $\delta$ after a whole auction, such as increasing the value of $\delta$ if its resource is over competitive in the market, and decreasing the value of $\delta$ 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 $i$th 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 $i$th AP, we call them the $i$th group of MUs. Different groups have different amount of tasks to offload. Let $m_j^i$ be the $j$th MU of the $i$th AP. Its valuation for each cloudlet may be different. The mobile user $m_j^i$ may give a higher valuation for the cloudlet it preferred. Then it will submit a much higher bid on those cloudlets based on their valuation. Instead, $m_j^i$ will submit a much lower bid on the cloudlets which $m_j^i$ do not like. Then, the valuations of $m_j^i$ on the kth cloudlet $C_k$ is $v_j^k(k)$, which is private information of $m_j^i$. The budget of $m_j^i$ for $C_k$ is $b_j^k(k)$, 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 $b_j^k(k) = v_j^k(k)$. 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 $p_i^k$ be clearing price of the MU $m_j^i$. If $m_j^i$ is a winner, then $m_j^i$ will be charged at $p_i^k$ after the auction. For the case of that $m_j^i$ bid truthfully, we define its utility $u_i^j$ as

$$u_i^j = \begin{cases} v_j^k(k) - p_i^k & \text{if } m_j^i \in w_i, \\ 0 & \text{otherwise}, \end{cases}$$

where $v_j^k(k)$ is the valuation of $m_j^i$ on the cloudlet $C_k$ it wins. This equation implies the $m_j^i$ 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

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

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 $P'$ be the clearing price of $C_k$. Its utility $u_i^k$ is defined as
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

\[
SW = \sum_{i=1}^{n} \sum_{j=1}^{n_i} u_{ij} + \sum_{i=1}^{n} u_i + \sum_{k=1}^{K} u_{k}.
\]

### 3.2 System model

Table 1 shows that the symbols of the three kinds of entities in our auction schemes. Figure 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 \( R^k_i \) for cloudlet \( C_k \). \( R^k_i \) is calculated according to the budget of the MU group in \( a_i \), and these budgets are their bids for \( C_k \), i.e., \( b^i_j(k) (j \in [1, \ldots, n_i]) \). 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 \( R^k_i \) 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 \( p^i_j(k) \) 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 based on the revenue for each cloudlet. For instance, the budget of \( a_i \) for \( C_k \) is \( B^k_i \) which is the price that \( a_i \) bid for \( C_k \). For each AP, its revenue \( R^k_i \) is provided by its MU group. It is a real value, and the budget \( B^k_i \) is generated by itself, we can easily find that both \( R^k_i \) and \( B^k_i \) 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 \( \sigma () \). 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.

### Table 1 Symbols of participant

| Symbol    | Definition                                      |
|-----------|-------------------------------------------------|
| \( C_k \) | Capacity or workload                            |
| \( a_i \) | Cost, revenue or valuation                      |
| \( m^i_j \) | Reserve price or budget                         |
| \( u^k \) | Clearing price                                  |
| \( W \) | Utility                                         |
| \( W' \) | Winner set                                      |

### Table 2 Symbols in algorithms

| Symbol    | Definition                                      |
|-----------|-------------------------------------------------|
| \( t^i(j) \) | \( m^i \)'s performance price ratio on \( C_k \) |
| \( A \)   | Array of MU sorted by \( t^i(k) \)              |
| \( l_x \) | The workload of the \( x \)th 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 \( L_x \leq C_k^i \) |
| \( S_k \) | The revenue of the first \( x \) MUs in \( A \)  |
| \( m \)   | The independent integer                         |
| \( w^i \) | The potential winner MUs in \( a_i \) for \( C_k \) |
| \( p \)   | The unit price of MUs                           |
| \( p^i_j(k) \) | \( m^i \)'s potential price on \( C_k \)      |
| \( x \)   | The top factor in ACRC, ASC                      |
| \( A' \)  | The randomly sorted AP set                      |
| \( D \)   | The profit matrix \( \{ B^k_i \} \)             |
| \( \sigma \) | Mapping function from \( a_i \) to \( C_k \)    |
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 \( m_i \) bid untruthfully, we define the utility \( \bar{u}_i \) as follows.

\[
\bar{u}_i = \begin{cases} 
  v_i(k) - p_i - \theta & \text{if } m_i \in w_i, \\
  -\theta & \text{otherwise}.
\end{cases}
\]  

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

\[
\bar{u}_i = \begin{cases} 
  R_i^k - P_i - \theta & \text{if } a_i \in W, \\
  -\theta & \text{otherwise}.
\end{cases}
\]  

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 \( u_i \geq \bar{u}_i \) and \( u_i \geq \bar{u}_i \) 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

\[
\sum_{i=1}^{n} \sum_{j=1}^{n} p_i^j \geq \sum_{k=1}^{K} R_k \left( \sum_{i=1}^{n} R_i^k - \sum_{j=1}^{n} p_j^k \right).
\]

3.3.3 Individual rationality

For sellers, they cannot benefit at a price smaller than it’s asked, i.e., \( p_i^k \geq B_i^k \). For buyers, they cannot be charged at a price bigger than it’s bid, i.e., \( b_i^j(k) \geq p_i^j \) 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. The symbols in our algorithms are shown in Table 2.

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 \( (R_i^k) \), 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 \( (B_i^k) \) and the reserve price vector \( \{r_k\} \), 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.

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 \( \{p_i^k\} \)

1: for \( k = 1 \) to \( K \) do
2:   Maximizing the number \( s \) subject to \( L_s \leq Cap^k \) and \( L_s + L_{s+1} > Cap^k \).
3:   If \( L_{n_i} \leq Cap^k \), then \( s = n_i \).
4:   The revenue set \( \{s_x\} = GTR(A, s) \), and the revenue of the first \( s - 1 \) cases is \( S_1, S_2, \ldots, S_{s-1} \).
5:   The integer \( m \) is randomly generated in \( [(s+1)/2, s-1] \).
6:   Then the revenue \( R_i^k = S_m \).
7:   \( a_i \)’s potential winner set for \( C_k \) is \( w_i^k = A_m \).
8:   Then the unit price \( p \) equals to the \( (m + 1) \)th MU’s performance price ratio in \( A \).
9:   if \( m_i^j \in w_i^k \) then
10:      \( p_i^j(k) = L_i \cdot p \)
11:   else
12:      \( p_i^j(k) = 0 \)
13:   end if
14: end for
15: return \( \{R_i^k\}, \{w_i^k\}, \{p_i^k\} \)
Algorithm 2 GTR: Getting the Revenue set

Input: $A$, $s$
Output: The revenue set $\{S_s\}$
1: Let $\{S_s\}$ 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: $L_s$ is total workload of the first $x$ MUs in $A$.
5: Then $S_x = p \cdot L_s$.
6: end for
7: return $\{S_s\}$

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 $R^i_c$ for all cloudlets. Obviously, the revenue $R^i_c$ is calculated from the small group of MUs in $a_i$. Let $t^j_i(k)$ be the performance price ratio of the MU $m^j_i$.

In other words, $t^j_i(k)$ is the unit budget of $m^j_i$ for the cloudlet $C_k$, and it is defined as follows:

$$t^j_i(k) = \frac{b^j_i(k)}{l^j_i}$$

where $l^j_i$ is the workload of $m^j_i$, and the value of $l^j_i$ is kept unchanged no matter which cloudlet receives the tasks offloaded by $m^j_i$. The value of $t^j_i(k)$ will increase with the increasing $b^j_i(k)$, i.e., $m^j_i$ will get a higher performance price ratio on $C_k$ if it has more budget than $C_k$.

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 $t^j_i(k)$. Let $A_r$ be the set of MUs which are the first $x$ ($x \leq n$) members of $A$. Let $l_x$ be the workload of the $x$th MU in $A$, i.e., $l_x$ is the workload of the first $x$ MUs in $A$. Let $L_x$ be the total workload of $A_x$, i.e., $L_x = l_1 + l_2 + \ldots + l_x$. We try to find the index $s$ in $A$ to maximize $L_s$, in which $L_s \leq \text{Cap}_k$ and $L_s + l_{s+1} > \text{Cap}_k$. If the total workload of the MUs in $a_i$ is less than or equal to $\text{Cap}_k$, i.e., $L_m \leq \text{Cap}_k$, then $s = n_r$.

Let $S_m$ be the revenue which is generated by the first $x$ MUs of $A$. Let $S_m = p \cdot L_s$, where $p$ is the unit 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 $\{S_s\}$, where $\{S_s\} = \{S_1, S_2, \ldots, S_{s-1}\}$. 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., $R^i_c = S_m$. The set of potential winner of $a_i$ for $C_k$ consists of the first $m$ MUs of $A$, i.e., $w^k_i = A_m$. The unit price $p$ equals to the performance price ratio of the $(m + 1)$th MU in $A$. For the MU $m^{j+1}_i$ in $a_i$, its potential price on $C_k$ is $p^{j+1}_i(k)$, and $p^{j+1}_i(k) = t^{j+1}_i \cdot p$ if $m^j_i \in w^k_i$, or $p^{j+1}_i(k) = 0$ if $m^j_i \notin w^k_i$. It means if $a_i$ is allocated with $C_k$ after the whole auction scheme, then the MUs which $m^j_i \in w^k_i$ are winners, and they will be charged at $p^{j+1}_i(k)$ by $a_i$. The sum of $\{p^{j+1}_i(k)\}$ equals to $R^{j+1}_i$, i.e., $R^{j+1}_i = \sum_{j=1}^{m} p^{j+1}_i(k)$, 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 $t^j_i(1)$ in descending order at first. Then the order of MUs in the sorted array $A$: $A = \{m^3_4, m^1_4, m^5_4, m^9_4, m^{10}_4, m^6_4, m^3_4, m^7_4, m^8_4\}$. Let $l_x$ be the workload of the $x$th MU in $A$. The workloads of the MUs in $A$ are $(l_1 = 1.4, l_2 = 1.5, l_3 = 1.6, \ldots)$, which are shown in Fig. 2. Let $L_s$ 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 $m^7_4$ and $m^8_4$ which are painted in red are losers in ACRC. Then we calculate the revenue for this $s - 1$ cases. The unit price $p$ for $S_5$ is the $(x + 1)$-th performance price ratio of MU in $A$, and $S_5 = p \cdot L_5$. For instance, the unit price $p$ for $S_5$ is the 6th performance price ratio of MU in $A$, i.e., $p = t^{10}_i(1) = 3.6$. Then, $S_5 = p \cdot L_5 = 3.6 \times 9.1 = 32.76$. We get a random number
within (4, 7). Assume that $m = 6$. We ‘sacrifice’ MUs $m^2, m^3$ which are painted in yellow to keep ACRC truthful. Therefore $R^1_i = S_0 = 32.8$ and the unit price $p = t_i^1(1) = 2.9$. The first 6 MUs in this example form the potential winner set for $C_1$, i.e., $w^1_i = \{m^4, m^1, m^5, m^9, m^8, m^10\}$. For these MUs, their potential price $p^1_i(k) = p \cdot t_i^1$. In this example, their potential price in $A$ is $p^1_i(1) = 2.9 \times 1.4 = 4.06$, $p^2_i(1) = 2.9 \times 1.5 = 4.35$, $p^3_i(1) = 2.9 \times 1.6 = 4.64$, and $p^4_i(1) = 6.96$, $p^5_i(1) = 6.38$, $p^6_i(1) = 6.38$. For the rest of MUs $m^j \notin w^1_i$, their potential price $p^1_i(k) = 0$. Then we can get the potential price set $\{p^1_i(1)\}$. 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 $\{R^k_i\}$ is formed. Then APs will bid for each cloudlet in the next stage. These bids constitute the budget matrix $\{B^k_i\}$ which means the APs’ budget for each cloudlet. All these APs have submitted their truthful bid if $\{B^k_i\} = \{R^k_i\}$, 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 matrix $D$ at first, where $D = \{B^k_j\} - \{r_k\}$, and $d^k_i = B^k_i - r_k$. 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 $B^k_i - r_k$ 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 $D$, which is the $i$th row of the matrix $D$. Then we select the largest element $d^k_i$ 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 $d^k_i$. Then, the algorithm ASC judges that if their profit is a positive value, i.e., whether $d^k_i > 0$. The budget of $a_i$ is bigger than the reserve price of $C_k$ if $d^k_i > 0$, i.e., if

---

**Table 3 Example for ACRC**

(a) MUs’ performance price ratio on each Cloudlet

| Cloudlet | $t^1_i(k)$ | $t^2_i(k)$ | $t^3_i(k)$ | $t^4_i(k)$ | $t^{10}_i(k)$ |
|----------|-------------|-------------|-------------|-------------|---------------|
| C_1      | 6           | 2.9         | 2.7         | 6.4         | 5.6           |
| C_2      | 6           | 2.5         | 4.5         | 5.7         | 3.1           |
| ...      | $t^1_i(k)$  | $t^2_i(k)$  | $t^3_i(k)$  | $t^4_i(k)$  | $t^{10}_i(k)$ |
| 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)

| $l^1_i$ | $l^2_i$ | $l^3_i$ | $l^4_i$ | $l^5_i$ | $l^6_i$ | $l^7_i$ | $l^{10}_i$ |
|---------|---------|---------|---------|---------|---------|---------|-----------|
| 1.5     | 2.7     | 2.2     | 1.4     | 1.6     | 2.2     | 2.5     | 2.3       | 2.4       | 2.2       |

(c) Cloudlets’ resource capacity

| $Cap^1$ | $Cap^2$ | $Cap^3$ | $Cap^4$ | $Cap^5$ | $Cap^b$ | $Cap^7$ | ... |
|---------|---------|---------|---------|---------|---------|---------|-----|
| 17      | 22      | 25      | 11      | 19      | 21      | 18     | ... |

---

**Fig. 2 Illustration of ACRC in TACD**

---

© Springer
Algorithm 3 ASC: APs' auction to Select suitable Cloudlet

Input: \{B\}_i, \{r_k\}, D
Output: W, W', \{P_i\}, \{P^k\}, \sigma
1: Distributing APs randomly into A'.
2: for \(x = 1 \to n\) do
3: Getting AP \(a_i\) and its matching cloudlet \(C_k\) by algorithm FRM(D, A', x).
4: if \(d^i_k > 0\) then
5: if \(B^i_k \geq B^j_k \geq \ldots \geq r_k\), which \(j \neq i\) then
6: \(\sigma(i) = k\)
7: \(P_i = P^k = B^i_k\)
8: \(W = W \cup a_i\)
9: \(W' = W' \cup C_k\)
10: Setting the values of elements in \(i\)th column of \(D\) to 0
11: Setting the values of elements in \(k\)th column of \(D\) to 0
12: else
13: \(d^i_k = 0\)
14: end if
15: end if
16: end for
17: return \(W, W', \{P_i\}, \{P^k\}, \sigma\)

Algorithm 4 FRM: Finding a Rational Matching to \(a_i\)

Input: D, A', x
Output: AP \(a_i\) and its matching cloudlet \(C_k\)
1: Let \(a_i\) denote the \(x\)th AP of A'.
2: Let vector \(D_i\) be the \(i\)th row of matrix D.
3: \(d^i_k\) is the maximum of \(D_i\).
4: return \(a_i, C_k\)
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 top1 must be larger than 1. In this case, let $S_{\text{max}}$ be the most profitable revenue of $S$, we cannot fix the revenue of AP at $S_{\text{max}}$. This is because, we cannot keep ACRC truthful if we always choose $R_i^c = S_{\text{max}}$. For instance, $S_{\text{max}} = S_7$, i.e., 40.3 in Fig 3 and the unit price $p = t_{i0}^b(2)$, i.e., 2.9 while MUs bid truthfully. For $m_i^p$, it’s valuation on $C_2$ is $v_i^c(2)$ where $v_i^c(2) = b_i^c(2)$, i.e., 4.96, its potential price $p_i^c = p \cdot t_i^c = 2.9 \cdot 1.6 = 4.64$. We assume that $a_i$ wins $C_2$ in stage II, then $m_i^2$ will be charged at clearing price $p_i^c = p_i^c(2) = 4.64$ in stage III. Therefore, the utility of $m_i^2$ is $u_i^c$ where $u_i^c = v_i^c(2) - p_i^c = 4.96 - 4.64 = 0.32$. However, if $m_i^2$ bid untruthfully, we assume that $m_i^2$ changes its budget on $C_2$ to $b_i^c(2) = 4.32$ which is less than its valuation on $C_2$. Then the performance price ratio of $m_i^2$ on $C_2$ is $t_i^c(2)$ where $t_i^c(2) = 2.7$ and it will be sorted behind $t_i^{c0}(2)$ according to ACRC. Let $L_7 = L_6 + t_{i0}^b = 12.3 + 2.2 = 14.5$, $L_6 = L_5 + t_i^c = 14.5 + 1.6 = 16.1$, and $S_8 = L_5 \times 2.9 = 35.67$, $S_7 = L_7 \times 2.7 = 39.15$, $S_8 = L_6 \times 2.5 = 40.25$, then $S_{\text{max}} = S_8$. If $R_i^c$ always equal to the most profitable revenue, then $R_i^c = S_8$ and its unit price $p = 2.5$. We assume that the matching result are the same in stage II, then $m_i^2$ will be charged at the clearing price $p_i^c = p_i^c(2) = t_i^c \cdot p = 1.6 \times 2.5 = 4$ in stage III. Then, if $m_i^2$ bids untruthfully, its utility is $\bar{u}_i^c$ where $\bar{u}_i^c = v_i^c(2) - p_i^c - \theta = 4.96 - 4 - \theta = 0.96 - \theta$. $m_i^2$ can improve its utility if the value of the extra cost $\theta$ is small enough, e.g., $\theta < 0.96$, when $m_i^2$ 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 top2 be a small number, it is the top factor in FRMG, its default value is 2. For each round, FRMG gets a random integer $\text{rnd}$ in $[1, \text{top2}]$, then selects the $\text{rnd}$-th 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 $\text{top2} > 1$ which is similar with $\text{top1}$. It will be discussed later.

The 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 $m$.

| 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 top1 cases | ASC+FRM  |
| TACDpp  | ACRC+GTR | One of top1 cases | ASC+FRM+FRMG |

5 Desired properties

5.1 Truthfulness

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

Proof To verify the truthfulness of ACRC, we only need to prove that MUs are truthful in our auction. In TACD, for the MU $m_i^j$, $b_i^c(k)$ is the truthful bid of $m_i^j$. Let $\bar{b}_i^c(k)$ be the untruthful bid. Then the utility of $m_i^j$ is $u_i^j$ when it bids truthfully. Let $\bar{u}_i^j$ 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., $\bar{u}_i^j \leq u_i^j$.

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

1. MU $m_i^j$ fails in the auction both in truthful bid $b_i^c(k)$ and untruthful bid $\bar{b}_i^c(k)$. Then, $u_i^j = 0$ and $\bar{u}_i^j = -\theta$.

2. MU $m_i^j$ wins the auction while bid truthfully and fails in the auction while bid untruthfully. In this case, $u_i^j \geq 0$, and $\bar{u}_i^j = -\theta$.

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 $\tilde{u}_i^j = u_i^j - 0$.

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., $c \geq b_i^j(k)$. And if $m_i^j$ wins the auction in another bid $\tilde{b}_i^j(k)$, it must have $\tilde{b}_i^j(k) \geq c$, so $\tilde{b}_i^j(k) > b_i^j(k)$ and $\tilde{b}_i^j(k) > v_i^j(k)$, then we have $\tilde{u}_i^j \leq u_i^j = 0$.

We have now discussed the truthfulness of MUs in TACD, while MU $m_i^j$ bid for the $k$th cloudlet. And the other cloudlets do not need care about whether $m_i^j$ cheat or not, if the $k$th 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 $m$.

\textbf{Theorem 2} The schemes TACD, TACDp and TACDpp are truthful in ASC.

\textbf{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 are 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. \hfill \Box

5.3 Individual rationality

\textbf{Theorem 4} The schemes TACD, TACDp and TACDpp are subject to the individual rationality.

\textbf{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 $p_i^j = p \cdot \cdot \cdot$ the performance price ratio of the $n$th MU in $A$, and $p \leq b_i^j(k)$. 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, according to the ASC, for the clearing price of AP, we have $P_i \leq B_i^k$. Therefore, $R_i^k = B_i^k \geq P_i$.

The proof of individual rationality for TACDp and TACDpp is the same as that of TACD. \hfill \Box

5.4 Computational efficiency

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

\textbf{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 $O(K \cdot n \log n)$. \hfill \Box

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

\textbf{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 FRM takes $O(n \cdot K)$ time, which is different from the algorithm FRM. Thus, the total time complexity of ASC in TACDpp is $O(K \cdot n^2)$. Therefore the total time complexity of TACDpp is $O(K \cdot n^2)$. \hfill \Box

\sum_{i=1}^{n} \sum_{j=1}^{n} p_i^j \geq \sum_{k=1}^{K} P_k^k + \left( \sum_{i=1}^{n} R_i^k - \sum_{i=1}^{n} P_i^k \right).
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 \( N(0.75, 0.1) \). 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 \( N(2, 1) \). 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 \( B_k^i \). 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 \( m \)th 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 \( B_k^i \geq r_k \), 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

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 Figs. 4, 5, and 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. Figure 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 \( B_j^i \) other than \( B_k^i \) and \( r_k \) to keep ASC truthful where \( B_j^k \geq B_k^k \geq r_k \). 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 \( B_j^k \geq r_k \), 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 $m$th 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 $s$th 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. 6. 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.

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 $B^*_1$. 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 $B^*_1$ are shown in Figs. 9, 10 and 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. $B^*_1 = 85.5$. The dotted line shows the profit of $a_1$ when it bids untruthfully from $B^*_1 = B^*_1 - 80$ to $B^*_1 = B^*_1 + 50$ with the increase unit of 1. The result is the averaged over 100 random instances. Figure 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_i 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_i 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 have 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_i is \( \bar{U}_1 \) where \( \bar{U}_1 = 22.7 - \theta \), while a_i bid untruthfully among \{64.5, 65.5, 66.5\}. \( \bar{U}_1 \) is larger than \( \bar{U}_1 \), if \( \theta < 4 \). It is because that, when \( B_1^i = \{64.5, 65.5, 66.5\} \), the profit \( B_1^i - r_1 \) is so big that a_i still wins C_1. Also, there is another AP a_i whose budget is \( B_1^i \) where \( B_1^i \leq 64.5 \), and it is the largest \( B_1^i \) in which \( B_j^i \leq B_1^i, j \in [1, n] \) and \( j \neq 1 \). Then, the clearing price will be much lower than that when it bids truthfully. Therefore, if AP a_i pays some extra price \( \theta \) 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_i 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 \( B_k^i - r_k \) is the most profitable one.
2. Matching cloudlet C_k with AP a_i whose profit \( B_k^i - r_k \) is the second profitable one.

So the utility of a_i is not a stable value, even a_i 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_i while it bids untruthfully.

There are also some more profitable cases while a_i bids untruthfully, such as \{64.5, 65.5, 66.5\} as occur ed as in the case of top_2 = 1. But the difference is that, if top_2 = 2, a_i can also benefit more in those cases with the probability of 50%. Otherwise, a_i will be matched with other less profitable cloudlets, and it also must pay an extra cost \( \theta \) to find those cases. Therefore, there is not any evident case in which a_i can get more utility than the truthful case. It is worthless for a_i to pay an extra cost \( \theta \) to determine how to bid untruthfully. Therefore, TACDpp is truthful while top_2 = 2.

Similarly, Fig. 11 shows the utility of a_i 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 \( B_j^i \). j in [1, 5]. The mixed strategy, the utility of a_i varies with a larger range than that in Fig. 10 while a_i 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 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. The comparison of the result in different cases is shown in Fig. 12. For truthfulness, there is no evidence that a_i 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 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.

Acknowledgements This work was supported by the National Natural Science Foundation of China under Grant Nos. 61702171, by the Major Research Project of Educational Commission Science Foundation of China under Grant Nos. 61702115 and This work was supported by the National Natural

References

1. Satyanarayanan, M., Bahl, P., Caceres, R., & Davies, N. (2009). The case for vm-based cloudlets in mobile computing. IEEE Pervasive Computing, 8(4), 14–23.
2. Kumar, K., Liu, J., Lu, Y.-H., & Bhargava, B. (2013). A survey of computation offloading for mobile systems. Mobile Networks and Applications, 18(1), 129–140.
3. Jia, M., Liang, W., Xu, Z., Huang, M. (2016). Cloudlet load balancing in wireless metropolitan area networks. In IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications, pp. 1–9.
4. Fang, F., Zhang, H., Cheng, J., Leung, V. C. M (May 2017). Energy-efficient resource scheduling for noma systems with imperfect channel state information. In 2017 IEEE International Conference on Communications (ICC), (pp. 1–5).
5. Xia, Q., Liang, W., & Xu, W. (Oct 2013). Throughput maximization for online request admissions in mobile cloudlets. In 38th Annual IEEE Conference on Local Computer Networks, (pp. 589–596).
6. Satyanarayanan, M. (2001). Pervasive computing: vision and challenges. IEEE Personal Communications, 8(4), 10–17.
7. Jia, M., Cao, J., & Liang, W. (2017). Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks. IEEE Transactions on Cloud Computing, 5(4), 725–737.
8. Xu, Z., Liang, W., Xu, W., Jia, M., & Guo, S. (Oct 2015). Capacitated cloudlet placements in wireless metropolitan area networks. In 2015 IEEE 40th Conference on Local Computer Networks (LCN), (pp. 570–578).
9. Jin, A. L., Song, W., & Zhuang, W. (2018). Auction-based resource allocation for sharing cloudlets in mobile cloud computing. IEEE Transactions on Emerging Topics in Computing, 6(1), 45–57.
10. Samimi, P., Teimouri, Y., & Mukhtar, M. (2016). A combinatorial double auction resource allocation model in cloud computing. Information Sciences, 357, 201–216.
11. Lin, P., Feng, X., Zhang, Q., & Hamdi, M. (2013). Groupon in the air: A three-stage auction framework for spectrum group-buying. In 2013 Proceedings IEEE INFOCOM, (pp. 2013–2021).
12. Liu, Y., Lee, M. J., & Zheng, Y. (2016). Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system. IEEE Transactions on Mobile Computing, 15(10), 2398–2410.
13. Rahimi, M. R., Ren, J., Liu, C. H., Vasilakos, A. V., & Venkatasubramanian, N. (2014). Mobile cloud computing: A survey, state of art and future directions. Mobile Networks and Applications, 19(2), 133–143.
14. Chen, M., Hao, Y., Li, Y., Lai, C. F., & Wu, D. (2015). On the computation offloading at ad hoc cloudlet: architecture and service modes. IEEE Communications Magazine, 53(6), 18–24.
15. Chen, M., Hao, Y., Lai, C. F., Wu, D., Li, Y., & Hwang, K. (2017). Opportunistic task scheduling over co-located clouds in mobile environment. IEEE Transactions on Services Computing, (pp. 1–13). https://doi.org/10.1109/TSC.2016.2589247.
16. Zhang, H., Jiang, C., Beaulieu, N. C., Chu, X., Wang, X., & Quek, T. Q. S. (2015). Resource allocation for cognitive small cell networks: A cooperative bargaining game theoretic approach. IEEE Transactions on Wireless Communications, 14(6), 3481–3493.
17. Parsons, S., Rodriguez-Aguilar, J. A., & Klein, M. (2011). Auctions and bidding: A guide for computer scientists. ACM Computing Surveys, 43(2), 10:1–10:59.
18. Zhang, Y., Lee, C., Niyato, D., & Wang, P. (2013). Auction approaches for resource allocation in wireless systems: A survey. IEEE Communications Surveys Tutorials, 15(3), 1020–1041.
19. Yang, D., Xue, G., & Zhang, X. (2014). Truthful group buying-based spectrum auction design for cognitive radio networks. In 2014 IEEE International Conference on Communications (ICC), (pp. 2295–2300).
20. Zhou, G., Wu, J., & Chen, L. (2017). Tacd: A three-stage auction scheme for cloudlet deployment in wireless access network. In L. Ma, A. Khreishah, Y. Zhang, & M. Yan (Eds.), Wireless Algorithms, Systems, and Applications (pp. 877–882). Cham: Springer.
21. Kang, X., & Sun, S. (June 2015). Incentive mechanism design for mobile data offloading in heterogeneous networks. In 2015 IEEE International Conference on Communications (ICC), (pp. 7731–7736).
22. Hartline, J. D., & Goldberg, A. V. (2001). Competitive auctions for multiple digital goods. Lecture Notes in Computer Science, 2161, 416–427.

Gangqiang Zhou received his Bachelor degree of Engineering in Communication from Hunan University of Science and Technology, China in 2013. He is working toward the Master degree in the School of Computer Science and Technology, Guangdong University of Technology, Guangzhou, China. His main research interests include cloud computing, incentive mechanisms and vehicular networks.
Jigang Wu received B.Sc. degree from Lanzhou University, China in 1983, and doctoral degree from the University of Science and Technology of China in 2000. He was with the Center for High Performance Embedded Systems, Nanyang Technological University, Singapore, from 2000 to 2010, as a research fellow. He was Dean, Tianjin distinguished professor of School of Computer Science and Software, Tianjin Polytechnic University, China, from 2010 to 2015. Now he is distinguished professor of School of Computer Science and Technology, Guangdong University of Technology as a postdoctoral fellow within the excellent postdoctoral research fellow program, under the supervision of Prof. Jigang Wu. His main research interests include network economics, mobile computing, cognitive radio, vehicular ad-hoc networks and energy harvesting.

Long Chen received his Bachelor degree of Engineering in Computer Science from Anhui University, Hefei, Anhui, China in 2011. After that, he was admitted to the University of Science and Technology of China (USTC) exempted from national graduate school entrance examination. In 2013, he became a Ph.D. candidate in the School of Computer Science and Technology, USTC. He received doctoral degree in 2016. Now he is in Guangdong University of Technology as a postdoctoral fellow within the.

Guiyuan Jiang received the B.S. degree from Northwest University for Nationalities, China, the M.Eng. degree from Tianjin Polytechnic University, China, and the doctoral degree from Tianjin University (TJU), all in Computer Science and Technology. Currently, he works as a research fellow at School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include data analytics for citywide transportation modeling and optimization, design methodologies for reconfigurable computing systems, resource optimization for data-center and sensor networks.

Siew-Kei Lam received his BASc, M.Eng. and Ph.D. from School of Computer Science and Engineering (SCSE), NTU. He was a Visiting Research Fellow in the Imperial College of London, University of Warwick, and RWTH Aachen, Germany. He is currently an Assistant Professor in SCSE and his research focuses on devising custom computing solutions to meet the challenging demands of energy-efficiency, reliability and security in embedded systems. His current projects include architecture-aware algorithms for vision-enabled sensing, design methodologies for secure and reliable embedded systems, and transportation analytics.