Int. J. Appl. Math. Comput. Sci., 2023, Vol. 33, No. 2, 327–339
DOI: 10.34768/amcs-2023-0024

A COMBINATORIAL AUCTION MECHANISM FOR TIME–VARYING MULTIDIMENSIONAL RESOURCE ALLOCATION AND PRICING IN FOG COMPUTING

SHIYONG LI [a], YANAN ZHANG [a], WEI SUN [a], JIA LIU [b,c,*]

[a] School of Economics and Management
Yanshan University
No. 438 West Hebei Avenue, Haigang District, Qinghuangdao 066004, China
e-mail: shiyongli,wsun@ysu.edu.cn,15831232797@163.com

[b] State Key Laboratory of Media Convergence and Communication
Communication University of China
No. 1 Dingfuzhuang East Street, Chaoyang District, Beijing 100024, China
e-mail: jialiu@cuc.edu.cn

[c] School of Economics and Management
Communication University of China
No. 1 Dingfuzhuang East Street, Chaoyang District, Beijing 100024, China

It is a hot topic to investigate resource allocation in fog computing. However, currently resource allocation in fog computing mostly supports only fixed resources, that is, the resource requirements of users are satisfied with a fixed amount of resources during the usage time, which may result in low utility of resource providers and even cause a waste of resources. Therefore, we establish an integer programming model for the time-varying multidimensional resource allocation problem in fog computing to maximize the utility of the fog resource pool. We also design a heuristic algorithm to approximate the solution of the model. We apply a dominant-resource-based strategy for resource allocation to improve resource utilization as well as critical value theory for resource pricing to enhance the utility of the fog resource pool. We also prove that the algorithm satisfies truthful and individual rationality. Finally, we give some numerical examples to demonstrate the performance of the algorithm. Compared with existing studies, our approach can improve resource utilization and maximize the utility of the fog resource pool.

Keywords: fog computing, combinatorial auction, time-varying multidimensional resource allocation, resource pricing.

1. Introduction

With the rapid development of IoT applications, the traditional cloud computing faces some challenges including latency, network bandwidth (Song *et al.*, 2021b; 2021a), and security problems (Ghobaei-Arani *et al.*, 2020). To overcome these challenges, a new distributed computing technology called fog computing has been proposed by Cisco, which extends the core of the cloud data center to the edge of the network (Sharghivand *et al.*, 2021). The basic architecture of fog computing has three layers, which include cloud servers, edge servers and mobile devices. Edge servers act as the middle layer between cloud servers and mobile devices to meet the computational requirement of latency-sensitive applications (Ghobaei-Arani *et al.*, 2020; Chang *et al.*, 2019). The advantage is that edge servers are placed closer to mobile devices to achieve better computing power and less transmission delay, which is a win-win situation for both fog resource providers and end users. Furthermore, fog computing is a service platform that can integrate CPU, memory, storage and other resources into fog nodes to form a fog resource pool to provide services for end users.

At present, resource allocation in fog computing has

[*] Corresponding author

been attracting much attention. Resource allocation approaches are categorized into two methods: auction-based and optimization (Ghobaei-Arani *et al.*, 2020, Li *et al.*, 2019; 2023; Li and Sun 2021). Many resource allocation problems can be transformed into a knapsack problem (Leao *et al.*, 2014; Angelelli and Filippi, 2011). For this reason, resource providers have introduced auction mechanisms in fog computing to obtain more profit, allowing idle resources to be sold at dynamic prices (Zhang *et al.*, 2020).

In the vehicle fog computing (VFC) field, Lee *et al.* (2020) proposed a VCG auction mechanism for actual service allocation. However, they did not consider the costs of users. Subsequently, for VFC parking assistance, Zhu *et al.* (2020) set forth a VFC parking assistance allocation strategy (RAFC) based on the reverse auction. This approach can reduce users' costs, but it only considers price attributes. Peng *et al.* (2020) proposed a double auction mechanism based on multiple features, but they did not take latency requirements into account. Then, Junior *et al.* (2021) established a request-processing-response-actuation programming model based on a distributed auction protocol to match clients and servers and meet latency requirements.

In the hybrid fog-cloud environment, in response to the low utilization of fog resources, Sun *et al.* (2020) proposed a general IoT-fog-cloud computing architecture based on a sealed-bid bilateral auction. Moreover, Besharati *et al.* (2021) set forth a second-price sealed-bid auction mechanism, which is significantly better than other methods in terms of execution time and energy consumption. Further, to reduce response time, Aggarwal *et al.* (2021) proposed a fog-integrated cloud auctioning model (FICAM). Then, Houshyar *et al.* (2021) applied the Nash equilibrium and auction to improve resource allocation, but they did not take energy consumption and execution time into account. Finally, Guo *et al.* (2020) presented an auction method to reduce electric energy consumption and execution time of fog nodes and servers.

In the mobile communication field, Han *et al.* (2019) proposed a combinatorial auction problem and suggested a joint processing strategy from a market perspective. Later, Tasiopoulos *et al.* (2018) constructed an auction-based resource allocation and supply mechanism. Still, the efficiency and network utilization were low. Therefore, Kayal and Liebeherr (2019) proposed a game-theoretic approximation method inspired by an iterative combinatorial auction. Then, Bandyopadhyay *et al.* (2020) set forth two types of truthful resource allocation and pricing mechanisms. Later, Bermbach *et al.* (2020) proposed an auction-based approach in which application developers bid for resources. In addition, Baranwal and Kumar (2020) recommended a decentralized auction to improve the utilization of fog resources.

We can find from previous studies above that there are some interesting research results on auction-based resource allocation of fog computing. However, most of them are dedicated to allocating fixed resources, which may result in low utility of resource providers and even cause a waste of resources (Ghobaei-Arani *et al.*, 2020). Also, few studies have considered the allocation of time-varying multidimensional resources. On the contrary, there are more studies in cloud computing. For example, Zhang *et al.* (2019) established an integer programming model based on limited task resource requirements at different times. However, users with more profit may fail to match the resource due to lower ranking. Therefore, Zhang *et al.* (2020) further proposed a waiting period strategy to improve social welfare and resource utilization. At the same time, they proposed a pricing algorithm according to the greedy mechanism offered by Zaman and Grosu (2012). It has the characteristics of high social welfare, high resource utilization and short execution time compared with the online virtual machine allocation and pricing mechanism (OVMAP) proposed by Mashayekhy *et al.* (2016).

In summary, time-varying multidimensional resource allocation in cloud computing has achieved some interesting results in the past years, which is equally essential in fog computing today. However, existing allocation algorithms in fog computing cannot be directly applied to time-varying multidimensional resource allocation. Therefore, we propose an integer programming model and a heuristic algorithm to calculate resource allocation and resource pricing. We also apply a dominant-resource-based strategy for resource allocation to improve resource utilization and propose a critical value theory for resource pricing to improve the utility of the fog resource pool. Finally, we give some numerical examples to demonstrate the performance of the algorithm. Compared with existing research results, our approach can improve resource utilization and maximize the utility of the fog resource pool.

2. Time-varying multidimensional resource allocation problem

2.1. Model description. As shown in Fig. 1, in the network, a complete resource allocation system consists of a cloud resource provider, a fog resource pool, a third-party platform and end users. In this paper, we mainly consider resource allocation among the fog resource pool, the third-party platform and the end users. The fog resource pool is a virtual integration of resources of fog nodes, which can provide services to end users. Specifically, it refers to a fog cluster composed of a cluster head and multiple cluster members. Among them, the cluster head is mainly responsible for collecting and summarizing the data of other nodes in the cluster and

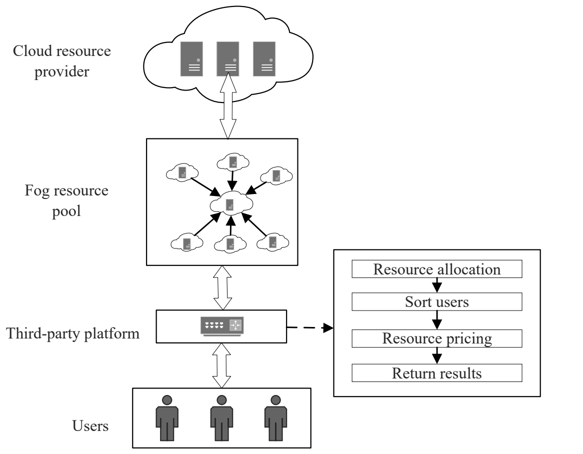

Fig. 1. Basic architecture of resource allocation in fog computing.

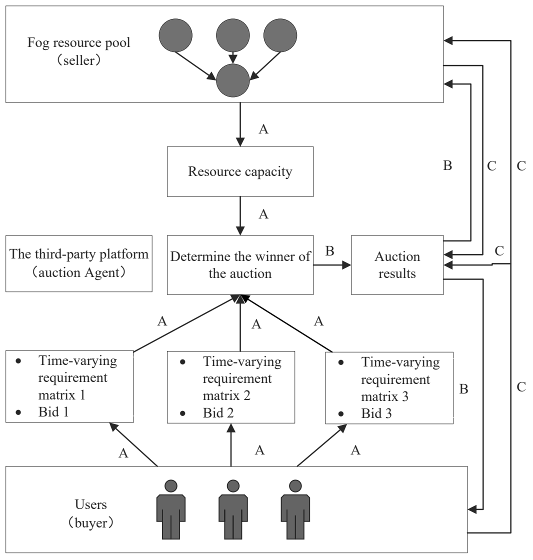
Fog resource pool (seller)
Resource capacity
The third-party platform (auction Agent)
Determine the winner of the auction
Auction results
Time-varying requirement matrix 1
Bid 1
Time-varying requirement matrix 2
Bid 2
Time-varying requirement matrix 3
Bid 3
Users (buyer)
A
B
C

Fig. 2. Schematic diagram of the relationship between three parties in a combinatorial auction.

transmitting the total resource data to the third-party platform.

In this paper, we investigate the time-varying multidimensional resource allocation problem (TMRAP) in fog computing, mainly focusing on offline resource allocation. When end users have the same arrival time slots and execution times, the fog resource pool provides service to end users through different types of resources. The model maximizes the utility of the fog resource pool within the scope of users' bids based on satisfying the resource requirements of end users, while ensuring that the consumption rate of each resource is as consistent as possible to improve the utilizations of resources and the success matching rates of end users. The relationship between the fog resource pool, the third-party platform, and the end users is shown in Fig. 2.

As shown in Fig. 2, in the combinatorial auction model, the end users act as buyers to submit their resource requirements and bids, and the fog resource pool acts as a seller to provide resource services. The third-party platform acts as an auction agent responsible for receiving information, coordinating allocation, and informing both parties about the final results. The auction process A indeed consists of two parts. One part means that the fog resource pool submits resource capacity to the third-party platform. The other part means that each user offers its time-varying requirement matrix and bid to the third-party platform. The process B indicates that the third-party platform determines the winner of the auction. It calculates resource allocation and pricing based on the information submitted by the fog resource pool and the end users, and it finally informs both buyers and sellers of auction results. The process C indicates that, when the auction is successful, the fog resource pool will pay the agency fee to the third-party platform, and the end users also pay the agency fees to the third-party platform. At the same time, each user needs to pay the transaction price to the fog resource pool. The notation used in this paper is summarized in Table 1.

2.2. Utility function.

2.2.1. Utility function of user m.

In Eqn. (1), for each user m, utility function U_m consists of three parts. The first part, H_m, is the maximum expense that user m is willing to pay. The second part, V_m^f, is the payment that user m offers to fog resource pool f when the transaction is successful, and the third part, V_m^l, is the payment that user m offers to third-party platform l:

$$U_m = H_m - V_m^f - V_m^l. \tag{1}$$

The second part is given by

$$V_m^f = a_m p_m, \tag{2}$$

in which a_m indicates the requirement satisfaction of user m, i.e., $a_m = 1$ indicates that the requirement is satisfied and otherwise $a_m = 0$, and p_m represents the final transaction price.

2.2.2. Utility function of fog resource pool f.

For fog resource pool f, utility function U_f consists of two parts

$$U_f = \sum_{m \in M} (V_m^f - V_f^l). \tag{3}$$

Table 1. Notation list.

Notation	Meaning
M	set of users, $M = \{1, 2, \dots, I\}$, $m \in M$
f	fog resource pool f
l	third-party platform l
R	set of resources, $R = \{1, 2, \dots, N\}$, $r \in R$
T	total time
B_m	bid requirement of user m
x_m	time-varying resource requirement matrix of user m
x_{mr}^t	amount of resources r required by user m at time t
H_m	maximum expense that user m is willing to pay
h_m	maximum bid that user m is willing to pay
p_m	transaction price of user m
c_r	capacity of resource r
c_r^t	remaining amount of resource r at time t
$\mathbf{c}$	vector of the capacities of different resources
$\mathbf{C}$	matrix of the remaining amount of resources at each time
a_m	matching indicator of user m
v_m	dominant resource proportion of user m
d_m	bid density of user m
U_m	utility generated by user m
U_f	utility generated by fog resource pool f
U_l	utility generated by third-party platform l
V_m^f	payment of user m to fog resource pool f
V_m^l	payment of user m to third-party platform l
V_f^l	payment of fog resource pool f to third-party platform l
C_l	cost generated by third-party platform l

The first part, $\sum_{m\in M} V_m^f$, is the resource revenue of providing resources, and the second part, V_f^l, is the payment that fog resource pool f offers to third-party platform l.

The first part, $\sum_{m\in M} V_m^f$, is the sum of all transaction prices. The second part is made specific as

$$V_f^l = \begin{cases} a_m V_1, & p_m \le p_1, \\ a_m\Big\{\frac{V_2-V_1}{p_2-p_1}(p_m-p_1)+V_1\Big\}, & p_1 < p_m < p_2, \\ a_m V_2, & p_m \ge p_2. \end{cases} \tag{4}$$

Cost V_f^l is a function of p_m, (p_1, V_1) and (p_2, V_2) are the coordinates of two points on curve V_f^l, and p_1, p_2, V_1, V_2 represent constants and $p_1 < p_2$, $V_1 < V_2$.

2.2.3. Utility function of third-party platform *l*.

In (5), for third-party platform l, utility function U_l consists of two parts. The first part, $\sum_{m\in M}(V_m^l + V_f^l)$, is the agency fee paid by users and fog resource pool f to third-party platform l when the transaction is successful, and the second part, C_l, is the cost of the allocation service provided by third-party platform l:

$$U_l = \sum_{m\in M}(V_m^l + V_f^l) - C_l. \tag{5}$$

2.3. Model formulation.

In this section, we introduce the objective function of the time-varying multidimensional resource allocation model based on a combinatorial auction in fog computing. Additionally, we assume that the set of users is $M = \{1, 2, \dots, I\}$, and each element is user $m \in M$. The set of resources is $R = \{1, 2, \dots, N\}$, and each element is resource $r \in R$. Additionally, we assume that fog resource pool f provides N types of resources, such as a CPU, memory and storage. The capacities of resources are represented by vector

$$\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{bmatrix}.$$

Then we can get the multidimensional resource capacity matrix as follows:

$$\mathbf{C} = \begin{bmatrix} c_1^1 & c_1^2 & \dots & c_1^T \\ c_2^1 & c_2^2 & \dots & c_2^T \\ \vdots & \vdots & \ddots & \vdots \\ c_N^1 & c_N^2 & \dots & c_N^T \end{bmatrix}. \tag{6}$$

We assume that the arrival time slots and execution times of users are the same, that is, each user $m \in M$ submits the resource requirement and bid $\mathbf{B}_m = (\mathbf{x}_m, h_m)$ at time slot $t = 1$ and the execution interval is $[1, T]$, where h_m represents the maximum bid that user m is willing to pay, and time-varying requirement matrix $\mathbf{x}_m$ is

$$\mathbf{x}_m = \begin{bmatrix} x_{m1}^1 & x_{m1}^2 & \dots & x_{m1}^T \\ x_{m2}^1 & x_{m2}^2 & \dots & x_{m2}^T \\ \vdots & \vdots & \ddots & \vdots \\ x_{mN}^1 & x_{mN}^2 & \dots & x_{mN}^T \end{bmatrix}. \tag{7}$$

For each user m, we introduce variable a_m to indicate the requirement satisfaction of user m, that is, $a_m = 1$ indicates that the requirement is satisfied and otherwise $a_m = 0$.

We denote by U_f the utility function generated by fog resource pool f, which is described in Eqn. (3). Therefore, the time-varying multidimensional resource allocation problem (TMRAP) in fog computing can be modeled as an integer programming problem as follows:

$$M: \quad \max \quad U_f \tag{8}$$

subject to

$$\sum_{m\in M} a_m x_{mr}^t \le c_r^t, \tag{8a}$$

$$\sum_{m\in M} a_m \le M, \tag{8b}$$

over

$$a_m \in \{0,1\}, \quad 0 \le p_m \le h_m, \tag{8c}$$

$$\forall m \in M, \quad \forall r \in R, \quad \forall t \in [1,T]. \tag{8d}$$

The objective (8) indicates that the goal of the time-varying multidimensional resource allocation problem is to maximize the utility of fog resource pool f. The inequality (8a) indicates that the resource allocation at any time t should not exceed the capacity of any resource. The inequality (8b) indicates that the user is allocated resources at most once during $[1, T]$. The constraint (8c) indicates that the value of a_m is 0 or 1. The actual transaction price p_m is not higher than bid h_m.

2.4. Model analysis.

Substituting (2) into (3), we can obtain utility function U_f of fog resource pool f,

$$U_f = \sum_{m\in M} (V_m^f - V_f^l) = \sum_{m\in M} (a_m p_m - V_f^l). \tag{9}$$

Then, substituting (4) into (9), we can analyze utility function U_f of fog resource pool f as follows:

(i) If $p_m \le p_1$,

$$U_f = \sum_{m\in M} (V_m^f - V_f^l) = \sum_{m\in M} (a_m p_m - a_m V_1) = \sum_{m\in M} a_m(p_m - V_1),$$

(ii) If $p_m \ge p_2$,

$$U_f = \sum_{m\in M} (V_m^f - V_f^l) = \sum_{m\in M} (a_m p_m - a_m V_2) = \sum_{m\in M} a_m(p_m - V_2).$$

(ii) If $p_1 < p_m < p_2$,

$$U_f = \sum_{m\in M} (V_m^f - V_f^l) = \sum_{m\in M} a_m\Big\{p_m - \Big\{\frac{V_2 - V_1}{p_2 - p_1}(p_m - p_1) + V_1\Big\}\Big\}.$$

Finally, we can obtain utility function U_f of fog resource pool f as

$$V_f^l = \begin{cases} \sum_{m\in M} a_m(p_m - V_1), & p_m \le p_1, \\ \sum_{m\in M} a_m\{p_m - \big\{\frac{V_2-V_1}{p_2-p_1}(p_m - p_1) + V_1\big\}\big\}, & p_1 < p_m < p_2, \\ \sum_{m\in M} a_m(p_m - V_2), & p_m \ge p_2. \end{cases} \tag{10}$$

In summary, the TMRAP is a 0-1 integer programming problem, which belongs to non-convex optimization problems. The goal is to maximize the utility of the fog resource pool. According to (10), we know that the goal of the TMRAP mainly depends on transaction price p_m. Accordingly, the TMRAP is to achieve resource allocation with an objective function of transaction price p_m. However, the 0-1 integer programming problem has been proven to be NP-hard. Most of the traditional algorithms for solving this kind of problem take too much computing time, and they even exhibit difficulty in obtaining an optimal result when there are a large number of users. Therefore, the traditional algorithms have limited practical application.

Furthermore, objective function U_f of the model is quadratic, and there are linear constraints and integer variable constraints, so the model is a mixed integer quadratic programming problem (MIQP), which belongs to nonlinear programming problems. Since objective function U_f of the model is nonlinear and the feasible region is not a polygonal region, it is difficult to obtain an optimal solution. In addition, traditional algorithms also hardly guarantee obtaining the optimal solution over the entire feasible region, that is, the global optimal solution. They can only obtain an optimal solution in a certain part of the feasible region, that is, the local optimal solution. It is also difficult to obtain a global optimal solution by using traditional tools such as LINGO to solve the above nonlinear programming problem.

Therefore, in order to better obtain a global optimum, inspired by the greedy algorithm idea, we propose a heuristic algorithm by using a combinatorial auction to solve the time-varying multidimensional resource allocation problem in the following section.

3. Resource allocation algorithm for the time-varying multidimensional resource allocation problem

3.1. Algorithm description. To solve the problem of time-varying multidimensional resource allocation, we will introduce a combinatorial auction algorithm (CAA-TMRAP) which includes three stages. The first stage is collecting bids. The third-party platform collects bids submitted by end users and resource capacity offered by the fog resource pool. The second stage is determining the resource allocation strategy and winners. The dominant-resource-based strategy is used as the basis for the resource allocation algorithm. We define the dominant resource proportion of the user as follows:

$$v_m = \max_r \frac{\max_t x_{mr}^t}{c_r}, \quad \forall m \in M, \quad \forall r \in R, \quad \forall t \in [1, T], \tag{11}$$

where x_{mr}^t is the amount of resource r required by user m at time t, $\max_t x_{mr}^t$ represents the maximum amount of resource r required by user m during different time slots, $(\max_t x_{mr}^t)/c_r$ is the ratio of the maximum amount of resource r to resource capacity c_r, and $\max_r (\max_t x_{mr}^t)/c_r$ is the maximum of the ratio.

Then, we define bid density d_m according to v_m as

$$d_m = \frac{h_m}{v_m}, \quad \forall m \in M. \tag{12}$$

The third stage of the algorithm is resource pricing, which uses a pricing method according to which bid is the transaction price.

Above all, in this algorithm, the resources are allocated in descending order according to bid density d_m, and the transaction price is set according to bid h_m to maximize the utility of the fog resource pool.

The general process of the algorithm is summarized as follows. The algorithm is invoked at $t = 1$, assuming that the arrive time slots and execution times of users are the same. Time-varying requirement matrix $\mathbf{x}_m$ records the resource requirement of user m in each period $[1, T]$. The users are arranged in descending order according to bid density d_m. If the resource requirement is satisfied, the transaction price is set. The remaining resource capacity matrix $\mathbf{C}$ will be updated after it is successfully allocated to a user.

3.2. Algorithm analysis. The cores of the algorithm are the second stage and the third stage, i.e., resource allocation and pricing. The utility of the fog resource pool is improved by determining the allocation order and transaction prices of users that the fog resource pool needs to supply.

In the second stage of the algorithm, we use a dominant-resource-based strategy as the basis for user ranking of resource allocation. This strategy is derived from dominant resource fairness (DRF), which calculates the ratio of the maximum demand of each resource required by users to the total amount of resources in the system. The idea of this strategy is to satisfy the requirements of users with a smaller share of dominant resources as much as possible and to ensure the fairness of resource allocation in the case of multiple resources coexisting. At the same time, the strategy can make the consumption rate of each resource as consistent as possible, and increase social welfare and conserve space to support the requirement allocation of other users. Under the dominant-resource-based strategy, the algorithm will be more inclined to the user with a higher bid and a smaller dominant resource proportion.

In the third stage of the algorithm, we use a pricing method according to which the bid is the transaction price. The pricing method can ensure that the maximum utility of the fog resource pool is achieved under the condition that transaction price p_m is not higher than highest price h_m that the user is willing to pay.

In addition, to illustrate the effectiveness of the CAA-TMRAP algorithm, we compare it with the CA-GREEDY algorithm (Zaman and Grosu, 2012), but CA-GREEDY does not support time-varying resource requirements of users in resource allocation. Therefore, in order to compare CA-GREEDY with our algorithm, we modify the bid density to order users in the CA-GREEDY algorithm.

3.3. Algorithm basic steps. The implementation steps of the algorithm are described as follows:

At time $t = 1, 2, \ldots, T$:

Step 1: Initialize variables and parameters. Input requirement: $\mathbf{B}_m = (\mathbf{x}_m, h_m)$, time-varying requirement matrix $\mathbf{x}_m$, bid h_m and the initial resource capacity matrix $\mathbf{C}$. Initialize the set a

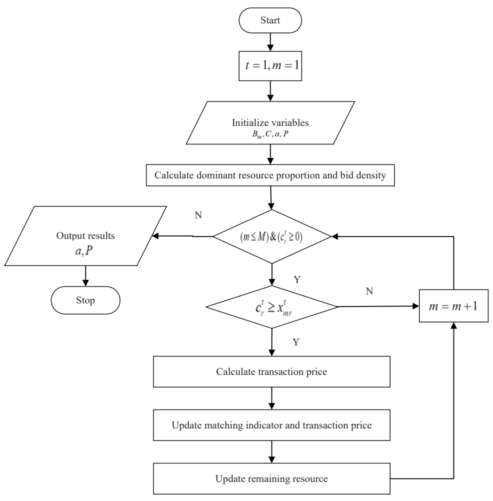
Start
$t = 1, m = 1$
Initialize variables B_m, C, a, P
Calculate dominant resource proportion and bid density
$(m \le M)\&(c_r^t \ge 0)$
N
Output results a, P
Stop
Y
$c_r^t \ge x_{mr}^t$
N
$m = m + 1$
Y
Calculate transaction price
Update matching indicator and transaction price
Update remaining resource

Fig. 3. Flowchart of the CAA-TMRAP algorithm.

of matching indicators of users and the set P of transaction prices.

Step 2: Calculate the dominant resource proportions and bid densities of users. Compute the dominant resource proportions v_m obtained by the third-party platform according to (11), and compute the bid densities d_m of users according to (12).

Step 3: Determine whether the remaining resources can satisfy the user requirement in period t. If fulfilled, the user will be allocated; otherwise, consider the next user.

Step 4: Calculate the prices paid by users to the fog resource pool. The third-party platform performs resource pricing that the bid is transaction price p_m.

Step 5: Update the set a of matching indicators of users and the set P of transaction prices. When the user is successfully allocated, set a of matching indicators of users and set P of transaction prices are updated.

Step 6: Update the remaining amount of resources of the fog resource pool. After a successful transaction, the remaining amount of resources $\mathbf{C}$ is updated.

Step 7: Check the stop criterion. The algorithm stops when the users are entirely traversed or the resources are completely allocated. Set a of matching indicators of users and set P of transaction prices are obtained.

The flowchart of the CAA-TMRAP algorithm is shown in Fig. 3, and its pseudocode is detailed in Algorithm 1.

Algorithm 1. CAA-TMRAP.

```
Require: $\mathbf{B}_m = (\mathbf{x}_m, h_m), \mathbf{C}$;
1: {The first stage: collecting information}
2: for all $m \in M$ do
3:     collect the requirements $\mathbf{B}_m = (\mathbf{x}_m, h_m)$ from users
4: end for
5: for $f$ do
6:     collect the resource capacity matrix $\mathbf{C}$ from fog resource pool $f$
7: end for
8: {The second stage: determining resource allocation strategy and winners}
9: $a \leftarrow \phi, P \leftarrow \phi$;
10: for all $m \in M, r \in R, t \in [1, T]$ do
11:     $v_m = \max_r \frac{\max_t x_{mr}^t}{c_r}$; {the dominant resource proportion}
12:     $d_m \leftarrow \frac{h_m}{v_m}$; {the bid density}
13: end for
14: for all $m \in M$ reordering, so that $d_1 \geq d_2 \geq \ldots \geq$
15: end for
16: for all $m \in M, r \in R, t \in [1, T]$ do
17:     if $x_{mr}^t \leq c_r^t$ then
18:         $a_m \leftarrow 1$;
19:         $a \leftarrow a \cup \{a_m\}$;
20:         $c_r^t \leftarrow c_r^t - x_{mr}^t$;
21:     end if
22: end for
23: {The third stage: resource pricing}
24: for all $m \in M$ do
25:     if $a_m = 1$ then
26:         $p_m \leftarrow h_m$;
27:     else
28:         $p_m \leftarrow 0$;
29:     end if
30:     $P \leftarrow P \cup \{p_m\}$;
31: end for
32: return $(a, P)$;
```

3.4. Algorithm properties. For mechanism design, truthful and individual rationality are two important features to be satisfied. In this section, we first introduce the preliminaries of mechanism design and then propose an offline optimal auction mechanism. The following definitions need to be met in mechanism design.

Definition 1. (*Individual rationality*) For a mechanism to ensure individual rationality, it should satisfy the condition that, when the user submits a requirement and bid, his utility value will be nonnegative, that is, $U_m \geq 0$. In other words, as long as the user participates in the auction and reports his requirement and bid truthfully, he will never incur losses.

Definition 2. (*Critical value*) Critical value theory means that all sellers are reluctant to provide services at a price lower than the user's bids, and all buyers are unwilling to purchase resources at a cost higher than the price. Thus, the price is the critical value.

Definition 3. (*Truthfulness*) If the allocation function of a mechanism satisfies monotonicity and the payment function satisfies critical value theory, then the mechanism is truthful.

Theorem 1. *The CAA-TMRAP algorithm satisfies individual rationality.*

Proof. From (1) and (2) it follows that the utility of user m is $U_m = H_m - a_m p_m - V_m^l$, p_m is the optimal solution, $H_m \geq a_m p_m + V_m^l$, thus $U_m \geq 0$. According to the CAA-TMRAP algorithm, for the winning users, the bids of the users are greater than or equal to the payment price, and the utility is nonnegative. By contrast, for the losing users, the utility is always 0, so the mechanism satisfies individual rationality. ■

Theorem 2. *The CAA-TMRAP algorithm is based on critical value theory.*

Proof. In the third stage of the algorithm, when fog resource pool f allocates resources to user m, it is unwilling to provide services at a price lower than bid h_m, reducing the resource revenue and its utility. User m is reluctant to purchase resources at a cost higher than bid h_m. Therefore, the transaction price satisfies $p_m = h_m$. In summary, for fog resource pool f and user m, the CAA-TMRAP algorithm satisfies critical value theory. ■

Theorem 3. *The CAA-TMRAP algorithm is correct.*

Proof. First, it is shown that the CAA-TMRAP algorithm is monotonic. User m can increase the probability of successful applications by increasing bid density. According to (12), under the premise that v_m remains unchanged, if $h_1 \leq h_2$, $d_1 \leq d_2$. Also, under the premise that h_m remains unchanged, if $v_1 \leq v_2$, $d_1 \geq d_2$. Therefore, if users want to rank higher when the CAA-TMRAP algorithm makes allocation, they can increase their bids or reduce the requirements for dominant resources. Thus, the CAA-TMRAP allocation algorithm is monotonic. Secondly, according to the pricing method of the CAA-TMRAP algorithm, we can know that the bid is the critical value, that is, $p_m = h_m$. Therefore, according to Definition 3, the CAA-TMRAP algorithm is truthful. ■

Theorem 4. *The time complexity of the CAA-TMRAP algorithm is polynomial.*

Proof. The CAA-TMRAP algorithm includes three stages: collecting bids, determining resource allocation strategy and winners, and resource pricing. The time complexity of the CAA-TMRAP algorithm is $o(2MRT)$. It will be invoked one time at $t = 1$ during the entire period $[1, T]$. As a result, the time complexity of the CAA-TMRAP algorithm is polynomial. ■

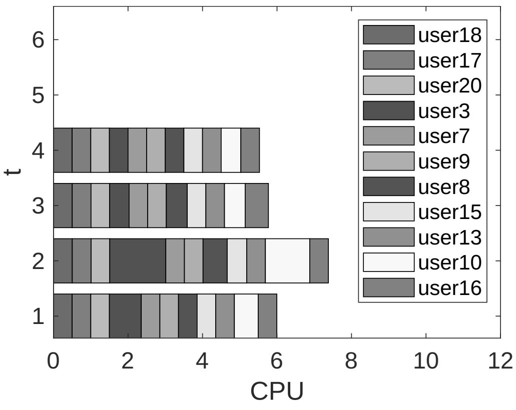

Fig. 4. CPU resource allocation.

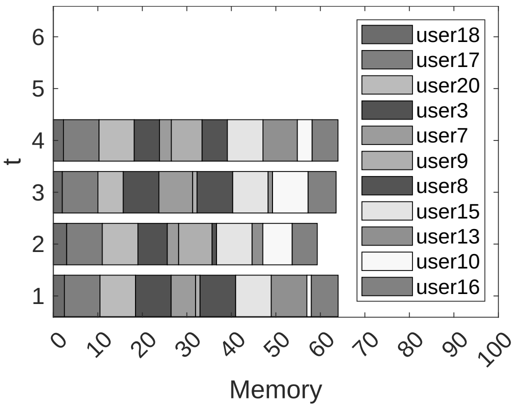

Fig. 5. Memory resource allocation.

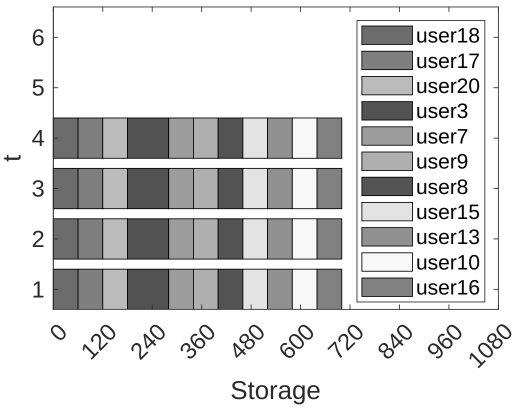

Fig. 6. Storage resource allocation.

4. Simulation and numerical examples

4.1. Experimental setup. We use the MATLAB simulation platform to verify the feasibility of the CAA-TMRAP algorithm, and give a numerical example to illustrate time-varying multidimensional resource allocation, which comes from the data set provided by the Alibaba Global Scheduling Algorithm Contest (Alibaba, 2019). Each requirement includes time-varying CPU, memory and storage demands for different time slots. The system consists of three types of resources whose capacities are represented by a vector of

$$\mathbf{c} = \begin{bmatrix} 32 \\ 64 \\ 1440 \end{bmatrix}$$

and 20 users whose requirements are $\mathbf{B}_m = (\mathbf{x}_m, h_m)$. We assume that users' arrival time slots and execution times are the same (all arrive at $t = 1$) and the execution time is $T = 4$, which is divided into four periods, namely $t = 1, 2, 3, 4$. Calculate the dominant resource proportions v_m and bid densities d_m to obtain the user requirements as shown in Table 2.

According to bid density d_m, the users are sorted in descending order for resource allocation. Thus, the solution to the TMRAP is shown in Figs. 4, 5, and 6, respectively.

Figures 4, 5, and 6 show one of the best solutions, which is an offline auction-based allocation that needs to know all users' requirements in advance. The users arrive at $t = 1$ and all are candidates. We sort them in descending order according to bid density, and finally we get the allocation result. From the vertical direction, we can see the allocation result of users at different times; from the horizontal direction, we can see the resource allocation result of different users at the same time. Specifically, in descending order of bid density d_m, the order of allocation should consists of users 18, 17, 20, 3, 7, 9, 8, 15, 13, 10, 2, 5, 11, 14, 6, 16, 19, 12, 4 and 1, but the final order of allocation consists of users 18, 17, 20, 3, 7, 9, 8, 15, 13, 10 and 16. Because the remaining resources do not satisfy the requirements of users 2, 5, 11, 14, 6, 19, 12, 4 and 1, and the objective of this paper is to maximize the utility of the fog resource pool, users 2, 5, 11, 14, 6, 19, 12, 4 and 1 fail to obtain the resources and user 16 succeeds. Then, the transaction price of each user in the CAA-TMRAP algorithm is shown in Table 3.

Furthermore, it is necessary to calculate the agency fee to obtain the utility of the fog resource pool. According to the Auction Law of the People's Republic of China, different proportions of commissions are charged for auctioning arbitrary items and public properties. If the client and the auctioneer have previously agreed, they will be charged according to the agreement. Otherwise, the auctioneer may charge the client a commission that does not exceed 5% of the transaction price. According to the law, we assume that the third-party platform charges an agency fee of 5% of the transaction price in each transaction. Therefore, the agency fee V_f^l can be specifically shown as follows:

$$V_f^l = \begin{cases} a_m, & p_m \le 20, \\ 0.05 p_m a_m, & 20 < p_m < 30, \\ 1.5 a_m, & p_m \ge 30. \end{cases} \tag{13}$$

Then, by calculating the agency fee for each transaction, Table 4 gives the utility of the fog resource pool in the CAA-TMRAP algorithm.

Furthermore, to prove the effectiveness of the CAA-TMRAP algorithm, we compare it with the CA-GREEDY algorithm. In addition to the pricing method based on the bid of the CAA-TMRAP algorithm, there is another pricing method of critical value in the CA-GREEDY algorithm. Many researchers use the highest bid density of failed users to calculate the transaction price (Zhang *et al*., 2020; Zaman and Grosu, 2012; Li *et al*., 2019; Mashayekhy *et al*., 2016), and it is regarded as the critical value in the CA-GREEDY algorithm.

First, the pricing method of critical value in the CA-GREEDY algorithm is to find the highest bid density of failed users m'. Then, the transaction price of each winner m is calculated as the product of its dominant resource proportion v_m and bid density $d_{m'}$, and all failed users do not need to pay. In the numerical example, this method determines that the failed user m' is user 2. Table 5 shows the transaction price of each user in the CA-GREEDY algorithm.

From Tables 3 and 5, we can observe that, under the premise of ranking in descending order according to bid density d_m, the resource revenue of the fog resource pool in the CAA-TMRAP algorithm is 249, but the resource revenue in the CA-GREEDY is 167.020. Therefore, we can state that the pricing method based on the bid of the CAA-TMRAP algorithm can better satisfy the maximum income of the fog resource pool. After calculating the agency fees, we can obtain the utility of the fog resource pool in the CA-GREEDY algorithm, as shown in Table 6.

From Tables 4 and 6, we can observe that, under the premise of ranking in descending order according to bid density d_m, the utility of the fog resource pool in the CAA-TMRAP algorithm is 235.600, and the utility of the resource pool in the CA-GREEDY algorithm is 156.020. Therefore, we can state that the pricing method based on the bid in the CAA-TMRAP algorithm can better satisfy the maximum utility of the fog resource pool, which proves the effectiveness of choosing the bid as the transaction price.

Moreover, the CA-GREEDY algorithm cannot satisfy the requirement that the transaction price p_m

Table 2. User requirements.

m	CPU	Memory	Storage	h_m	v_m	d_m
1	(11.988,6.278,2.800,0.500)	(16.000,4.314,16.000, 12.763)	(200,200,200, 200)	6	0.375	16.000
2	(5.842,6.000,0.500,3.399)	(8.000,1.000,8.000,5.675)	(40,40,40,40)	25	0.188	132.979
3	(0.855,1.505,0.522,0.500)	(8.000,6.600,8.000, 5.667)	(100, 100, 100, 100)	28	0.125	224.000
4	(0.653,0.500,0.700,0.500)	(32.000,13.400,1.000,32.000)	(80,80,80,80)	9	0.500	18.000
5	(0.500,0.552,0.629,0.500)	(8.000,8.000,8.000,8.000)	(60,60,60,60)	14	0.125	112.000
6	(0.500,0.500,0.500,0.500)	(8.000,7.837, 7.935,8.000)	(60,60,60,60)	8	0.125	64.000
7	(0.500,0.500,0.500,0.500)	(5.495,2.560,7.590,2.668)	(60,60,60,60)	26	0.119	218.487
8	(0.500,0.649,0.562,0.500)	(8.000,1.000,8.000,5.667)	(60,60,60,60)	23	0.125	184.000
9	(0.500,0.500,0.500,0.500)	(1.000,7.510,1.000,6.921)	(60,60,60,60)	24	0.117	205.128
10	(0.640,1.191,0.555,0.529)	(1.000,6.600,8.000,3.333)	(60,60,60,60)	17	0.125	136.000
11	(1.283,0.500,1.247,0.500)	(8.000,7.718,7.888,8.000)	(60,60,60,60)	13	0.125	104.000
12	(0.886,0.509,1.766,0.500)	(16.000,16.000,16.000,16.000)	(60,60,60,60)	5	0.250	20.000
13	(0.500,0.500,0.500,0.500)	(8.000,2.400,1.000,7.723)	(60,60,60,60)	20	0.125	160.000
14	(2.333,0.500,4.878,0.500)	(8.000,1.000,5.200,8.000)	(60,60,60,60)	12	0.152	78.947
15	(0.500,0.525,0.500,0.500)	(8.000,8.000,7.952,8.000)	(60,60,60,60)	21	0.125	168.000
16	(0.500,0.500,0.624,0.500)	(6.023,5.631,6.280,5.800)	(60,60,60,60)	4	0.098	40.816
17	(0.500,0.506,0.504,0.500)	(8.000,8.000,8.000,8.000)	(60,60,60,60)	30	0.125	240.000
18	(0.500,0.503,0.502,0.500)	(2.490,3.008,2.020,2.275)	(60,60,60,60)	27	0.047	574.468
19	(0.500,0.519,0.515,0.500)	(8.000,8.000,6.826,7.059)	(60,60,60,60)	3	0.125	24.000
20	(0.500,0.500,0.500,0.500)	(7.990,8.000,5.702,7.910)	(60,60,60,60)	29	0.125	232.000

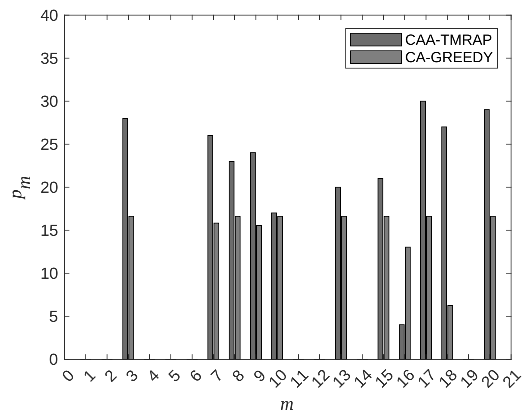

Fig. 7. Transaction price under different users.

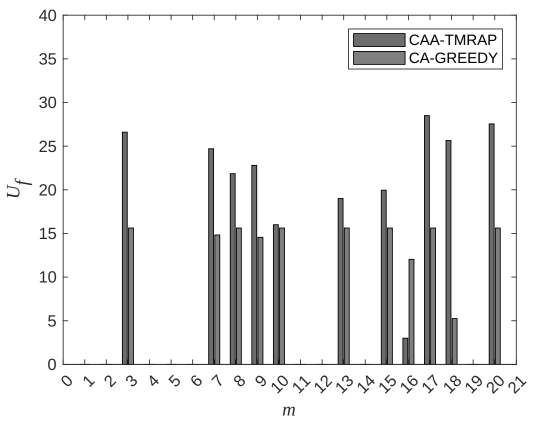

Fig. 8. Utility of fog resource pool under different users.

should not be higher than the highest price h_m that the user is willing to pay. For instance, the highest bid density of failed users is that of user 2. But it will result in a problem in which the transaction price of user 16 will be higher than the bid. During the process of allocation, there will be a situation where the bid density is higher than others, but the resource requirement is not satisfied. For example, users 2, 5, 11, 14 and 6 fall into the situation mentioned above. Therefore, if the transaction price is calculated according to the critical value, it cannot be guaranteed that the value of p_m of each successful user is not higher than h_m. Thus, the method of pricing with critical value in CA-GREEDY is not advisable.

4.2. Analysis of results. From Tables 3 and 5, we can compare the transaction price under different users of the CAA-TMRAP and CA-GREEDY algorithms in Fig. 7.

We can find that, under the premise of ranking in descending order according to bid density d_m, the resource revenue of the fog resource pool in the CAA-TMRAP algorithm is larger than in the CA-GREEDY algorithm. Furthermore, from Tables 4 and 6, we also compare the utility of the fog resource pool of CAA-TMRAP and CA-GREEDY algorithms in Fig. 8.

We can find that, under the premise of ranking in descending order according to bid density d_m, the method according to which the bid is the transaction price in the CAA-TMRAP algorithm has larger excellent utility to the

Table 3. Transaction price of each user in the CAA-TMRAP algorithm.

m	h_m	v_m	d_m	a_m	p_m
1	6	0.375	16.000	0	0
2	25	0.188	132.979	0	0
3	28	0.125	224.000	1	28
4	9	0.500	18.000	0	0
5	14	0.125	112.000	0	0
6	8	0.125	64.000	0	0
7	26	0.119	218.487	1	26
8	23	0.125	184.000	1	23
9	24	0.117	205.128	1	24
10	17	0.125	136.000	1	17
11	13	0.125	104.000	0	0
12	5	0.250	20.000	0	0
13	20	0.125	160.000	1	20
14	12	0.152	78.947	0	0
15	21	0.125	168.000	1	21
16	4	0.098	40.816	1	4
17	30	0.125	240.000	1	30
18	27	0.047	574.468	1	27
19	3	0.125	24.000	0	0
20	29	0.125	232.000	1	29
$\sum_{m\in M} a_m p_m$	–	–	–	–	249

Table 4. Utility of the fog resource pool in the CAA-TMRAP algorithm.

m	p_m	V_f^l	$a_m p_m - V_f^l$
1	0	0	0
2	0	0	0
3	28	1.400	26.600
4	0	0	0
5	0	0	0
6	0	0	0
7	26	1.300	24.700
8	23	1.150	21.850
9	24	1.200	22.800
10	17	1	16
11	0	0	0
12	0	0	0
13	20	1	19
14	0	0	0
15	21	1.050	19.950
16	4	1	3
17	30	1.500	28.500
18	27	1.350	25.650
19	0	0	0
20	29	1.450	27.550
$U_f = \sum_{m\in M} (a_m p_m - V_f^l)$	–	–	235.600

fog resource pool. Therefore, we can further prove the effectiveness of choosing the bid as the transaction price. In summary, the CAA-TMRAP algorithm of resource allocation and pricing is effective.

5. Conclusion

In this paper, we investigated the time-varying multidimensional resource allocation problem in fog computing. To achieve the goal of resource allocation in maximizing the satisfaction of the fog resource pool, we established an integer programming model with the objective of maximizing network utility of the fog resource pool. Furthermore, we designed a heuristic algorithm to find an approximate solution. Finally, we gave some numerical examples to demonstrate the performance of the algorithm. Theoretical analysis and experimental results show that the algorithm can improve resource utilization and maximize the utility of the fog resource pool while ensuring truthful and individual rationality. In this paper, we just considered an offline auction-based time-varying multidimensional resource allocation problem in fog computing. In the future, we will consider an on-line auction-based time-varying multidimensional resource allocation problem in fog computing and formulate an actual on-line auction mechanism for resource allocation in a competitive environment.

Acknowledgment

The authors would like to thank the anonymous reviewers for very detailed and helpful comments and suggestions to improve this article. This work was supported by the National Natural Science Foundation of China (no. 71971188), the Humanity and Social Science Foundation of the Ministry of Education of China (no. 22YJCZH086), the Natural Science Foundation of Hebei Province (nos. G2022203003, G2020203005, G2020203012), the S&T Program of Hebei (no. 22550301D), and a project funded by the Hebei Education Department (no. ZD2022142).

References

Aggarwal, A., Kumar, N., Vidyarthi, D. and Buyya, R. (2021). Fog-integrated cloud architecture enabled multi-attribute combinatorial reverse auctioning framework, *Simulation Modelling Practice and Theory* **109**: 102307.

Alibaba (2019). Alibaba cloud, `https://tianchi.aliyun.com/`.

Angelelli, E. and Filippi, C. (2011). On the complexity of interval scheduling with a resource constraint, *Theoretical Computer Science* **412**(29): 3650–3657.

Bandyopadhyay, A., Roy, T., Sarkar, V. and Mallik, S. (2020). Combinatorial auction-based fog service allocation mechanism for IoT applications, *2020 10th International Conference on Cloud Computing, Data Science and Engineering (Confluence), Noida, India*, pp. 518–524.

Table 5. Transaction price of each user in the CA-GREEDY algorithm.

m	h_m	v_m	d_m	a_m	p_m
1	6	0.375	16.000	0	0
2	25	0.188	132.979	0	0
3	28	0.125	224.000	1	16.622
4	9	0.500	18.000	0	0
5	14	0.125	112.000	0	0
6	8	0.125	64.000	0	0
7	26	0.119	218.487	1	15.825
8	23	0.125	184.000	1	16.622
9	24	0.117	205.128	1	15.559
10	17	0.125	136.000	1	16.622
11	13	0.125	104.000	0	0
12	5	0.25	20.000	0	0
13	20	0.125	160.000	1	16.622
14	12	0.152	78.947	0	0
15	21	0.125	168.000	1	16.622
16	4	0.098	40.816	1	13.032
17	30	0.125	240.000	1	16.622
18	27	0.047	574.468	1	6.250
19	3	0.125	24.000	0	0
20	29	0.125	232.000	1	16.622
$\sum_{m \in M} a_m p_m$	–	–	–	–	167.020

Table 6. Utility of the fog resource pool in the CA-GREEDY algorithm.

m	p_m	V_m^l	$p_m - V_m^l$
1	0	0	0
2	0	0	0
3	16.622	1	15.622
4	0	0	0
5	0	0	0
6	0	0	0
7	15.825	1	14.825
8	16.622	1	15.622
9	15.559	1	14.559
10	16.622	1	15.622
11	0	0	0
12	0	0	0
13	16.622	1	15.622
14	0	0	0
15	16.622	1	15.622
16	13.032	1	12.032
17	16.622	1	15.622
18	6.250	1	5.250
19	0	0	0
20	16.622	1	15.622
$U_f = \sum_{m \in M} (a_m p_m - V_f^l)$	–	–	156.020

Baranwal, G. and Kumar, D. (2020). DAFNA: Decentralized auction based fog node allocation in 5G era, *2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS), Rupnagar, Punjab, India*, pp. 575–580.

Bermbach, D., Maghsudi, S., Hasenburg, J. and Pfandzelter, T. (2020). Towards auction-based function placement in serverless fog platforms, *2020 IEEE International Conference on Fog Computing (ICFC), Sydney, Australia*, pp. 25–31.

Besharati, R., Rezvani, M. and Sadeghi, M. (2021). An incentive-compatible offloading mechanism in fog-cloud environments using second-price sealed-bid auction, *Journal of Grid Computing* **19**(3): 37.

Chang, B.-J., Hwang, R.-H., Tsai, Y.-L., Yu, B.-H. and Liang, Y.-H. (2019). Cooperative adaptive driving for platooning autonomous self driving based on edge computing, *International Journal of Applied Mathematics and Computer Science* **29**(2): 213–225, DOI: 10.2478/amcs-2019-0016.

Ghobaei-Arani, M., Souri, A. and Rahmanian, A. (2020). Resource management approaches in fog computing: A comprehensive review, *Journal of Grid Computing* **18**(1): 1–42.

Guo, Y., Saito, T., Oma, R., Nakamura, S., Enokido, T. and Takizawa, M. (2020). Distributed approach to fog computing with auction method, *Advanced Information Networking and Applications* **1151**: 268–275.

Han, C., Zhang, P., Wang, W., Wang, W., Wang, Y. and Zhang, Z. (2019). Delay-optimal joint processing in computation-constrained fog radio access networks, *IEEE Access* **7**: 58857–58865.

Houshyar, M., Seyyed, J., Hamidreza, N. and Afshin, R. (2021). A new resource allocation method in fog computing via non-cooperative game theory, *Journal of Intelligent & Fuzzy Systems* **41**(2): 3921–3932.

Junior, F., Dias, K., d'Orey, P. and Kokkinogenis, Z. (2021). Fogwise: On the limits of the coexistence of heterogeneous applications on fog computing and Internet of vehicles, *Transactions on Emerging Telecommunications Technologies* **32**(1): e4145.

Kayal, P. and Liebeherr, J. (2019). Distributed service placement in fog computing: An iterative combinatorial auction approach, *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Richardson, USA,* pp. 2145–2156.

Leao, A., Cherri, L. and Arenales, M. (2014). Determining the k-best solutions of knapsack problems, *Computers & Operations Research* **49**: 71–82.

Lee, Y., Jeong, S., Masood, A., Park, L., Dao, N. and Cho, S. (2020). Trustful resource management for service allocation in fog-enabled intelligent transportation systems, *IEEE Access* **8**: 147313–147322.

Li, S., Liu, H., Li, W. and Sun, W. (2023). An optimization framework for migrating and deploying multiclass enterprise applications into the cloud, *IEEE Transactions on Services Computing* **16**(2): 941–956.

Li, S. and Sun, W. (2021). Utility maximisation for resource allocation of migrating enterprise applications into the cloud, *Enterprise Information Systems* **15**(2): 197–229.

Li, S., Zhang, Y., Wang, Y. and Sun, W. (2019). Utility optimization-based bandwidth allocation for elastic and inelastic services in peer-to-peer networks, *International Journal of Applied Mathematics and Computer Science* **29**(1): 111–123, DOI: 10.2478/amcs-2019-0009.

Mashayekhy, L., Nejad, M., Grosu, D. and Vasilakos, A. (2016). An online mechanism for resource allocation and pricing in clouds, *IEEE Transactions on Computers* **65**(4): 1172–1184.

Peng, X., Ota, K. and Dong, M. (2020). Multiattribute-based double auction toward resource allocation in vehicular fog computing, *IEEE Internet of Things Journal* **7**(4): 3094–3103.

Sharghivand, N., Derakhshan, F. and Siasi, N. (2021). A comprehensive survey on auction mechanism design for cloud/edge resource management and pricing, *IEEE Access* **9**: 126502–126529.

Song, F., Ai, Z., Zhang, H., You, I. and Li, S. (2021a). Smart collaborative balancing for dependable network components in cyber-physical systems, *IEEE Transactions on Industrial Informatics* **17**(10): 6916–6924.

Song, F., Li, L., You, I. and Zhang, H. (2021b). Enabling heterogeneous deterministic networks with smart collaborative theory, *IEEE Network* **35**(3): 64–71.

Sun, H., Yu, H. and Fan, G. (2020). Contract-based resource sharing for time effective task scheduling in fog-cloud environment, *IEEE Transactions on Network and Service Management* **17**(2): 1040–1053.

Tasiopoulos, A., Onur, A., Ioannis, P. and George, P. (2018). Edge-map: Auction markets for edge resource provisioning, *2018 IEEE 19th International Symposium “A World of Wireless, Mobile and Multimedia Networks (WoWMoM)”, Chania, Greece*, pp. 14–22.

Zaman, S. and Grosu, D. (2012). Combinatorial auction-based allocation of virtual machine instances in clouds, *Journal of Parallel and Distributed Computing* **73**(4): 495–508.

Zhang, J., Li, J., Li, W. and Zhang, X. (2019). A fair distribution strategy based on shared fair and time-varying resource demand, *Journal of Computer Research and Development* **56**(7): 1534–1544.

Zhang, J., Yang, X., Xie, N., Zhang, X., Vasilakos, A. and Li, W. (2020). An online auction mechanism for time-varying multidimensional resource allocation in clouds, *Future Generation Computer Systems* **111**: 27–38.

Zhu, L., Sun, L. and Yan, Y. (2020). Parking assistance scheme based on reverse auction in vehicle fog computing, *Computer Engineering* **46**(7): 14–20.

Shiyong Li received his PhD degree from Beijing Jiaotong University, China, in 2011. Currently he is a full professor in the School of Economics and Management at Yanshan University. He is the (co)author of more than 60 papers in mathematics, engineering and management journals. His research interests include cloud migration for enterprise applications, resource allocation of cloud/edge computing, information systems and electronic commerce.

Yanan Zhang received her BSc degree from Hebei Finance University, Baoding, in 2019 and is currently working toward her MSc degree at the School of Economics and Management, Yanshan University, Qinhuangdao, China. Her research interests include resource allocation in edge computing and fog computing.

Wei Sun received her PhD degree from Yanshan University, Qinhuangdao, China, in 2010. Currently she is a full professor in the School of Economics and Management at Yanshan University. She has published more than 50 papers in leading international journals in the areas of operations research and applied mathematics. Her research interests include economics of queues, and queueing systems with vacations.

Jia Liu received her PhD degree from the Beijing Institute of Technology, China, in 2011. Currently she is an associate professor in the School of Economics and Management at the Communication University of China. She is the (co)author of more than 40 papers in engineering and management journals. Her research interests include technology innovation management, data mining, and technology evaluation.

Received: 23 December 2021
Revised: 9 April 2022
Re-revised: 14 August 2022
Accepted: 19 August 2022