

## A Reverse Auction Mechanism for Time-Varying Multidimensional Resource Allocation in Vehicular Fog Computing with Cloud and Edge Collaboration

**Shiyong LI**

*School of Economics and Management, Yanshan University, Qinhuangdao 066004, China*  
*E-mail: shiyongli@ysu.edu.cn*

**Yanan ZHANG**

*School of Economics and Management, Yanshan University, Qinhuangdao 066004, China*  
*E-mail: 15831232797@163.com*

**Wei SUN**

*School of Economics and Management, Yanshan University, Qinhuangdao 066004, China*  
*E-mail: wsun@ysu.edu.cn*

**Abstract** It is a hot issue to allocate resources using auction mechanisms in vehicular fog computing (VFC) with cloud and edge collaboration. However, most current research faces the limitation of only considering single type resource allocation, which cannot satisfy the resource requirements of users. In addition, the resource requirements of users are satisfied with a fixed amount of resources during the usage time, which may result in high cost of users and even cause a waste of resources. In fact, the actual resource requirements of users may change with time. Besides, existing allocation algorithms in the VFC of cloud and edge collaboration cannot be directly applied to time-varying multidimensional resource allocation. Therefore, in order to minimize the cost of users, we propose a reverse auction mechanism for the time-varying multidimensional resource allocation problem (TMRAP) in VFC with cloud and edge collaboration based on VFC parking assistance and transform the resource allocation problem into an integer programming (IP) model. And we also design a heuristic resource allocation algorithm to approximate the solution of the model. We apply a dominant-resource-based strategy for resource allocation to improve resource utilization and obtain the lowest cost of users for resource pricing. Furthermore, we prove that the algorithm satisfies individual rationality and truthfulness, and can minimize the cost of users and improve resource utilization through comparison with other similar methods. Above all, we combine VFC smart parking assistance with reverse auction mechanisms to encourage resource providers to offer resources, so that more vehicle users can obtain services at lower prices and relieve traffic pressure.

**Keywords** reverse auction; time-varying multidimensional resource allocation; resource pricing; cloud and edge collaboration; vehicular fog computing

---

Received June 18, 2022, accepted February 14, 2023

Supported by the National Natural Science Foundation of China (71971188), the Humanities and Social Science Fund of Ministry of Education of China (22YJCZH086), the Natural Science Foundation of Hebei Province (G2022203003), the S&T Program of Hebei (22550301D)

## 1 Introduction

With the rapid growth of smart connected devices in the internet of things (IoT)-based network systems, there is an explosion of data in smart cities, smart healthcare, smart grids, 5G networks and the internet of vehicles<sup>[1]</sup>. Edge computing, also known as fog computing, extends the resources from the cloud data center to the edge of the network, alleviating network congestion and reducing the response time of services<sup>[2, 3]</sup>. Cloud computing has the advantages of strong computing and storage capabilities, and edge computing is characterized by low latency and low power consumption. Therefore, many resource providers have begun to jointly use the advantages of cloud and edge computing to provide better quality of service (QoS) in cloud and edge collaboration<sup>[4]</sup>.

Vehicular fog computing (VFC), as an extension of edge computing, has become a research hotspot in the field of cloud and edge collaboration<sup>[4, 5]</sup>. It combines cloud and edge collaboration with traditional in-vehicle networks by applying a cloud resource provider and multiple edge devices which act as fog nodes. The vehicle users are the infrastructures for communication and computing. The edge devices, which close to vehicle users, provide real-time response services. And the cloud resource provider, which is far away from vehicle users, offers long-term storage services<sup>[6, 7]</sup>. For instance, the in-vehicle network deploys road side units (RSU) to provide real-time response services such as communication and computing services for vehicle users, and takes advantage of cloud resource provider to offer storage services for vehicle users<sup>[8]</sup>.

However, the severe parking problems were caused by limited parking places due to the increased population and limited spatial resources of the city<sup>[6]</sup>. Previous research has shown that traffic jams in many cities were caused by vehicles searching for parking slots<sup>[6, 9]</sup>. Additionally, unnecessary time and energy consumption of vehicles were wasted during their searching for parking. Therefore, smart parking systems (SPS) have been widely studied to guide the vehicles to the available parking slots with less time and energy consumption<sup>[10, 11]</sup>. For instance, the automatic parking system can provide the services of real-time parking navigation for drivers, thereby alleviating traffic congestion<sup>[12]</sup>.

In VFC, most current research faces the limitation of only considering single type resource allocation, which cannot satisfy the resource requirements of users. For instance, with the increase of service requirements of vehicle users, the data processing pressure of RSU and cloud resource provider continue to increase, and it becomes difficult to satisfy the variety requirements of vehicle users<sup>[13, 14]</sup>. In addition, the current resource allocation in VFC is implemented with the fixed resource requirements of vehicle users. But the actual resource requirements of users may change with time. It will result in high costs of users, low utility of resource providers and even cause a waste of resources if the resource requirements of users are fixed during the usage time<sup>[15]</sup>. Therefore, it is also an emerging issue for VFC to realize optimal time-varying resource allocation and pricing mechanisms. Time-varying resources imply that the resource requirements of users vary over time. Specifically, it can reduce the cost of end users by providing appropriate resources and also improve the utility of resource providers by increasing resource utilization. Thus, the approximate solution of time-varying resource requirements is a win-win solution for both users and resource providers. Besides, existing allocation algorithms in the VFC of cloud and edge collaboration cannot be directly applied to

time-varying multidimensional resource allocation.

Therefore, in this paper, we propose a reverse auction mechanism for the time-varying multidimensional resource allocation problem (TMRAP) in cloud and edge collaboration based on VFC parking assistance and transform the resource allocation problem into an integer programming model to minimize the cost of users. And we also design a heuristic algorithm to approximate the solution of the model. The reverse auction mechanisms can make the market more competitive and help buyers get services at lower prices. Therefore, we combine VFC smart parking assistance with reverse auction mechanisms to encourage resource providers to offer resources, so that more vehicle users can obtain services at lower prices and relieve traffic pressure.

## 2 Related Works

At present, resource allocation in cloud and edge collaboration has attracted much attention. The resource allocation approaches are categorized into two types: Auction-based<sup>[15]</sup> and optimization<sup>[2, 16, 17]</sup>. Many resource allocation problems can be transformed into a knapsack problem<sup>[18]</sup>. For this reason, resource providers have introduced auction mechanisms in cloud and edge collaboration to obtain more profit, allowing idle resources to be sold at dynamic prices<sup>[15]</sup>. And many researchers have initially explored optimal solutions for resource allocation problems, which are often proven to be NP-hard<sup>[18]</sup>. It is difficult to obtain an optimal solution in polynomial time. Therefore, heuristic algorithms are often used to apply for the design of resource allocation and pricing mechanisms.

In the VFC field, Lee, et al.<sup>[19]</sup> proposed the VCG auction mechanism for actual service allocation. The mechanism offers service stability while ensuring the credibility of the service. But they did not consider the cost of users. Subsequently, for VFC parking assistance, Zhang, et al.<sup>[20]</sup> designed the VFC-aware parking reservation auction to guide the on-the-move vehicles to the available parking places with less effort. The simulation results confirm the win-win performance enhancement to the fog node controller, the smart vehicles, and the parking places from the proposed design. And Zhu, et al.<sup>[21]</sup> proposed a VFC parking assistance allocation strategy RAFC based on the reverse auction. This approach can reduce users' costs and improve social welfare. But they only considered single resource. Then, Alamer, et al.<sup>[22]</sup> presented a proposal for a tendering-based incentive framework in order to encourage vehicle owners to join in announced tasks in the vehicular fog computing, which is designed based on heterogeneous vehicular resource types. The detailed performance analysis demonstrates that the communication and computational overheads of this privacy-preserving scheme are significantly more efficient than the available alternatives. But they only considered price attributes. Peng, et al.<sup>[23]</sup> proposed a double auction mechanism based on multiple features. They believed that some non-price characteristics (location, reputation, and computing power) were crucial for providing fair resource allocation in VFC.

However, the above studies did not take latency requirements into account. Then, Junior, et al.<sup>[24]</sup> established a request-processing-response-actuation programming model based on distributed auction protocol to match clients and servers and satisfy latency requirements. The simulation results show that VFC can satisfy application requirements under mobility. And

Zhou, et al.<sup>[25]</sup> provided a contract-matching approach to minimize the network delay and maximize the expected utility of the base station. And they transformed the task assignment problem into a two-sided matching problem between vehicles and users, which is solved by a pricing-based stable matching algorithm. The numerical results demonstrate that significant performance improvement can be achieved by the proposed scheme. But they did not take energy efficiency into account. Then, Shojafar, et al.<sup>[26]</sup> proposed and tested an energy-efficient adaptive resource scheduler for Networked Fog Centers to maximize the overall communication-plus-computing energy efficiency, while meeting the application-induced hard QoS requirements on the minimum transmission rates, maximum delays and delay-jitters. But they did not consider the task offloading and task scheduling. Then, Chen, et al.<sup>[27]</sup> proposed a hybrid dynamic scheduling scheme (HDSS) that can optimize the task scheduling dynamically based on the changeable system environments. The analysis results in a superior performance of HDSS in the unstable vehicular edge computing (VEC) environments. And Liao, et al.<sup>[28]</sup> exploited blockchain and smart contract to facilitate fair task offloading and mitigate various security attacks, designed a subjective logic-based trustfulness metric to quantify the possibility of task offloading success, and developed a trustfulness assessment mechanism. The extensive theoretical analysis and simulations are carried out to demonstrate the reliability, feasibility, and efficiency of the proposed secure and intelligent task offloading scheme.

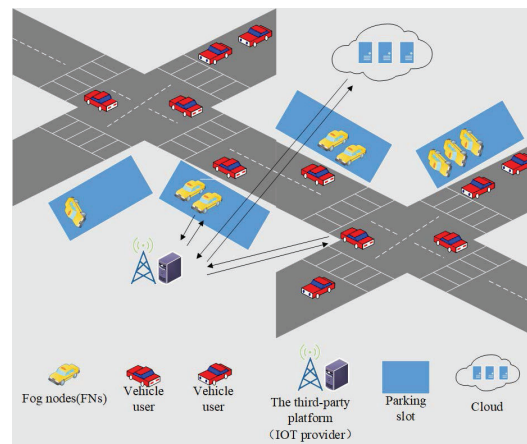
We can find from previous studies above that there are some interesting research results on resource allocation in VFC. However, most of them are dedicated to allocating fixed resources, and few studies have considered the allocation of time-varying multidimensional resources. On the contrary, there are more studies in other fields of cloud computing. Among them, Li, et al.<sup>[29]</sup> established an integer programming model based on the limited task resource requirements at different times. In addition, they proposed a heuristic algorithm, which has achieved good results in ensuring the fairness of resource allocation and high resource utilization. However, users with more profit may fail to match due to lower ranking. Therefore, Zhang, et al.<sup>[15]</sup> further proposed a waiting period strategy and a dominant-resource-based strategy to improve social welfare and resource utilization. At the same time, they proposed a pricing algorithm based on the critical value theory according to the greedy mechanism offered by Zaman, et al.<sup>[30]</sup>. 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.<sup>[31]</sup>. But they did not consider the priorities of arriving jobs. Then, Zhang, et al.<sup>[32]</sup> also proposed a creative integer programming model for time-varying resource allocation problems and designed a strategy-proof online auction mechanism that allows jobs to be scheduled in a preemptive-restart mode. Their approach outperforms existing algorithms in terms of execution time, social welfare, and resource utilization. But they only considered the time-varying resource allocation problem of cloud computing. Furthermore, Zhang, et al.<sup>[33]</sup> designed an auction mechanism for online resource allocation in cloud-edge collaboration based on live video webcast services and transformed the resource allocation problem into a constrained integer programming model. Compared with others, their approach is characterized by high social welfare, high resource utilization and short execution time.

In summary, the time-varying multidimensional resource allocation in other fields of cloud computing has achieved some interesting results in the past years, which is equally essential in the VFC of cloud and edge collaboration today. However, existing allocation algorithms in the VFC of cloud and edge collaboration cannot be directly applied to time-varying multidimensional resource allocation. Therefore, we consider the problem of online time-varying multidimensional resource allocation in cloud and edge collaboration, and transform it into an integer programming model to minimize the cost of users. And we also design a heuristic algorithm to approximate the solution of the model. Finally, we give some numerical examples to demonstrate the performance of the algorithm. And we also prove that the algorithm is individual rationality and truthful. Compared with existing research results, our approach can minimize the cost of users and improve resource utilization.

### 3 Time-Varying Multidimensional Resource Allocation Problem

#### 3.1 Model Description

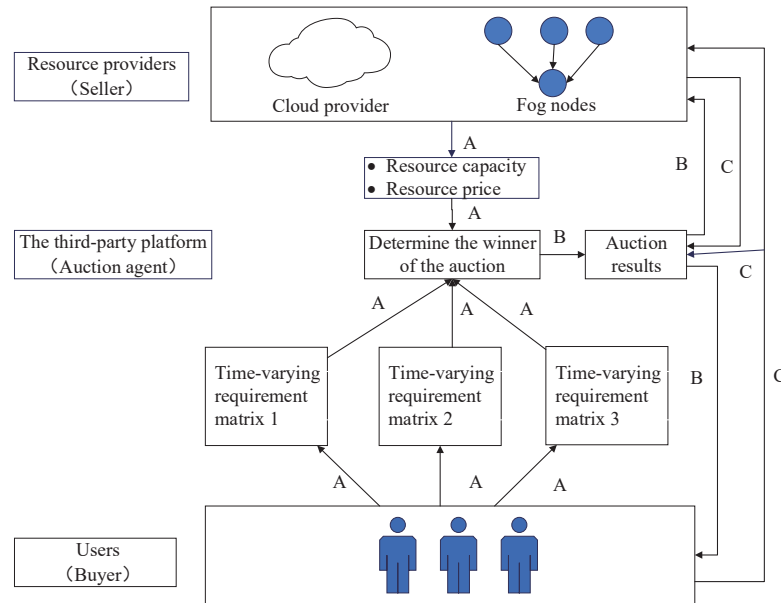
As shown in Figure 1, we mainly consider the resource allocation among vehicle users, resource providers and an IoT provider in the VFC parking assistance system. And the resource providers consist of a cloud resource provider and some smart vehicles. In the system, the vehicle users act as end users to request the resources of VFC parking assistance. The cloud resource provider supplies resources for vehicle users with low delay sensitivity. And the smart vehicles act as fog nodes to provide resources for vehicle users with high delay sensitivity. In addition, the IoT provider acts as the third-party platform to receive information and coordinate allocation. Besides, we assume that each user is independent in the system. And the requirement of each user should be migrated to a resource provider at most, but each resource provider can receive and process multiple requirements.



**Figure 1** The framework of VFC parking assistance system

In this paper, we investigate the TMRAP in the VFC parking assistance system, mainly focusing on online resource allocation in cloud and edge collaboration. When end users have different arrival time slots and the same execution times, the resource providers, which consist of a cloud provider and some fog nodes, provide services to end users through different types of

resources. The model minimizes the cost of users 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 utilization of resources. The relationship between the end users, the third-party platform, and the resource providers is shown in Figure 2.



**Figure 2** The relationship between three parties in the reverse auction

As shown in Figure 2, in the reverse auction model, the end users act as buyers to submit their resource requirements. And the resource providers, which include a cloud resource provider and some fog nodes, act as sellers 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 resource providers submit resource capacities and unit prices to the third-party platform. The other part means that each user offers its time-varying requirement matrix 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 resource providers and end users. And it finally informs both buyers and sellers of auction results. The process C indicates that when the auction is successful, the resource providers will pay the agency fees to the third-party platform, and the end users will also pay the agency fees to the third-party platform. At the same time, each user needs to pay the transaction price to the corresponding resource provider.

The objectives of the TMRAP based on the reverse auction include:

- 1) To satisfy the requirements of time-varying multidimensional resources of users.
- 2) To minimize the cost of users.
- 3) To improve the utilization of resources.

The notations used in this paper are summarized in Table 1.

**Table 1** Notation list

| Notations      | Meanings  |
|----------------|---|
| $I$            | the set of users, $I = \{1, 2, \dots, m\}, i \in I$   |
| $j$            | the cloud resource provider $j$   |
| $F$            | the set of fog nodes, $F = \{1, 2, \dots, n\}, f \in F$                                     |
| $l$            | the third-party platform $l$  |
| $R$            | the set of resources, $R = \{1, 2, \dots, q\}, r \in R$                                     |
| $T$            | the total time of system  |
| $t$            | the time slot of system   |
| $\tau$         | the time slot of user   |
| $e_i$          | the execution time of user $i$  |
| $\theta_i$     | the requirement of user $i$   |
| $A_i$          | the arrival time slot of user $i$   |
| $t_i$          | the allocation time of user $i$   |
| $\varphi_i$    | the delay sensitivity of user $i$   |
| $\mathbf{z}_i$ | the time-varying resource requirement matrix of user $i$                                    |
| $z_{ir}^t$     | the amount of resource $r$ required by user $i$ at time $t$                                 |
| $H_i$          | the maximum expense that user $i$ is willing to pay   |
| $p_i$          | the transaction price of user $i$   |
| $\mathbf{p}_j$ | the vector of the unit prices of different resources of cloud resource provider $j$         |
| $\mathbf{p}_f$ | the vector of the unit prices of different resources of fog node $f$                        |
| $p_{jr}$       | the unit price of resource $r$ of cloud resource provider $j$                               |
| $p_{fr}$       | the unit price of resource $r$ of fog node $f$  |
| $\mathbf{c}_j$ | the vector of the capacities of different resources of cloud resource provider $j$          |
| $\mathbf{c}_f$ | the vector of the capacities of different resources of fog node $f$                         |
| $c_{jr}$       | the capacity of resource $r$ of cloud resource provider $j$                                 |
| $c_{fr}$       | the capacity of resource $r$ of fog node $f$  |
| $\mathbf{C}_j$ | the matrix of the remaining amount of resources at each time of cloud resource provider $j$ |
| $\mathbf{C}_f$ | the matrix of the remaining amount of resources at each time of fog node $f$                |
| $c_{jr}^t$     | the remaining amount of resource $r$ at time $t$ of cloud resource provider $j$             |
| $c_{fr}^t$     | the remaining amount of resource $r$ at time $t$ of fog node $f$                            |
| $v_i$          | the dominant resource proportion of user $i$  |
| $a_{ij}$       | the matching indicator of user $i$ in cloud resource provider $j$                           |
| $a_{if}$       | the matching indicator of user $i$ in fog node $f$  |
| $U_i$          | the utility generated by user $i$   |
| $U_j$          | the utility generated by cloud resource provider $j$  |
| $U_f$          | the utility generated by fog node $f$   |
| $U_l$          | the utility generated by the third-party platform $l$                                       |
| $V_{ij}$       | the payment of user $i$ to cloud resource provider $j$                                      |

**Table 1** (Continued)

| Notations | Meanings   |
|-----------|--|
| $V_{if}$  | the payment of user $i$ to fog node $f$                                    |
| $V_{il}$  | the payment of user $i$ to the third-party platform $l$                    |
| $V_{jl}$  | the payment of cloud resource provider $j$ to the third-party platform $l$ |
| $V_{fl}$  | the payment of fog node $f$ to the third-party platform $l$                |
| $C_i$     | the cost generated by user $i$   |
| $C_l$     | the cost generated by the third-party platform $l$                         |

### 3.2 Utility Function

#### 3.2.1 The Utility Function of User $i$

In Equation (1), for each user  $i$ , the utility function  $U_i$  consists of four parts. The first part  $H_i$  is the maximum expense that user  $i$  is willing to pay. The second part  $V_{ij}$  is the payment that user  $i$  offers to cloud resource provider  $j$  if the auction is successful, and the third part  $V_{if}$  is the payment that user  $i$  offers to fog node  $f$  when the transaction is successful, and the fourth part  $V_{il}$  is the payment that user  $i$  offers to the third-party platform  $l$ .

$$U_i = H_i - V_{ij} - V_{if} - V_{il}, \quad (1)$$

The second part  $V_{ij}$  is given in (2), in which  $\varphi_i$  represents the delay sensitivity of user  $i$ , i.e.,  $\varphi_i=1$  indicates that the user has high delay sensitivity and it should be allocated to fog nodes. And otherwise  $\varphi_i=0$  indicates that the user has low delay sensitivity and it should be allocated to the cloud resource provider. The parameter  $a_{ij}$  indicates the requirement satisfaction of user  $i$  to the cloud resource provider  $j$ , i.e.,  $a_{ij}=1$  indicates that the requirement is satisfied with the cloud resource provider  $j$  and otherwise  $a_{ij}=0$ . And  $p_i$  represents the final transaction price of user  $i$ .

$$V_{ij} = (1 - \varphi_i)a_{ij}p_i. \quad (2)$$

Specifically, the final transaction price  $p_i$  is given in (3), in which  $z_{ir}^\tau$  represents the amount of resource  $r$  required by user  $i$  in time  $\tau$ , and  $p_{jr}$ ,  $p_{fr}$  represent the unit prices of resource  $r$  of cloud resource provider  $j$  and fog node  $f$ , and  $e_i$  represents the execution time of each user.

$$p_i = \begin{cases} \sum_{r \in R} \sum_{\tau=1}^{e_i} z_{ir}^\tau p_{jr}, & a_{ij} = 1, \\ \sum_{r \in R} \sum_{\tau=1}^{e_i} z_{ir}^\tau p_{fr}, & a_{if} = 1. \end{cases} \quad (3)$$

The third part  $V_{if}$  is given in (4), in which  $a_{if}$  indicates the requirement satisfaction of user  $i$  to fog node  $f$ , i.e.,  $a_{if}=1$  indicates that the requirement is satisfied with fog node  $f$  and otherwise  $a_{if}=0$ . And  $p_i$  represents the final transaction price of user  $i$ .

$$V_{if} = a_{if}p_i. \quad (4)$$

The fourth part  $V_{il}$  is shown in (5). The cost  $V_{il}$  is a function of  $p_i$ . The  $(p_1, V_1)$  and  $(p_2, V_2)$  are the coordinates of two points on the curve  $V_{il}$ . The  $p_1, p_2, V_1, V_2$  represent constants and  $p_1 < p_2, V_1 < V_2$ .

$$V_{il} = \begin{cases} \left( a_{ij} + \sum_{f \in F} a_{if} \right) V_1, & p_i \leq p_1, \\ \left( a_{ij} + \sum_{f \in F} a_{if} \right) \left\{ \frac{V_2 - V_1}{p_2 - p_1} (p_i - p_1) + V_1 \right\}, & p_1 < p_i < p_2, \\ \left( a_{ij} + \sum_{f \in F} a_{if} \right) V_2, & p_i \geq p_2. \end{cases} \quad (5)$$

In summary, taking (2) and (4) into (1), the utility function  $U_i$  of each user  $i$  can be specifically shown in the following form.

$$U_i = H_i - [(1 - \varphi_i)a_{ij} + a_{if}]p_i - V_{il}. \quad (6)$$

Then, the cost  $C_i$  of each user  $i$  is given in (7).

$$C_i = V_{ij} + V_{if} + V_{il}. \quad (7)$$

Taking (2) and (4) into (7), we can obtain the cost  $C_i$  of each user  $i$  as shown in (8) as follows.

$$C_i = [(1 - \varphi_i)a_{ij} + a_{if}]p_i + V_{il}. \quad (8)$$

### 3.2.2 The Utility Function of Cloud Resource Provider $j$

For cloud resource provider  $j$ , the utility function  $U_j$  consists of two parts, as shown in (9). The first part  $\sum_{i \in I} V_{ij}$  is the resource revenue of providing resources, and the second part  $V_{jl}$  is the payment that cloud resource provider  $j$  offers to the third-party platform  $l$ .

$$U_j = \sum_{i \in I} V_{ij} - V_{jl}, \quad (9)$$

The first part  $\sum_{i \in I} V_{ij}$  is the sum of all transaction prices. The second part  $V_{jl}$  is further shown in (10). We assume that when the transaction is successful, the cloud resource provider  $j$  and the user  $i$  pay the same agency fee. So the agency fee paid by the cloud resource provider  $j$  to the third-party platform  $l$  can be represented by the sum of the agency fees of users.

$$V_{jl} = \sum_{i \in I} a_{ij} V_{il}. \quad (10)$$

In summary, taking (2) and (10) into (9), the utility function of the cloud resource provider  $j$  can be specifically shown in the following form.

$$U_j = \sum_{i \in I} (1 - \varphi_i) a_{ij} p_i - \sum_{i \in I} a_{ij} V_{il}. \quad (11)$$

### 3.2.3 The Utility Function of Fog node $f$

For each fog node  $f$ , the utility function  $U_f$  consists of two parts, as shown in (12). The first part  $\sum_{i \in I} V_{if}$  is the resource revenue of providing resources, and the second part  $V_{fl}$  is the payment that fog node  $f$  offers to the third-party platform  $l$ .

$$U_f = \sum_{i \in I} V_{if} - V_{fl}. \quad (12)$$

The first part  $\sum_{i \in I} V_{if}$  is the sum of all transaction prices. The second part  $V_{fl}$  is further shown in (13). We assume that when the transaction is successful, the fog node  $f$  and the user  $i$  pay the same agency fee. So the agency fee paid by the fog node  $f$  to the third-party platform  $l$  can be represented by the sum of the agency fees of users.

$$V_{fl} = \sum_{i \in I} a_{if} V_{il}. \quad (13)$$

In summary, taking (4) and (13) into (12), the utility function of the fog node  $f$  can be specifically shown in the following form.

$$U_f = \sum_{i \in I} a_{if} p_i - \sum_{i \in I} a_{if} V_{il}. \quad (14)$$

### 3.2.4 The Utility Function of the Third-Party Platform $l$

In Equation (15), for the third-party platform  $l$ , the utility function  $U_l$  consists of four parts. The first, second and third parts  $\sum_{i \in I} V_{il} + V_{jl} + \sum_{f \in F} V_{fl}$  are the agency fees paid by users, cloud resource provider and fog nodes to the third-party platform  $l$  when the transaction is successful, and the fourth part  $C_l$  is the cost of the allocation service provided by the third-party platform  $l$ .

$$U_l = \sum_{i \in I} V_{il} + V_{jl} + \sum_{f \in F} V_{fl} - C_l. \quad (15)$$

Taking (10) and (13) into (15), we can obtain the utility function  $U_l$  of the third-party platform  $l$  as shown in (16) as follows.

$$U_l = \sum_{i \in I} V_{il} + \sum_{i \in I} a_{ij} V_{il} + \sum_{f \in F} \sum_{i \in I} a_{if} V_{il} - C_l. \quad (16)$$

## 3.3 Model Formulation

In this section, we introduce the objective function of the time-varying multidimensional resource allocation model based on reverse auction in cloud and edge collaboration. We assume that resource providers, which include a cloud resource provider and some fog nodes, offer  $q$  types of resources, such as CPU, memory and storage. The unit prices of resource providers are

represented by vectors  $\mathbf{p}_j = \begin{bmatrix} p_{j1} \\ p_{j2} \\ \vdots \\ p_{jq} \end{bmatrix}$ , and  $\mathbf{p}_f = \begin{bmatrix} p_{f1} \\ p_{f2} \\ \vdots \\ p_{fq} \end{bmatrix}$ . And the resource capacities of them are

indicated by vectors  $\mathbf{c}_j = \begin{bmatrix} c_{j1} \\ c_{j2} \\ \vdots \\ c_{jq} \end{bmatrix}$ , and  $\mathbf{c}_f = \begin{bmatrix} c_{f1} \\ c_{f2} \\ \vdots \\ c_{fq} \end{bmatrix}$ . Then we can get these multidimensional resource capacity matrices as follows.

$$\mathbf{C}_j = \begin{bmatrix} c_{j1}^1 & c_{j1}^2 & \cdots & c_{j1}^T \\ c_{j2}^1 & c_{j2}^2 & \cdots & c_{j2}^T \\ \vdots & \vdots & \ddots & \vdots \\ c_{jq}^1 & c_{jq}^2 & \cdots & c_{jq}^T \end{bmatrix}, \quad (17)$$

$$\mathbf{C}_f = \begin{bmatrix} c_{f1}^1 & c_{f1}^2 & \cdots & c_{f1}^T \\ c_{f2}^1 & c_{f2}^2 & \cdots & c_{f2}^T \\ \vdots & \vdots & \ddots & \vdots \\ c_{fq}^1 & c_{fq}^2 & \cdots & c_{fq}^T \end{bmatrix}. \quad (18)$$

And we assume that the arrival time slots of users are different and the execution times of users are the same, that is, each user  $i \in I$  submits the resource requirement  $\theta_i = (A_i, \varphi_i, \mathbf{z}_i)$  at time slot  $A_i$  and the execution time is  $[1, e_i]$ , where  $\varphi_i$  represents the delay sensitivity of user  $i$  which values 1 or 0, and the time-varying resource requirement matrix  $\mathbf{z}_i$  is shown in Equation (19) as follows.

$$\mathbf{z}_i = \begin{bmatrix} z_{i1}^1 & z_{i1}^2 & \cdots & z_{i1}^{e_i} \\ z_{i2}^1 & z_{i2}^2 & \cdots & z_{i2}^{e_i} \\ \vdots & \vdots & \ddots & \vdots \\ z_{iq}^1 & z_{iq}^2 & \cdots & z_{iq}^{e_i} \end{bmatrix}. \quad (19)$$

For each user  $i$ , we introduce two variables  $a_{ij}$  and  $a_{if}$  to indicate the requirement satisfaction of user  $i$ , that is,  $a_{ij}=1$  or  $a_{if}=1$  indicates that the requirement is satisfied with the cloud resource provider  $j$  or fog node  $f$  respectively. Otherwise  $a_{ij}=0$  and  $a_{if}=0$ .

We denote the cost generated by each user  $i$  as  $C_i$  which is described in Equation (8). Therefore, the time-varying multidimensional resource allocation problem in cloud and edge collaboration can be modelled as an integer programming problem as follows.

$$M : \quad \min \quad \sum_{i \in I} C_i \quad (20)$$

$$\text{s.t.} \quad \sum_{i \in I} a_{ij} z_{ir}^{t-t_i+1} \leq c_{jr}^t, \quad (20A)$$

$$\sum_{i \in I} a_{if} z_{ir}^{t-t_i+1} \leq c_{fr}^t, \quad (20B)$$

$$a_{ij} + \sum_{f \in F} a_{if} \leq 1, \quad (20C)$$

$$\text{over } a_{ij}, a_{if}, \varphi_i \in \{0, 1\}, \quad 0 \leq p_i \leq H_i, \quad 1 \leq t - t_i + 1 \leq T, \quad (20D)$$

$$\forall i \in I, \forall r \in R, \forall t \in [1, T], \quad t_i \in [A_i, T - e_i + 1]. \quad (20E)$$

The objective (20) indicates that the goal of the time-varying multidimensional resource allocation problem is to minimize the cost of users. The inequalities (20A) and (20B) indicate that the resource allocation at any time  $t$  should not exceed the capacity of any resource. The inequality (20C) indicates that the user is allocated resources at most once during  $[1, T]$ . The constraint (20D) indicates that the value of  $a_{ij}$ ,  $a_{if}$  and  $\varphi_i$  is 0 or 1.

### 3.4 Model Analysis

The TMRAP in cloud and edge collaboration is a 0-1 integer programming problem. It aims to minimize the cost of users. The TMRAP is a function of the transaction price  $p_i$  because the agency fee for each transaction is a function of  $p_i$ . Thus, the goal of TMRAP mainly depends on the transaction price  $p_i$ . Therefore, we design a heuristic algorithm to solve the time-varying multidimensional resource allocation problem by following the greedy algorithm.

## 4 Resource Allocation Algorithm for Time-varying Multidimensional Resource Allocation Problem

### 4.1 Algorithm Description

To solve the problem of time-varying multidimensional resource allocation, we will introduce a reverse auction algorithm (RAA-TMRAP) which includes three stages. The first stage is collecting information from users and resource providers. The third-party platform collects requirements submitted by end users, and collects resource capacities and unit prices offered by resource providers. 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, and the matching indicator of each user is finally determined. We define the dominant resource proportion of each user as follows.

$$v_i = \begin{cases} \max_r \frac{\max_{\tau} z_{ir}^{\tau}}{c_{jr}}, & \varphi_i = 0, \\ \max_r \frac{\max_{\tau} z_{ir}^{\tau}}{\sum_{f \in F} c_{fr}}, & \varphi_i = 1, \end{cases} \quad (21)$$

where  $z_{ir}^{\tau}$  is the amount of resource  $r$  required by user  $i$  at time  $\tau$ ,  $\max_{\tau} z_{ir}^{\tau}$  represents the maximum amount of resource  $r$  required by user  $i$  during different time slots,  $\frac{\max_{\tau} z_{ir}^{\tau}}{c_{jr}}$  and  $\frac{\max_{\tau} z_{ir}^{\tau}}{\sum_{f \in F} c_{fr}}$  are the ratios of the maximum amount of resource  $r$  to the resource capacity  $c_{jr}$  and  $\sum_{f \in F} c_{fr}$  respectively. Finally, the  $\max_r \frac{\max_{\tau} z_{ir}^{\tau}}{c_{jr}}$  and  $\max_r \frac{\max_{\tau} z_{ir}^{\tau}}{\sum_{f \in F} c_{fr}}$  are the maximum of the corresponding ratio among all resources.

The third stage of the algorithm is resource pricing, which uses a pricing method that the transaction price is the sum of the products of the unit prices and the resource quantities.

Above all, in this algorithm, the resources are allocated in ascending order according to the dominant resource proportion  $v_i$ . And the transaction price is set according to the unit prices  $p_{jr}$  and  $p_{fr}$  to minimize the cost of users.

The general process of the algorithm is summarized as follows. The algorithm is invoked at  $t = 1$ . The time-varying requirement matrix  $\mathbf{z}_i$  records the resource requirement of user  $i$  in each period  $[1, e_i]$ . The users are arranged in ascending order according to the dominant resource proportion  $v_i$ . If the resource requirement is satisfied, the transaction price is set. The remaining resource capacity  $c_{jr}^t$  or  $c_{fr}^t$  will be updated until when the auction is successful.

## 4.2 Algorithm Analysis

The cores of the algorithm are the second stage and the third stage, i.e., resource allocation and pricing. The cost of users is decreased by determining the allocation order and transaction prices of them that resource providers need to supply.

In the second stage, we use a dominant-resource-based strategy as the basis for user ranking of resource allocation and finally determine the matching indicators of users. This strategy is derived from the dominant resource fairness (DRF), which calculates the ratio of the maximum demand of each resource required by the users to the total amount of resources in the system. The ratio is the dominant share (DS), and the resource with the largest ratio is the dominant resource (DR). This strategy aims 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, decrease the cost of users and conserve space to support the resource allocation of other users. Under the dominant-resource-based strategy, the algorithm will be more inclined to the user with a smaller dominant resource proportion. And in the third stage, we use a pricing method in which the transaction price is the sum of the products of the unit prices and the resource quantities.

In addition, we compare the RAA-TMRAP algorithm with the Random matching algorithm<sup>[21]</sup> to illustrate its effectiveness. Still, the Random matching algorithm does not support the time-varying resource requirements of users in resource allocation. So, we modify the dominant resource proportion to allocate users in the Random matching algorithm to compare it with our algorithm.

## 4.3 Algorithm Basic Steps

The implementation steps of the algorithm are described as follows:

At time  $t = 1, 2, \dots$

**Step 1** Initialize variables and parameters. Input the requirement  $\theta_i = (A_i, \varphi_i, \mathbf{z}_i)$  including the arrival time slot  $A_i$ , the delay sensitivity  $\varphi_i$  and the time-varying resource requirement matrix  $\mathbf{z}_i$  of user  $i$ , the vectors of the initial resource capacities  $\mathbf{c}_j$  and  $\mathbf{c}_f$ , and the vectors of the unit prices  $\mathbf{p}_j$  and  $\mathbf{p}_f$  of resource providers. Initialize the matching indicators  $a_{ij}$  and  $a_{if}$  of users and the set  $P$  of transaction prices.

**Step 2** Calculate the dominant resource proportion of each user  $i \in W_t$  at each time  $t$  and sort them to user sets  $W_t^1$  and  $W_t^2$  according to  $\varphi_i$ . Compute the dominant resource proportions  $v_i$  according to (21) and sort users in ascending order according to the dominant resource proportion  $v_i$ .

**Step 3** Determine whether the remaining resources of resource providers can satisfy the user requirement in period  $t$  according to  $\varphi_i$ . If fulfilled, the user  $i$  (i.e.,  $\varphi_i=0$ ) will be allocated

to the cloud resource provider, and the user  $i$  (i.e.,  $\varphi_i=1$ ) will be allocated to the fog node which has the lowest cost for it. Otherwise, consider the next user  $i \in W_t$ . If the users  $i \in W_t$  are entirely traversed, consider the users  $i \in W_{t+1}$  of next time  $t + 1$ .

**Step 4** Calculate the transaction price paid by each user to the corresponding resource provider. The third-party platform performs resource pricing that the transaction price  $p_i$  is the sum of the products of the unit prices and the resource quantities.

**Step 5** Update the matching indicators  $a_{ij}$  or  $a_{if}$  of users and the set  $P$  of transaction prices. When the user is successfully allocated, the matching indicators  $a_{ij}$  or  $a_{if}$  of users and the set  $P$  of transaction prices are updated.

**Step 6** Update the remaining amount of resources of the corresponding resource provider. After a successful transaction, the remaining amount  $c_{jr}^t$  or  $c_{fr}^t$  of resources is updated.

**Step 7** Set the stop criterion. The algorithm stops when the candidate users are entirely traversed at time slot  $t \in \{1, \dots, T\}$  or the resources are completely allocated. And the matching indicators  $a_{ij}$  and  $a_{if}$  of users and the set  $P$  of transaction prices are obtained.

The RAA-TMRAP algorithm is summarized as follows.

---

**Algorithm 1** RAA-TMRAP

---

**Input:**  $\theta_i = (A_i, \varphi_i, \mathbf{z}_i), \mathbf{c}_j, \mathbf{c}_f, \mathbf{p}_j, \mathbf{p}_f$ ;

**Output:**  $a_{ij}, a_{if}, P$ ;

```

1: {The first stage: Collecting information from users and resource providers}
2: for all  $i \in I$  do
3:   collect the requirements  $\theta_i = (A_i, \varphi_i, \mathbf{z}_i)$  from users
4: end for
5: for all  $j, f \in F$  do
6:   collect the vectors of the resource capacities  $\mathbf{c}_j$  and  $\mathbf{c}_f$ , and the vectors of the unit prices  $\mathbf{p}_j$  and  $\mathbf{p}_f$  from resource providers
7: end for
8: {The second stage: Determining resource allocation strategy and winners}
9:  $P \leftarrow \phi$ ;
10: for all  $t \in \{1, \dots, T\}, i \in I$  do
11:   if  $(A_i \leq t) \ \& \ (a_{ij} == 0) \ \& \ (a_{if} == 0)$  then
12:     collect the requirements  $\theta_i = (A_i, \varphi_i, \mathbf{z}_i)$  to user sets  $W_t^1, W_t^2$  according to  $\varphi_i$ ;
13:   end if
14:   for all  $i \in W_t^1, r \in R, \tau \in \{1, \dots, e_i\}$  do
15:      $v_i = \max_r \frac{\max_{\tau} z_{ir}^{\tau}}{c_{jr}}$ 
16:   else
17:      $v_i = \max_r \frac{\max_{\tau} z_{ir}^{\tau}}{\sum_{f \in F} c_{fr}}$ 
18:   end for

```

---

**Algorithm 1** RAA-TMRAP

---

```

19:  for all  $i \in W_t^1$  reordering, so  $v_1 \geq v_2 \geq \dots$ 
20:  end for
21:  for all  $i \in W_t^2$  reordering, so  $v_1 \geq v_2 \geq \dots$ 
22:  end for
23:  for all  $i \in W_t^1, r \in R, \tau \in \{1, \dots, e_i\}$  do
24:    if  $z_{ir}^\tau \leq c_{jr}^{\tau+t-1}$  then
25:       $t_i \leftarrow t$ ;
26:       $a_{ij} \leftarrow 1$ ;
27:       $c_{jr}^{\tau+t-1} \leftarrow c_{jr}^{\tau+t-1} - z_{ir}^\tau$ ;
28:    end if
29:  end for
30:  for all  $i \in W_t^2, r \in R, \tau \in \{1, \dots, e_i\}$  do
31:    for each  $f \in F$ 
32:      if  $z_{ir}^\tau \leq c_{fr}^{\tau+t-1}$  then
33:         $F_c \leftarrow F_c \cup f$ ;
34:      end if
35:    end for
36:    calculate  $C_i$  for each  $f \in F_c$  to find the lowest  $f$ 
37:     $t_i \leftarrow t$ ;
38:     $a_{if} \leftarrow 1$ ;
39:     $c_{fr}^{\tau+t-1} \leftarrow c_{fr}^{\tau+t-1} - z_{ir}^\tau$ ;
40:  end for
41: end for
42: {The third stage: Resource pricing}
43: for all  $i \in I$  do
44:  if  $a_{ij} == 1$  then
45:     $p_i \leftarrow \sum_{r \in R} \sum_{\tau=1}^{e_i} z_{ir}^\tau p_{jr}$ ;
46:  else if  $a_{if} == 1$  then
47:     $p_i \leftarrow \sum_{r \in R} \sum_{\tau=1}^{e_i} z_{ir}^\tau p_{fr}$ ;
48:  else
49:     $p_i \leftarrow 0$ ;
50:  end if
51:   $P \leftarrow P \cup \{p_i\}$ ;
52: end for
53: return  $a_{ij}, a_{if}, P$ ;

```

---

#### 4.4 Algorithm Properties

For resource allocation mechanism design, it is necessary to satisfy individual rationality and truthfulness. In this section, we first introduce the definitions which need to be satisfied in the mechanism design and then prove them.

**Definition 1** Individual rationality. To ensure individual rationality, the mechanism should satisfy the condition that when the user submits a requirement, its utility value will be larger than or equal to 0, that is,  $U_i \geq 0$ . In other words, as long as the user participates in the auction and reports its requirement truthfully, it 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 transaction price, and all buyers are unwilling to purchase resources at a cost higher than the transaction price. Thus, the transaction price is the critical value.

**Definition 3** Truthful. 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 RAA-TMRAP algorithm satisfies individual rationality.*

*Proof* From the utility function formula (6), the utility of user  $i$  is  $U_i = H_i - [(1 - \varphi_i)a_{ij} + a_{if}]p_i - V_{il}$ . When  $p_i$  is the optimal solution,  $H_i \geq [(1 - \varphi_i)a_{ij} + a_{if}]p_i + V_{il}$ , thus  $U_i \geq 0$ . According to the RAA-TMRAP algorithm, for each winner, the utility is larger than or equal to 0. By contrast, for each loser, the utility is always 0, so the mechanism satisfies individual rationality.

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

*Proof* In the third stage of the algorithm, when user  $i$  purchases resources from resource providers, it is reluctant to purchase resources at a cost higher than the transaction price. And each resource provider is unwilling to offer services at a price lower than the transaction price. In summary, for each resource provider and each user, the RAA-TMRAP algorithm satisfies critical value theory.

**Theorem 3** *The RAA-TMRAP algorithm is truthful.*

*Proof* First, it is shown that the RAA-TMRAP algorithm is monotonic. During resource allocation process, user  $i$  can increase the probability of successful applications by decreasing dominant resource proportion. According to (21), under the premise that  $c_{jr}$  and  $\sum_{f \in F} c_{fr}$  remain unchanged, if  $\max_{\tau} z_{1r}^{\tau} \leq \max_{\tau} z_{2r}^{\tau}$ ,  $v_1 \leq v_2$ . Therefore, if users want to rank higher when the RAA-TMRAP algorithm makes allocation, they can reduce the requirements for dominant resources. Thus, the RAA-TMRAP allocation algorithm is monotonic. Secondly, according to the pricing method of the RAA-TMRAP algorithm, we can know that the transaction price is the critical value. Therefore, according to Definition 3, the RAA-TMRAP algorithm is truthful.

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

*Proof* The RAA-TMRAP algorithm includes three stages: Collecting information from users and resource providers, determining resource allocation strategy and winners, and resource pricing. In the first stage, the time complexity of the lines 2~4 of the algorithm is  $o(m)$ , and the time complexity of the lines 5~7 of the algorithm is  $o(n + 1)$ . In the second stage, the

time complexity of the lines 14~18 of the algorithm is  $o(W_1 q T \max(e_i))$ , the time complexity of the lines 23~29 of the algorithm is  $o(W_1 q T \max(e_i))$ , and the time complexity of the lines 30~35 of the algorithm is  $o(W_2 q T \max(e_i))$ . In the third stage, the time complexity of the lines 43~52 of the algorithm is  $o(m)$ . So the time complexity of the RAA-TMRAP algorithm is  $o(m q T \max(e_i))$ . It will be invoked one time at  $t = 1$  during the entire period  $[1, T]$ . As a result, the time complexity of the RAA-TMRAP algorithm is polynomial.

## 5 Simulation and Numerical Examples

### 5.1 Experimental Setup

In this section, we provide the simulation results to verify the performance of the proposed algorithm of the practical VFC parking assistance system. As shown in Figure 1 of Section 3.1, we mainly consider the resource allocation among vehicle users, resource providers and an IoT provider in the VFC parking assistance system. In this system, the vehicle users act as end users to request the resources of VFC parking assistance. The resource providers include a cloud resource provider and some smart vehicles. Specifically, the cloud resource provider supplies resources for vehicle users with low delay sensitivity. And the smart vehicles act as fog nodes to provide resources for vehicle users with high delay sensitivity. In addition, the IoT provider acts as the third-party platform to receive information and coordinate allocation.

In the VFC parking assistance system, we use the MATLAB simulation platform to confirm the feasibility of the RAA-TMRAP algorithm and give a numerical instance to illustrate the time-varying multidimensional resource allocation problem of VFC. In this system, each requirement of users includes time-varying CPU, Memory and Storage demands for different time slots. The system consists of four resource providers, which include a cloud resource provider and three fog nodes, whose resource capacities are represented by (CPU, Memory, Storage). And the unit prices of their resources are indicated by  $p_{\text{CPU}}$ ,  $p_{\text{Memory}}$  and  $p_{\text{Storage}}$  respectively. Specifically, the information of resource providers is shown in Table 2.

**Table 2** The information of resource providers

| Resource provider | (CPU, Memory, Storage) | $p_{\text{CPU}}$ | $p_{\text{Memory}}$ | $p_{\text{Storage}}$ |
|-------------------|------------------------|------------------|---------------------|----------------------|
| Cloud             | (400, 500, 5600)       | 0.10             | 0.05                | 0.0009               |
| Fog node 1        | (30, 40, 520)          | 0.12             | 0.06                | 0.0009               |
| Fog node 2        | (30, 50, 520)          | 0.10             | 0.07                | 0.0010               |
| Fog node 3        | (40, 50, 560)          | 0.14             | 0.05                | 0.0011               |

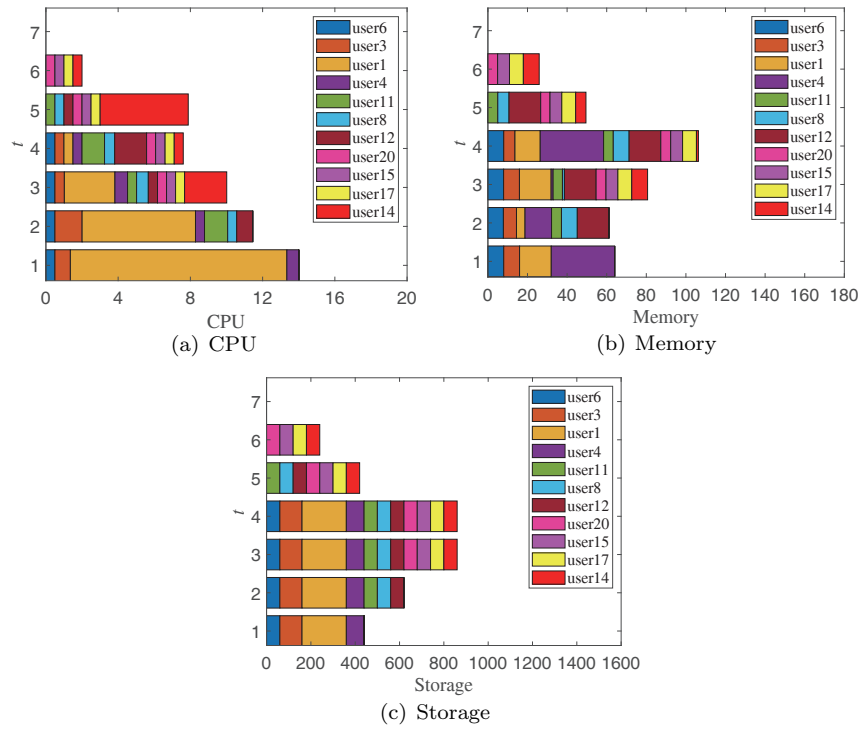
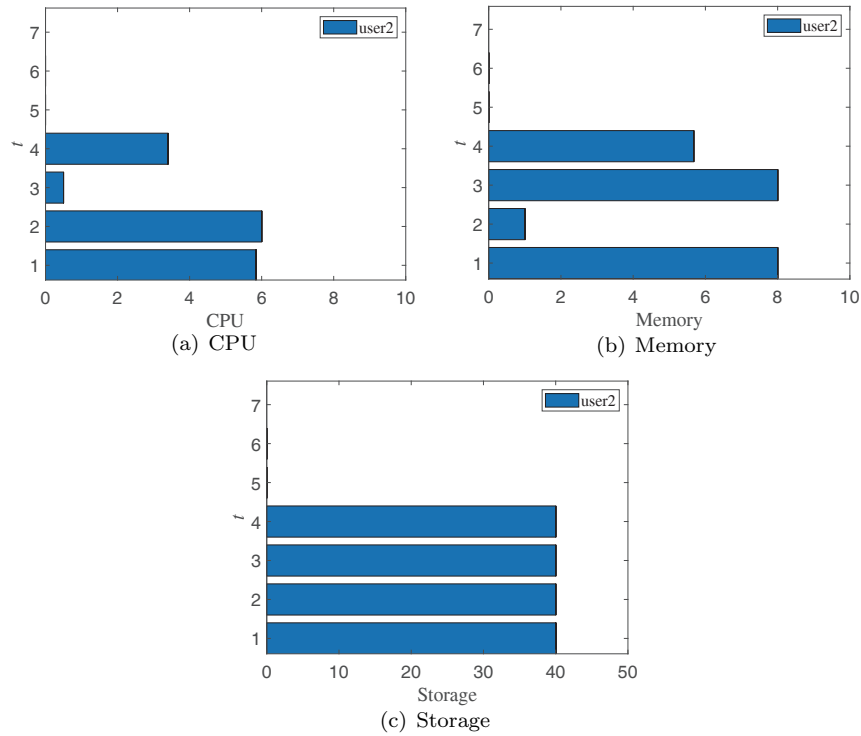
And the system also includes ten users whose requirements are  $\theta_i = (A_i, \varphi_i, \mathbf{z}_i)$ , where  $A_i$  represents the arrival time slot of user  $i$ ,  $\varphi_i$  represents the delay sensitivity of user  $i$  and  $\mathbf{z}_i$  represents the time-varying resource requirement matrix of user  $i$ . We assume that the arrival time slots of users are different and the execution times of users are the same, that is, each user arrives at  $A_i$  and the execution time is  $[1, 4]$ . Calculate the dominant resource proportions  $v_i$  to obtain the user requirements as shown in Table 3.

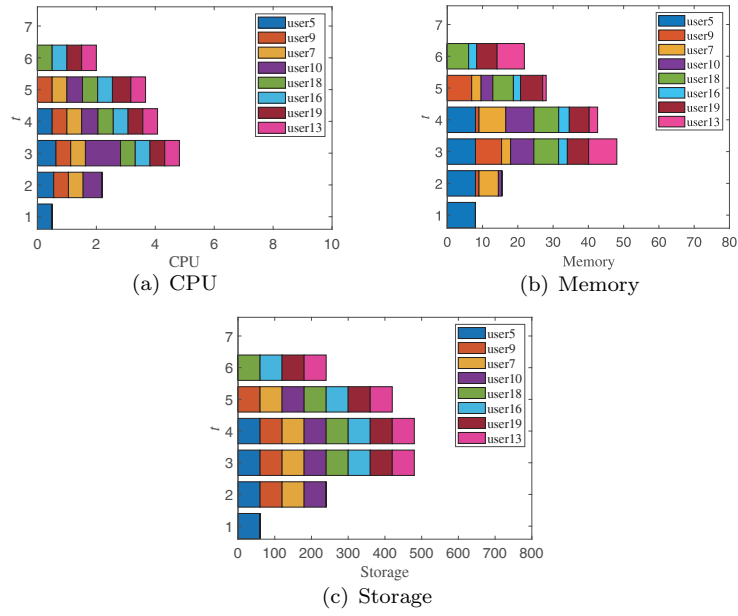
**Table 3** The user requirements

| $i$ | CPU                           | Memory                           | Storage              | $A_i$ | $\varphi_i$ | $v_i$ |
|-----|-------------------------------|----------------------------------|----------------------|-------|-------------|-------|
| 1   | (11.988, 6.278, 2.800, 0.500) | (16.000, 4.314, 16.000, 12.763)  | (200, 200, 200, 200) | 1     | 0           | 0.036 |
| 2   | (5.842, 6.000, 0.500, 3.399)  | (8.000, 1.000, 8.000, 5.675)     | (40, 40, 40, 40)     | 1     | 1           | 0.060 |
| 3   | (0.855, 1.505, 0.522, 0.500)  | (8.000, 6.600, 8.000, 5.667)     | (100, 100, 100, 100) | 1     | 0           | 0.018 |
| 4   | (0.653, 0.500, 0.700, 0.500)  | (32.000, 13.400, 1.000, 32.000)  | (80, 80, 80, 80)     | 1     | 0           | 0.064 |
| 5   | (0.500, 0.552, 0.629, 0.500)  | (8.000, 8.000, 8.000, 8.000)     | (60, 60, 60, 60)     | 1     | 1           | 0.057 |
| 6   | (0.500, 0.500, 0.500, 0.500)  | (8.000, 7.837, 7.935, 8.000)     | (60, 60, 60, 60)     | 1     | 0           | 0.016 |
| 7   | (0.500, 0.500, 0.500, 0.500)  | (5.495, 2.560, 7.590, 2.668)     | (60, 60, 60, 60)     | 2     | 1           | 0.054 |
| 8   | (0.500, 0.649, 0.562, 0.500)  | (8.000, 1.000, 8.000, 5.667)     | (60, 60, 60, 60)     | 2     | 0           | 0.016 |
| 9   | (0.500, 0.500, 0.500, 0.500)  | (1.000, 7.400, 1.000, 6.921)     | (60, 60, 60, 60)     | 2     | 1           | 0.053 |
| 10  | (0.640, 1.191, 0.555, 0.529)  | (1.000, 6.600, 8.000, 3.333)     | (60, 60, 60, 60)     | 2     | 1           | 0.057 |
| 11  | (1.283, 0.500, 1.247, 0.500)  | (5.000, 4.718, 4.888, 5.000)     | (60, 60, 60, 60)     | 2     | 0           | 0.011 |
| 12  | (0.886, 0.509, 1.766, 0.500)  | (16.000, 16.000, 16.000, 16.000) | (60, 60, 60, 60)     | 2     | 0           | 0.032 |
| 13  | (0.500, 0.500, 0.500, 0.500)  | (8.000, 2.400, 1.000, 7.723)     | (60, 60, 60, 60)     | 3     | 1           | 0.057 |
| 14  | (2.333, 0.500, 4.878, 0.500)  | (8.000, 1.000, 5.200, 8.000)     | (60, 60, 60, 60)     | 3     | 0           | 0.016 |
| 15  | (0.500, 0.525, 0.500, 0.500)  | (6.000, 6.000, 5.952, 6.000)     | (60, 60, 60, 60)     | 3     | 0           | 0.012 |
| 16  | (0.500, 0.500, 0.624, 0.500)  | (6.023, 5.631, 6.280, 5.800)     | (60, 60, 60, 60)     | 3     | 1           | 0.045 |
| 17  | (0.500, 0.506, 0.504, 0.500)  | (7.000, 7.000, 7.000, 7.000)     | (60, 60, 60, 60)     | 3     | 0           | 0.014 |
| 18  | (0.500, 0.503, 0.502, 0.500)  | (2.490, 3.008, 2.020, 2.275)     | (60, 60, 60, 60)     | 3     | 1           | 0.038 |
| 19  | (0.500, 0.519, 0.515, 0.500)  | (7.000, 7.000, 5.826, 6.059)     | (60, 60, 60, 60)     | 3     | 1           | 0.050 |
| 20  | (0.500, 0.500, 0.500, 0.500)  | (4.990, 5.000, 4.702, 4.910)     | (60, 60, 60, 60)     | 3     | 0           | 0.011 |

According to the dominant resource proportion  $v_i$ , the users are sorted in ascending order for resource allocation. Thus, the results of the RAA-TMRAP algorithm to the TMRAP are shown in Figures 3, 4, and 5 respectively.

Figures 3, 4 and 5 show one of the best solutions, which is a reverse auction-based allocation in cloud and edge collaboration. The users arrive at  $t = A_i$ . We sort candidates in ascending order according to the dominant resource proportion at each time, and finally get the result of resource allocation. In each figure, from the vertical direction, we can find the allocation result of users at different times; from the horizontal direction, we can observe the resource allocation result of different users at the same time. Specifically, at  $t = 1$ , the order of allocation is users 6, 3, 1, 4, 2 and 5. Users 6, 3, 1 and 4 are deployed to the cloud resource provider. User 2 is assigned to the fog node 2 because of the lowest cost. And user 5 is assigned to the fog node 3. At  $t = 2$ , the order of allocation is users 11, 8, 12, 9, 7 and 10. Users 11, 8 and 12 are allocated to the cloud resource provider. Users 9, 7 and 10 are deployed to the fog node 3. At  $t = 3$ , the order of allocation is users 20, 15, 17, 14, 18, 16, 19 and 13. Users 20, 15, 17 and 14 are allocated to the cloud resource provider. User 18, 16, 19 and 13 are assigned to the fog node 3. Then, the transaction price of each user in the RAA-TMRAP algorithm is shown in Table 4. The  $p_{ij}$  represents the transaction price of user  $i$  with low delay sensitivity to cloud resource

**Figure 3** The resource allocation of cloud resource provider**Figure 4** The resource allocation of fog node 2

**Figure 5** The resource allocation of fog node 3**Table 4** The transaction price of each user in the RAA-TMRAP algorithm

| $A_i$ | $\varphi_i$ | $i$ | $v_i$ | $p_{ij}$ | $p_{i1}$ | $p_{i2}$ | $p_{i3}$ |
|-------|-------------|-----|-------|----------|----------|----------|----------|
| 1     | 0           | 6   | 0.016 | 2.005    | -        | -        | -        |
| 1     | 0           | 3   | 0.018 | 2.112    | -        | -        | -        |
| 1     | 0           | 1   | 0.036 | 5.330    | -        | -        | -        |
| 1     | 0           | 4   | 0.064 | 4.443    | -        | -        | -        |
| 1     | 1           | 5   | 0.057 | -        | 2.398    | 2.698    | 2.169    |
| 1     | 1           | 2   | 0.060 | -        | 3.393    | 3.321    | 3.514    |
| 2     | 0           | 11  | 0.011 | 1.549    | -        | -        | -        |
| 2     | 0           | 8   | 0.016 | 1.570    | -        | -        | -        |
| 2     | 0           | 12  | 0.032 | 3.782    | -        | -        | -        |
| 2     | 1           | 9   | 0.053 | -        | 1.435    | 1.583    | 1.360    |
| 2     | 1           | 7   | 0.054 | -        | 1.555    | 1.722    | 1.460    |
| 2     | 1           | 10  | 0.057 | -        | 1.702    | 1.857    | 1.619    |
| 3     | 0           | 20  | 0.011 | 1.396    | -        | -        | -        |
| 3     | 0           | 15  | 0.012 | 1.616    | -        | -        | -        |
| 3     | 0           | 17  | 0.014 | 1.817    | -        | -        | -        |
| 3     | 0           | 14  | 0.016 | 2.147    | -        | -        | -        |
| 3     | 1           | 18  | 0.038 | -        | 1.044    | 1.126    | 1.034    |
| 3     | 1           | 16  | 0.045 | -        | 1.895    | 2.114    | 1.748    |
| 3     | 1           | 19  | 0.050 | -        | 2.013    | 2.255    | 1.843    |
| 3     | 1           | 13  | 0.057 | -        | 1.603    | 1.779    | 1.500    |

provider  $j$ , and the  $p_{i1}$ ,  $p_{i2}$  and  $p_{i3}$  represent the transaction price of user  $i$  with high delay sensitivity to fog nodes 1, 2, 3 respectively. Then we can get the final transaction price which is the lowest for each user in the RAA-TMRAP algorithm in Table 5.

**Table 5** The final transaction price of each user in the RAA-TMRAP algorithm

| $i$   | 1     | 2     | 3     | 4     | 5     | 6     | 7     | 8     | 9     | 10    |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| $p_i$ | 5.330 | 3.321 | 2.112 | 4.443 | 2.169 | 2.005 | 1.460 | 1.570 | 1.360 | 1.619 |
| $i$   | 11    | 12    | 13    | 14    | 15    | 16    | 17    | 18    | 19    | 20    |
| $p_i$ | 1.549 | 3.782 | 1.500 | 2.147 | 1.616 | 1.748 | 1.817 | 1.034 | 1.843 | 1.396 |

Furthermore, it is necessary to calculate the agency fee to obtain the cost of each user. 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 shall 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, and the successful user and the resource provider pay the same agency fee in each transaction. Therefore, the agency fee  $V_{il}$  can be shown explicitly as follows.

$$V_{il} = \begin{cases} 0.10 \left( a_{ij} + \sum_{f \in F} a_{if} \right), & p_i \leq 2, \\ 0.05 p_i \left( a_{ij} + \sum_{f \in F} a_{if} \right), & 2 < p_i < 3, \\ 0.15 \left( a_{ij} + \sum_{f \in F} a_{if} \right), & p_i \geq 3. \end{cases} \quad (22)$$

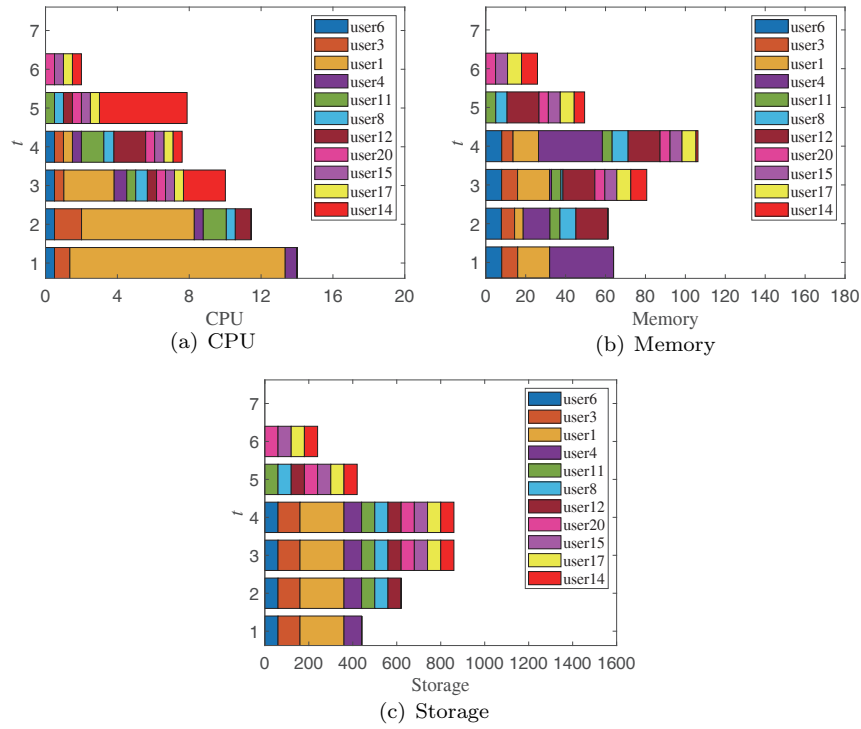
Then, by calculating the agency fee for each transaction, Table 6 gives the cost of each user in the RAA-TMRAP algorithm.

**Table 6** The cost of each user in the RAA-TMRAP algorithm

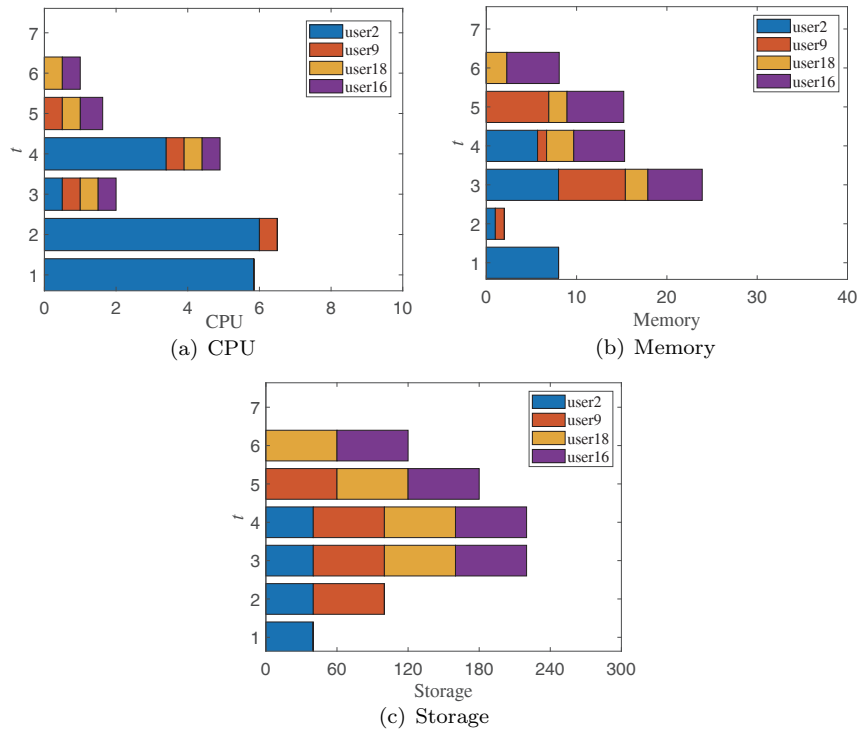
| $i$   | 1     | 2     | 3     | 4     | 5     | 6     | 7     | 8     | 9     | 10    |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| $C_i$ | 5.480 | 3.471 | 2.218 | 4.593 | 2.277 | 2.105 | 1.560 | 1.670 | 1.460 | 1.719 |
| $i$   | 11    | 12    | 13    | 14    | 15    | 16    | 17    | 18    | 19    | 20    |
| $C_i$ | 1.649 | 3.932 | 1.600 | 2.254 | 1.716 | 1.848 | 1.917 | 1.134 | 1.943 | 1.496 |

Furthermore, we compare the RAA-TMRAP algorithm with the Random matching algorithm to confirm its effectiveness. According to the dominant resource proportion  $v_i$ , the users are sorted in ascending order for resource allocation. Thus, the results of the Random matching algorithm to the TMRAP are shown in Figures 6, 7, 8 and 9 respectively.

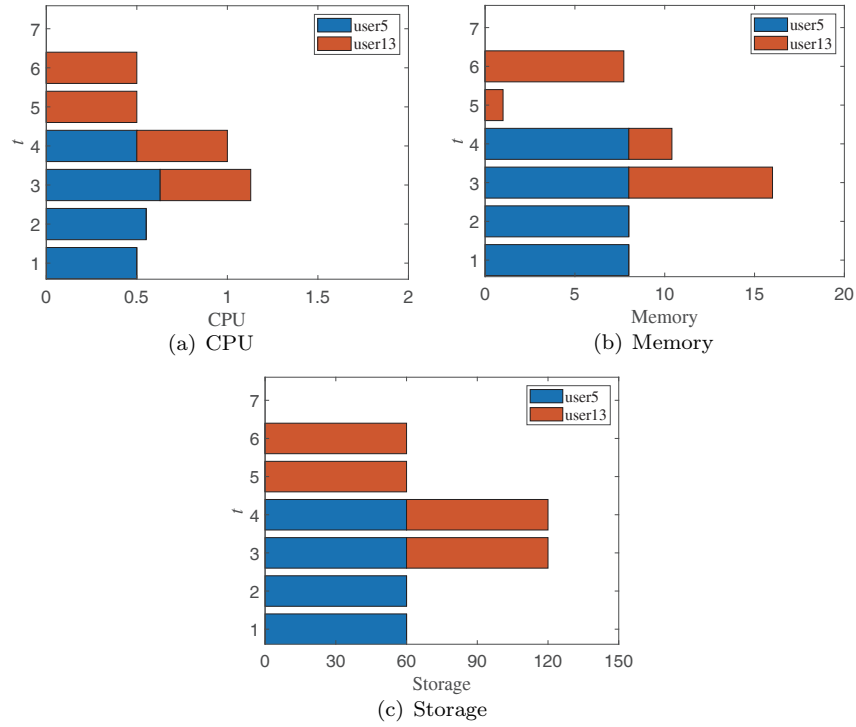
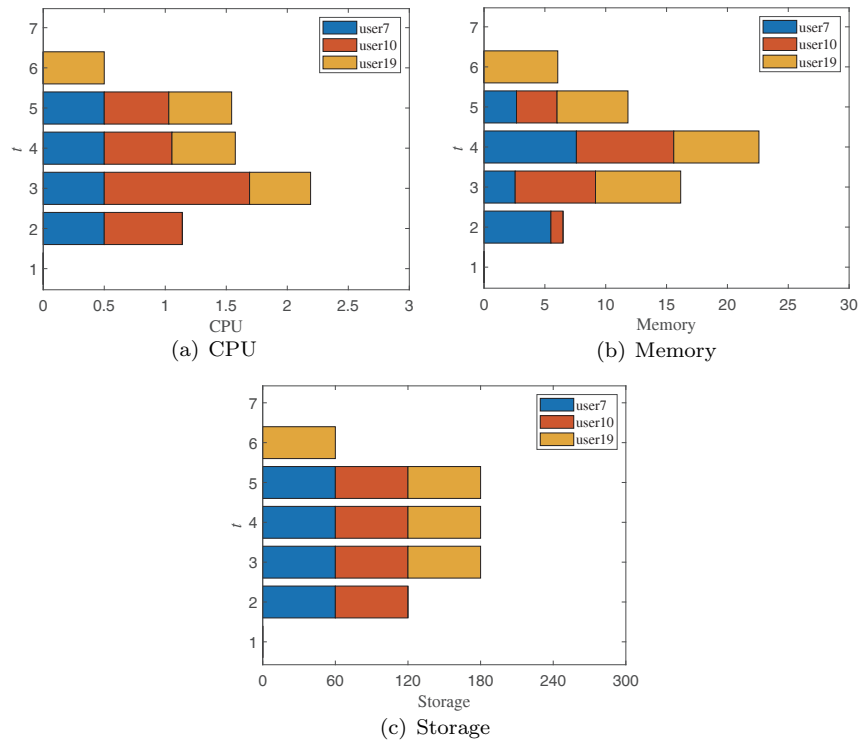
Specifically, at  $t = 1$ , the order of allocation is users 6, 3, 1, 4, 2 and 5. Users 6, 3, 1 and 4 are allocated to the cloud resource provider. User 2 is deployed to the fog node 1 and user 5 is assigned to the fog node 2 due to the lowest cost. At  $t = 2$ , the order of allocation is users 11,



**Figure 6** The resource allocation of cloud resource provider



**Figure 7** The resource allocation of fog node 1

**Figure 8** The resource allocation of fog node 2**Figure 9** The resource allocation of fog node 3

8, 12, 9, 7 and 10. Users 11, 8 and 12 are deployed to the cloud resource provider. User 9 is allocated to the fog node 1. Users 7 and 10 are assigned to the fog node 3. At  $t = 3$ , the order of allocation is users 20, 15, 17, 14, 18, 16, 13 and 19. Users 20, 15, 17 and 14 are deployed to the cloud resource provider. Users 18 and 16 are allocated to the fog node 1. User 13 is deployed to the fog node 2. User 19 is allocated to the fog node 3. Then, the final transaction price of each user in the Random matching algorithm is shown in Table 7.

**Table 7** The final transaction price of each user in the Random matching algorithm

| $i$   | 1     | 2     | 3     | 4     | 5     | 6     | 7     | 8     | 9     | 10    |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| $p_i$ | 5.330 | 3.393 | 2.112 | 4.443 | 2.698 | 2.005 | 1.460 | 1.570 | 1.435 | 1.619 |
| $i$   | 11    | 12    | 13    | 14    | 15    | 16    | 17    | 18    | 19    | 20    |
| $p_i$ | 1.549 | 3.782 | 1.779 | 2.147 | 1.616 | 1.895 | 1.817 | 1.044 | 1.843 | 1.396 |

From Tables 5 and 7, we can observe that under the premise of ranking in ascending order according to the dominant resource proportion  $v_i$ , the sum of final transaction price of each user in the RAA-TMRAP algorithm is 43.821. Still, the sum in the Random matching algorithm is 44.933. Therefore, we can prove that the RAA-TMRAP algorithm can better satisfy the minimum transaction price of each user. Then, we can obtain the cost of each user in the Random matching algorithm in Table 8 after calculating the agency fees.

**Table 8** The cost of each user in the Random matching algorithm

| $i$   | 1     | 2     | 3     | 4     | 5     | 6     | 7     | 8     | 9     | 10    |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| $C_i$ | 5.480 | 3.543 | 2.218 | 4.593 | 2.833 | 2.105 | 1.560 | 1.670 | 1.535 | 1.719 |
| $i$   | 11    | 12    | 13    | 14    | 15    | 16    | 17    | 18    | 19    | 20    |
| $C_i$ | 1.649 | 3.932 | 1.879 | 2.254 | 1.716 | 1.995 | 1.917 | 1.144 | 1.943 | 1.496 |

From Tables 6 and 8, we can observe that under the premise of ranking in ascending order according to the dominant resource proportion  $v_i$ , the cost of users in the RAA-TMRAP algorithm is 46.042, but the cost in the Random matching algorithm is 47.181. Therefore, we can find that the RAA-TMRAP algorithm can better satisfy the minimum cost of users. In summary, we can prove the effectiveness of the RAA-TMRAP algorithm.

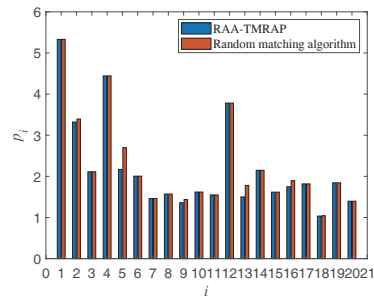
## 5.2 Analysis of Results

From Tables 5 and 7, we can compare the final transaction price of each user obtained by the RAA-TMRAP and the Random matching algorithm in Figure 10.

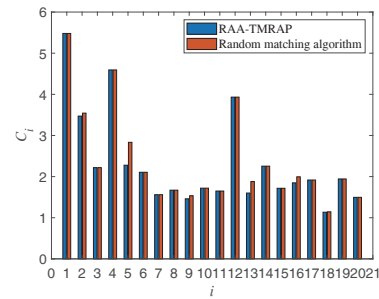
We can find that under the premise of ranking in ascending order according to the dominant resource proportion  $v_i$ , the final transaction price of each user in the RAA-TMRAP algorithm is not higher than that in the Random matching algorithm. Furthermore, from Tables 6 and 8, we can also compare the cost of each user obtained by the RAA-TMRAP and the Random matching algorithm in Figure 11.

We can find that under the premise of ranking in ascending order according to the dominant resource proportion  $v_i$ , the cost of each user in the RAA-TMRAP algorithm is not larger than

that in the Random matching algorithm. Therefore, we can further prove the effectiveness of the RAA-TMRAP algorithm.



**Figure 10** The final transaction price of each user



**Figure 11** The cost of each user

## 6 Conclusion

In this paper, we investigate the time-varying multidimensional resource allocation problem in the VFC parking assistance system of cloud and edge collaboration. We establish an integer programming model to achieve the goal of resource allocation in minimizing the cost of users. Furthermore, we design a heuristic algorithm to find an approximate solution. Theoretical analysis and experimental results both show that the algorithm can minimize the cost of users while ensuring individual rationality and truthfulness. However, we just consider price attributes of the reverse auction-based time-varying multidimensional resource allocation problem in the VFC parking assistance system of cloud and edge collaboration. Specifically, we just take the unit prices of different resource providers into account, but we did not care about some non-price characteristics of them which is important in VFC. Therefore, in the future, we will consider some non-price characteristics, such as location and reputation of different resource providers, which are crucial for providing fair resource allocation in VFC.

## References

- [1] Wang T, Lu Y, Wang J, et al. EIHPD: Edge-intelligent hierarchical dynamic pricing based on cloud-edge-client collaboration for IoT systems. *IEEE Transactions on Computers*, 2021, 70(8): 1285–1298.
- [2] Ghobaei-Arani M, Souri A, Rahmanian A A. Resource management approaches in fog computing: A comprehensive review. *Journal of Grid Computing*, 2020, 18: 1–42.
- [3] Sharghivand N, Derakhshan F, Siasi N. A comprehensive survey on auction mechanism design for cloud/edge resource management and pricing. *IEEE Access*, 2021, 9: 126502–126529.
- [4] Ren J, Yu G, He Y, et al. Collaborative cloud and edge computing for latency minimization. *IEEE Transactions on Vehicular Technology*, 2019, 68(5): 5031–5044.
- [5] Waheed A, Shah M A, Mohsin S M, et al. A comprehensive review of computing paradigms, enabling computation offloading and task execution in vehicular networks. *IEEE Access*, 2022, 10: 3580–3600.
- [6] Ning Z, Huang J, Wang X. Vehicular fog computing: Enabling real-time traffic management for smart cities. *IEEE Wireless Communications*, 2019, 26(1): 87–93.
- [7] Hou X, Li Y, Chen M, et al. Vehicular fog computing: A viewpoint of vehicles as the infrastructures. *IEEE Transactions on Vehicular Technology*, 2016, 65(6): 3860–3873.
- [8] Sethi V, Pal S, Vyas A. Online energy-efficient scheduling algorithm for renewable energy-powered roadside units in VANETs. *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, 2020, 506–514.
- [9] Xu X, Huang Q, Yin X, et al. Intelligent offloading for collaborative smart city services in edge computing. *IEEE Internet of Things Journal*, 2020, 7(9): 7919–7927.

- [10] Pham T N, Tsai M F, Nguyen D B, et al. A cloud-based smart-parking system based on internet-of-things technologies. *IEEE Access*, 2015, 3: 1581–1591.
- [11] Lin T, Rivano H, Le M F. A survey of smart parking solutions. *IEEE Transactions on Intelligent Transportation Systems*, 2017, 18(12): 3229–3253.
- [12] Sookhak M, Yu F R, He Y, et al. Fog vehicular computing: Augmentation of fog computing using vehicular cloud computing. *IEEE Vehicular Technology Magazine*, 2017, 12(3): 55–64.
- [13] Zhang W, Zhang Z J, Chao H C. Cooperative fog computing for dealing with big data in the internet of vehicles: Architecture and hierarchical resource management. *IEEE Communications Magazine*, 2017, 55(12): 60–67.
- [14] Ibrar M, Akbar A, Jan S, et al. ARTNet: AI-based resource allocation and task offloading in a reconfigurable internet of vehicular networks. *IEEE Transactions on Network Science and Engineering*, 2022, 9(1): 67–77.
- [15] Zhang J, Yang X, Xie N, et al. An online auction mechanism for time-varying multidimensional resource allocation in clouds. *Future Generation Computer Systems*, 2020, 111: 27–38.
- [16] Li S, Sun W. Utility maximisation for resource allocation of migrating enterprise applications into the cloud. *Enterprise Information Systems*, 2021, 15(2): 197–229.
- [17] Li S, Liu H, Li W, et al. An optimization framework for migrating and deploying multiclass enterprise applications into the Cloud. *IEEE Transactions on Services Computing*, 2022.
- [18] Angelelli E, Filippi C. On the complexity of interval scheduling with a resource constraint. *Theoretical Computer Science*, 2011, 412: 3650–3657.
- [19] Lee Y, Jeong S, Masood A, et al. Trustful resource management for service allocation in fog-enabled intelligent transportation systems. *IEEE Access*, 2020, 8: 147313–147322.
- [20] Zhang Y, Wang C, Wei H. Parking reservation auction for parked vehicle assistance in vehicular fog computing. *IEEE Transactions on Vehicular Technology*, 2019, 68(4): 3126–3139.
- [21] Zhu L, Sun L, Yan Y. Parking assistance scheme based on reverse auction in vehicle fog computing. *Computer Engineering*, 2020, 46(7): 14–20.
- [22] Alamer A, Basudan S. An efficient truthfulness privacy-preserving tendering framework for vehicular fog computing. *Engineering Applications of Artificial Intelligence*, 2020, 91: 103583–103593.
- [23] Peng X, Ota K, Dong M. Multiattribute-based double auction toward resource allocation in vehicular fog computing. *IEEE Internet of Things Journal*, 2020, 7(4): 3094–3103.
- [24] Junior F F, Dias K L, d'Orey P M, et al. FogWise: On the limits of the coexistence of heterogeneous applications on fog computing and Internet of Vehicles. *Transactions on Emerging Telecommunications Technologies*, 2021, 32(1): 1–21.
- [25] Zhou Z, Liu P, Feng J, et al. Computation resource allocation and task assignment optimization in vehicular fog computing: A contract-matching approach. *IEEE Transactions on Vehicular Technology*, 2019, 68(4): 3113–3125.
- [26] Shojafar M, Cordeschi N, Baccarelli E. Energy-efficient adaptive resource management for real-time vehicular cloud services. *IEEE Transactions on Cloud Computing*, 2019, 7(1): 196–209.
- [27] Chen X, Thomas N, Zhan T, et al. A hybrid task scheduling scheme for heterogeneous vehicular edge systems. *IEEE Access*, 2019, 7: 117088–117099.
- [28] Liao H, Mu Y, Zhou Z, et al. Blockchain and learning-based secure and intelligent task offloading for vehicular fog computing. *IEEE Transactions on Intelligent Transportation Systems*, 2021, 22(7): 4051–4063.
- [29] Li J, Zhang J, Li W, et al. A fair distribution strategy based on shared fair and time-varying resource demand. *Journal of Computer Research and Development*, 2019, 56(7): 1534–1544.
- [30] Zaman S, Grosu D. Combinatorial auction-based allocation of virtual machine instances in clouds. *Journal of Parallel and Distributed Computing*, 2012, 73: 495–508.
- [31] Mashayekhy L, Nejad M M, Grosu D, et al. An online mechanism for resource allocation and pricing in clouds. *IEEE Transactions on Computers*, 2016, 65(4): 1172–1184.
- [32] Zhang J, Xie N, Zhang X, et al. Strategy-proof mechanism for online time-varying resource allocation with restart. *Journal of Grid Computing*, 2021, 19(25): 1–20.
- [33] Zhang J, Chi L, Xie N, et al. Strategy-proof mechanism for online resource allocation in cloud and edge collaboration. *Computing*, 2022, 104: 383–412.