Real-time Pricing-based Resource Allocation in Open Market Environments

PANKAJ MISHRA, Nagoya Institute of Technology, Japan and University of Wollongong, Australia
AHMED MOUSTAFA, Nagoya Institute of Technology, Japan
TAKAYUKI ITO, Kyoto University, Japan

Open market environments consist of a set of participants (vendors and consumers) that dynamically leave or join the market. As a result, the arising dynamism leads to uncertainties in supply and demand of the resources in these open markets. In specific, in such uncertain markets, vendors attempt to maximise their revenue by dynamically changing their selling prices according to the market demand. In this regard, an optimal resource allocation approach becomes immensely needed to optimise the selling prices based on the supply and demand of the resources in the open market. Therefore, optimal selling prices should maximise the revenue of vendors while protecting the utility of buyers. In this context, we propose a real-time pricing approach for resource allocation in open market environments. The proposed approach introduces a priority-based fairness mechanism to allocate the available resources in a reverse-auction paradigm. Finally, we compare the proposed approach with two state-of-the-art resource allocation approaches. The experimental results show that the proposed approach outperforms the other two resource allocation approaches in its ability to maximise the vendors' revenue.

CCS Concepts: • **Theory of computation → Multi-agent reinforcement learning**; **Computational pricing and auctions;**

Additional Key Words and Phrases: Open market environments, resource allocation, reinforcement learning, real-time pricing

ACM Reference format:
Pankaj Mishra, Ahmed Moustafa, and Takayuki Ito. 2023. Real-time Pricing-based Resource Allocation in Open Market Environments. *ACM Trans. Internet Technol.* 23, 1, Article 1 (April 2023), 22 pages.
https://doi.org/10.1145/3465237

1 INTRODUCTION

Resource allocation in an **open market environment (OME)** is a challenging and widely studied problem in many domains, not limited to manufacturing supply chains [55, 60], open cloud environments [38], wireless sensor networks [17], and so on. In such OMEs, multiple vendors and buyers exist that service and request different kinds of resources, respectively. Developing

This research was supported by the Joint Degree Program (JDP) and [KAKENHI Young Researcher] Grant No. 20K19931.
Authors' addresses: P. Mishra (corresponding author), Nagoya Institute of Technology, Nagoya, Aichi, Japan, 466-8555 and University of Wollongong, Wollongong, Australia; emails: m.pankaj.890@stn.nitech.ac.jp, ppm468@uowmail.edu.au; A. Moustafa, Nagoya Institute of Technology, Nagoya, Aichi, Japan, 466-8555; email: ahmed@nitech.ac.jp; T. Ito, Kyoto University, Kyoto, Kyoto, Japan, 606-8501; email: ito@i.kyoto-u.ac.jp.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2023 Association for Computing Machinery.
1533-5399/2023/04-ART1 $15.00
https://doi.org/10.1145/3465237

an optimal allocation policy and pricing policy in OME is a challenging problem because of the contradicting goal of the participants. In specific, vendors focuses on maximising their revenue by selling most of their available resources at the highest possible price. However, buyers focus on maximising their utility by minimising their cost for the maximum possible quality of resources. In addition, in OME, the dynamic arrival and departure of participants (i.e., vendors and buyers) leads to uncertainty of supply and demand of the resources in these environments. Further, the self-interested buyers tend to behave strategically and misreport their undisclosed preference information [36]. So, developing optimal pricing in such a dynamic environment with strategically behaving buyers becomes a complicated and challenging problem.

In this regard, many resource allocation approaches with various pricing policies have been proposed for different OMEs in the past. For instance, the authors in References [41, 62, 66] proposed a set of auction-based resource allocation approaches that considered cloud environments. However, in these auction-based approaches, resources were allocated based on fixed-pricing policies. Meanwhile, dynamic pricing approaches were also proposed in References [5, 24, 59], which were based on statistical mathematical models. However, these approaches fail to address the dynamic change in the supply and demand of resources in the OMEs. To handle the dynamism of these environments, machine learning-based dynamic pricing [15] approaches have been adopted. However, these approaches focus mainly on maximising the revenues of vendors, based on the change in demand, and fail to optimise the prices exclusively for each buyer. Also, existing machine learning fails to incorporate undisclosed preferences of the participants, which they do not reveal to each other while resource allocation [36].

Further, in a service-based OME, apart from supply and demand in the market, to cope with the competition, resource prices are influenced by the offered price of the other vendors in the market. Moreover, OME's optimal resource allocation policy should foster fair participation and competitiveness [34, 53] of vendors to avoid monopoly in the market, which is known as bidder drop problem [6]. Therefore, an auction paradigm should adapt to dynamically changing supply and demand while maintaining the equilibrium in an open-market environment [34]. In specific, designing an auction paradigm in such an open market has three key challenges, as follows: (1) *participation*: encouraging the participation of vendors in the auction, (2) *efficiency*: the allocation of resources with optimal resource utilisation, and (3) *fairness*: each vendor should be given a fair chance to win the auction. In addition to the above considerations, the knowledge of buyers' *quality uncertainty*,[1] i.e., their *private values* [36], is used in determining the appropriate vendor for the buyer's request.

To the best of our knowledge, none of the existing approaches addresses all the challenges effectively. Therefore, to address the above-mentioned challenges, this research proposes an efficient resource allocation approach for OMEs. In particular, the proposed approach implements a learning model to optimise the pricing policy in a dynamically changing OMEs, then fairly allocates the requested resources based on a proposed priority mechanism in a reverse-auction paradigm. Moreover, in such dynamic OMEs, the behaviour of the participants loosely depends on the pre-existing datasets; rather, it depends on the real-time updates in the open market. In this regard, learning techniques, such as supervised or unsupervised learning are not appropriate, due to the inappropriate datasets. Therefore, we implement a **reinforcement learning (RL)** technique [48] for the proposed real-time pricing approach. The contributions of this research are as follows:

- First, a novel reinforcement-learning-based real-time pricing algorithm is proposed, which optimises the base prices of the requested resources for the vendors.

[1]Preference of a buyer for a particular vendor over other vendors.

- Second, we introduce a preference labelling scheme, which categorises all the dynamically arriving buyers based on their past behaviour. This further aids the real-time pricing algorithm to learn the undisclosed choices of different buyers.
- Third, we introduce a priority-based fairness mechanism for the selection of vendors to give fair chances to all vendors and to avoid monopoly in the environment.

The rest of this article is organised as follows: The formulation of the resource allocation problem in OMEs is introduced in Section 2. Section 3 presents the modelling of the dynamic pricing problem as a Markov decision process. In Sections 4 and 5, the proposed real-time bidding algorithm and vendor elicitation strategy are discussed, respectively. In Section 6, the experimental results are presented for evaluating the proposed approach. In Section 7, the related works are discussed. The article is concluded in Section 8.

2 PROBLEM FORMULATION

This section presents a resource allocation problem formulation into a *common value* [42] reverse-auction paradigm. In our setting, a broker intermediates the resource allocation, through a multi-agent environment. In this context, we intend to design an optimal auction paradigm, wherein the broker efficiently allocates the resources based on a set of learning agents. In this setting, resources are allocated in an episodic manner, such that each episode has many auctions. Each auction is about the allocation of a single resource request. The length of the episode t_{max} represents the maximum time after which the market closes and resets to its initial setting, i.e., total available resources, potential vendors, and requesting buyers. However, the trained agents are not initialised at each episode.

In the following subsections, we would briefly introduce all the three stakeholders in the resource allocation problem, i.e., the buyers, the vendors, and the broker.

2.1 Buyer

We consider an OME with set of buyers B, denoted as $b_j = \{req_{b_j}, l_{b_j}, d_{b_j}\}$, where req_{b_j} is the set of non-perishable[2] resources denoted as $req_{b_j} = \{res_{1,j}, res_{2,j}, \ldots, res_{k,j}\}$ for set K of k different types of resources, where $res_{i,j}$ represents the quantity of resource type $i \in K$; whereas l_{b_j} is length of request for which resource is being requested, and d_{b_j} is the deadline within which the request is to be satisfied, $\forall b_j \in B$. In addition, each buyer is characterised by their undisclosed preferences, which are the *private values* [36] and not known by the vendors. It should be noted, we assume that buyers do not have any budget constraints, and buyer focus is on acquiring the resources at minimum possible price.

2.2 Vendor

We consider an OME a set of n bidding vendors V, denoted as $v_i = \{bid(v_i, b_j), req_{b_j}\}$, where $bid(v_i, b_j)$ denotes the offered bid value for the resource request req_{b_j} from the buyer b_j, $b_j \in B$, $\forall v_i \in V$. In this context, each bidding vendor has maximum available set of available resources in the OME, i.e., the capacity of the vendors is denoted as $\mathbf{C} = \{C_{v_1}, \ldots, C_{v_n}\}$ and capacity of vendor $v_i \in V$ is denoted as $C_{v_i} = \{c_{1,i}, \ldots, c_{k,i}\}$, where $k \in K$. Also, it should be noted that, to lower the complexity, we assume that vendors do not join or leave the OME in the middle of auction.

2.3 Broker

In the proposed reverse-auction paradigm, the broker intermediates the whole resource allocation problem in the OME. Figure 1 depicts the architecture of the broker, which consists of three key

[2]Resources can be re-used once released by a buyer.

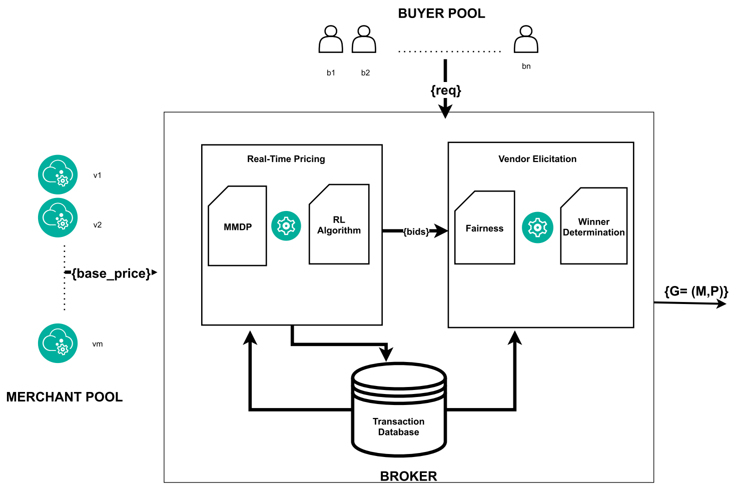

Fig. 1. Architecture of the broker.

modules, namely: the *Real-Time Pricing* module, the *Vendor Elicitation* module, and the *Transaction Database*. In this context, first, from *pool of buyers B*, buyers arrive dynamically and submit their resource requests to *pool of vendors* through a broker. Then, for every resource request, all the potential vendors submit their base prices to the broker. Hereafter, vendors authorise the brokers to bid on their behalf. Upon receiving all these base prices, the broker generates vendor agents, which bids on behalf of all the potential vendors. Then, vendor agents compute optimal bid values using *Real Time Pricing* module. This *Real Time Pricing* module is implemented using a novel RL Algorithm, which takes historical auction data from the *Transaction Database* as input. Then, finally, based on these optimised base prices, the broker determines an allocation rule $\alpha(v_i, b_j)$ and pricing rule $\rho(v_i, b_j)\ \forall v_i \in V, b_j \in B$ using *Vendor Elicitation* module. This *Vendor Elicitation* module implements a novel fairness technique and winner determination strategy. Also, after each allocation, *Transaction Database* is updated.

In this way, the broker intermediates the resource allocation and payment between vendors and buyers, through vendor agents. Note that, within each episode, allocation rule $\alpha(v_i, b_j)$ makes only feasible allocations, i.e., resource request will be allocated to a vendor only if a vendor has enough resources and that complete before the start of t_{max} as shown in Equation (1).

$$\sum_{t \in 1}^{t_{max}} \alpha(v_i, b_j) \times \mathbf{1}(t + l_{b_j} \leq t_{max}) \times \mathbf{1}(t \leq d_{b_j}) \times \mathbf{1}(req_{b_j} \leq C_{v_i}) \forall v_i, b_j, \tag{1}$$

where $\mathbf{1}(.)$ is the indicator function, such that $\mathbf{1}(.) = 1$ if condition inside is true or else $\mathbf{1}(.) = 0$. In this manner, broker plays the crucial role in whole resource allocation mechanism and also ensures the stability in the OME, i.e., competitiveness [53], *fairness* [34], truthfulness [14], and so on.

In this regard, all three stakeholders in the proposed reverse-auction paradigm interact with each other in the OME. In the next section, we would introduce the novel learning-based real-time pricing policy, which optimises the vendors' offered bid in real-time.

3 REAL-TIME PRICING AS MARKOV DECISION PROCESS

In this section, we model the real-time pricing problem in reverse-auction paradigm as **multi-agent Markov decision process (*MMDP*)**, wherein vendors are represented as autonomous and independent vendor agents.[3] In this regard, for set of N vendor agents, *MMDP* is defined as joint state-space s, which represents the state of possible status of all the agents and a set of action space A_i, $\forall i \in N$ representing the state space for all the agents. In this *MMDP*, first all the agents $i \in N$ perform an action $a_i \in A_i$ based on their decision policy $\pi_{\theta_i} : s \times A_i \mapsto [0, 1]$. Then, after execution of action a_i, agents are transferred to next state (s') based on joint transition function (τ), s.t. $\tau : s \times A_1 \times \cdots \times A_n \mapsto s'$. Finally, all the agents receive reward (r_i) from set of rewards R based on the current state and the performed action, such that, $r_i : S \times A_i \longrightarrow R$. Also, the private observation of each agent changes, such that, $s_i : S' \longrightarrow s_i$. In this setting, the initial state-space of all the agents are determined by a predefined distribution. In the real-time pricing problem, the goal is to compute the optimal action value for all the agents, which optimises their base-prices. To achieve that, agent i focuses on updating its policy π_i, which maximises the total expected long term reward R_i in each episode of length t_{max}, i.e., $R_i = \sum_{t=0}^{t_{max}} \lambda^t r_i^t$, where λ is the discount factor[4] [48].

In the following subsections, we explain the proposed profiling scheme and present the three entities *states*, *actions,* and *rewards* in the the proposed *MMDP*-based model.

3.1 Profiling Scheme

In the OME, buyers with different volume of resource requests arrive dynamically, and agent aims to model its pricing model following the arriving buyer. So, agents focus on adapting their selling prices based on the potential buyers (demand) and all the potential agents (supply) in the OME. Also, agents are not aware of all the other potential vendor agents in the market. Therefore, to estimate and model the supply and demand in the OME, we deploy this novel profiling scheme. In specific, vendors utilise the different parameters from the past auctions, stored in the *Transaction Database*, namely: set of requested resources, a deadline of the request, winning agent, the payment received by the agent, and selling price of all the vendors. However, participants (vendors and buyers) in the OME are changing, and it is not possible to observe a pattern for each participant. Therefore, we categorise vendors and buyers with the same volume of available resources or requested resources under the same profile. In this regard, we categorise vendor agents into u different profiles denoted as $V' = \{v'_1, \ldots, v'_u\}$. Similarly, dynamically arriving similar buyers are categorised into w different profiles, denoted as $B' = \{b'_1, \ldots, b'_w\}$. Then, for each pair of vendor profile v'_i and buyer profile b'_j, different auction parameters are stored in the *transaction database*, where $v'_i \in V'$ and $b'_j \in B'$. For instance, $revenue(v'_i, b'_j)$ represents the revenue that is earned by vendor profile v'_i and buyer profile b'_j. Similarly, other auction parameters are also recorded in the transaction database after each auction. Then, based on these past auction info, the welfare of allocation for every pair of vendor and buyer profiles in the OME is represented as a quality feature vector. In specific, a quality feature vector $F(v'_i, b'_j$ representing the welfare for buyer profile b'_j and vendor profile v'_i. This is denoted as, $F(v'_i, b'_j) \equiv [r_{mean}(v'_i, b'_j), p_{mean}(v'_i, b'_j), u_{mean}(v'_i, b'_j)]$, where $r_{mean}(v'_i, b'_j)$, $p_{mean}(v'_i, b'_j)$, and $u_{mean}(v'_i, b'_j)$ denote the quality parameters; mean revenue, mean penalty, and mean utility (profit), respectively. Intuitively, higher values of mean revenue and mean profit and lower values of mean penalty suggest allocation is optimal, and these parameters are computed using Equations (2)–(4).

$$r_{mean}\left(v'_i, b'_j\right) = \frac{revenue(v'_i, b'_j)}{total\_allocated(v'_i, b'_j)}, \tag{2}$$

[3]Vendor agents and agents are used interchangeably.
[4]It is commonly used in RL algorithms, which is originally a financial term to represent the significance of future or delayed returns.

where $revenue(v'_i, b'_j)$ and $total\_allocated(v'_i, b'_j)$ denote the average revenue and total potential buyers' requests allocated for a pair of vendor and buyer, respectively.

$$p_{mean}\left(v'_i, b'_j\right) = \frac{penalty(v'_i, b'_j)}{total\_rejected(v'_i, b'_j)}, \tag{3}$$

where $penalty(v'_i, b'_j)$ and $total\_rejected(v'_i, b'_j)$ denote the average penalty imposed and total potential buyers' request rejected for a pair of vendor and buyer, respectively.

$$u_{mean}\left(v'_i, b'_j\right) = \frac{revenue(v'_i, b'_j) - penalty(v'_i, b'_j)}{total\_request(v'_i, b'_j)} \tag{4}$$

Apart from above quality parameters, the *transaction database* tuples are used in computing the acceptance ratio (Υ), such that $0 \le \Upsilon \le 1$, as denoted by Equation (5).

$$\Upsilon(v_k, b_l) = \frac{acceptance\_rate(v_k, b'_j)}{acceptance\_rate(v'_i, b'_j)}, \tag{5}$$

where $v_k \in V$, $b_l \in B$ and $v'_i \in V'$, $b'_j \in B'$ are the corresponding profiles of vendor v_k and b_l, respectively. Whereas, acceptance rate is the ratio of number of times a buyer is allocated to total number of times requested. So, $acceptance\_rate(v_k, b'_j)$ denotes the acceptance rate of buyer profile $b'_j \in B'$ by vendor $v_k \in V$, whereas, $acceptance\_rate(v'_i, b'_j)$ denotes the acceptance rate of buyer profile $b'_j \in B'$ by vendor profile v'_i. Intuitively, acceptance ratio $\Upsilon(b'_j)$ resembles ratio of past acceptance decisions made by vendor v_k for similar kind of buyer b_k, which aid the broker in modelling the pricing rule for the vendors (see Section 3.3)

3.2 State

In *MMDP*, a state-space represents the status of all the agents. In our setting, status is represented in the form of the supply and demand of resources and the profiles of the requesting buyer. Also, since in an OME, multiple buyers are being allocated in parallel at any instance of time or within a single episode of length t_{max}. Therefore, the joint state space is represented by concatenating the status of all the agents for every interaction with all the buyers within an episode. In this regard, the state-space is represented as $s = [H, F]$, where F represents the profile of the resource requesting buyers throughout an episode (see Section 3.1). However, vector H is obtained by concatenating the *status vector* of all the agents within an episode, wherein, *status vector* of an agent v_i is represented as $\eta(v_i) \equiv [available_{v_i}, requested_{v_i}, revenue_{v_i}, penalty_{v_i}]$; where $available_{v_i}$, $requested_{v_i}$, $revenue_{v_i}$, and $penalty_{v_i}$ represent the available resources, total requested resources by all the buyers, total revenue earned, and total penalty imposed, respectively.

3.3 Action

In the considered setting, initially, all the vendors independently set the base prices $base\_price(v_i, b_j)$ for every resource request req_{b_j} from buyer b_j and submit it to the broker. Then, the broker attempts to optimise these base prices by modelling a set of exclusive adjustment multipliers $act(v_i, b_j)$ for all the pair of potential vendors and potential buyers in the OME using Equation (6).

$$bid(v_i, b_j) = base\_price(v_i, b_j) \times (1 + act(v_i, b_j) \times \Upsilon(b_j)) \tag{6}$$

In this research,to control the minimum and maximum values, we set $act \in [-0.2, 0.8]$. Also, recall that $\Upsilon \in [0, 1]$, so, the optimal bid values for the vendors would range in the range, $bid = [base\_price \times 0.8, base\_price \times 1.8]$.

3.4 Reward

In the OME, vendors are competing among each other to maximise their revenues. In this setting, the payments from buyers are the reward for the winning vendor. However, losing vendors have to bear a negative reward (penalty). Intuitively, this penalty resembles the cost incurred on losing vendors for reserving the requested resources. In this context, vendors' utility is improved in two ways: (1) selling the resource at the maximum possible price; (2) selling the resources to more number of buyers at an optimal price. In this research, we focus on maximising the revenue by serving the maximum possible buyers; also, selective participation to avoid negative rewards. Therefore, we model a reward function based on *difference-reward* technique [31], which takes the number of times a vendor *won*, *lose,* and was *out* of auction by choice, possibly due to less available resources or lesser utility, within a single episode. In specific, the reward function for vendor v_i is represented as $r(v_i) \equiv (I_{v_i}^{win} \times win(v_i), I_{v_i}^{lose} \times lose(v_i), I_{v_i}^{out} \times out(v_i))$. Wherein, $win(v_i)$, $lose(v_i)$, and $out(v_i)$ denote the number of times vendor v_i won, lose, and was out of auction for all the auctions within a single episode, whereas, $I_{v_i}^{win}$, $I_{v_i}^{lose}$, and $I_{v_i}^{out}$ represent the impact of each of these three variables, which are private values of the vendors. Briefly, in each episode, the m potential buyers denoted as $b_1, \ldots, b_m$ lodge their requests for different bundles of resources, denoted as $req(b_1), \ldots, req(b_m)$. In turn, all the available vendor agents offer their optimised bids to all the potential buyers. In this context, soon after a certain buyer selects a winning vendor, then the reward is computed for each vendor based on the reward function. Finally, all the rewards gained within an episode are added to get an episodic reward.

4 REAL-TIME PRICING ALGORITHM

In a competitive resource market (OME), each vendor has a limited volume of resources leased/sold at the maximum possible price to maximise its revenue. In this regard, the vendor earns revenue only when it wins the auction. Considering these constraints and the dynamism of the OMEs, an optimal dynamic pricing algorithm is required. Therefore, in this research, we propose a learning-based real-time pricing algorithm that optimises the bid values of the vendors based on the changing real-time supply and demand of the resources. Besides, we propose a profiling scheme to train agents to optimise base price on behalf of vendors, based on the estimated undisclosed preferences. In this manner, the bid values for the requested bundle of resources for each vendor are optimised based on a trained neural network model using a RL algorithm. In specific, the proposed real-time pricing algorithm is implemented using the multi-agent actor-critic RL architecture [28]. The proposed real-time pricing algorithm is demonstrated in Algorithm 1, which takes the concatenated status of the set of all vendors *H*, tuples from the transaction database, and the revenue of all the vendors as input. Then, the algorithm provides the adjustment multipliers $act(v_i, b_j)$ for pair of vendors v_i and buyer b_j, $\forall v_i \in V, b_j \in B$ as output. Algorithm 1 trains each vendor agent to select and leverage a certain adjustment multiplier (action), to maximise its total expected future revenues (R). These future revenues are discounted by the factor γ per each timestep. In this regard, the future revenue at each timestep $t \in t_{max}$ for vendor v_i is denoted as $R(v_i) = \sum_{t=0}^{t_{max}} \gamma^t r_i^t$, where t_{max} is the episode length, i.e., timestep at which the bidding ends. In this work, we assume the action space to be continuous, therefore, we adopt a deterministic policy gradient [28] for learning the optimal bid values. Therefore, for the considered **multi-agent reinforcement learning (MARL)** setting, the Q function for the vendor agent i is denoted by Equation (7).

$$Q_i^{\pi}(s, act) = \mathcal{E}_{\pi,\tau}\left[\sum_{t=0}^{T} \gamma^t r_i^t | s_0 = s, act\right], \tag{7}$$

ALGORITHM 1: Real-Time Pricing (RTP)

```
1: Initialise: $Q_i(s, act(v_1, b_j), \dots, act(v_n, b_j)|\theta_i^Q)$
2: Initialise: replay memory $D$
3: Initialise: actor $\mu_i$, target actor $\mu_i^{'}$
4: Initialise: target network $Q^{'}$ with $\theta_i^{Q^{'}} \leftarrow \theta_i^Q$, $\theta_i^{\mu^{'}} \leftarrow \theta_i^{\mu}$ for each agent.
5: for episode = 1 to $e$ do                                          ▷ $e$ total episodes
6:     Initialise: $s_0$ for all the vendors
7:     for $t = 0$ to $t_{max}$ do                                    ▷ length of each episode
8:         for each buyer within $t_{max}$ do
9:             Select $act(v_i, b_j)$ using Equation (9) for all vendor $v_i$ and buyer $b_j$
10:            Compute $\Upsilon$ using Equation (5)
11:            Execute actions $a = \{act(v_1, b_j), \dots, act(v_n, b_j)\}$
12:            Record reward $r$ new state $s$'
13:            Compute reward $r(v_i, b_j)^t$, and update distribution $F$
14:            Update distribution $F(v_i, b_j)$ for each buyer and vendor pair
15:        end for
16:        Merge $r(v_i) = \sum_{i=1}^{n} \sum_{t=0}^{t_{max}} r(v_i, b_j)^t$ rewards within $t_{max}$   ▷ merge the reward of all
    vendors for each buyer
17:        Push $(s, act(v_i, b_j), r(v_i), s^{'})$ into $D$            ▷ s' is the next state
18:        $s^{'} \leftarrow s$
19:        for agent $i = 1$ to $n$ do
20:            POP mini batch $(s, act(v_1, b_j), \dots, act(v_n, b_j), r(v_i), s^{'})$ from $D$
21:            Update critic using Equation (11)
22:            Update actor using Equation (13)
23:            Update target network: $\theta^{'} \leftarrow \tau\theta + (1 - \tau)\theta$
24:        end for
25:    end for
26: end for
```

where $\pi = \{\pi_1, \dots, \pi_n\}$ is the set of joint-policies of all the vendors and $act = [act(v_1, b_j), \dots, act(v_n, b_j)]$ denotes the joint action of all the vendors for a certain buyer b_j. Further, the next state s' and the next joint action act' are computed using Bellman equation as shown in Equation (8):

$$Q_i^{\pi}(s, act) = \mathcal{E}_{r,s'}[r(s, act) + \gamma \mathcal{E}_{act' \sim \pi}[Q_i^{\pi}(s^{'}, act^{'})]. \tag{8}$$

However, the mapping function $\mu_i()$ maps each state s to action $act(v_i, b_j)$ based on Equation (9), where $\mu_i()$ is known as actor in the actor-critic architecture.

$$act(v_i, b_j) = \mu_i(s) = \mu_i([H, F]) \tag{9}$$

Further, we derive Equation (10) from Equations (8) and (9).

$$\begin{aligned} Q_i^{\mu}(s, act(v_1, b_j), \dots, act(v_n, b_j)) = \mathcal{E}_{r,s'}[r(s, act(v_1, b_j), \dots, act(v_n, b_j)) \\ + \gamma Q_i^{\mu}(s', \mu_1(s', \dots, \mu_n(s'))], \end{aligned} \tag{10}$$

where $\mu = \{\mu_1, \dots, \mu_n\}$ is the joint deterministic policy space of all the vendor agents. In this regard, the goal of the proposed algorithm becomes to learn an optimal policy for each vendor agent to attain the Nash equilibrium [20]. In addition, in this multi-agent stochastic environment, each vendor agent learns to behave optimally by learning an optimal policy of μ_i, which is also based on the optimal policies of the other co-existing agents.

Further, this equilibrium is achieved by gradually reducing the loss function $L(\theta_i^Q)$ of the critic Q_i^μ with the parameter θ_i^Q as denoted in Equations (11) and (12).

$$L\left(\theta_i^Q\right) = \mathbb{E}_{s,act,r,s'}\left[(Q_i^\mu(s, act(v_1, b_j), \dots, act(v_n, b_j)) - y)^2\right] \tag{11}$$

$$y = r_i + \gamma Q_i^{\mu'}(s', \mu_1'(s'), \dots, \mu_n'(s')) \tag{12}$$

In Equations (11) and (12), $\mu' = \{\mu_1', \dots, \mu_n'\}$ represents the set of target actors; each of these actors has a delayed parameter $\theta_i^{\mu'}$. Meanwhile, $Q_i^{\mu'}$ represents the target critic, which also has a set of delayed parameters $\theta_i^{Q'}$ for each actor, and $(s, act(v_1, b_j), \dots, act(v_n, b_j), r_i, s')$ represents the transition tuple that is pushed into a replay memory D. In this regard, each vendor's policy μ_i, with parameters θ_i^μ, is trained based on Equation (13). In the next section, we present a proposed vendor selection algorithm.

$$\nabla_{\theta_i^\mu} J \approx \mathbb{E}_s\left[\sum_w \nabla_{\theta_i^\mu}\mu_i(s)\nabla_{act_i}Q_i(s, act(v_1, b_j), \dots, act(v_n, b_j))|_{act(v_i,b_j)=\mu_i(s)}\right] \tag{13}$$

5 VENDOR ELICITATION

In this section, we present a novel vendor elicitation mechanism to determine a single winning agent for every resource request. In the proposed vendor elicitation mechanism, we employ a novel priority-based *fairness* mechanism to handle the primitive drop in bidders' participation in auctions [34]. The vendor elicitation mechanism is divided into two key steps, namely, (1) *priority labelling*: in this step, we propose a priority-based *fairness* mechanism; and (2) *resource allocation*: in this step, based on the vendors' priority labels and their bid values, we determine the winning agent and the payment paid by each buyer.

5.1 Priority Labelling

In a competitive market, vendors lose because of their non-competitive (too low) bidding strategy, which leads to a classical bidder drop problem [25]. That is, bidders (vendors) refrain from further participation when they repeatedly lose in the auctions, which further leads to vendor monopoly due to scarcity of resources. Therefore, it is crucial to handle the bidder drop problem to maintain the resource supply and demand equilibrium [53]. To handle this, we introduce a priority-based fairness mechanism, wherein, broker attaches a priority label (pr) to all the bidding vendors in the auction. This priority label pr is computed based on Equation (14) as follows:

$$pr(v_i) = (1 + \zeta(v_i))/\left(1 + \sum_{auc=1}^{o} BidRatio(v_i)_{auc}\right), \tag{14}$$

where ζ denotes the number of times a certain vendor v_j loses in the last o auctions (auc). $BidRatio(v_i)$ is the ratio of each vendor's bid to the maximum bid in a certain auction, which is calculated using Equation (15) as follows:

$$BidRatiov_i = bid(v_i, b_j)/maxBid(b_j), \tag{15}$$

where $maxBid(b_j)$ is the maximum bid that is offered for the resource request $req(b_j)$ from buyer b_j. In this regard, a priority label is attached to each of the bidding vendor agents; wherein $0 \leq pr \leq 1$, 0 being the highest priority and 1 being the lowest priority.

5.2 Resource Allocation

In this subsection, we present the computation of final allocation rule α and pricing rule ρ for the reverse-auction mechanism $G = (\alpha, \rho)$ in the OME. First, the broker computes the *bid*s for all the bidding vendors based on Equation (6) and their corresponding priority label *pr* using Equation (14). Then, based on agent's bid and its priority label, we implement a **Simple Additive Weighting (*SAW*)** [63] technique to determine the winning bidder. Implemented *SAW* technique has two phases, that is, (1) Scaling Phase and (2) Scoring Phase, discussed as follows.

5.2.1 *Scaling Phase.* In this phase, the broker normalises the bid values and priority label values for all the bidding vendors. Specifically, the normalised bid value for vendor v_i and buyer b_j, where $i \in n$ and $j \in m$ is denoted as $bid'(v_i, b_j)$. Similarly, the normalised priority label for vendor v_i is denoted as $pr'(v_i)$. Finally, the normalised bid value and priority label are calculated using Equations (16) and (17), respectively.

$$bid'(v_i, b_j) = \begin{cases} \frac{bid^{max} - bid(v_i, b_j)}{bid^{max} - bid^{min}}, & \text{if } bid^{max} - bid^{min} \neq 0. \\ 1, & \text{if } bid^{max} - bid^{min} = 0. \end{cases}, \tag{16}$$

$$pr'(v_i) = \begin{cases} \frac{pr^{max} - pr(v_i, b_j)}{pr^{max} - pr^{min}}, & \text{if } pr^{max} - pr^{min} \neq 0. \\ 1, & \text{if } pr^{max} - pr^{min} = 0. \end{cases}, \tag{17}$$

where bid^{max} and bid^{min} denote the maximum and the minimum bid among all the vendors in a certain auction, respectively. Similarly, pr^{max} and pr^{min} denote the maximum and the minimum priority label among all the vendors in a certain auction, respectively.

5.2.2 *Scoring Phase.* In this phase, the broker computes the bid score using the normalised bids and the priority labels, based on Equation (18).

$$bid\_score(v_i) = pr'(v_i) \times W^{pr}(b_j) + bid'(v_i, b_j) \times W^{bid}(b_j), \tag{18}$$

wherein, $W^{pr}(b_j)$ and $W^{bid}(b_j)$ denote the preference weight of priority label and selling price for buyer b_j, respectively. It should be noted that these preference weights are buyers' *private values* of the buyers [36], so vendors are not aware of these values. In this setting, the vendor with the minimum bid score *bid_score* is the winning vendor, denoted as v_{win}, where $v_{win} \in V$. Then, the requested resource $req(b_j)$ from buyer b_j is allocated to vendor v_{win}. Also, to observe truthfulness in mechanism G, similar to Reference [18], payment rule is based on generalised second price mechanism, such that, $\rho(v_{win}, b_j) \equiv bid(v_{min}, b_j)$, where v_{min} denotes the vendor with second lowest bid value, where $v_{min} \in V$ but $v_{min} \neq v_{win}$. Algorithm 2 represents the pseudo-code of the proposed vendor selection algorithm, which takes the optimised bid values of the vendors and the preference weights of the buyers' preferences as input. Finally, gives the allocation $\alpha(v_{win}, b_j)$ and the payment $\rho(v_{win}, \rho)$ rule as output.

Towards this end, based on the proposed **real-time resource-allocation (*RTRA*)** algorithm, the broker dynamically allocates the resource requests to the potential vendors in OMEs, as shown in Algorithm 3, which dynamically takes resource requests from B buyers as input in OME initialised with a set of potential vendors and their respective base prices. Then, computes the selling price and priority labels for all the available potential vendors in the OME or wait until the vendor is available. Finally, based on the computed selling price and the priority labels, the allocation rules $M(v_{win})$ and payment rule $P(v_{win})$ are given as output. In specific, the resource is allocated to the winning vendor and the second-lowest selling price is received by the winning vendor. In the next section, we present the results of the extensive experiments that were conducted to evaluate the proposed resource allocation approach in a simulated OME.

ALGORITHM 2: Vendor Selection Algorithm

```
Input: $\{bid(v_i, b_j)\}$, where $v_i \in V$, $b_j \in B$
Output: $\alpha(v_{win}, b_j)$, $\rho(v_{win}, b_j)$
for GET buyer do                                    ▷ generate random buyers from pool of buyers
    for vendor = 1 to $n$ do                        ▷ list of participating vendors
        Calculate $BidRatio^{v_n}$                  ▷ using Equation (15)
        Calculate $pr^{v_n}$                        ▷ using Equation (14)
        Calculate $bid'(v_n, b_m)$                  ▷ using Equation (16)
        Calculate $pr'(v_n)$                        ▷ using Equation (17)
    end for
    Calculate $bid\_score(v_n, b_m)$                ▷ using Equation (18)
    $\alpha(v_{win}, b_m) = min(bid\_score(v_n, b_m))$
    $\rho(v_{win}, b_m) = min(bid_{v_i, b_m})$, where $i \in N$ but $v_i \neq v_{win}$
end for
```

ALGORITHM 3: Real-Time Resource-Allocation (RTRA)

```
Initialise: $V = \{v_1, \dots, v_n\}$               ▷ potential vendors
Initialise: $C = \{C_{v_1}, \dots, C_{v_n}\}$       ▷ available resources
Initialise: $\{base\_price_1, \dots, base\_price_n\}$
Initialise: $L_B$, $L_W$ $L_V$  ▷ queue of waiting buyers, queue of buyers waiting for vendors, & list of participating vendors, respectively
Initialise: $T_{max}$                               ▷ maximum timestep in one episode
for t = 1 to $T_{max}$ do
    Push all the requesting buyers in $L_B$         ▷ set of buyer(s) are arriving randomly at each timestep
    /* Update List of Participating Vendors */
    if participating                                ▷ check for participating vendors
        Push set of participating vendor(s) in $L_V$
    else
        penalty on non-participating vendor(s)
    /* First Allocate the Waiting Buyers */
    if $L_W$ not empty
        POP a single buyer $b_i$ from $L_W$
    else
        POP a single buyer $b_i$ from $L_B$
    if not CANSCHEDULE($b_i$)   ▷ CANSHEDULE() checks if there exists a vendor $v_j \in L_V$, s.t. $C_{v_j} \geq req(b_i)$
        if $d(b_i) \leq t$
            PUSH $b_i$ to $L_W$
    else
        Compute $bid(v_j, b_i)$ $\forall v_j \in L_V$ using Algorithm 1
        Compute allocation and payment using Algorithm 2
end for
```

Table 1. Types of Cloud Vendors

# vendor-type	*CPU (MIPS)*	*RAM (MBs)*	*Storage (MBs)*
A	50	250	2,500
B	100	500	5,000
C	200	1,000	10,000
D	400	2,000	20,000

6 EXPERIMENTAL SETUP AND RESULTS

This section presents the results of the extensive simulation experiments performed to investigate the performance of the novel *RTRA* algorithm. Throughout the experiment, we use the following hyper-parameters for *RTRA*: (discount factor)$\lambda = 0.9$; *learning-rate*$= 3e^{-4}$; as these hyper-parameters gave good results. Towards this end, we compare the novel *RTRA* algorithm with the following benchmarks:

- The **combinatorial double auction resource allocation approach (*CDARA*)** [41]—This approach implements a fixed pricing strategy, wherein the vendor with the lowest bid is the winner.
- The **indicator-based combinatorial auction-based approach (ICAA)** [24]—This approach implements a demand-based dynamic pricing strategy. Also, the vendor with the lowest bid is the winner.

6.1 Experimental Setup

In the performed simulation experiment, we target to build an open-market cloud environment with multiple cloud vendors. In this context, we considered a pool of cloud-vendors sampled among four types of cloud vendors, characterised based on their total available resources, as listed in Table 1.

In addition, similar to Reference [41], we set the base prices (*base_price*) for one unit of cloud resources based on sampling the price from preset range of values. In particular, we set $[10, 20]$\$, $[1, 5]$\$, and $[5, 10]$\$ for *CPU*, *RAM*, and *Disk Storage*, respectively. Further, we extract the resource request of the buyers from the Google Cluster Trace (i.e., Task Events Tables) [57], in which the requested resource (*CPU*, *RAM*, *Disk Storage*) quantities are re-scaled values as discussed in Reference [57]. Therefore, we considered the maximum re-scaled value as one unit for each resource type and then scaled all the requested units by it. Also, since the values of *arrival-time*, *execution-length*, and *deadline* for each resource request is not publicly available in the dataset (i.e., *Task Events Tables*). Therefore, similar to Reference [64], we simulate the *arrival-time*, *execution-length,* and *deadline* using three random generators that takes values $[1, 24]$ *timesteps*, $[0, 12]$ *timesteps,* and $[1, 12]$ *timesteps*, respectively.

In this setting, we train the *RTRA* algorithm for $10,000$ training episodes, each of length $T_{max} = 2,000$ timesteps. Then, we evaluate the performance of the *RTRA* algorithm in *four* different experimental settings concerning the number of available vendors, i.e., with *four*, *six*, *eight,* and *twelve* cloud-vendors. For all the above four experimental settings, we evaluate the performance based on average performance in 150 cycles of $1,000$ episodes each. Finally, all the algorithms are implemented in *Python 3*-based framework *SimPy* [32], and the experiments are performed on *Intel Xeon 3.6 GHz* six-core processor with *32 GB RAM.*

As the objective of this research is to not only enhance the performance of the vendors but also safeguard the preference and utility of the buyers, therefore, we evaluate the performance of the novel *RTRA* algorithm based on the performance of vendors and buyers, as discussed in the next subsections.

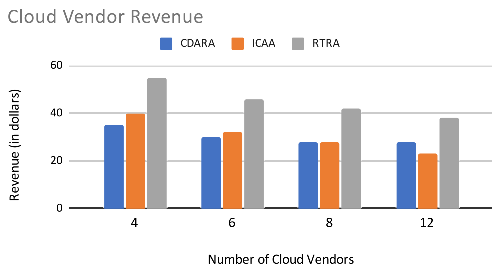

Fig. 2. Cloud vendor revenue.

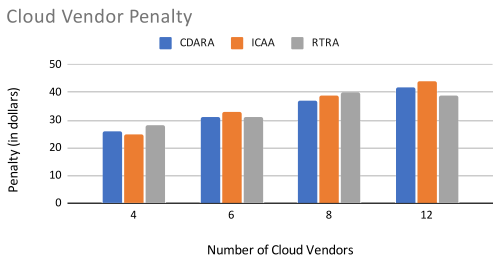

Fig. 3. Cloud vendor penalty.

6.2 Evaluation Based on the Performance of Vendors

In our experimental setting, the performance of the cloud-vendors are measured based on three parameters as follows: (1) *average revenue*: which is the average of all the revenues earned by each vendor during the *150* evaluation cycles, (2) *average penalty*: which is the average of all the penalties imposed on each vendor during the *150* evaluation cycles; (3) *fairness*: which is the ratio of the number of never-won vendors to the total number of vendors. This ratio indicates the active participation of vendors in the auction, and its lower value depicts the existence of the bidder drop issue; (4) the *participation rate* of vendors: which is the total number of buyers to whom the vendor has offered its resources; (5) the *Non-participation rate* of vendors: which is the total number of buyers to whom the vendor could not offer its resources because of not having enough available resources; (6) the *losing rate* of vendors: which is the total number of buyers to whom vendor has offered its resources but lose in the auction.

From Figure 2, it can be observed that when the number of cloud vendors increases, the revenue in all algorithms decreases simultaneously. This is observed because revenue is divided among the cloud vendors. Overall, it can be observed that, for all four cases, revenue is maximum in the *RTRA* algorithm. In particular, when cloud vendors $n = 12$, revenue in *RTRA* algorithm is twice as compared to revenue in *ICAA* algorithm. An interesting observation here is, although in *CDARA* and *ICAA* the prices are fixed, revenue for both the algorithms are different in all the cases, except when $n = 8$. This seems to suggest that allocation rule plays an important rule in such a dynamic environment.

However, Figure 3 shows the average penalty imposed on cloud vendors. It can be observed that, with the increase in cloud vendors, competitiveness increases, so the penalty also increases. In all three algorithms, the difference in their average penalty does not have a significant change.

Further, we evaluate the *fairness* in all the algorithms with the change in the number of cloud vendors. From Figure 4, it is evident that *fairness ratio* is highest for *RTRA* algorithm in all the four cases. This suggests that the available resources in *RTRA* algorithm are allocated optimally

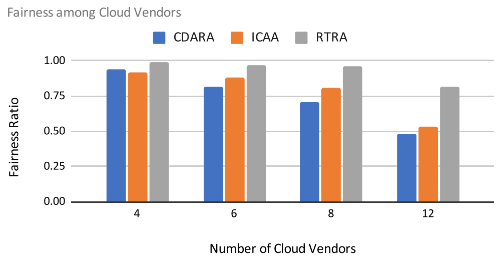

Fig. 4. Fairness among cloud vendors.

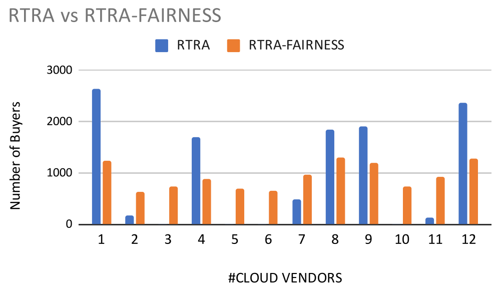

Fig. 5. Rate of participation in *RTRA* vs. *RTRA-Fairness*.

by giving non-partisan chance to all the cloud vendors, thus increases the availability of resources with the cloud vendors. Also, with the increase in the number of cloud vendors, *fairness ratio* in *RTRA* increases more rapidly as compared to the other two benchmarks. In particular, for $n = 12$, the *fairness ratio* value is significantly (*60%*) higher as compared to the other two algorithms.

To provide one more evidence of a reduction in bidder drop problem with the inclusion of priority based *fairness* mechanism, we compare the participation rate of *12* vendors in *RTRA* algorithm with and without fairness mechanism, wherein all other experimental settings were the same. Also, in both scenarios, all the vendors had the same resource capacity and resource base prices. From Figure 5, it is evident that in *RTRA* without fairness, vendors 1, 2, 4, 7, 8, 9, 11, 12 are the participating and others are not participating. Out of which, vendors 2, 7, 11 have the minimum participation rate. However, in *RTRA-FAIRNESS* with fairness mechanism, vendors 3, 5, 6, 10 are also starting to participate, which were not in the previous scenario. Also, the participation rate of vendors 2, 7, 11 are improved as compared to the previous scenario.

Furthermore, to provide an evidence of optimal resource utilisation, we would observe the *participation/non-participation* of cloud vendors in all the algorithms, wherein $n = 12$. From Figures 6 and 7, it can be observed that *participation/non-participation* of the vendors are fluctuating in different algorithm. However, the behaviour of any particular vendor is observed similarly in all three algorithms. For instance, vendors *2* and *3* are the least participating and maximum non-participation, whereas participation of vendors *1*, *8*, and *12* are highly participating (least non-participation) in all the three algorithms. This is possible because certain vendors have higher available resources as compared to other vendors. Overall, the participation of cloud vendors in *RTRA* algorithm is boosted as compared to the benchmarks, which again reflects the optimal utilisation of the available resources.

Finally, from Figure 8, cloud vendors in *RTRA* have a lower rate of losing in the auction as compared to other algorithms. This suggests that *RTRA* effectively models the allocation rule and

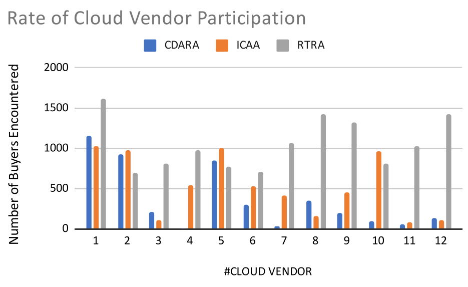

Fig. 6. Rate of participation.

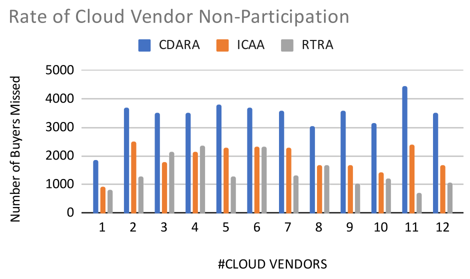

Fig. 7. Rate of non-participation.

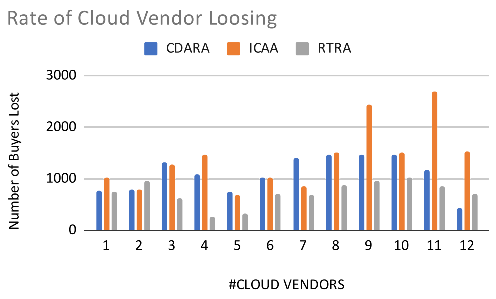

Fig. 8. Rate of losing.

pricing rule for the cloud vendors, such that cloud vendors selectively participate in the auction with an optimal bid that increases their chances of winning. Also, an interesting observation is cloud vendors *9* and *11*, which have worst performance in *ICAA* algorithm, brilliantly perform in *RTRA* algorithm. Therefore, from the above results, it becomes clear that the proposed *RTRA* approach is capable of boosting the performance of cloud vendors in the open market cloud environment.

6.3 Evaluation Based on the Performance of Buyers

In this subsection, we evaluate the performance of all potential buyers based on three parameters, namely, (1) **paid to max-bid (*PMB*)**: which is the average ratio of the prices that are paid by the buyers for their requested resources to the maximum bidding price. This parameter denotes the margin between resource cost that buyer bears with resource allocation mechanism to the cost that buyer would have paid without any mechanism. Since, in this research, we consider a free market, wherein buyers do not have any resource valuation, buyers focus on buying the resources at the minimum possible price in the considered market. In this context, the utility of the buyer cannot be directly computed, so it is computed based on this ratio, which intuitively resembles the price paid with the mechanism to price paid without any mechanism. This parameter is significant, especially for *RTRA* algorithm, where the final selling prices (i.e., bids) are optimised by the broker. Also, in our experimental setting, the base price of the resources for every vendor are sampled from the same range of values in all three algorithms. Therefore, vendors cannot manipulate their final selling prices (i.e., bids) and it purely depends on RL-based Equation (6); (2) average waiting time: which is the average time all the potential buyers wait before their resource request is allocated to one of the potential vendors; and (3) Success Rate: which is the ratio of the average number of buyer's requests which were finished to rejected during the *150* evaluation

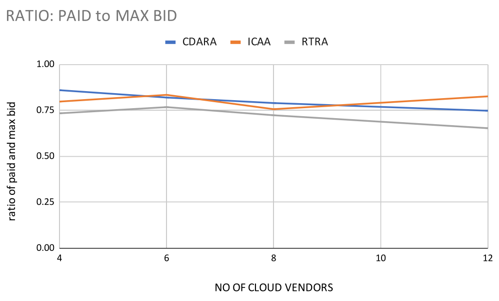

Fig. 9. Paid per max bid ratio.

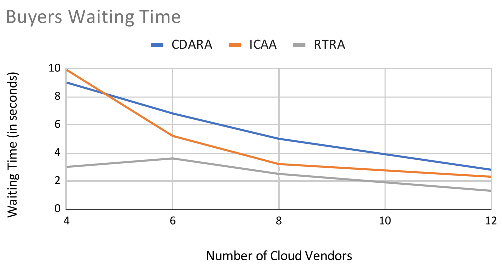

Fig. 10. Average waiting time.

cycles. Note that a higher success rate does not show the effectiveness of the mechanisms, it only denotes the reliability of the OME.

From Figure 9, it becomes evident that the potential buyers that use the *RTRA* approach pay marginally lesser prices for the requested resources as compared to the maximum offered prices by the bidding vendors. Although there is very little change in ratio with the change in the number of cloud vendors, the ratio decreases with the increase in the number of cloud vendors, except in the *ICAA*, wherein the sinusoidal-like pattern is observed. Overall, this highlights that price paid by the buyers is not affected in the *RTRA* algorithm, since the base price was the same for all the algorithms.

On the contrary, from Figure 10, the waiting time for the potential providers decreases with the increase in the number of cloud vendors. Overall, it is visible from Figure 10 that waiting time is least in the *RTRA* algorithm as compared to other algorithms.

Finally, from Figure 11, the ratio rejection rate decreases with an increase in the number of cloud vendors. This suggests that the gap between the finished and rejected buyers decreases with the increase in the number of cloud vendors. However, it is interesting to observe that, at $n = 8$, the ratio reaches its lowest level and hereafter becomes steady. This suggests the gap between the finished and rejected remains steady and does not fall below 0.1, and there is no effect of change in the number of cloud vendors. In this context, except for the threshold point, the ratio remains higher in the *RTRA* algorithm as compared to the other two algorithms. This demonstrates the capability of the *RTRA* approach to achieve a higher success rate, thus higher resource reliability in the OME.

Thus, we also showed that the proposed approach is capable of boosting the performance of buyers in an OME. However, from the experimental results, it is observed that in the *RTRA* algorithm, the revenue of the vendors is higher, so one might conclude that buyers in *RTRA* algorithm have to bear the extra cost. However, this is not fully true, as the success rate in the *RTRA* algorithm is also higher, i.e., the number of allocated buyers is more, which would

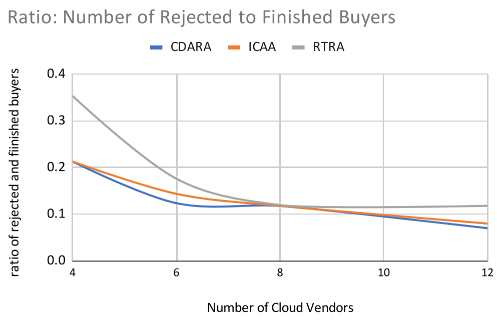

Fig. 11. Rejection rate: Ratio of buyers' resource request rejected to finished.

have also led to higher revenue. In the end, even if prices in *RTRA* algorithm are proved to be marginally higher as compared to other algorithms, it benefited with lower waiting time and higher reliability, as compared to other algorithms. Because maximising the revenue for vendors along with minimising the cost for buyers at the same time is not purely practical. Therefore, in the *RTRA* algorithm, we showed that price paid by the buyers are the best price within a particular setting, which was reflected using *PMB* ratio. Overall, it highlights that the proposed approach demonstrates its ability to address both the real-time pricing and fair vendor elicitation problems while boosting the overall performance of the *open market cloud environment.*

7 RELATED WORK

In the past, a mechanism based on different techniques, such as game theory [3, 56], stochastic programming [12], genetic algorithm [46], and so on, have been proposed to address the challenges related to optimal resource allocation. However, this research utilises an auction mechanism to allocate a bundle of requested resources in OMEs. In specific, we deploy a combinatorial auction paradigm, as we focus on multiple types of resource allocation, i.e., instances of bandwidth, CPU time and memory space, and so on. In such an auction paradigm, the focus is to model optimal allocation policy and payment policy to maximise social welfare. In specific, we adopt first-come-first-serve allocation policy and optimal pricing policy for the bidding vendors. In designing both of the policies, the primary objective is to ensure social-welfare in the market.

In 1981, *Myserson* [36], in its nominal work, put forward the necessary conditions for designing an optimal auction paradigm. In this context, different properties associated with an optimal auction paradigm are Incentive-compatibility, competitiveness, fairness, and so on. Later, based on the nominal work, different class auction paradigms were studied, namely, generalised second-price auction [4], *Vickrey* auction [47], double auction [49] for combinatorial auction [19]. These different classes of auction paradigms were used in different types of OME, not limited to wireless networks [8, 17, 35], energy grids [10, 23], advertisements [9, 30], cloud markets [2, 33], vehicular network [27], and so on. However, in the existing approaches, all the challenges in optimal allocation were not addressed. For instance, Bo et al. [2], proposed a decentralised approach for multi-resource allocation in a market-based environment. However, in this research, buyers were expected to buy each resource separately in sequence of auctions instead of single auction.

Further, Schwind et al. [44] proposed a combinatorial auction-based mechanism for simultaneous resource allocation in grids. This mechanism improved the resource utilisation of the grid, but at the cost of providers' utility. Then, Teo et al. [51] proposed *VCG*based mechanism. In this mechanism, participants' dominant strategy was to reveal the true valuation. Therefore, this mechanism was stable and efficient. However, later, *Rothkopf* [40] listed the proofs to show that *VCG* mechanism was practically impossible to implement without any strict assumptions. In this regard, recently, in 2016, Prasad et al. [38] proposed combinatorial auction-based resource allocation approaches in cloud computing. These approaches focused on maximising the utilities of both the participants. However, these approaches were based on a fixed pricing policy. Therefore, it led to lesser revenue for the vendors when there are fewer supplies and higher demands in the market. Consequently, adapting the pricing-rule based on the dynamics of supply and demand is very crucial and also a challenging task [17, 33]. In this regard, Pourebrahimi et al. [37] proposed a continuous double auction mechanism for resource allocation in the grid. This approach was adopted by stock markets worldwide, and it is a very popular mechanism. However, it could only handle a single resource request instead of multiple resources; also, later, Reference [41] found that it has lower resource utilisation.

Besides, fixed-pricing leads to less competition because of monopoly [26] of a set of bidding vendors, which has the lowest possible prices, as those vendors keep winning in consecutive auctions. Therefore, to maintain competitiveness [29] through avoiding monopoly in the environment, vendors should be able to adjust their prices based on the market situation. In the past, dynamic pricing policies were studied using different techniques. For instance, statistical models based on dynamic policies [24, 26, 54]. However, those statistical models adapt to static or gradually changing environments, thus fail to adapt to the overall dynamics of supply and demand in the OMEs. Later, to address these issues, Schoenig et al. [43] proposed Single-Item auctions, wherein the vendor is not aware of all the upcoming future requests. However, this approach focused only on the dynamics of the buyers (i.e., demand) and did not address the dynamics of vendors (i.e., demand).

Furthermore, to handle the dynamism of OMEs, machine learning techniques [5, 13, 15] were implemented for resource provisioning within a single business market. In this context, a machine learning-based non-linear approximation function was used as a pricing function to maximise the social welfare of the vendors. As those techniques were vendor-centric and failed to address different buyers-centric preferences, such as resource quality, completion deadline, and so on. Also, all those approaches model an optimal approximation function, purely based on the past data, and considered important game-theoretical parameters for designing an optimal auction mechanism [36], such as buyers' *quality uncertainty*,[5] i.e., their *private values*, is used in determining the appropriate vendor for a certain buyer. Besides, the selling price of resources depends on the overall supply and demand in the market. However, the vendor is not aware of the resource availability with the other vendors in the market. In this regard, knowledge of buyers' preferences and the total estimated supply in the market plays a key role is real-time pricing and vendor selection. Therefore, learning techniques, especially RL-based [16, 45, 50], attempted to estimate and include these parameters. Besides, RL-based model-free algorithms give similar performance as a model-based framework and are easily scalable [52]. However, existing RL-based approaches only considered the dynamics of the buyers and assumed an existence of constant supply of resources.

Similar to the real-time problem discussed in this research, *display advertisement* domain is also time-sensitive. In this domain, designing the price for dynamically generated advertisements is a challenging task. In this context, Yuan et al. [61] and Zhang et al. [65] proposed a real-time bid optimisation approach based on linear programming. However, those approaches could not handle the real-time complexity of the pricing policy. Lately, to handle this real-time complexity, many RL-based approaches have been proposed for display advertisement [11, 22, 58]. Therefore, RL techniques proved to be suitable for such real-time problems. However, those models considered a single-auction paradigm with a single vendor and dynamically arriving bidders. However, in this research, we focus on markets with dynamically arriving and leaving vendors and buyers. Apart

[5]Preference of a buyer for a particular vendor over other vendors.

from the above challenges, a mechanism should observe fairness in resource allocation. In specific, the mechanism should give a fair chance to all the participants, especially the bidders, to improve the social welfare in the environment [6] and also the resource availability in the market [39]. In this regard, Jiang et al. [21] proposed a Nash equilibrium-based cooperative mechanism, which focused on enhancing the overall resource utilisation. However, using this approach utility of individual vendors was not improved. Apart from this, the inclusion of fairness in the mechanism is essential to handle the bidder drop problem. Bidder drop problem [7] in the market, thus again leads to monopoly.

To sum up, all the existing pricing policies are based on either static mathematical models or inefficient learning techniques, thus fail to address all the challenges associated with designing an optimal pricing policy for OMEs. To handle the complexity and dynamism for the resource allocation problem in OMEs, in this work, we propose a reinforcement-learning-based pricing mechanism enabled with *SAW* [1]-based fairness strategy for fair real-time resource allocation problems.

8 CONCLUSION

This article proposes a novel real-time fair resource allocation learning approach in OMEs. In this approach, on behalf of bidding vendors, the broker optimises the selling price based on the real-time pricing algorithm. This real-time pricing algorithm is built using the RL technique. Also, we introduce a profiling scheme that models the overall uncertain supply and demand in the market based on past allocations. Also, to address the bidder drop problem in the market, we deployed a priority-based fairness mechanism. Towards this end, based on experimental results, we demonstrate the performance boost of all the participants in the novel *RTRA* mechanism.

As our future work, we would like to focus on some issues left that have not yet been addressed in this research. In this context, the future direction is as follows: (1) Scalability: investigating the performance of the real-time pricing algorithm and impact on learning model (i.e., convergence) with the increase in the number of participants; (2) Selective Bidding: considering different social and quality preferences of the vendors while bidding for the requesting buyers; (3) Buyers' Social Welfare: investigate the effect of the pricing scheme on the budget of the buyers, as in this research, we assume that buyers do not have any budget constraints; and (4) Coalition: it would be interesting to encourage cooperation among the competing participants. For example, vendors share their resources or buyers combine their resource request and maximise their overall utility.

ACKNOWLEDGEMENT

The authors would like to thank two anonymous reviewers for their insightful suggestions and careful reading of the manuscript.

REFERENCES

[1] Alireza Afshari, Majid Mojahed, and Rosnah Mohd Yusuff. 2010. Simple additive weighting approach to personnel selection problem. *Int. J. Innov., Manag. Technol.* 1, 5 (2010), 511.

[2] Bo An, Victor Lesser, and Kwang Mong Sim. 2011. Strategic agents for multi-resource negotiation. *Auton. Agents Multi-Agent Syst.* 23, 1 (2011), 114–153.

[3] Danilo Ardagna, Barbara Panicucci, and Mauro Passacantando. 2011. A game theoretic formulation of the service provisioning problem in cloud systems. In *Proceedings of the 20th International Conference on World Wide Web.* 177–186.

[4] Junjik Bae, Eyal Beigman, Randall A. Berry, Michael L. Honig, and Rakesh Vohra. 2008. Sequential bandwidth and power auctions for distributed spectrum sharing. *IEEE J. Select. Areas. Commun.* 26, 7 (2008), 1193–1203.

[5] Gaurav Baranwal, Dinesh Kumar, Zahid Raza, and Deo Prakash Vidyarthi. 2018. A negotiation based dynamic pricing heuristic in cloud computing. *Int. J. Grid Util. Comput.* 9, 1 (2018), 83–96.

[6] Gaurav Baranwal and Deo Prakash Vidyarthi. 2015. A fair multi-attribute combinatorial double auction model for resource allocation in cloud computing. *J. Syst. Softw.* 108 (2015), 60–76.

[7] Gaurav Baranwal and Deo Prakash Vidyarthi. 2019. A truthful and fair multi-attribute combinatorial reverse auction for resource procurement in cloud computing. *IEEE Trans. Serv. Comput.* 12, 06 (2019), 851–864.

[8] Randall Berry, Michael L. Honig, and Rakesh Vohra. 2010. Spectrum markets: Motivation, challenges, and implications. *IEEE Commun. Mag.* 48, 11 (2010), 146–155.

[9] Anand Bhalgat, Jon Feldman, and Vahab Mirrokni. 2012. Online allocation of display ads with smooth delivery. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1213–1221.

[10] Markus Burger, Bernhard Klar, Alfred Müller, and Gero Schindlmayr. 2004. A spot market model for pricing derivatives in electricity markets. *Quantit. Finan.* 4 (2004), 109–122.

[11] Han Cai, Kan Ren, Weinan Zhang, Kleanthis Malialis, Jun Wang, Yong Yu, and Defeng Guo. 2017. Real-time bidding by reinforcement learning in display advertising. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*. ACM, 661–670.

[12] Sivadon Chaisiri, Rakpong Kaewpuang, Bu-Sung Lee, and Dusit Niyato. 2011. Cost minimization for provisioning virtual servers in Amazon elastic compute cloud. In *Proceedings of the IEEE 19th Annual International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*. IEEE, 85–95.

[13] Mingxi Cheng, Ji Li, and Shahin Nazarian. 2018. DRL-cloud: Deep reinforcement learning-based resource provisioning and task scheduling for cloud service providers. In *Proceedings of the 23rd Asia and South Pacific Design Automation Conference (ASP-DAC'18)*. IEEE, 129–134.

[14] Wolfram Conen and Tuomas Sandholm. 2002. Partial-revelation VCG mechanism for combinatorial auctions. In *Proceedings of the AAAI Conference on Artificial Intelligence and Innovative Applications of Artificial Intelligence Conference*. 367–372.

[15] Bingqian Du, Chuan Wu, and Zhiyi Huang. 2019. Learning resource allocation and pricing for cloud profit maximization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 7570–7577.

[16] Paul Dütting, Zhe Feng, Harikrishna Narasimhan, David C. Parkes, and Sai Srivatsa Ravindranath. 2017. Optimal auctions through deep learning. *arXiv preprint arXiv:1706.03459* (2017).

[17] Neda Edalat, Wendong Xiao, Nirmalya Roy, Sajal K. Das, and Mehul Motani. 2011. Combinatorial auction-based task allocation in multi-application wireless sensor networks. In *Proceedings of the IFIP 9th International Conference on Embedded and Ubiquitous Computing*. IEEE, 174–181.

[18] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. 2007. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *Amer. Econ. Rev.* 97, 1 (2007), 242–259.

[19] Ikki Fujiwara, Kento Aida, and Isao Ono. 2010. Applying double-sided combinational auctions to resource allocation in cloud computing. In *Proceedings of the 10th IEEE/IPSJ International Symposium on Applications and the Internet*. IEEE, 7–14.

[20] Junling Hu, Michael P. Wellman, et al. 1998. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proceedings of the International Conference on Machine Learning*, Vol. 98. Citeseer, 242–250.

[21] Congfeng Jiang, Liangcheng Duan, Chunlei Liu, Jian Wan, and Li Zhou. 2013. VRAA: Virtualized resource auction and allocation based on incentive and penalty. *Cluster Comput.* 16, 4 (2013), 639–650.

[22] Junqi Jin, Chengru Song, Han Li, Kun Gai, Jun Wang, and Weinan Zhang. 2018. Real-time bidding with multi-agent reinforcement learning in display advertising. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2193–2201.

[23] Ashkan R. Kian and Jose B. Cruz Jr. 2005. Bidding strategies in dynamic electricity markets. *Decis. Supp. Syst.* 40, 3–4 (2005), 543–551.

[24] Yan Kong, Minjie Zhang, and Dayong Ye. 2015. An auction-based approach for group task allocation in an open network environment. *Comput. J.* 59, 3 (2015), 403–422.

[25] Dinesh Kumar, Gaurav Baranwal, Zahid Raza, and Deo Prakash Vidyarthi. 2019. Fair mechanisms for combinatorial reverse auction-based cloud market. In *Information and Communication Technology for Intelligent Systems*. Springer, 267–277.

[26] Xuejun Li, Ruimiao Ding, Xiao Liu, Xiangjun Liu, Erzhou Zhu, and Yunxiang Zhong. 2016. A dynamic pricing reverse auction-based resource allocation mechanism in cloud workflow systems. *Scient. Program.* 2016, Article ID 7609460, 13 pages. https://doi.org/10.1155/2016/7609460

[27] Minghui Liwang, Shijie Dai, Zhibin Gao, Yuliang Tang, and Huaiyu Dai. 2018. A truthful reverse-auction mechanism for computation offloading in cloud-enabled vehicular network. *IEEE Internet Things J.* 6, 3 (2018), 4214–4227.

[28] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275* (2017).

[29] Mario Macías and Jordi Guitart. 2011. A genetic model for pricing in cloud computing markets. In *Proceedings of the ACM Symposium on Applied Computing*. 113–118.

[30] Mohammad Mahdian, Hamid Nazerzadeh, and Amin Saberi. 2007. Allocating online advertisement space with unreliable estimates. In *Proceedings of the 8th ACM Conference on Electronic Commerce*. 288–294.

[31] Kleanthis Malialis, Sam Devlin, and Daniel Kudenko. 2019. Resource abstraction for reinforcement learning in multiagent congestion problems. *arXiv preprint arXiv:1903.05431* (2019).

[32] Norm Matloff. 2008. Introduction to discrete-event simulation and the simpy language. *Davis, CA. Dept of Computer Science. University of California at Davis. Retrieved on August* 2, 2009 (2008), 1–33.

[33] Pankaj Mishra, Ahmed Moustafa, Takayuki Ito, and Minjie Zhang. 2020. Optimal auction based automated negotiation in realistic decentralised market environments. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 13726–13727.

[34] Javier Murillo, Víctor Muñoz, Beatriz López, and Dídac Busquets. 2008. A fair mechanism for recurrent multi-unit auctions. In *Proceedings of the German Conference on Multiagent System Technologies*. Springer, 147–158.

[35] Huseyin Mutlu, Murat Alanyali, and David Starobinski. 2009. Spot pricing of secondary spectrum access in wireless cellular networks. *IEEE/ACM Trans. Netw.* 17, 6 (2009), 1794–1804.

[36] Roger B. Myerson. 1981. Optimal auction design. *Math. Operat. Res.* 6, 1 (1981), 58–73.

[37] Behnaz Pourebrahimi, Koen Bertels, G. M. Kandru, and Stamatis Vassiliadis. 2006. Market-based resource allocation in grids. In *Proceedings of the 2nd IEEE International Conference on e-Science and Grid Computing (e-Science'06)*. IEEE, 80–80.

[38] G. Vinu Prasad, Abhinandan S. Prasad, and Shrisha Rao. 2016. A combinatorial auction mechanism for multiple resource procurement in cloud computing. *IEEE Trans. Cloud Comput.* 6, 4 (2016), 904–914.

[39] Ricardo J. Rabelo, Luis M. Camarinha-Matos, and Hamideh Afsarmanesh. 1998. Multiagent perspectives to agile scheduling. In *Proceedings of the International Conference on Information Technology for Balanced Automation Systems*. Springer, 51–66.

[40] Michael H. Rothkopf. 2007. Thirteen reasons why the Vickrey-Clarke-Groves process is not practical. *Operat. Res.* 55, 2 (2007), 191–197.

[41] Parnia Samimi, Youness Teimouri, and Muriati Mukhtar. 2016. A combinatorial double auction resource allocation model in cloud computing. *Inf. Sci.* 357 (2016), 201–216.

[42] Tuomas W. Sandholm. 1996. Limitations of the Vickrey auction in computational multiagent systems. In *Proceedings of the 2nd International Conference on Multiagent Systems (ICMAS'96)*. 299–306.

[43] Adrian Schoenig and Maurice Pagnucco. 2010. Evaluating sequential single-item auctions for dynamic task allocation. In *Proceedings of the Australasian Joint Conference on Artificial Intelligence*. Springer, 506–515.

[44] Michael Schwind, Oleg Gujo, and Tim Stockheim. 2006. Dynamic resource prices in a combinatorial grid system. In *Proceedings of the 8th IEEE International Conference on E-Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)*. IEEE, 49–49.

[45] Weiran Shen, Binghui Peng, Hanpeng Liu, Michael Zhang, Ruohan Qian, Yan Hong, Zhi Guo, Zongyao Ding, Pengjun Lu, and Pingzhong Tang. 2017. Reinforcement mechanism design, with applications to dynamic pricing in sponsored search auctions. *arXiv preprint arXiv:1711.10279* (2017).

[46] Gurmeet Singh, Carl Kesselman, and Ewa Deelman. 2006. Application-level resource provisioning on the grid. In *Proceedings of the 2nd IEEE International Conference on e-Science and Grid Computing (e-Science'06)*. IEEE, 83–83.

[47] Igor Stanojev, Osvaldo Simeone, Umberto Spagnolini, Yeheskel Bar-Ness, and Raymond L. Pickholtz. 2010. Cooperative ARQ via auction-based spectrum leasing. *IEEE Trans. Commun.* 58, 6 (2010), 1843–1856.

[48] Richard S. Sutton, Andrew G. Barto, et al. 1998. *Introduction to Reinforcement Learning*. Vol. 2. MIT Press, Cambridge, MA.

[49] Zhu Tan and John R. Gurd. 2007. Market-based grid resource allocation using a stable continuous double auction. In *Proceedings of the 8th IEEE/ACM International Conference on Grid Computing*. IEEE, 283–290.

[50] Pingzhong Tang. 2017. Reinforcement mechanism design. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 5146–5150.

[51] Yong Meng Teo and Marian Mihailescu. 2009. A strategy-proof pricing scheme for multiple resource type allocations. In *Proceedings of the International Conference on Parallel Processing*. IEEE, 172–179.

[52] Gerald Tesauro et al. 2005. Online resource allocation using decompositional reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 5. 886–891.

[53] Adel Nadjaran Toosi, Kurt Vanmechelen, Farzad Khodadadi, and Rajkumar Buyya. 2016. An auction mechanism for cloud spot markets. *ACM Trans. Auton. Adapt. Syst.* 11, 1 (2016), 2.

[54] Hongyi Wang, Qingfeng Jing, Rishan Chen, Bingsheng He, Zhengping Qian, and Lidong Zhou. 2010. Distributed systems meet economics: Pricing in the cloud. In *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*. 6–6.

[55] Jian Wang, Huimin Miao, and Mingzhu Yu. 2019. Interdependent order allocation in the two-echelon competitive and cooperative supply chain. *Int. J. Product. Res.* 57, 4 (2019), 1190–1213.

[56] Guiyi Wei, Athanasios V. Vasilakos, Yao Zheng, and Naixue Xiong. 2010. A game-theoretic method of fair resource allocation for cloud computing services. *J. Supercomput.* 54, 2 (2010), 252–269.

[57] John Wilkes. 2011. More Google Cluster Data. Google research blog. Retrieved from: http://googleresearch.blogspot.com/2011/11/more-google-cluster-data.html.

[58] Di Wu, Xiujun Chen, Xun Yang, Hao Wang, Qing Tan, Xiaoxun Zhang, Jian Xu, and Kun Gai. 2018. Budget constrained bidding by model-free reinforcement learning in display advertising. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management.* ACM, 1443–1451.

[59] Bo Yang, Zhiyong Li, Shilong Jiang, and Keqin Li. 2018. Envy-free auction mechanism for VM pricing and allocation in clouds. *Fut. Gen. Comput. Syst.* 86 (2018), 680–693.

[60] Samuel Yousefi, Mustafa Jahangoshai Rezaee, and Maghsud Solimanpur. 2021. Supplier selection and order allocation using two-stage hybrid supply chain model and game-based order price. *Oper Res Int J* 21 (2021), 553–588. https://doi.org/10.1007/s12351-019-00456-6

[61] Shuai Yuan, Jun Wang, and Xiaoxue Zhao. 2013. Real-time bidding for online advertising: Measurement and analysis. In *Proceedings of the 7th International Workshop on Data Mining for Online Advertising*. 1–8.

[62] Sharrukh Zaman and Daniel Grosu. 2013. Combinatorial auction-based allocation of virtual machine instances in clouds. *J. Parallel Distrib. Comput.* 73, 4 (2013), 495–508.

[63] Liangzhao Zeng, Boualem Benatallah, Anne H. H. Ngu, Marlon Dumas, Jayant Kalagnanam, and Henry Chang. 2004. QoS-aware middleware for web services composition. *IEEE Trans. Softw. Eng.* 30, 5 (2004), 311–327.

[64] Jixian Zhang, Xutao Yang, Ning Xie, Xuejie Zhang, Athanasios V. Vasilakos, and Weidong Li. 2020. An online auction mechanism for time-varying multidimensional resource allocation in clouds. *Fut. Gen. Comput. Syst.* 111 (2020), 27–38.

[65] Weinan Zhang, Shuai Yuan, and Jun Wang. 2014. Optimal real-time bidding for display advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1077–1086.

[66] Yeru Zhao, Zhiwu Huang, Weirong Liu, Jun Peng, and Qianqian Zhang. 2016. A combinatorial double auction based resource allocation mechanism with multiple rounds for geodistributed data centers. In *Proceedings of the IEEE International Conference on Communications (ICC'16).* IEEE, 1–6.

Received 15 July 2020; revised 9 March 2021; accepted 6 May 2021