Reinforcement Learning Based Real-Time Pricing in Open Cloud Markets

Pankaj Mishra(✉), Ahmed Moustafa, and Takayuki Ito

Nagoya Institute of Technology, Nagoya, Japan
pankaj.mishra@itolab.nitech.ac.jp, {ahmed,ito}@nitech.ac.jp

Abstract. Recently, with the rise in demand for reliable and economical cloud services, there is a rise in the number of cloud providers competing among each other. In such a competitive open market of multiple cloud providers, providers aim to model the selling prices of their requested resources in real-time to maximise their revenue. In this regard, there is a pressing need for an efficient real-time pricing mechanism, that effectively considers a change in the supply and demand of the resources in a certain open cloud market. In this research, we propose a reinforcement learning-based real-time pricing mechanism for dynamically modelling the prices of the requested resources. In specific, the proposed real-time pricing mechanism in a reverse-auction based resource allocation paradigm, which utilises the supply/demand of the resources and undisclosed preferences of the cloud users. Further, we compare the proposed approach with two state-of-the-art resource allocation approaches and the proposed approach outperforms the other two resource allocation approaches.

Keywords: Open cloud markets · Resource allocation · Real-time pricing

1 Introduction

In recent years, with a tremendous rise in several cloud consumers and cloud providers, open cloud markets has become a very competitive and complex business market. In such a competitive and complex market, each providers aim at maximising their revenue. In this regard, providers aim to serve their resources to the highest possible number of consumers and also selling their resources with the maximum possible prices. On the other hand, consumers aim to satisfy their resource requests with minimum possible price and also within the desired deadline. Thus, there is a pressing need to design an efficient pricing mechanism, that efficiently handles the conflicting objectives of the participants. In this regard, to handle the above-mentioned conflicting objectives of the participants, we propose a learning-based dynamic pricing mechanism. In specific, we propose an efficient learning-based real-time pricing mechanism, that models the selling prices in real-time with least possible delay. Moreover, a dynamic pricing

H. Fujita et al. (Eds.): IEA/AIE 2020, LNAI 12144, pp. 419–430, 2020.
https://doi.org/10.1007/978-3-030-55789-8_37

mechanism, which not only considers the supply/demand of the resources but also the preferences of the consumers, while modelling the selling price. In this regard, the proposed pricing mechanism consists of two main steps, which are, (1) tracking the supply-demand resources in the market (i.e. tracking the availability and demand of resources) and (2) mechanism to analyse the undisclosed preferences of the dynamically arriving consumers. Besides, an optimal pricing mechanism must maintain the equilibrium and social welfare [8] in any auction paradigm.

In the literature, Samimi et al. [7] and Zaman et al. [9] proposed efficient combinatorial auction-based resource allocation approaches in cloud computing environments and geo-distributed data centres, respectively. However, these two approaches are consumer-centric, which maximise the utilities of consumer alone, i.e., by selecting a provider with the minimum selling price. Meanwhile, Kong et al. [3] and Li et al. [4] proposed two auction-based pricing mechanism for cloud resource allocation and cloud workflow scheduling, respectively. However, both of these pricing mechanisms were based on static mathematical model, which could only adapt to static or gradually changing open cloud markets. Besides, there exist certain behavioural interdependencies [6] among vendors and consumers when modelling their bid values (i.e. resource prices). However, analysing and incorporating these interdependencies is a challenging problem that is not yet practically addressed. Therefore, there arises the need for real-time pricing approaches, that can incorporate the behavioural interdependencies among vendors and consumers. Secondly, buying and selling in such open markets takes place in an auction paradigm, therefore maintaining the equilibrium in an auction paradigm is also a challenging problem. In specific, maintaining the equilibrium in open cloud markets requires promoting two main characteristics, which are competitiveness [8] and fairness [5]. In specific, the optimal pricing policy needs to maintain competitiveness by dynamically modelling the resource selling prices based on supply/demand in the market. Meanwhile, a provider selection strategy needs to observe fairness and gives equal winning opportunities to all the bidding vendors in the market. However, to the best of our knowledge, none of the existing resource allocation approaches focus on addressing these three challenges simultaneously.

Therefore, it becomes clear that the existing resource allocation approaches exhibit two key limitations, such that 1) they are only suitable for static or gradually changing environments, whereas, they fail to adapt to dynamically changing environments; 2) they fail to consider the fairness in the open cloud market. In order to address the above-mentioned challenges and limitations, we propose a novel two-stage resource allocation approach that employs a reverse auction paradigm in open cloud markets. The proposed approach works in two stages as follows: 1) learning-based real-time pricing policy for consumers, and 2) fairness-based provider selection strategy for consumers. Accordingly, the contributions of this research are as follows:

- First, a new learning-based real-time pricing approach is proposed which optimises the true resource valuation of provider, based on the supply and demand in the open cloud market.
- Second, fairness mechanism for provider selection policy is proposed, to aid potential cloud users to select the best available cloud providers.
- Third, an open cloud market simulated environment is developed, that can simulate the dynamic arrival and departure of cloud users with different resource requests.

The rest of this paper is organised as follows. The problem formulation of learning the resource allocation in open cloud markets is introduced in Sect. 2. Section 3 presents the modelling of the cloud open market into a Markov process. In Sect. 4 and 5, the proposed real-time pricing algorithm and cloud provider selection strategy is discussed. Whereas, the experimental results are presented for evaluating the proposed approach in Sect. 6. Finally, the paper is concluded in Sect. 7.

2 Problem Formulation

This section presents the formulation of the proposed real-time reverse-auction (*RTRA*) pricing approach in an open-cloud market. In this context, the proposed approach is a two-stage process, as depicted in Fig. 1, which are, the real-time pricing algorithm and the cloud provider selection policy. Usually, in a reverse-auction paradigm for open cloud markets, there are three participants, which are, the cloud auctioneer, the cloud consumers and the cloud providers. The auctioneer conducts the auction by coordinating between the providers and the users. Specifically, it accepts the cloud resource requests from all the dynamically arriving users at one hand and their respective selling prices from all providers on the other hand. Then, based on these values, auctioneer determines the resource allocation rule and the payment rule. Further, this allocation and payment rule is broadcast to all the participants in the reverse-auction. Besides, it is the responsibility of the auctioneer to maintain the stability in the auction paradigm by observing confidentiality and non-partisan in the open cloud markets.

In this regard, we consider an open cloud market with one auctioneer and a set of n providers denoted as $p_1, \ldots, p_n$. Furthermore, each provider is represented by an autonomous agent. In addition, each provider has g types of resources, denoted as $pr_1, \ldots, pr_g$; where pr_k represents the quantity of resource type $k \in g$. These resources are requested by m dynamically arriving consumers denoted as $c_1, \ldots, c_m$, each with resource request req_j, where $j \in [1, m]$. Further, each resource request is denoted as $req \equiv (\{rb_k\}, dl)$, where rb_k represents the quantity of the requested resource of type $k \in g$, whereas, dl denotes a strict deadline for serving each consumer's request. By following a reverse-auction paradigm, various cloud resources are dynamically allocated using the proposed *RTRA* approach as follows. Firstly, upon receiving a set of resource requests, a cloud auctioneer broadcasts these requests to all the available providers in the market. Then, each available provider sets its optimal selling price (i.e. bid) using a real-time pricing policy and submits it to the auctioneer. Finally, the auctioneer elicits a suitable provider to serve the lodged request by each consumer, based on a provider selection strategy. Besides, it should be noted that the proposed *RTRA* approach sustains with the assumption that multiple independent provider can presumably participate in the proposed auction platform to sell their resources through an unbiased auctioneer. As a result, each provider is expected to agree to share its information to maximise its revenues in the long term. Towards this end, we model the open cloud market into multi-agent Markov decision process *(MMDP)*. In this context, we structure the state space of the proposed *MMDP* model based on the feature vector of the requesting consumers stored in the transaction database. In the further section, we explain the proposed preference labelling scheme along with three other key aspects of the proposed *MMDP* model, i.e., states, actions, and rewards.

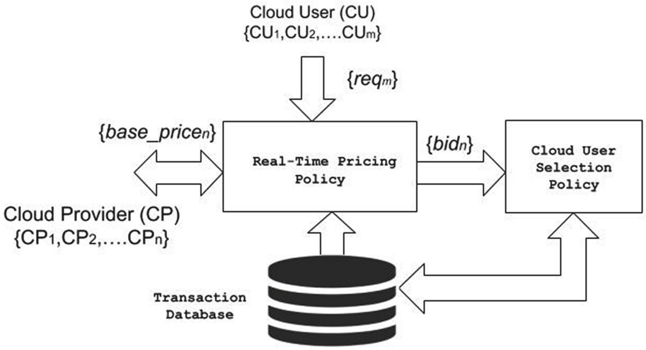


Fig. 1. The proposed architecture of real-time resource allocation

3 Modelling Cloud Open Markets

In this section, we model the resource allocation problem in a multi-agent Markov decision process *(MMDP)*. In this regard, the open cloud market is the Markov environment, wherein m dynamically arriving consumers change the state of the environment with their resource requests req. In turn, n autonomous agents on behalf of each provider optimises the selling price for each resource request req from consumers. In this context, the action space is defined as $A \equiv \{A_1, \dots A_n\}$, wherein, A_i represents the action space for p_i, $i \in n$. Further, based on the pricing policy $\pi_i : s_i \mapsto A_i$ of p_i, the action $act_i \in A_i$ is determined. After executing the action act_i (after resource allocation), the state s is transferred to the next state s_{i+1}; which is based on the transition function $\tau * s \times A_1 \times \cdots \times A_n \mapsto \Omega(s)$, where $\Omega(s)$ represents the set of probability distributions and s represents the shared state. Finally, at the end of each episode, all the agents, which took part in the auction, receive rewards based on the outcome of the auction (win or loss), such that, reward $r_i : s_i \times A_i \times ... \times A_n \mapsto R_i$, where R is the resultant reward. In this regard, each agent (i.e. provider) models their selling price based on the demand-supply of the resources in the market. In further subsections, we explain the three key aspects of the proposed *MMDP* model; i.e., states, actions, and rewards.

3.1 State

In the proposed *MMDP* model, each state s represents the status of all the agents (i.e. providers) in the market. In this context, the existence of multiple consumers within each episode leads to multiple resource allocation transactions within each single episode. Therefore, the state s is represented by concatenating all the transactional states within an episode, that is denoted as $s \equiv [H, F]$. In this state representation, H represents the vector of concatenated transactional states, and F represents the feature vector of each consumer. In this regard, each transactional state is represented by a vector η which is represented based on the categories of providers and consumers. Specifically, η is represented as $\eta_{p_i} \equiv [rev_{p_i}, rb_{c_j}, avg\_win_{p_i}]$, where, rev_{p_i}, rb_{c_i}, and $avg\_win_{p_i}$ denote the revenue, requested resources, and average winning rate of the provider p_i, respectively. In this regard, at the end of each episode, all the transactional state vectors η are concatenated together, which is represented as concatenated transactional state vector H. Whereas, feature vector represents the information of the requesting consumers stored in the transaction database.

3.2 Action

Initially, each provider independently sets a base price for each bundle of the requested resources. This base price is denoted as $base\_price^i_j$ by the provider p_i for the requested bundle of resources from consumer c_j, where $i \in [1, n]$ and $j \in [1, m]$. In this context, different providers optimise their base prices based on a set of exclusive adjustment multipliers for each resource request. In this regard, these adjustment multipliers represent the optimised action values *act*, which are obtained based on a proposed RL-based algorithm. Specifically, based on the learnt adjustment multipliers (i.e. action values), the optimal selling price for each provider *sp* is computed using Eq. 1.

$$bid = sp(i) = base\_price^i_j \times (1 + act_i) \tag{1}$$

In this context, the action space for n different providers is denoted as $A \equiv \{A_1, \ldots A_n\}$, where A_i represents the action space for provider p_i, where $i \in n$. Further, based on provider p_i's pricing policy $\pi_i : s_i \mapsto A_i$, the action value $act_i \in A_i$ is determined. Finally after executing the chosen actions (allocating the resources), the proposed *MMDP* model transfers to the next state s_{i+1}. This state change occurs based on the transition function $\tau * s \times A_1 \times \cdots \times A_n \mapsto \Omega(s)$, where $\Omega(s)$ represents a set of probability distributions. Finally, at the end of each episode, all the bidding providers receive rewards based on their chosen actions, such that; $r_i : s_i \times A_i \times \cdots \times A_n \mapsto R$.

3.3 Reward

In open cloud markets, several providers are competing amongst each other to maximise their profits (rewards). In this context, the profit of each provider is maximised by winning the highest possible number of auctions, while simultaneously minimising its loss and non-participation rates. Therefore, in this work, the reward function is formulated to represent the win, the loss and the non-participation of each provider. In this regard, we chose to model the reward function r for each provider p_i as $r_i \equiv (\alpha \times win_i, \beta \times loss_i, \gamma \times non\_participation_i)$, wherein, win_i, $loss_i$ and $non\_participation_i$ denote the number of times provider p_i won, lost and did not participate in the auctions within a single episode. In this context, α, β and γ represent the impact of each of these outcomes, which are set independently by each provider. Finally, the episodic reward of each provider is computed via cumulatively adding all the rewards that were earned within each episode. In this regard, to reduce the complexity of updating the reward values after each auction; these reward values are updated only at the end of each episode. Finally, the proposed *MMDP* model transfers to the next state.

4 Real-Time Pricing

In this section, we discuss the proposed reinforcement learning-based real-time pricing algorithm. As mentioned before the proposed algorithm aims at learning the supply/demand of the resources as well as the consumers' preferences in the open cloud markets. In any competitive open markets, providers have a limited volume of resources, and they make profit only serving the resources to the consumers. Therefore, to maximise their profit, providers would have to offer their resources wisely at the right time, and importantly at the right price. In this regard, the provider is expected to carefully decide whether to bid for any particular request from the consumer by evaluating their undisclosed preferences. Considering this dynamism of open cloud markets, there is a need for a learning-based real-time pricing approach. In addition, given the presumed real-time scenario, with no available training data sets, supervised learning becomes infeasible. Therefore, the proposed real-time pricing algorithm adopts a reinforcement learning scheme [2] in order to handle the presumed real-time scenario. As a result, the proposed real-time pricing algorithm enables the bidding providers to dynamically optimise their bid values. Specifically, the proposed algorithm optimises the bid values based on the dynamically changing supply/demand in an open market, and the characteristics of the requesting consumers. In doing so, the proposed algorithm takes three inputs, which are (i) the characteristics of consumers in the form of feature vector (F), (ii) the concatenated categorical state H, and (iii) the cumulative rewards of all providers. Then, the proposed algorithm provides the adjustment multipliers for all the providers as output. Meanwhile, the initial state (s_0) is determined by a predefined distribution. Further, as depicted in Algorithm 1, each provider aims to select and leverage a certain adjustment multiplier (action), to maximise its total expected future revenues. These future revenues are discounted by the factor γ each time-step. In this regard, the future reward at time-step t for provider i is denoted as $R_i = \sum_{t=0}^{T_e} \gamma^t r_i^t$, where T_e is the time-step at which the bidding process ends. In addition, the Q function for provider i is denoted by Eq. 2.

$$Q_i^{\pi}(s,a) = \mathcal{E}_{\pi,\mathcal{T}}[\sum_{t=0}^{T} \gamma^t r_i^t | s_0 = s, a] \tag{2}$$

where $\pi = \{\pi_1, \ldots, \pi_n\}$ is the set of joint-policies of all the $CB - agent$ and $a = [a_1, \ldots, a_n]$ denotes the list of bid-multipliers (actions) of all the $CB-agent$. Further, the next state s' and the next action a' are computed using Bellman equation as shown in Eq. 3:

$$Q_i^{\pi}(s,a) = \mathcal{E}_{r,s'}[r(s,a) + \gamma \mathcal{E}_{a' \sim \pi}[Q_i^{\pi}(s^{'}, s^{'})]. \tag{3}$$

On the other hand, the mapping function μ_i maps each shared state $s[H, F]$ of provider p_i to the selected action act_i, based on Eq. 4. This mapping function μ is the actor in the actor-critic architecture.

$$a_i = \mu(s) = \mu_i([H, F]) \tag{4}$$

Further, we get Eq. 5 from Eqs. 3 and 4. As shown in Eq. 5, $\mu = \{\mu_1, \ldots, \mu_n\}$ are the joint deterministic policies of all the providers.

$$Q_i^{\mu}(s, a_1, \ldots, a_n) = \mathcal{E}_{r,s'}[r(s, a_1, \ldots, a_n) + \gamma Q_i^{\mu}(s', \mu_1(s', \ldots, \mu_n(s'))] \tag{5}$$

In this regard, the goal of the proposed algorithm becomes to learn an optimal policy for each provider to attain the Nash equilibrium [1]. In addition, in such stochastic environments, each provider learns to behave optimally by learning an optimal policy (μ_i), which is also based on the optimal policies of the other co-existing providers. Further, in the proposed algorithm, the equilibrium of all providers are achieved by gradually reducing the loss function $LOSS(\theta_i^Q)$ of the critic Q_i^{μ} with the parameter θ_i^Q as denoted in Eqs. 6 and 7. In specific, in Eqs. 6 and 7, $\mu' = \{\mu_1^{'}, \ldots, \mu_n^{'}\}$ 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 $(s, act_i, \ldots, act_n, r_i, s')$ represents the transition tuple that is pushed into the replay memory D. In this regard, each provider's policy μ_i, with parameters θ_i^{μ}, is learned based on Eq. 8, as demonstrated in Algorithm 1.

$$LOSS(\theta_i^Q) = (y - \gamma Q_i(s, act_1, \ldots, act_n))^2 \tag{6}$$

$$y = r_i + \gamma Q_i^{'}(s^{'}, act_1^{'}, \ldots, act_n^{'})|_{act_0' = [\mu_0'([Z', F_1']), \ldots, \mu_0'([Z', F_u'])]} \tag{7}$$

$$\nabla_{\theta_i^{\mu} J} \approx \sum_{w} \nabla_{\theta_i^{\mu}} \mu_i([Z, F_w]) \nabla_{act_{iw}^q} Q_i(s, act_1, \ldots, act_n) \tag{8}$$

Finally, the optimal bid values are utilised by the proposed provider selection algorithm, which is discussed in the next section.

Algorithm 1. Real-Time Pricing Algorithm

```
**Initialise:** $Q_i(s, act_1, \ldots, act_n|\theta_i^Q)$, replay memory $D$
**Initialise:** actor $\mu_i$, target actor $\mu_i^{'}$
**Initialise:** target network $Q^{'}$ with $\theta_i^{Q^{'}} \leftarrow \theta_i^Q$,
$\theta_i^{\mu^{'}} \leftarrow \theta_i^{\mu}$ for each provider $i$.
**for** episode = 1 to $E$ **do**
    **Initialise:** $s_0$
    **for** $t = 1$ to $T$ **do**
        **for** arriving $CU$ within $T$ **do**
            Get $a_i$ using Equation 4
            compute $\varUpsilon \in [0, 1]$
            Get $r_i^t$, where $i \in N$ and $F$
        $r_i = \sum_{i=1}^{n} \sum_{t=1}^{T} r_i^t$ rewards within $T$
        Push $(s, act_i, r_i, s^{'})$ into D // s' is the next state
        $s^{'} \leftarrow S$
        **for** provider $i = 1$ to $n$ **do**
            Sample mini batch $(s,act_i,\ldots,act_n,r_i,s^{'})$ from $D$
            Update critic and actor using Equation 6 and 8
            Update target network: $\theta^{'} \leftarrow \tau\theta + (1 - \tau)\theta$
```

5 Cloud Provider Selection Strategy

In this subsection, we discuss the proposed provider selection strategy for every resource requests from consumers. The proposed strategy handles the primitive drop in bidder drop problem in the auction [5] by employing a priority-based fairness mechanism. Firstly, in order to remove the bidder drop problem in the proposed resource approach, we include a priority-based fairness mechanism. In this regard, on receiving the bids from all the participating providers in the auction, auctioneer attaches the priority label (pr) to every bid from the providers. In specific, on receiving the request $req_j = (rb_j, dl_j)$ from consumer c_j and all the corresponding bids $bid_i^j = (rb_j, sp_i)$ from all the provider, where $i \in n$ and $j \in n$, priority label pr_i is computed using Eq. 9 as follows.

$$pr = 1 - (loss/O) \tag{9}$$

wherein, $loss$ is the number of times consumers lost in the last O auctions, such that provider with $pr \in [0, 1]$, $pr = 1$ has highest priority and $pr = 0$ has the lowest priority. Secondly, auctioneer computes the normalised values of priority label pr and optimised selling price sp (bid values) based on the based on the simple additive weighting (SAW) technique [10] using Eq. 10 as follows:

$$S = \begin{cases} \frac{C^{max} - C}{C^{max} - C^{min}}, & \text{if } C^{max} - C^{min} \neq 0. \\ 1, & \text{if } C^{max} - C^{min} = 0. \end{cases} \tag{10}$$

wherein, where C denotes value of either sp or pr. Then, finally, bid score (*bid-score*) is calculated using Eq. 11, and provider's bid with minimum $bid - score$ is the winner.

$$bid - score = S_{pr} \times W_{pr} + S_{sp} \times W_{sp} \tag{11}$$

wherein, S_{pr} and S_{sp} denote the scaled values of priority label and selling price respectively. Whereas, W_{sp} and W_{sp} denote the preference weight of priority label and selling price respectively. These weight preferences are decided independently by each consumer. In this regard, the winning provider serves the requesting consumer. In this regard, auctioneer attempts to select one provider for each resource request req with maximum $bid - score$ and consumer pay the corresponding sp.

6 Experimental Setup and Results

This section presents the results of the performed experiments, that are conducted in order to evaluate the proposed real-time reverse-auction ($RTRA$) pricing approach in an open cloud market. The open cloud environment is implemented in *SimPy* python library, wherein, each cloud provider is initialised with quantities of four types of resources; namely, computer processing speed, memory, storage, and bandwidth (BW). We compare the proposed $RTRA$ approach with two other notable bidding-based resource allocation approaches, which are as follows: (1) the combinatorial double auction resource allocation approach ($CDARA$) [7]; and (2) the indicator-based combinatorial auction-based approach ($ICAA$) [3]. The $CDARA$ has fixed selling price strategy and no fairness based provider selection strategy, whereas, $ICAA$ the indicator-based combinatorial has a static mathematical model to model the price and also with no fairness baaed provider selection strategy. Finally, We run the three resource allocation approaches for *120* episodes, each episode of length *1500* s, wherein *25* consumers arrive dynamically in each episode and 12 cloud providers.

The performance of the provider is evaluated based on two parameters, which are: (1) *average revenue*; (2) *average unavailability* of resources with the provider; and (3) active participation of the provider computed as a ratio of never-won provider to total-provider in the auction denoted as *fairness*, as depicted in Table 1. Firstly, from Table 1, it is clear that the average revenue $RTRA$ is approximately 55% higher as compared to the other two approaches. Moreover, in the $RTRA$, unavailability of the provider is decreased by $1/3^{rd}$ of that observed in $CDARA$. In addition, $RTRA$ increases the participation of the provider through the proposed priority-based allocation which is reflected through the fairness values in Table 1. Therefore, the proposed $RTRA$ approach is capable of maximising the performance of the provider in the open cloud markets.

Similarly, the performance of consumers is evaluated based on three parameters, which are, (1) the ratio of resource price paid and maximum offered price by all the providers (PMB), (2) waiting time (avg_wait_time) of the

Table 1. Performance of cloud providers

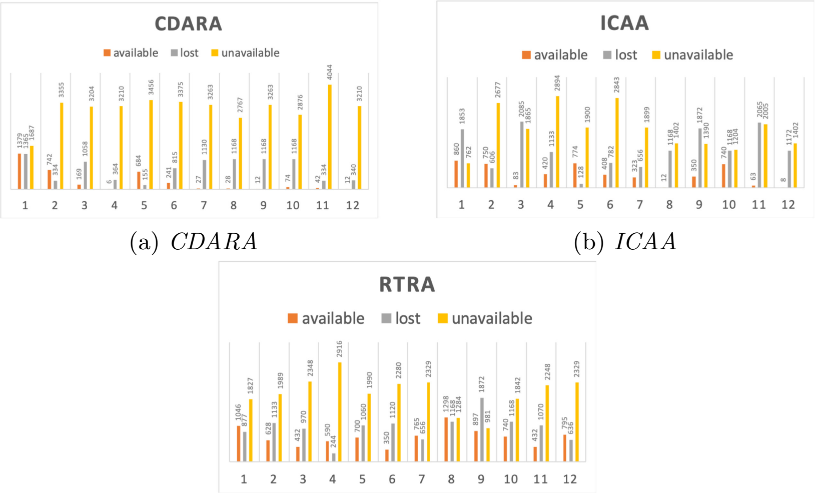
Performance	*CDARA*	*ICAA*	*RTRA**
Average revenue	25000	23000	**40000**
Average unavailability	64%	42%	**28%**
Fairness	48%	53%	**65%**

served consumers, and (3) scalability (σ) of the serving providers. Firstly from Table 2, the consumer in the cloud market with *RTRA* approach pays marginally lesser prices as compared to the maximum offered prices among all the other bids. Secondly, the cloud market with *RTRA* approach has lesser waiting time as compared to the other two approaches. Finally, *RTRA* allocates the consumers request to more scalable providers, where, scalability($sigma$) is defined as $\sigma = 1 - (requested\ resource/available\ resource)$. Therefore, the proposed *RTRA* approach is capable of maximising the performance of the *CU* in the open cloud markets.

Table 2. Performance cloud user

Utilities	*CDARA*	*ICAA*	*RTRA**
PMB	0.86	0.92	**0.76**
avg_wait_time	24	22	**12**
σ	0.4	0.6	**1.2**

Finally, the overall performance of the open cloud market is evaluated based on three parameters, which are; (1) the participation rate of providers in the auction (availability); (2) the non-participation rate of providers in auction (unavailability); and (3) the lost rate of provider in the auction. As shown in Fig. 2, providers in the proposed *RTRA* approach has a higher participation rate, lower non-participation rate and also lower lost rate, as compared to the other two approaches. This proves the capability of the *RTRA* approach to serve higher numbers of consumer as compared to the other two approaches. In conclusion, the proposed *RTRA* approach outperforms the other two approaches by serving the highest number of consumer while enabling higher participation in the auctions.



(a) *CDARA*

(b) *ICAA*

(c) *RTRA*

Fig. 2. The availability, unavailability and loss of providers in auction, based on the number of buyers handled

7 Conclusion

This paper proposes a learning-based real-time pricing approach for resource allocation in dynamic and complex open cloud markets. The proposed approach models the selling price of the requested resources in real-time, based on the proposed pricing approach, which considers the supply/demand of the resources in the market and the characteristics of the cloud users. As a result, the proposed approach enables both the cloud providers and cloud consumers to maximise their performances at the same time. The experimental results demonstrate the efficiency and fairness of the proposed resource allocation approach and its ability to maximise the overall performance of all the participants in an open cloud market. The future work is set to develop a mechanism that enables multiple preferences of the consumers in the provider selection strategy.

References

1. Hu, J., Wellman, M.P., et al.: Multiagent reinforcement learning: theoretical framework and an algorithm. In: ICML, vol. 98, pp. 242–250. Citeseer (1998)
2. Konda, V.R., Tsitsiklis, J.N.: Actor-critic algorithms. In: Advances in Neural Information Processing Systems, pp. 1008–1014 (2000)
3. Kong, Y., Zhang, M., Ye, D.: An auction-based approach for group task allocation in an open network environment. Comput. J. **59**(3), 403–422 (2015)
4. Li, X., Ding, R., Liu, X., Liu, X., Zhu, E., Zhong, Y.: A dynamic pricing reverse auction-based resource allocation mechanism in cloud workflow systems. Sci. Program. **2016**, 17 (2016)

5. Murillo, J., Muñoz, V., López, B., Busquets, D.: A fair mechanism for recurrent multi-unit auctions. In: Bergmann, R., Lindemann, G., Kirn, S., Pěchouček, M. (eds.) MATES 2008. LNCS (LNAI), vol. 5244, pp. 147–158. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87805-6_14
6. Myerson, R.B.: Optimal auction design. Math. Oper. Res. **6**(1), 58–73 (1981)
7. Samimi, P., Teimouri, Y., Mukhtar, M.: A combinatorial double auction resource allocation model in cloud computing. Inf. Sci. **357**, 201–216 (2016)
8. Toosi, A.N., Vanmechelen, K., Khodadadi, F., Buyya, R.: An auction mechanism for cloud spot markets. ACM Trans. Auton. Adapt. Syst. (TAAS) **11**(1), 2 (2016)
9. Zaman, S., Grosu, D.: Combinatorial auction-based allocation of virtual machine instances in clouds. J. Parallel Distrib. Comput. **73**(4), 495–508 (2013)
10. Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J., Sheng, Q.Z.: Quality driven web services composition. In: Proceedings of the 12th International Conference on World Wide Web, pp. 411–421. ACM (2003)