2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications

Pricing Cloud Resource based on Multi-Agent Reinforcement Learning in the Competing Environment

Bing Shi[1,2], Han Yuan[1], Rongjian Shi[1]
[1] *School of Computer Science and Technology, Wuhan University of Technology, Wuhan, P.R. China*
[2] *State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, P.R. China*
bingshi@whut.edu.cn

***Abstract*—Multiple cloud providers compete against each other in order to attract cloud users and make profits in the cloud market. In doing so, each provider needs to charge fees to users in a proper way. In this paper, we analyze how a cloud provider sets price effectively when competing against other cloud providers. The price set by the cloud provider is affected by its opponent's price, and as well as the prices set in the last round. Specifically, we model this problem as a Markov game by considering two cloud providers competing against each other. We then adopt two different solution concepts in game theory, minimax and Nash equilibrium, to solve this problem. Specifically, we use two different multi-agent reinforcement learning algorithms, minimax-Q and Nash-Q, which correspond to those two solution concepts respectively, to design the pricing policies. Furthermore, we improve the Nash-Q learning algorithm by taking into account the probability of each Nash equilibrium happening. Based on this, we run extensive experiments to analyze the effectiveness of minimax-Q and Nash-Q based pricing policies in terms of making long-term profits. We find that the pricing policy based on Nash-Q learning algorithm with selecting Nash equilibrium according to the probability can beat other Nash-Q based pricing polices with selecting Nash equilibrium according to the maximal payoff. However, in the further experimental analysis, we find that minimax-Q based pricing policies can beat all Nash-Q based pricing policies. This is because the minimax solution concept is more suitable in this competing environment. Our experimental results provide useful insights on designing practical pricing policies for competing cloud providers.**

***Index Terms*—Competing Cloud Providers, Pricing Policy, Minimax-Q Learning, Nash-Q Learning**

I. Introduction

During the past few years, the development of cloud computing has achieved significant success in the industry since it can provide economical, scalable, and elastic access to computing resources, thus liberating people from installing, configuring, securing, and updating a variety of hardware and software [1]–[3] . More and more firms and personal users have been using cloud computing services over Internet, which contribute to the development of the cloud computing market. The global cloud computing market is expected to grow at a 30% compound annual growth rate(CAGR) reaching \$270 billion in 2020. To compete for hundreds of billions of dollars, many firms as service providers have been participating in the cloud market [4]. Now, there exist many dominating cloud platforms offering cloud services, such as Microsoft's Azure, IBM's SoftLayer and Amazon's AWS. In the cloud market with multiple cloud providers, cloud users have various choices, and they usually participate in the provider which can satisfy their demands and charge the lowest price to them. Actually, when multiple providers offer similar quality of service [5]–[7], the price will significantly affect users' choices and thus providers' profits. Therefore, cloud providers need to set effective prices to compete against each other. Furthermore, the competition among providers usually lasts for a long time, i.e. the providers compete against each other repeatedly, and thus they need to maximize the long-term profits. In this paper, we analyze how a cloud provider designs an appropriate pricing policy to maximize the long-term profit and also remain attractive to cloud users.

There exist some works on designing pricing policies for cloud providers. In [8], a game-theoretic approach was proposed to allocate the computing resources dynamically in response to fluctuations of the workload. In [9], a non-cooperative competing model based on game theory has been proposed which computes the equilibrium price for one-shot game and does not consider the long-term profits. In [10], [11], the authors assume that there is only one provider, while in today's cloud market multiple providers exist and compete against each other. Then the authors in [12], [13] analyze the user behavior with respect to the providers' prices, but ignore the competition among providers. In [14], the authors analyze the pricing policy in the competing environment by assuming that there is only one proactive provider, and other providers just follow the proactive one's pricing policy. Some other works, such as [15], [16], consider the competition among providers but does not capture the market dynamics, and their algorithms can only be applied to a very small market with few users.

To the best of our knowledge, few works have considered the situation of multiple providers competing against each other repeatedly. In this paper, we analyze how the competing cloud provider sets price effectively to maximize the long-term profits in the context with two competing providers.[1] In more detail, we first describe basic settings of cloud users

[1]Our model can be easily extended to the case with more than two cloud providers.

and providers. Specifically, we consider the uncertainty of users choosing cloud providers in the setting, which is consistent with the realistic user behavior. Furthermore, how users choosing cloud providers is affected by the prices, and how cloud providers setting prices is affected by users' choices, and therefore it is a sequential-decision problem. Moreover, this problem involves two self-interested cloud providers, and thus it is a Markov game [17]. In this paper, we model the competition between cloud providers as a Markov game. In such a game, we adopt two different solution concepts, minimax (which means that the cloud provider tries to maximize its payoff in the worst case) and Nash equilibrium (which means that in the equilibrium, no provider can gain more by individually deviating its current pricing policy), to solve this problem. Specifically, we use two different multi-agent reinforcement learning algorithms, minimax-Q (corresponding to minimax solution concept) [18] and Nash-Q (corresponding to Nash equilibrium solution concept) [19], to solve this game and design the pricing policy. Moreover, we improve the traditional Nash-Q learning algorithm by taking into account the probability of each Nash equilibrium appearing. We show that the pricing policy based on the improved Nash-Q learning algorithm can beat other Nash-Q based pricing policies. However, in the further investigation, we find that minimax-Q based pricing policies can beat all Nash-Q based pricing policies (even the improved one). This may suggest that in the competing environment, the cloud provider should adopt the pricing policy which can guarantee the maximal payoff in the worst case (i.e. minimax solution concept).

The structure of the paper is as follows. In Section II, we describe basic settings of cloud users and providers. In Section III, we describe how to use minimax-Q and Nash-Q learning algorithms to design the pricing policy. We run extensive experiments to evaluate the pricing policies in different situations in Section IV. Finally, we conclude the paper in Section V.

II. Basic Settings

In this section, we describe the basic settings of cloud providers and users. We assume that there are N users and two cloud providers, A and B. Cloud providers compete against each other repeatedly, i.e. the competition consists of multiple stages. At the beginning of each stage, each provider publishes its price, and then each user chooses to be served by which provider based on its choice model. According to users' choices, the two providers compute the obtained profits at the current stage, and the competition enters into the next stage.

A. Cloud Providers

Cloud providers can make profits by charging fees to users, while they also need to pay for the cost of offering services (e.g. power, hardware, infrastructure maintenance cost and so on). At stage t, provider i should pay for the cost of offering per unit service [20], which is denoted as $c_{i,t}$. We assume that each user only requests one-unit service.[2] Therefore, the amount of requested service at stage t is equal to the number of users. At the beginning of the competition, the initial marginal cost of provider i is $c_{i,0}$. At stage t, the amount of users choosing provider i is $N_{i,t}$, and then at this stage, the marginal cost is:

$$c_{i,t} = c_{i,0}(N_{i,t})^{-\beta} e^{-\theta t} \tag{1}$$

This equation indicates that as more users requiring the service and as time goes, the marginal cost decreases [21]. Specifically, when the provider receives more demands of services, its marginal cost would be decreased because of economics of scale, where β is the parameter for the economics of scale, and $\beta > 0$. Furthermore, the reduction of hardware cost and the development of technology contribute to the temporal decaying factor of the marginal cost, where θ is the parameter of temporal decaying factor, and $\theta > 0$.

We assume that the price per unit service is denoted as p, and all allowable prices constitute a finite set P. The price is actually the *action* used in Section III. After providers publishing the prices, users make the choices of providers. We calculate the immediate reward of each provider, which is the immediate profit made at the current stage t:

$$r_{i,t} = N_{i,t}(p_{i,t} - c_{i,t}) \tag{2}$$

where $p_{i,t}$ is the price set by provider i at stage t.

B. Cloud Users

Each user has a marginal value on per-unit requested service, which is denoted as δ. At stage t, after all providers publish the prices, user j can calculate its expected revenue when entering provider i, which is:

$$R_{j,i}^{t} = \delta_j - p_{i,t} \tag{3}$$

Intuitively, based on Equation 3, cloud users can determine in which provider they can obtain the maximal revenue at the current stage, and then choose that provider. However, in the real world, users keep requiring cloud services, and they usually take into account the prices at previous stages. Specifically, in this paper, we assume that the users consider the prices at the current stage t and the last stage $t-1$ when choosing the cloud providers. We do not need to consider the prices at all previous stages since in this paper, the providers' prices and the users' choices are affected by each other, and thus the price of the last stage actually implies the dynamic interaction of all previous stages. Therefore, the expected utility that user j can make when entering provider i is:

$$v_{j,i}^{t} = \xi R_{j,i}^{t} + (1-\xi) R_{j,i}^{t-1} \tag{4}$$

where ξ is the weight of the price considered by the user at this stage. Furthermore, in reality, when agents make decisions, their choices are affected by some unobservable factors [22], such as customers' loyalty on some product brand, which is

[2]Users can request multiple units of service. However, since the price charged by the cloud provider to users is based on per unit service, we can simply assume that each user only requests one-unit service.

denoted as $\eta_{j,i}$. This part introduces the uncertainty of users' choice. Now the utility that cloud user j makes in provider i at stage t is defined as follows:

$$u_{j,i}^t = v_{j,i}^t + \eta_{j,i} \tag{5}$$

We assume that the random variable $\eta_{j,i}$ is an independently, identically distributed extreme value, i.e. it follows Gumbel and type I extreme value distribution [22], and the density of $\eta_{j,i}$ is

$$f(\eta_{j,i}) = e^{-\eta_{j,i}} e^{-e^{-\eta_{j,i}}} \tag{6}$$

and the cumulative distribution is

$$F(\eta_{j,i}) = e^{-e^{-\eta_{j,i}}} \tag{7}$$

The probability of user j choosing provider i at stage t, which is denoted as $P_{j,i}^t$

$$\begin{aligned} P_{j,i}^t &= Prob(u_{j,i}^t > u_{j,i'}^t, \forall i' \neq i) \\ &= Prob(v_{j,i}^t + \eta_{j,i} > v_{j,i'}^t + \eta_{j,i'}, \forall i' \neq i) \\ &= Prob(\eta_{j,i'} < \eta_{j,i} + v_{j,i}^t - v_{j,i'}^t, \forall i' \neq i) \end{aligned} \tag{8}$$

According to (7), $P_{j,i}^t$ is

$$P_{j,i}^t = e^{-e^{(\eta_{j,i}+v_{j,i}^t - v_{j,i'}^t)}} \tag{9}$$

Since $\eta_{j,i}$ is independent, the cumulative distribution over all $i \neq i'$ is the product of the individual cumulative distributions

$$P_{j,i}^t \mid \eta_{j,i} = \prod e^{-e^{-(\eta_{j,i}+v_{j,i}^t - v_{j,i'}^t)}} \tag{10}$$

And $\eta_{j,i}$ is unknown to the providers, so the choice probability is the integral of $P_{j,i}^t \mid \eta_{j,i}$ over all values of $\eta_{j,i}$ weighted by its density

$$P_{j,i}^t = \int (\prod_{i' \neq i} e^{-e^{-(\eta_{j,i}+p_{i,t}-p_{i',t})}}) e^{-\eta_{j,i}} e^{-e^{-\eta_{j,i}}} d\eta_{j,i} \tag{11}$$

The closed-form expression is

$$P_{j,i}^t = \frac{e^{v_{j,i}^t}}{\sum_{i'} e^{v_{j,i'}^t}} \tag{12}$$

which is the probability of user j choosing to be served by provider i at stage t.

III. Multi-Agent Reinforcement Learning Algorithms

After describing the basic settings, we now introduce how to design a pricing policy for the cloud provider. How to set an effective price is a decision-making problem, and reinforcement learning algorithms have been widely used to solve similar issues. Since our problem actually involves two providers competing against each other repeatedly, it can be modeled as a Markov game [17]. We intend to adopt two well-known solution concepts to solve this game. The first solution concept is minimax, which means that the cloud provider tries to maximize the minimum gain (i.e. tries to do best in the worst case). The reason for considering this concept is that in realistic cloud market, some providers want to guarantee the payoff in the worst case. Another solution concept is Nash equilibrium, which has been widely used in game theory. In such an equilibrium, each provider has to maximize its payoff given other providers' strategies, and no one can gain more by individually deviating its strategy. Specifically, we use minimax-Q learning algorithm to find the minimax solution concept in this game, and use Nash-Q learning algorithm to find the Nash equilibrium solution. In the following, we introduce how to design the pricing policy based on minimax-Q learning and Nash-Q learning algorithms respectively.

At stage t, provider A sets price according to its own and the opponent B's price at the last stage $t-1$, which is denoted as state $s_{t-1} = (p_{A,t-1}, p_{B,t-1})$. Note that the state does not involve the amount of users participating in each provider since the price has implied users' choices and therefore we only use the prices to represent the state. The state space is denoted as $S = P \times P$. For simplicity, in the following, we use $a \in P$ and $b \in P$ to represent the actions of providers A and B respectively. The pricing policies of provider A and B are denote as Π_A and Π_B respectively. Based on these notations, minimax-Q learning algorithm is shown in Algorithm 1, and Nash-Q learning algorithm is shown in Algorithm 2. In this setting, it is guaranteed that both algorithms converge [18], [19].

In Nash-Q learning algorithm, one important step is to compute Nash equilibrium at each state. We adopt a typical Lemke-Howson algorithm [23] to compute Nash equilibrium in Algorithm 2. Another problem is that there usually exist multiple Nash equilibria in the game, and we have to use some way to decide the selection of Nash equilibrium. In [19], Nash equilibrium selection is based on the sequence of finding each Nash equilibrium, which is not reasonable. The authors also suggest selecting the Nash equilibrium which has the maximal total payoff of all players. In this paper, we adopt this selection method. However, this selection implies that all players in the game need to coordinate with each other before taking the Nash equilibrium actions, which is infeasible in real world. Another possible way is to select Nash equilibrium according to the probability of each equilibrium happening. In this selection, we first need to know the probability of each Nash equilibrium happening, which is also a trick problem to address.

In this paper, we adopt fictitious play algorithm [24], [25] to approximate the probability of each equilibrium appearing. In the fictitious play algorithm, opponents are assumed to play a fixed mixed strategy. Then by observing relative appearance frequencies of different actions, the player can estimate their opponents' mixed strategies, and take a best response. The observed frequencies of opponents' actions are termed FP beliefs. In each round, all players estimate their opponents' mixed strategies and update their FP beliefs, and play a best response to their FP beliefs. All players continually iterate this process until it converges. The converged FP beliefs constitute a Nash equilibrium. When starting from different initial FP beliefs, the game often converges to different Nash equilibrium. Therefore, by sampling initial FP beliefs randomly, we record

Algorithm 1 minimax-Q learning

```
Input: pricing space $P$; $B$'s pricing policy $\Pi_B$
Output: $A$'s pricing policy $\Pi_A$
1: for $\forall s \in S, V(s) = 0, and\ \forall a \in P, Q(s,a,b) = 0$
2: for $\forall s \in S, \forall a \in P, \Pi_A(s,a) \leftarrow 1/|P|$
3: repeat
4:     at the current state $s$, given $\epsilon$, generate a random number $rand$ ($0 < rand < 1$); when $rand \leq \epsilon$, $A$ chooses a price $a \in P$ randomly; when $rand > \epsilon$, $A$ chooses a price $a \in P$ according to the pricing policy $\Pi_A$
5:     $B$ chooses the price $b$ (according to the pricing policy $\Pi_B$), the next state is $s' = (a,b)$
6:     $Q(s,a,b) = (1-\alpha) * Q(s,a,b) + \alpha * (r_{i,t} + \gamma * V(s'))$
7:     $\Pi_A(s,\cdot) = argmax_{\Pi'_A(s,\cdot)}(min_{b'} \sum_{a'}(\Pi_A(s,a') * Q(s,a',b')))$
8:     $V(s) = min_{b'}(\sum_{a'}(\Pi_A(s',a') * Q(s',a',b')))$
9: until ($\Pi_A(s,\cdot)$ is converged)
```

the frequency of each Nash equilibrium being converged, and this frequency is the approximated probability of each equilibrium happening. In this paper, we randomly choose 1000000 different initial FP beliefs to estimate the probability of each Nash equilibrium happening.

In so doing, we consider the following four different methods to do the equilibrium selection in this paper, and generate four different Nash-Q based pricing policies:

- $MESP$: For each Nash equilibrium, we compute A and B's payoff, and then select the Nash equilibrium which has the maximal sum of payoffs of A and B to update the Q value in the algorithm. When the algorithm is converged, we have the pricing policy, named $MESP$. Note that in this pricing policy, both providers need to coordinate each other to decide the equilibrium selection.
- BEP: Each provider selects the Nash equilibrium which can maximize its own payoff, but not the sum of both providers' payoffs. Each provider selects the equilibrium separately (and no pre-coordination is required), and then use the selected equilibrium action to update the Q value. When the algorithm is converged, we have the pricing policy, named BEP.
- $FP1$: We use fictitious play algorithm to approximate the probability of each Nash equilibrium appearing. A and B can coordinate with each other or select the equilibrium separately according to the probability. In this pricing policy, we assume that both providers select the same Nash equilibrium according to the probability. When the algorithm is converged, we have the pricing policy, named $FP1$.
- $FP2$: This pricing policy is similar to $FP1$. However, in equilibrium selection, each provider selects the Nash equilibrium separately according to the probability. In such a situation, the selected equilibrium action may be not from a paired Nash equilibrium. When the algorithm is converged, we have the pricing policy, named $FP2$.

Algorithm 2 Nash-Q learning

```
Input: pricing space $P$
Output: Nash equilibrium pricing policy of $A$ and $B$, ($\Pi_A$,$\Pi_B$)
1: for $\forall s \in S, and\ \forall a,b \in P, Q_A(s,a,b) = 0, Q_B(s,a,b) = 0$
2: repeat
3:     at the current state $s$, given $\epsilon$, generate a random number $rand$ ($0 < rand < 1$); when $rand \leq \epsilon$, $A$ chooses a price $a \in P$ randomly; when $rand > \epsilon$, $A$ chooses a price $a \in P$ according to the pricing policy $\Pi_A$. $B$ choose the price $b$ similarly. The next state is $s' = (a,b)$
4:     compute $A$'s immediate reward $r_{A,t}$, and $B$'s immediate reward $r_{B,t}$
5:     compute all Nash equilibria according to $Q_A(s')$ and $Q_B(s')$ based on Lemke-Howson algorithm
6:     $A$ and $B$ select Nash equilibrium according to the above four methods, which constitute equilibrium pricing action ($\Pi_A(s')$,$\Pi_B(s')$), and then compute the corresponding payoff $NashQ_{A,t}$ and $NashQ_{B,t}$ where $NashQ_{A,t} = \Pi_A(s') * \Pi_B(s') * Q_{A,t}(s')$, $NashQ_{B,t} = \Pi_A(s') * \Pi_B(s') * Q_{B,t}(s')$
7:     $Q_{A,t+1}(s,a,b) = (1-\alpha) * Q_{A,t+1}(s,a,b) + \alpha * (r_{A,t} + \gamma * NashQ_{A,t+1}(s'))$, $Q_{B,t+1}(s,a,b) = (1-\alpha) * Q_{B,t+1}(s,a,b) + \alpha * (r_{B,t} + \gamma * NashQ_{B,t+1}(s'))$
8: until ($\Pi_A(s,\cdot)$ and $\Pi_B(s,\cdot)$ are converged)
```

IV. Experimental Analysis

In this section, we run numerical simulations to evaluating pricing policies in different situations. In the evaluation, we consider winning percentage and average profits as the evaluation metrics, and repeat the experiments for 1000 times to do the average. In the following, we first describe the parameter setup in the experiments.

A. Experimental Parameters

First, we assume that each cloud provider has the same initial marginal cost, i.e. $c_{\cdot,0} = 5$, and the marginal cost is decreased as the demand increases. In addition, we assume that the set of allowable prices P chosen by cloud providers is $\{10, 20, ..., 100\}$. Furthermore, we assume that there are $N = 10000$ cloud users in total. The marginal values of users δ are independent random variable, and for illustrative purpose, we assume that they are drawn from a uniform distribution with support $[50, 150]$. Other parameters used in the following simulations follow the typical setting in the related literature, and are shown in Table I.

B. Evaluation

We first analyze the pricing policies trained by Nash-Q learning algorithm since Nash equilibrium is one of the most important solution concepts in game theory. We investigate the four different pricing policies generated from different equilibrium selection methods. We also evaluate Nash-Q based pricing policies with some practical pricing policies.

TABLE I
EXPERIMENTAL PARAMETERS

Parameter Setup	*Description*
$c_{\cdot,0} = 5$	cloud provider's marginal cost at the initial stage
$P = \{10, 20, ..., 100\}$	the set of allowable prices
$N = 10000$	the amount of users in the market
$\delta \in [50, 150]$	the cloud user's marginal value δ follows a uniform distribution supported on [50,150]
$\beta = 0.01$	parameter β in Equation 1
$\theta = 0.001$	parameter θ in Equation 1
$\xi = 0.8$	parameter ξ in Equation 4
$\epsilon = 0.2$	exploration rate in Q learning and minimax-Q learning algorithms
$\gamma = 0.8$	discount factor in Q learning and minimax-Q learning algorithms

According to different Nash equilibrium selection methods, we have four different Nash-Q based pricing policies, $MESP, BEP, FP1$ and $FP2$. Table II shows the winning percentage and standard deviation of the column-pricing policy competing against the row-pricing policy. From Table II we find that when pricing policy $MESP$ competes against BEP, the winning percentage of BEP is 46.7%. This means that $MESP$ is slightly better than BEP. When $MESP$ and BEP compete against $FP1$ or $FP2$, we also find that $MESP$ is slightly better than BEP. This may indicate that it is better for providers to pre-coordinate the equilibrium selection during the Nash-Q learning process, which may be infeasible in real world. From this table, we also find that $FP1$ and $FP2$ significantly beat $MESP$ and BEP. This means that in the equilibrium selection, it is more reasonable to select the Nash equilibrium according the probability of each equilibrium appearing. Furthermore, when $FP2$ competing $FP1$, we find that the winning percentage of $FP2$ is 54.4%, i.e. $FP2$ beats $FP1$. This means that when selecting the Nash equilibrium according to the probability separately (without any coordination), the pricing policy performs better. We also compare the average profit made by each pricing policy. The results are shown in Figure 1 We find that the winner of average profit is consistent with Table II. We can see that $FP1$ and $FP2$ make more profits than other pricing policies, and $FP2$ can beat $FP1$ as well in terms of average profits. All these experimental results suggest that when using Nash-Q learning algorithm, each provider should select the Nash equilibrium according to the probability. $FP2$ outperforming $FP1$ suggests that the provider does not need to coordinate with each other in the equilibrium selection (and actually no pre-coordination is more feasible in the realistic competing market).

We now evaluate the pricing policies with some practical pricing policies. In the real world, cloud providers usually attract cloud users by decreasing the price continuously. For example, when a fresh cloud provider enters the market, it may keep decreasing the price to attract users. We also evaluate these four pricing policies against the cloud provider which keeps reducing the price. Specifically, we consider two typical price reduction policies, Linear Reduction and Exp Reduction.

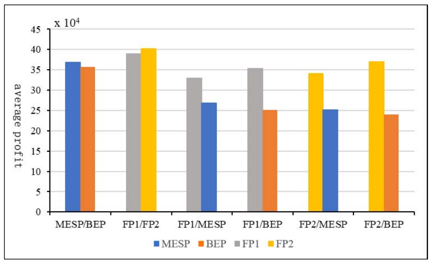

Fig. 1. Average profit of Nash-Q based pricing policies.

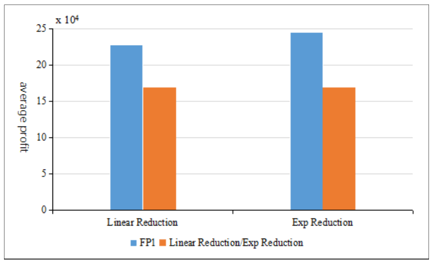

Fig. 2. Average profit of $FP1$ pricing policy competing with Linear and Exp Reduction pricing policies.

In Linear Reduction policy, the price decreases linearly with respect to time, where at stage t the price is $p_t = p_0 - 0.01t$ (p_0 is the initial price, and we set it as the maximal price, i.e. 100), while in Exp Reduction, the price decreases exponentially with time, where $p_t = p_0 * e^{-0.0003t}$ ($p_0 = 100$ which is the same as before). Since $FP1$ and $FP2$ outperform $MESP$ and BEP, in this experiment, we only evaluate $FP1$ and $FP2$ against Linear Reduction policy and Exp Reduction policy. The results are shown in Table III, and the average profits made by different pricing policies are shown in Figures 2 and 3. We find that $FP1$ and $FP2$ can significantly beat Linear Reduction and Exp Reduction pricing policies.

In addition to designing Nash-Q based pricing policy, we also design the pricing policy based on minimax-Q learning algorithm. We consider the case that the cloud provider takes minimax-Q learning algorithms against the opponent choosing actions randomly, and the case that both cloud providers are trained in minimax-Q learning algorithms. We name the trained pricing policies as MR and MM respectively, and for example MM means that both cloud providers adopt minimax-Q learning algorithm and are trained against each other. We then evaluate these two minimax-Q based pricing policies against Nash-Q based pricing policies. The results are shown in Table IV. We find that minimax-Q based pricing policies can significantly beat Nash-Q based pricing policies. For example, when $FP2$ competes against MR, the winning percentage of $FP2$ is only 27.5%, which is significantly less than MR. We believe this is in the minimax-Q learning algorithm, the cloud provider tries to maximize the payoff in the worst case. The competition between cloud providers is very fierce, and the payoff made by one provider is actually the loss of another provider. Therefore, the minimax solution

TABLE II
NASH-Q BASED PRICING POLICIES.

	$FP2$		$MESP$		BEP	
	win	standard dev.	win	standard dev.	win	standard dev.
vs. $FP1$	54.4%	0.011436	41.3%	0.010728	39.8%	0.009372
vs. $FP2$			28.8%	0.009720	27.9%	0.011393
vs. $MESP$					46.7%	0.012936

TABLE III
$FP1$ AND $FP2$ PRICING POLICIES VS. LINEAR REDUCTION AND EXP REDUCTION PRICING POLICIES.

	Linear Reduction		Exp Reduction	
	win	standard dev.	win	standard dev.
vs. $FP1$	25.6%	0.007316	26.0%	0.010690
vs. $FP2$	24.9%	0.006583	25.8%	0.008476

TABLE IV
NASH-Q BASED PRICING POLICIES VS. MINIMAX-Q BASED PRICING POLICIES.

	$FP1$		$FP2$		$MESP$		BEP	
	win	standard dev.	win	standard dev.	win	standard dev.	win	standard dev.
vs. MR	23.9%	0.006728	27.5%	0.010301	20.4%	0.007853	19.0%	0.003821
vs. MM	22.0%	0.007164	24.6%	0.007281	18.9%	0.005381	16.1%	0.006099

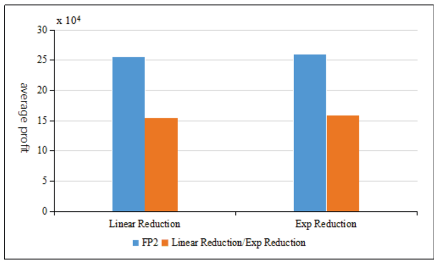

Fig. 3. Average profit of $FP2$ pricing policy cometing with Linear and Exp Reduction pricing policies.

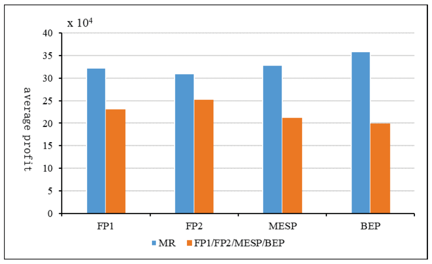

Fig. 4. Average profit of MR pricing policy competing with Nash-Q based pricing policies.

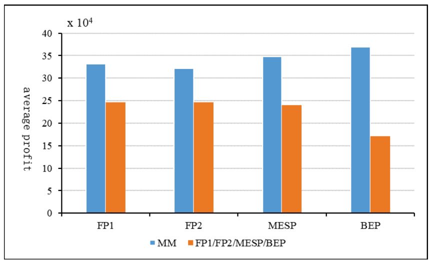

Fig. 5. Average profit of MM pricing policy competing with Nash-Q based pricing policies.

concept is more suitable in this competing setting.

We also compare the average profits made by cloud providers using minimax-Q based pricing policies and Nash-Q based pricing policies. The results are shown in Figures 4 and 5. We find that MM and MR can make more profits than Nash-Q based pricing policies. This future proves that minimax-Q based pricing policies perform better than Nash-Q based pricing policies, and may suggest that minimax solution concept is more suitable for analyzing the game of multiple cloud providers competing against each other.

V. CONCLUSIONS

How to set prices effectively is an important issue for the cloud provider, especially in the environment with multiple cloud providers competing against each other. In this paper, we use multi-agent reinforcement learning algorithms to address this issue. Specifically, we model the issue as a Markov game, and use two well-known solution concepts with Nash-Q and minimax-Q learning algorithms to design the pricing policies respectively. We then run extensive experiments to analyze the pricing policies. We find that when selecting Nash equilibrium according to the probability of appearing, the Nash-Q based pricing policies can beat other Nash equilibrium selection methods. Furthermore, when evaluating minimax-Q based pricing policies against Nash-Q based pricing policies, we find that minimax-Q based pricing policies can beat Nash-Q based pricing policies. This may suggest that in the competing environment, the cloud provider should adopt the pricing policy which can guarantee the maximal payoff in the worst case (i.e. using minimax solution concept). The experimental results provide useful insights on designing practical pricing policies for competing cloud providers.

Acknowledgment

This paper was funded by the National Natural Science Foundation of China (Grant No. 61402344) and the Fundamental Research Funds for the Central Universities(Grant No. WUT:2017II14XZ).

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing: clearing the clouds away from the true potential and obstacles posed by this computing capability," *Communications of The ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandicl, "Cloud computing and emerging it platforms: vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[3] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Grid Computing Environments Workshop*, 2008, pp. 1–10.

[4] R. Buyya, C. Yeo, and S. Venugopal, "Market-oriented cloud computing: vision,hype,and reality for delivering it services as computing utilities," in *The 10th IEEE International Conference on High Performance Computing and Communications*, 2008, pp. 5–13.

[5] G. Laatikainen, A. Ojala, and O. Mazhelis, "Cloud services pricing models," in *International Conference of Software Business*, 2013, pp. 117–129.

[6] B. Sharma, R. K. Thulasiram, P. Thulasiraman, S. K. Garg, and R. Buyya, "Pricing cloud compute commodities: a novel financial economic model," in *The 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2012, pp. 451–457.

[7] H. Wang, Q. Jing, R. Chen, B. He, Z. Qian, and L. Zhou, "Distributed systems meet economics: pricing in the cloud," in *The 2nd USENIX Conference on Hot Topics in Cloud Computing*, 2010, pp. 1–6.

[8] G. Mencagli, "A game-theoretic approach for elastic distributed data stream processing," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 11, no. 2, pp. 13:1–13:34, 2016.

[9] Y. Feng, B. Li, and B. Li, "Price competition in an oligopoly market with multiple iaas cloud providers," *IEEE Transactions on Computers*, vol. 63, no. 1, pp. 59–73, 2014.

[10] V. Kantere, D. Dash, G. Francois, S. Kyriakopoulou, and A. Ailamaki, "Optimal service pricing for a cloud cache," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 9, pp. 1345–1358, 2011.

[11] H. Xu and B. Li, "Maximizing revenue with dynamic cloud pricing: the infinite horizon case," in *IEEE International Conference on Communications*, 2012, pp. 2929–2933.

[12] D. Vengerov, "A gradient-based reinforcement learning approach to dynamic pricing in partially-observable environments," *Future Generation Computer Systems*, vol. 24, no. 7, pp. 687–693, 2008.

[13] H. Xu and B. Li, "Dynamic cloud pricing for revenue maximization," *IEEE Transactions on Cloud Computing*, vol. 1, no. 2, pp. 158–171, 2013.

[14] B. Xu, T. Qin, G. Qiu, and T.-Y. Liu, "Optimal pricing for the competitive and evolutionary cloud market," *The 24th International Joint Conference on Artificial Intelligence*, pp. 139–145, 2015.

[15] T. H. Tram and C. K. Tham, "A game-theoretic model for dynamic pricing and competition among cloud providers," in *The 6th International Conference on Utility and Cloud Computing*, 2013, pp. 235–238.

[16] T.-H. Tram and C.-K. Tham, "A novel model for competition and cooperation among cloud providers," *IEEE Transactions on Cloud Computing*, vol. 2, no. 3, pp. 251–265, 2014.

[17] J. Wal, *Stochastic dynamic programming*. Methematisch Centrum, 1980.

[18] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *11th International Conference on Machine Learning*, 1994, pp. 157–163.

[19] J. Hu and M. P. Wellman, "Nash q-learning for general-sum stochastic games," *Journal of Machine Learning Research*, vol. 4, pp. 1039–1069, 2003.

[20] I. F. Adams, D. D. E. Long, E. L. Miller, S. Pasupathy, and M. W. Storer, "Maximizing efficiency by trading storage for computation," in *The 1st USENIX Conference on Hot Topics in Cloud Computing*, vol. 7, 2009.

[21] H. Jung and C. M. Klein, "Optimal inventory policies under decreasing cost functions via geometric programming," *European Journal of Operational Research*, vol. 132, no. 3, pp. 628–642, 2001.

[22] K. E. Train, *Discrete choice methods with simulation.* Cambridge University Press, 2003.

[23] C. E. Lemke and J. T. Howson, "Equilibrium points of bimatrix games," *Journal of the Society for Industrial and Applied Mathematics*, vol. 12, no. 2, p. 413C423, 1964.

[24] G. W. Brown, "Iterative solutions of games by fictitious play," *Activity Analysis of Production and Allocation*, 1951.

[25] von Neumann J. and G. W. Brown, "Solutions of games by differential equations," *Contributions to the Theory of Games*, pp. 73–79, 1950.