Meta-game Equilibrium for Multi-agent Reinforcement Learning

Yang Gao[1], Joshua Zhexue Huang[2], Hongqiang Rong[2], and Zhi-Hua Zhou[1]

[1] National Laboratory for Novel Software Technology,
Nanjing University, Nanjing 210093, China
{gaoy, zhouzh}@nju.edu.cn
[2] E-Business Technology Institute,
The University of Hong Kong, Hong Kong, China
{jhuang, hrong}@eti.hku.hk

Abstract. This paper proposes a multi-agent Q-learning algorithm called meta-game-Q learning that is developed from the meta-game equilibrium concept. Different from Nash equilibrium, meta-game equilibrium can achieve the optimal joint action game through deliberating its preference and predicting others' policies in the general-sum game. A distributed negotiation algorithm is used to solve the meta-game equilibrium problem instead of using centralized linear programming algorithms. We use the repeated prisoner's dilemma example to empirically demonstrate that the algorithm converges to meta-game equilibrium.

1 Introduction

Recently there have been growing interests in extending reinforcement learning to the multi-agent domain. Based on the Markov (or stochastic) game models, many multi-agent reinforcement learning algorithms have been proposed. Littman suggested the minimax-Q learning algorithm for zero-sum stochastic games [5]. A second approach was pursued by Claus and Boutilier to deal with common-payoff stochastic games[1]. Hu et al. in 1998 made a pivotal contribution by introducing Nash-Q learning to general-sum games[3][4]. Littman replaced Nash-Q learning by Friend-and-Foe-Q learning in some special stochastic games[6]. Furthermore, Greenwald et al. introduced the correlated equilibrium concept and proposed CE-Q learning to generalize both Nash-Q and Friend-and-Foe-Q learning methods[2].

Shoham et al. have raised the question of the justification of using Nash equilibrium in multi-agent setting[7]. To answer this question, we think that Nash equilibrium is not optimal in general-sum games. In dealing with the collective rationality, new solutions can be adopted to replace the Nash equilibrium. When agents can consider their own preferences and predict actions of other agents correctly, they can reach meta-game equilibrium that is the optimal joint policy in the general-sum game. Based on this concept, we discuss the meta-game equilibrium and introduce the meta-game-Q learning algorithm in this paper.

G.I. Webb and Xinghuo Yu (Eds.): AI 2004, LNAI 3339, pp. 930–936, 2004.

In the next section we briefly review Markov game and multi-agent reinforcement learning. In Section 3, we introduce the meta-game equilibrium concept. Then, we present a distributed negotiation algorithm to solve the meta-game equilibrium problem in the general-sum game in Section 4. In Section 5, we discuss our experimental results. Finally, in Section 6 we draw some conclusions.

2 Multi-agent Reinforcement Learning

Game theory is one of the most important mathematical foundations to formulate a multi-agent system. When we use the games to model a multi-agent system, all discrete states in the MDP model are regarded as some distinguishing games. Therefore, the immediate rewards of one agent received from the world not only depend on its own action chosen by itself, but also depend on other actions by other agents. As a result, the single agent reinforcement learning algorithm fails in the multi-agent domain.

When the probability transitions between different games satisfy the Markov property, the MDP model for single agent can be generalized to the Markov game for the multi-agent system.

Definition 1. A Markov game is a tuple $\langle I, S, (A_i(s))_{s\in S, 1\le i\le n}, T, (R_i)_{1\le i\le n}\rangle$, where I is a set of n agents, S is a finite set of states of the world, $A_i(s)$ is a finite set of the ith agent's actions at state s, T is the probability function that describes state-transition conditioned on past states and joint actions, and $R_i(s, \vec{a})$ is the ith agent's reward for state $s \in S$ and joint actions $\vec{a} \in A_1(s) \times \ldots \times A_n(s)$[2].

In order to find the optimal action sequence policy $\pi : S \rightarrow \vec{a}$, a state-action value function $Q_i^\pi(s, \vec{a})$ is defined as the agent i's value of taking action $\vec{a}$ in state s under a policy π, i.e.,

$$Q_i^\pi(s, \vec{a}) = E_\pi\{\Sigma_{j=0}^{\infty}\gamma^j r_i(s_{j+1})\} \tag{1}$$

where γ is a discounter factor. The optimal policy Q_i^* is defined as

$$Q_i^*(s, \vec{a}) = \max_\pi Q_i^\pi(s, \vec{a}) \tag{2}$$

Because the parameters of the Markov game model are unknown, the agent can only get its experiences by trial-and-error to approximate the optimal policy through reinforcement learning. In the Markov games, the ith agent's Q-values are updated on states and the action-vector $\vec{a}$ as

$$Q_i(s, \vec{a}) = Q_i(s, \vec{a}) + \alpha[r_i(s, \vec{a}) + \gamma \max_{\vec{a'}\in A'} Q_i(s', \vec{a'}) - Q_i(s, \vec{a})] \tag{3}$$

3 Meta-game Equilibrium

Fig. 1 shows the Prisoner's Dilemma game which is an abstraction of social situations where each agent is faced with two alternative actions: *c*ooperation and

Agent 2 \ Agent 1	d	c
d	(r_1=-9,r_1=-9)	(r_3=0,r_2=-10)
c	(r_2=-10,r_3=0)	(r_4=-1,r_4=-1)

Fig. 1. Prisoner's Dilemma game G

Agent 2 \ Agent 1	f_1	f_2	f_3	f_4
d	(-9,-9)	(0,-10)	(-9,-9)	(0,-10)
c	(-10,0)	(-1,-1)	(-1,-1)	(-10,0)

Fig. 2. Meta-game 1G of Prisoner's Dilemma game G

*d*efecting. Prisoners will receive different payoffs r_1, r_2, r_3, r_4 for different combinations of actions, where $r_3 > r_4 > r_1 > r_2, 2r_4 > r_2 + r_3 > 2r_1$. It is well-known that the joint policy (d, d) holds Nash equilibrium in the Prisoner's Dilemma problem. However, the optimal joint policy is (c, c) because every prisoner can get more rewards than the rewards under Nash equilibrium (d, d). From the Prisoner's Dilemma game, we can see that the combination of individual agents' optimal policies may not be the optimal joint policy of the entire multi-agent system in the general-sum game. In other words, Nash equilibrium may not be the optimal strategy if collective rationality is considered.

When the agents' payoff for different action combinations become common knowledge, one agent can get the optimal policy of the entire system by means of revising its own policy through deliberating its preference and predicting others' policies. This approach is the most important principle of the meta-game theory.

Meta-game is a hypothetical game derived from the original game situation by assuming that other agents have taken their actions first. Meta-game can be presented as an extended strategic form. When extending the ith agent's strategy to a function of other agents' strategies in game G, the meta-game K_iG is constructed, where K_i is the sign of the ith agent. Obviously, the recursive meta-game can be derived from the meta-game too. Fig. 2 presents the meta-game 1G as an extended form game in Fig. 1.

In this extended form, agent 1 has four different actions, f_1, f_2, f_3 and f_4. $f_1(*) =' d'$ means that agent 1 always chooses action $'d'$ regardless of agent 2's action. Similar to f_1, the second action of agent 1 is $f_2(*) =' c'$. The third action $f_3(*) =' *'$ means that agent 1 chooses the same action as agent 2's. If agent 1 always chooses the action opposite to the agent 2's action, it is $f_4(*) =' \neg *'$. So, the game shifts to the new stable equilibrium called meta-game equilibrium if all agents can predict other's actions correctly.

Definition 2. In a multi-agent system with n agents, the meta-game $12 \ldots nG$, or $n(n-1) \ldots 21G$, or one meta-game whose prefix is any kind of permutation of $1, 2, \ldots, n$ is the complete game of the origin game G.

Definition 3. A joint policy $\vec{a}_m = (a_{1,m}, a_{2,m}, \ldots, a_{n,m})$ is called meta-game equilibrium in a complete game $K_1K_2 \ldots K_nG$ if every agent satisfies

$$\min_{a_{P_i}} \max_{a_i} \min_{a_{F_i}} R_i(a_{P_i}, a_i, a_{F_i}) \leq R_i(a_{P_i,m}, a_{i,m}, a_{F_i,m}) \tag{4}$$

where P_i is the set of the agents listed in front of the sign i and F_i is the set of the agents listed after the sign i in prefixes $K_1K_2 \ldots K_n$[8].

4 Meta-game-Q Reinforcement Learning

In many multi-agent settings, the reward function and the probability-transition function are unknown in advance. We cannot find the optimal Q-value by means of linear programming directly. Every agent needs to approximate the correct Q-value through trial-and-error. As shown in Eq. 5, agents must update their current Q-values with meta-game equilibrium.

$$Q_i(s, \vec{a}) = Q_i(s, \vec{a}) + \alpha[r_i(s, \vec{a}) + \gamma Q_i(s', \vec{a}_m\ (s')) - Q_i(s, \vec{a})] \tag{5}$$

We cannot use a centralized algorithm to compute any equilibrium in multi-agent learning since each agent cannot know anything about other agents' rewards in advance. Instead, we have designed a distributed negotiation algorithm for meta-game-Q learning. Assume that there are only two agents a and b in a multi-agent system. The algorithm of agent a is given Table 1, where $Q_a(s, a, b)$ is agent a's current Q-value after it chooses the joint policy (a, b) in state s.

Table 1. Negotiation algorithm for solving meta-game equilibrium

Initial $J_a = NULL$;
Step1: agent a chooses a joint policy $(a, b) \notin J_a$;
Step2: If $Q_a(s, a, b) \geq max_{a' \in A} Q_a(s, a', b)$, $J_a = J_a + (a, b)$, return step1; else, record (a', b), goto step3;
Step3: Agent a broadcasts the message to agent b, ask agent b to judge, $Q_b(s, a', b) \geq max_{b' \in B}(s, a', b')$.
Step3.1: If it is satisfied, agent b returns 'SUCCESS MESSAGE' to agent a. After agent a receives the 'SUCCESS MESSAGE', $J_a = J_a + (a, b)$, return step1.
Step3.2: If it isn't satisfied, agent b return 'FAIL MESSAGE' to agent a. After agent a receives the 'FAIL MESSAGE', record the (a', b'), goto step4.
Step4: Agent a judges whether $Q_a(s, a', b') \geq Q_a(s, a, b)$. If it is satisfied, return step1, else, return $J_a = J_a + (a, b)$ and return step1.
Step5: When all agents get their final joint action sets, the meta-game equilibrium could be obtained through computing the intersection of all joint action sets, viz. $J_a \cap J_b$.

A template for meta-game-Q reinforcement learning is presented in Table 2.

5 Experimental Results and Analysis

We used the repeated prisoner's dilemma(RPD) game to test our meta-game-Q reinforcement learning algorithm. The RPD game consists of ten independent prisoner's dilemma games. The immediate reward of each independent prisoner's

Table 2. Algorithm for meta-game-Q reinforcement learning

Step1: Initialization. $\forall s \in S, \forall i \in I, \forall \vec{a}, Q_i(s, \vec{a}) = 0; \alpha = 1, \gamma = 0.9, Pr = 0.9$;
Step2: Select action. The ith agent chooses the action $a_{i,m}$ with probability Pr, or randomly chooses any other action with probability $1 - Pr$;
Step3: Observation. The ith agent observes the reward r_i and next state s';
Step4: Negotiation to get the meta-game equilibrium $\vec{a}_m(s')$ at s';
Step5: Learning. Update the Q-value $Q_i(s, \vec{a})$ using Eq.5.
Step6: Adjust the learning rate α and selection factor Pr. And return step2 until end.

dilemma game is given in Fig. 1. The state transitions between each independent game is deterministic. The value of the game for one agent is defined as its accumulated reward when both agents follow their meta-game equilibrium strategies in Fig. 3.

$$\begin{array}{cc} & \text{Agent 1} \\ & \begin{array}{cc} d & c \end{array} \\ \text{Agent 2} \begin{array}{c} d \\ c \end{array} & \left| \begin{array}{cc} r_1 + \sum_{j=1}^{10-i} \gamma^j r_4, r_1 + \sum_{j=1}^{10-i} \gamma^j r_4 & r_3 + \sum_{j=1}^{10-i} \gamma^j r_4, r_2 + \sum_{j=1}^{10-i} \gamma^j r_4 \\ r_2 + \sum_{j=1}^{10-i} \gamma^j r_4, r_3 + \sum_{j=1}^{10-i} \gamma^j r_4 & r_4 + \sum_{j=1}^{10-i} \gamma^j r_4, r_4 + \sum_{j=1}^{10-i} \gamma^j r_4 \end{array} \right| \end{array}$$

Fig. 3. The Q-value matrix of the ith game in the repeated prisoner's dilemma game, where the sum of the game's number is 10 and $r_1 = -9, r_2 = -10, r_3 = 0, r_4 = -1$

Similar to the single prisoner's dilemma game, the joint policy (c, c) is the optimal solution in RPD because of $r_3 + \sum_{j=1}^{10-i} \gamma^j r_4 > r_4 + \sum_{j=1}^{10-i} \gamma^j r_4 > r_1 + \sum_{j=1}^{10-i} \gamma^j r_4 > r_2 + \sum_{j=1}^{10-i} \gamma^j r_4, 2r_4 + 2\sum_{j=1}^{10-i} \gamma^j r_4 > r_2 + r_3 + 2\sum_{j=1}^{10-i} \gamma^j r_4 > 2r_1 + 2\sum_{j=1}^{10-i} \gamma^j r_4$. All Q-values of the matrix of the RPD game are unknown before agents begin to learn the optimal action sequence.

We ran 10 trails and calculated the difference between the current Q-value and the optimal Q-value in Fig. 3. In our experiment, we employed a training period of 100 episodes. The performance of the test period was measured by the Q-value difference when agents followed their learned strategies, starting from the first prisoner's dilemma game to the last game. The experimental result for RPD is shown in Fig. 4. From Fig. 4, we can see that when both agents were meta-game-Q learners and followed the same meta-game updating rule, they ended up with the meta-game equilibrium 100% of the time.

6 Conclusions

In this paper, we have discussed algorithms for learning optimal Q-values in the Markov game, given the meta-game equilibrium solution concept. Different

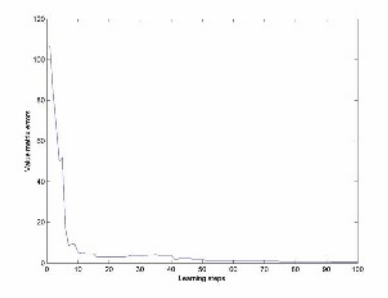

Fig. 4. On-line performance of meta-game-Q agents in RPD

from the Nash-Q learning algorithm, we have used the meta-game equilibrium instead of Nash equilibrium in the general-sum game. Specifically, we have replaced the centralized linear programming algorithms with a distributed negotiation algorithm to solve the meta-game equilibrium under incomplete common knowledge. This adaptive meta-game-Q reinforcement learning algorithm can learn the meta-game equilibrium in Markov game.

Acknowledgements

The paper is supported by the Natural Science Foundation of China (No.6010301 2), the National Outstanding Youth Foundation of China (No.60325207), the National Grand Fundamental Research 973 Program of China (No.2002CB312002) and the Natural Science Foundation of Jiangsu Province, China(No.BK2003409). The comments and suggestions from the anonymous reviewers greatly improved this paper.

References

1. Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the Fifteenth National Conference on Artificail Intelligence*, pages 746–752, 1998.
2. Amy Greenwald, Keith Hall, and Roberto Serrano. Correlated-q learning. In *Proceedings of the Twentieth International Conference on*, pages 242–249, Washington DC, 2003.
3. Junling Hu and Michael P. Wellman. Multiagent reinforcement learning: theoretical framework and an algorithm. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 242–250, 1998.
4. Junling Hu and Michael P. Wellman. Nash q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4:1039–1069, 2003.
5. Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Eleventh International Conference on Machine Learning*, pages 157–163, New Brunswick, 1994.

6. Michael L. Littman. Friend-or-foe q-learning in general-sum games. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 322–328. Williams College, Morgan Kaufman, June 2001.
7. Yoav Shoham, Rob Powers, and Trond Grenager. Multi-agent reinforcement learning: a critical survey. Technical report, Stanford University, 2003.
8. L. C. Thomas. *Games, Theory and Applications*. Halsted Press, 1984.