

A multi-agent reinforcement learning method with the estimation of the other agent's actions

Yasuo Nagayuki^{†1} Shin Ishii^{†1†4†5} Minoru Ito^{†1} Katsunori Shimohara^{†2†4} Kenji Doya^{†3†5†1}

^{†1} Graduate School of Information Science, Nara Institute of Science and Technology
8916-5, Takayama-cho, Ikoma-shi, Nara 630-0101, Japan

^{†2} NTT Communication Science Laboratories
2-4, Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0237, Japan

^{†3} ATR International

^{†4} ATR Human Information Processing Research Laboratories
^{†5} CREST, Japan Science and Technology Corporation
2-2, Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0288, Japan

Abstract¹

The application of reinforcement learning to multi-agent systems has attracted recent attention. In a multi-agent environment, whether one agent's action is good or not depends on the other agents' actions. In traditional reinforcement learning methods, which are based on a stationary environment, it is hard to take account of the other agent's actions which may dynamically vary due to learning. In this article, we consider a two-agent cooperation problem, and propose a multi-agent reinforcement learning method based on the estimation of the other agent's actions. In our learning method, one agent estimates the other agent's action based on the internal model of the other agent. The internal model is acquired by the observation of the other agent's actions. Through experiments, we demonstrate that good cooperative behaviors are achieved by the use of the internal model of the other agent.

keywords: reinforcement learning, Q-learning, multi-agent systems, internal model, action estimation, cooperation problem.

1 Introduction

The realization of behaviors in multi-agent systems is an interesting topic from the viewpoint of engineering and cognitive science. In particular, reinforcement

learning [1] of behaviors has attracted recent attention because of its adaptability to dynamic environments. Reinforcement learning has been applied to multi-agent problems such as pursuit games [2, 3, 4], soccer [5, 6, 7], prisoner's dilemma games [8], and coordination games [9, 10, 11].

Traditional reinforcement learning methods have been developed in a single-agent environment which is modeled as a Markov decision process (MDP). In an MDP, the transition of an environment's state is defined by a transition probability function that does not change with time. When considering a multi-agent environment in which the agents autonomously learn, the transition function often changes with time, because the policies of the other agents, which are regarded as a part of the environment, change according to their learning. Thus, the environment cannot be modeled as an MDP. In many multi-agent reinforcement learning studies, however, reinforcement learning methods based on the MDP are applied without much modification. In other words, those approaches ignore the difference between learning agents and a stationary environment.

In this article, we propose a multi-agent reinforcement learning method that explicitly takes the policy of the other agents into account. We consider a multi-agent environment with two agents and assume that they synchronously execute their actions and there is no inter-agent communication except for the observation of each other's actions. In our learning method, each agent learns to take actions based

¹Paper for AROB2000 (Fifth International Symposium on Artificial Life and Robotics)

on the estimation of the other agent’s action. For this purpose, each agent updates an internal model of the other agent’s policy using the observation of the other agent’s actions. We experimentally evaluate our learning method by applying it to a variant of the pursuit problem [12], which is a typical multi-agent cooperation problem.

2 Markov decision process and Q-learning

Our multi-agent reinforcement learning method is based on Q-learning [13]. Q-learning was originally defined to deal with an MDP, and hence it should be used in an environment including only one learning agent.

An MDP is defined by a set of finite states of the environment, S , and a set of finite actions, A . At each time step, an agent observes a state $s(\in S)$, executes an action $a(\in A)$ and receives a reward r from the environment. The state transition of the environment is modeled by a transition probability function:

$$P_{ss'}^a \equiv \Pr(s'|s, a), \quad (1)$$

where $\Pr(s'|s, a)$ is the probability that the environment changes to a new state $s'(\in S)$ when the agent executes action a in state s . The reward r is a variable, possibly probabilistic, which depends only upon the current state s and action a .

Q-learning [13] is an incremental reinforcement learning method. According to Q-learning, the agent selects an action based on an action value function (called the Q-function), $Q(s, a)$, which defines the expected sum of the discounted reward attained when executing action $a(\in A)$ in $s(\in S)$, and the subsequent actions are determined by the current policy. The Q-function is updated using the agent’s experience. The learning flow is as follows:

1. For the current state s , the agent executes an action $a_i(\in A)$ with the probability:

$$\pi[s, a_i] = \frac{e^{Q(s, a_i)/T}}{\sum_{a_k \in A} e^{Q(s, a_k)/T}}. \quad (2)$$

T is the parameter called the temperature, which determines the randomness of the stochastic policy (2). Although several methods for selecting an action have been proposed, we adopt this Boltzmann selection method.

2. The agent executes action a selected in step 1, and the environment changes to a new state s'

according to $P_{ss'}^a$. After a reward r is given from the environment, the Q-function is updated as follows:

$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha(r + \gamma \max_{a' \in A} Q(s', a')), \quad (3)$$

where $\alpha(0 \leq \alpha < 1)$ is the learning rate and $\gamma(0 \leq \gamma \leq 1)$ is the discount factor.

3. If the new state s' satisfies a terminal condition, then a single episode ends. Otherwise let $s' \rightarrow s$ and go back to step 1.

When the temperature T is very small and the environment is an MDP (namely, $P_{ss'}^a$ is fixed), the above Q-learning converges and the optimal Q-function is obtained [13].

3 Multi-agent reinforcement learning

In this study, we consider a multi-agent environment that consists of learning agents and a stationary environment. Such an environment cannot be formulated as an MDP, because the transition probability $P_{ss'}^a$ changes over time due to the learning of the other agents. In such a case, the original Q-learning is not appropriate. In this article, we intend to extend Q-learning so as to deal with such a multi-agent environment. As a step in this direction, we propose a multi-agent reinforcement learning method based on the estimation of the other agents’ action which may dynamically vary due to learning.

We consider a multi-agent environment with two agents and assume that they synchronously execute their actions and there is no inter-agent communication except for the observation of each other’s actions.

In order to explicitly express the other agent actions, the Q-function is rewritten as $Q(s, a_{self}, a_{other})$. Here, $s(\in S)$ is an environment state that involves the two agents, $a_{self}(\in A_s)$ is an action of the agent under consideration and $a_{other}(\in A_o)$ is the other agent’s action. A_s and A_o are the sets of actions which are possible for the agent under consideration and the other agent, respectively.

3.1 Estimation of internal model

The new definition of the Q-function clarifies that a_{other} is a hidden variable in our environment. In our reinforcement learning method, a_{other} is estimated based on the internal model $I(s, a_{other})$ of the other agent’s policy, which is estimated from the past observation of the other agent’s actions.

When the agent observes that the other agent executes an action $a_{other}^* (\in A_o)$ in s , function $I(s, a_{other})$ is updated for every executable action $a_{other} (\in A_o)$ in s :

$$I(s, a_{other}) \leftarrow (1-\theta)I(s, a_{other}) + \begin{cases} \theta & (a_{other} = a_{other}^*) \\ 0 & (\text{otherwise}) \end{cases}, \quad (4)$$

where $\theta (0 \leq \theta < 1)$ is the parameter that controls the effect of the actual action a_{other}^* . If the other agent's policy is fixed and parameter θ decreases as an inverse of the number of observations, $I(s, a_{other})$ converges to the empirical probability that the agent executes a_{other} in s [1]. Since the other agent's policy changes over time due to learning, the learning of $I(s, a_{other})$ puts emphasis on recent observation, by using a constant for θ .

3.2 Reinforcement learning process

Here, we propose a multi-agent reinforcement learning method based on an estimation of the other agent's policy. The learning flow is as follows:

1. The agent under consideration observes the current state s , estimates the other agent's action a_{other} by $I(s, a_{other})$, and selects an action a_{self} based on its stochastic policy:

$$\pi[s, a_{self_i}] = \frac{e^{\bar{Q}(s, a_{self_i})/T}}{\sum_{a_{self_k} \in A_s} e^{\bar{Q}(s, a_{self_k})/T}} \quad (5)$$

$$\begin{aligned} \bar{Q}(s, a_{self}) &\equiv \langle Q(s, a_{self}, a_{other}) \rangle_{I(s, a_{other})} \\ &= \sum_{a_{other} \in A_o} I(s, a_{other}) Q(s, a_{self}, a_{other}), \end{aligned} \quad (6)$$

where $\bar{Q}(s, a_{self})$ is the expected value of the Q-function with respect to $I(s, a_{other})$.

2. The agent executes the action a_{self} selected in step 1. Here, the other agent synchronously executes an action a_{other}^* . The environment changes to a new state s' and the agent receives a reward r from the environment. After that, function $I(s, a_{other})$ is updated according to equation (4), and $Q(s, a_{self}, a_{other})$ is updated as follows:

$$\begin{aligned} Q(s, a_{self}, a_{other}^*) &\leftarrow (1 - \alpha)Q(s, a_{self}, a_{other}^*) \\ &\quad + \alpha(r + \gamma \max_{a'_{self} \in A_s} \bar{Q}(s', a'_{self})). \end{aligned} \quad (7)$$

3. If the new state s' satisfies a terminal condition, then a single episode ends. Otherwise let $s' \rightarrow s$ and go back to step 1.

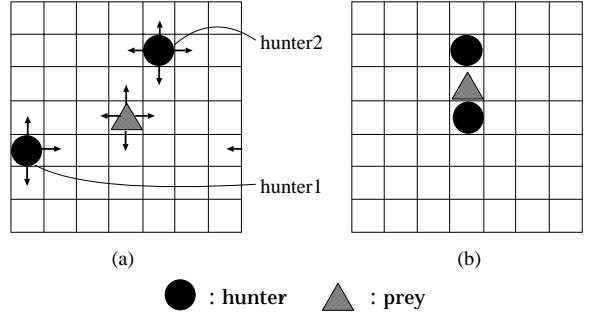


Figure 1: (a) Grid world of pursuit problem: Arrows show the directions which prey/hunters can move to, (b) An example of goal states.

4 Experiments and results

4.1 Pursuit problem

In order to evaluate our multi-agent reinforcement learning method, we prepared a variant of the well-known pursuit problem [12], which is a typical testbed for multi-agent cooperation algorithms.

- In a 7×7 toroidal grid world, two hunter agents and a prey exist, as shown in Figure 1(a). At the beginning of each episode, they are placed at random.
- At a discrete time step in an episode, the prey and the two hunters each synchronously executes one out of five actions: moving up, down, left, or right from the current position, or staying in the current position.
- The hunters perceive the current state, which is represented by the location of the prey relative to each hunter. For example, the state shown in Figure 1(a) is represented as $s = ([3, 1], [-1, -2])$, where, $[3, 1]$ and $[-1, -2]$ are the relative locations of hunter1 and hunter2 to the prey, respectively. The number of possible $s (\in S)$ is $2304 (= 48 \times 48)$.
- When the two hunters capture the prey, the episode ends. The capture is achieved when the two hunters are positioned on both sides of the prey as shown in Figure 1(b).
- The prey's actions are independent of the hunters' positions. The probability of moving up, down or right is $\frac{1}{3}$, and the probability of the other actions, i.e., moving left or staying, is 0. This action definition implies that the state transition of this stationary environment is stochastic.

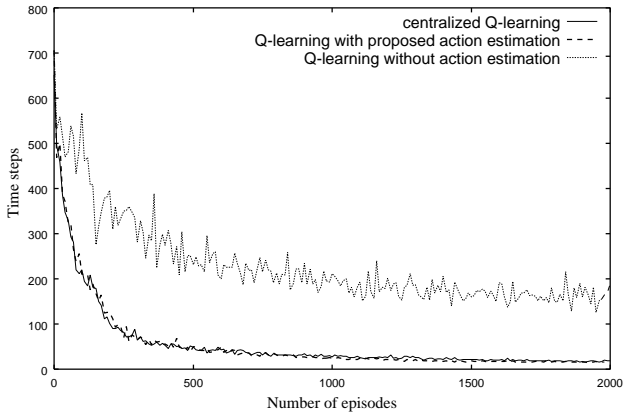


Figure 2: Time steps to capture the prey. Abscissa denotes number of learning episodes, and ordinate denotes average time steps to capture the prey. After every ten learning episodes, 100 evaluation episodes that vary the initial placements are done, and this figure shows their average.

4.2 Experimental results

Here, a “centralized Q-learning” (CQ) method is defined in order to evaluate our proposed learning method. In CQ, a centralized learning component is able to observe the whole grid world, and control the two hunter agents. The learning scheme is the usual Q-learning. Namely, its flow is almost identical to the one described in Section 2, except that an action is given by $\mathbf{a} = (a^1, a^2)$, where, a^1 and a^2 are the actions of hunter1 and hunter2, respectively. In this CQ, the transition of the environment’s states is stochastic, thus the environment can be represented as an MDP.

Figure 2 shows the average time steps needed to capture the prey. The parameters for the learning are: $\alpha = 0.3, \gamma = 0.9, T = 0.1$ and $\theta = 0.1$. The reward is designed such that if the two hunters capture the prey, then $r = 1.0$, else $r = 0$. The initial values of the Q-function are 0, and the initial values of $I(s, a_{other})$ are 0.2. In figure 2, the label “Q-learning without action estimation” (QwoAE) denotes a method in which $I(s, a_{other}) = 0.2$ for every s and a_{other} ; namely, the estimation of the other agent’s action is completely random. Figure 2 shows that the cooperative behaviors are well acquired by CQ or Q-learning with action estimation (QwAE), while QwoAE does not work well.

Figure 3 shows the average time steps needed to capture the prey where the two hunters have reward functions different from each other. One hunter re-

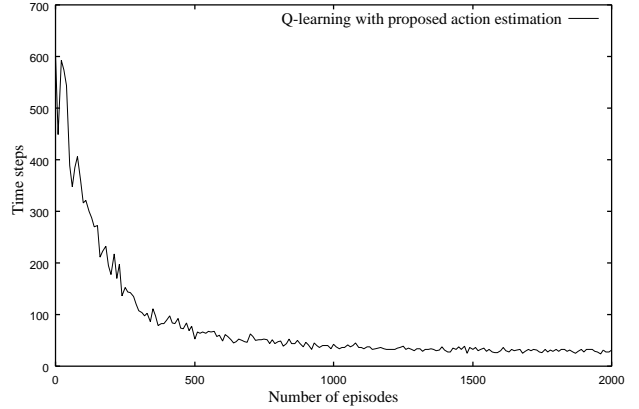


Figure 3: Time steps needed to capture the prey. One hunter receives $r = 1.0$ when capturing, $r = -0.01$ otherwise, and the other hunter receives $r = 0.5$ when capturing, $r = 0$ otherwise.

ceives $r = 1.0$ when it captures the prey, $r = -0.01$ otherwise. The other hunter receives $r = 0.5$ when capturing, $r = 0$ otherwise. The other parameters are the same as those in the previous experiment. It should be noted that it is difficult for CQ to deal with such heterogeneous agents.

5 Discussion

By comparing QwAE with QwoAE, the advantage of our action estimation method is obvious. In QwoAE, although the hunters learn to approach the prey, they do not learn cooperative capture movements. The capture thus occurs by chance.

By comparing CQ with QwAE, the learning performance of QwAE is similar to that of CQ. CQ is expected to learn effectively, because the environment can be represent as an MDP. In QwAE, on the other hand, there exists a hidden variable, i.e., the other agent’s action, and the environment cannot be represented as an MDP. If the estimation of the hidden variable is not good, the learning will not proceed well, such as QwoQE. The similarity of the learning performance between CQ and QwAE implies that our estimation method works effectively.

In actual situations, the environment is often heterogeneous among agents. We believe such heterogeneity induces self-organization in multi-agent systems. An important factor of this heterogeneity is the difference in rewards given to the agents. In central-

ized reinforcement learning, however, such an environment is hard to be considered. The result shown in Figure 3 implies that our new learning scheme is able to deal with such an environment.

6 Conclusion

In this article, we proposed a multi-agent reinforcement learning method based on an estimation of the other agent's action. In our learning method, one agent estimates the other agent's action based on the internal model of the other agent. The internal model is acquired by the observation of the other agent's actions. When applied to the pursuit problem, the experimental results showed that good cooperation behaviors were achieved by our new learning method. Furthermore, we showed that our new learning method could be applied to a heterogeneous environment, in which the rewards given to the agents were different from each other.

References

- [1] R.S. Sutton, and A.G. Barto. *Reinforcement Learning: An Introduction*, MIT Press, 1998.
- [2] S. Arai, K. Miyazaki, and S. Kobayashi. "Methodology in multi-agent reinforcement learning – approaches by Q-learning and profit sharing – ". *Journal of Japanese Society for Artificial Intelligence*, Vol.13, No.4, pp.609-618, 1998. (in Japanese)
- [3] N. Ono, and K. Fukumoto. "Multi-agent reinforcement learning: a modular approach". In *Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS-96)*, pp.252-258, 1996.
- [4] M. Tan. "Multi-agent reinforcement learning: independent vs. cooperative agents". In *Proceedings of the Tenth International Conference on Machine Learning (ICML-93)*, pp.330-337, 1993.
- [5] T. Balch. "Learning roles: behavioral diversity in robot teams". In *Collected papers from the AAAI-97 workshop on multiagent learning*, AAAI Press, 1997.
- [6] M.L. Littman. "Markov games as framework for multi-agent reinforcement learning". In *Proceedings of the Eleventh International Conference on Machine Learning (ICML-94)*, pp.157-163, 1994.
- [7] R. Salustowicz, M. Wiering, and J. Schmidhuber. "Learning team strategies: soccer case studies". *Machine Learning*, Vol.33, No.2/3, pp.263-282, 1998.
- [8] T.W. Sandholm, and R.H. Crites. "Multiagent reinforcement learning in the iterated prisoner's dilemma". *Biosystems*, Vol.37, pp.147-166, 1995.
- [9] C. Claus, and C. Boutilier. "The dynamics of reinforcement learning in cooperative multiagent systems". In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, pp.746-752, 1998.
- [10] J. Hu, and M.P. Wellman. "Multiagent reinforcement learning: theoretical framework and an algorithm". In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML-98)*, pp.242-250, 1998.
- [11] S. Sen, M. Sekaran, and J. Hale. "Learning to coordinate without sharing information". In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, pp.426-431, 1994.
- [12] M. Benda, V. Jagannathan, and R. Dodhiawalla. "On optimal cooperation of knowledge sources ". *Technical Report BCS-G2010-28*, Boeing AI Center, 1985.
- [13] C.J.C.H. Watkins, and P. Dayan. "Technical note Q-learning". *Machine Learning*, Vol.8, No.3, pp.279-292, 1992.