Multi-agent Reinforcement Learning based on Multi-channel ART Networks

Hitomi Morishita, Hiroaki Ueda and Kenichi Takahashi
Graduate School of Information Sciences, Hiroshima City University, Hiroshima, 731-3194, Japan

Keywords: Multi-agent Reinforcement Learning, FALCON, Multi-channel ART Networks, Hearts.

Abstract: 3-channel fuzzy ART network *FALCON* is a good solution to combine reinforcement learning with state segmentation, where it learns the relations among percepts, actions and rewards. FALCON, however, does not have a mechanism to predict behavior of other agents, and thus it is difficult for FALCON to learn the optimal agent's behavior in a multi-agent circumstance. In this paper, an action prediction module based on 2-channel fuzzy ART network is proposed, and FALCON is modified in order to be able to register the output of the action prediction module. The modified FALCON is called *FALCON_AP*. Moreover, *FALCON_ER* that estimates the expected value of rewards and selects an action according to the value is proposed. Through experiments in which FALCON, FALCON_AP and FALCON_ER are applied to a card game Hearts, it is shown that FALCON_ER receives less penalty points and learns better rules.

1 INTRODUCTION

Many researchers have developed and proposed methods to solve various problems in the real world by modeling them with autonomous agent systems (RoboCup, 2012; CST, 2010). It is required that those systems should be able to observe information from environments and to determine an optimal action that leads a solution of the problem from the observed information. When we model an environment close to the real world with an agent system, information that agents perceive tends to be given in the form of continuous real numbers. In order to learn an utility function for the pair of discretized states and actions like Q-learning, methods that segment a percept-state space adaptively by Voronoi maps and Adaptive Resonance Theory(ART) and obtain efficiently optimal action rules for agents by reinforcement learning using the discrete space have been developed (G.A. Carpenter and Rosen, 1991; Hamagami and Hirata, 2003; H.Ueda and T.Miyahara, 2008). In recent years, 3-channel fuzzy ART network FALCONia Fusion Architecture for Learning, COgnition and Navigationj(Tan, 2004; Tan and Xiao, 2005) has been proposed to combine reinforcement learning with state segmentation and learn the relations among percepts, actions and rewards. However, it is difficult to apply these methods directly to multi-agent environments where prediction of other agents' behavior and acquisition of cooperated actions are required.

In this paper we modify FALCON so as to be able to predict other agents' behavior and apply the method to a multi-player game with imperfect information; we call the method FALCON_AP (FALCON considering action probability of other agents). In FALCON_AP, we implement an action prediction module based on 2-channel fuzzy ART network and add fourth channel to FALCON so that it can receive the output of the action prediction module. Moreover, we implement FALCON_ER that estimates the expected value of rewards and selects an action according to the value. We apply these methods to a card game Hearts and examine whether the methods are effective or not.

2 FALCON WITH PREDICTION

FALCON (a Fusion Architecture for Learning, COgnition and Navigation) (Tan, 2004) is an extended fuzzy ART (fuzzy Adaptive Resonance Theory) so that it can learn the relations among percepts, actions and rewards. In multi-agent environments, prediction of other agents' behavior is important for choosing optimal actions. Thus, we extend FALCON so that it can receive the predictive information of other agents and can learn the relations among the four kinds of in-

Morishita H., Ueda H. and Takahashi K..
Multi-agent Reinforcement Learning based on Multi-channel ART Networks.
DOI: 10.5220/0004239104610464
In *Proceedings of the 5th International Conference on Agents and Artificial Intelligence* (ICAART-2013), pages 461-464
ISBN: 978-989-8565-39-6

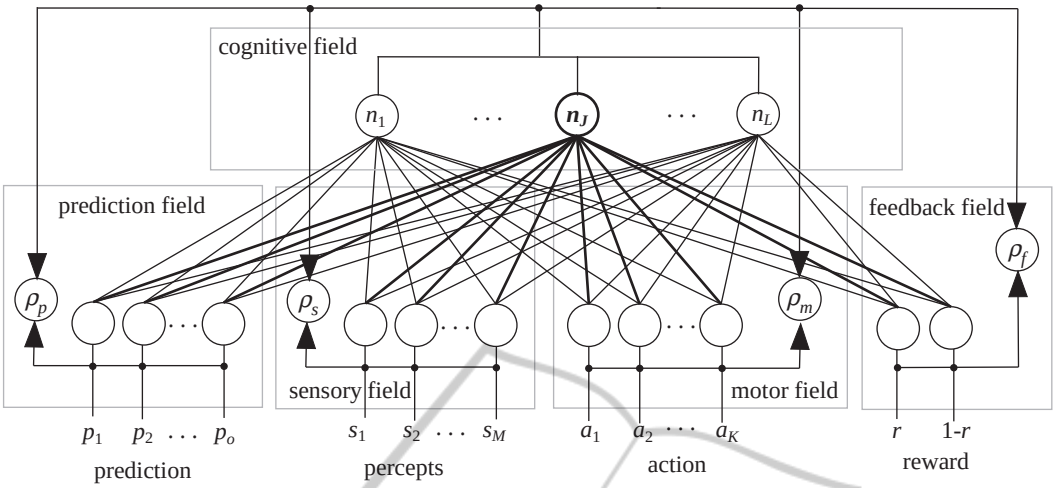

Figure 1: A neural network architecture of FALCON_AP.

formation, namely, predictions, percepts, actions, and rewards. We propose and examine two extensions: FALCON_AP and FALCON_ER.

2.1 FALCON_AP

An architecture of an extended version FALCON_AP (FALCON considering action probability of other agents) of FALCON is shown in Figure 1. FALCON_AP has an architecture in which a prediction field (PF), a sensor field (SF), a motor field (MF), and a feedback field (FF) are connected at a cognitive field (CF). When an autonomous agent has M sensors, M neurons that receive inputs $s_i \in [0,1] (i = 1,...M)$ from M sensors are built in at SF. Motor field MF receives a vector $A = (a_1,...,a_K)$ corresponding to an action chosen by the agent, where K is the number of actions. Elements a_k's of the vector A are set as follows: $a_k = 1$ if action k is chosen, and $a_k = 0$ otherwise. FF has 2 neurons; one receives reward r and the other receives $1-r$, where $r \in [0,1]$ is the reward that the agent receives. Prediction field PF receives information of predictive behavior of other agents. Vector $P = (p_1,...,p_O)$ in Figure 1 is a prediction vector to other agents' behavior, where O is the number of actions other agents can choose. Elements p_o's in P take real values of $0 \leq p_o \leq 1$. We calculate the probability that another agent takes action o as $p_o / \sum_k p_k$.

The cognitive field CF has L neurons. Neuron n_J in CF is connected to neurons in PF, SF, MF, and FF with weighting vectors $W_J^p = (w_{1,J}^p,...,w_{O,J}^p)$, $W_J^s = (w_{1,J}^s,...,w_{M,J}^s)$, $W_J^m = (w_{1,J}^m,...,w_{K,J}^m)$, and $W_J^f = (w_{1,J}^f, w_{2,J}^f)$, respectively, where elements $w_{x,J}^y \in [0,1]$ of weighting vectors indicate the degree of relations between n_J and neurons in PF, SF, MF, and FF, respectively. FALCON_AP learns the relations among predictions, percepts, actions, and rewards by updating weighting vectors W_J^p, W_J^s, W_J^m, and W_J^f, respectively, in the same manner as FALCON does.

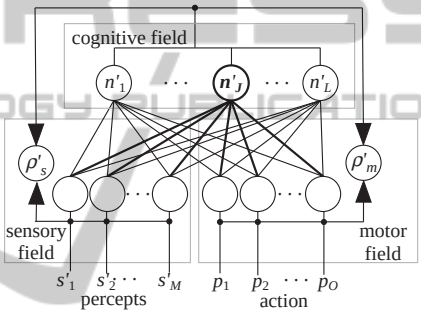

Figure 2: A network architecture of an action prediction module.

Similar to FALCON, an action is chosen in the action choice phase, and weighting vectors of FALCON_AP are updated in the action learning phase. Hereafter, we call neuron n_J in CF category n_J.

In the action choice phase, a percept vector $S = (s_1,...,s_M)$ obtained from sensors is inputted to SF, a K-dimensional vector $N = (1,...,1)$ whose all elements are 1's is inputted to AF, and the reward vector $R = (1,0)$ is inputted to FF. We calculate prediction vector P for the prediction field using an action prediction module shown in Figure 2. In this paper, we build an action prediction module by 2-channel fuzzy ART, i.e. FALCON without FF. Action prediction vector P for a target agent to be predicted is calculated as follows. First, information $S = (s_1,...,s_M)$ that the learning agent can perceive for a target agent is inputted to SF, the K-dimensional vector $N = (1,...,1)$ whose all elements are 1's is inputted to MF in the

prediction module, and then a category n'_J whose choice intensity is the largest is determined. Next, we use weighting vector $W_J^{\prime m}$ between n'_J and MF as prediction vector P for the input to the prediction field of FALCON_AP. Then category n_J whose intensity for choice is the largest in CF of FALCON_AP is chosen as a winning category. An action is chosen according to the weighting vector W_J^m; action k whose weighting value is the largest, i.e. $\text{maxarg}_k \left(w_{k,J}^m \right)$ in W_J^m, is usually chosen.

In the action learning phase, either reinforcement or reset of the relations among predictions, percepts, actions, and rewards is performed depending on the reward the learning agent obtained. When the learning agent received a positive reward, the prediction vector $P = (p_1, ..., p_O)$ obtained from the action prediction module is inputted to PF, the percept vector $S = (s_1, ...s_M)$ obtained from sensors is inputted to SF, the action vector A that indicates the action the agent chose is inputted to MF, the reward vector $R = (1, 0)$ is inputted to FF. Then the weighting vectors between the winning category n_J and each of vectors in PF, SF, MF, FF are updated. In the action learning phase of the action prediction module shown in Figure 2, the percepts vector is inputted to SF, the action vector A that indicates the action the agent chose is inputted to MF, and the weighting vectors between n'_J and each of vectors in SF and MF are updated. When the learning agent received a negative reward, the weighting vectors are updated to weaken the relations among input vectors.

2.2 FALCON_ER

We propose another extended version of FALCON; we call the extended version FALCON_ER (FALCON considering the expected reward). FALCON_ER predicts other agents' behavior for each action the learning agent can take using its action prediction module, and determines the action of the learning agent according to the expected value of rewards calculated according to the prediction. For example, assume that actions the learning agent can take are a_1 and a_2. Also assume that there are two other agents, and agent 1 and agent 2 choose and carry out actions according to the action the learning agent carried out. Then, we calculate the expected reward in the following. FALCON_ER first predicts action $p_{1,1}$ of agent 1 when the learning agent choose action a_1 and then predicts action $p_{2,1}$ of agent 2 after actions a_1 and $p_{1,1}$ are taken. Next, we calculate expected reward r_1 that the learning agent receives after actions a_1, $p_{1,1}$ and $p_{2,1}$ are taken. Expected reward r_2 for action a_2 the learning agent chooses is also calculated in the similar manner to a_1 by predicting actions of agents 1 and 2.

FALCON_ER predicts other agents' behavior from the moment the learning agent chooses an action to the moment it receives a reward and determines the action of the learning agent based on the expected reward. In experiments, we use a card game Hearts for performance evaluation. When we apply FALCON_ER to Hearts, FALCON_ER predicts cards other agents play until one trick ends and then calculates the expected value of penalty points obtained at the trick.

3 EXPERIMENTS

We employ a card game Hearts for performance evaluation. In the experiments, our learning agents play the game against rule-based agents. We compare the performance of FALCON, FALCON_AP and FALCON_ER. Based on feature extraction by heuristics (Fujita, 2004), we determine the percept vector S for FALCON, FALCON_AP and FALCON_ER. For the experiments, we implement a rule-based agent and use it as players opponent to the learning agent. The rule-based agent determines an action with rules extracted from gnome-hearts(Hearts, 2012).

3.1 Hearts

The number of players of Hearts is normally four. Hearts uses a standard deck of 52 playing cards. The higher card of the suit wins; the strength of cards is as follows: in the descending order, A, K, Q, ... , 4, 3, and 2. There is no superiority or inferiority among suits. Each player is delivered 13 cards, and must play a card from his hand at his turn. Starting from a player and playing a card in clockwise direction until the four players play is called a trick. One game is completed after successive 13 tricks. In each trick, the card played by the first player is called a leading card, and the player is called the dealer. The objective of Hearts is to obtain fewest penalty points at the completion of the game. The penalty points of cards are as follows: Q♠ = 13 points, and every card of suit ♡ = 1 point.

3.2 Experimental Results

In this subsection, we show experimental results for game Hearts. The maximum number of categories of FALCON, FALCON_AP and FALCON_ER is limited to 1000. Parameter values for them are chosen by preliminary experiments. We use in the following Figures the average penalty ratio obtained through 1000

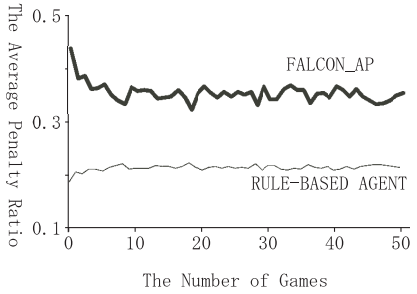

Figure 3: Results of FALCON_AP.

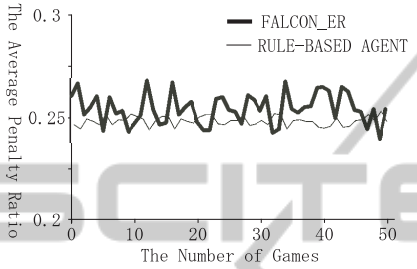

Figure 4: Results of FALCON_ER.

simulation runs for 50 games.

Figure 3 plots the average penalty ratios obtained by the learning agent using FALCON_AP and three rule-based agents for 50 games. We see in Figure 3 that the average penalty ratios of FALCON_AP are decreasing according to progress of learning. The penalty ratios of FALCON_AP were smaller than those of FALCON; the action prediction is effective. However, the average penalty ratio of FALCON_AP is 1.6 times larger than that of the rule-based agent.

Figure 4 plots the average penalty ratios obtained by the learning agent using FALCON_ER and three rule-based agents for 50 games. The expected reward r_x for action x chosen by the learning agent is calculated as follows.

$$r_x = \frac{1}{penalty_x \frac{1}{trick} + queen_x + 1}, \qquad (1)$$

where $penalty_x$ is the predicted penalty points for action x chosen by the learning agent, and $trick$ is the number of remaining tricks (including the current trick) in a game. Parameter $queen_x$ is set 1 if the learning agent without Q♠ becomes the dealer, and 0 otherwise.

We see in Figure 4 that the average penalty ratios of FALCON_ER are slightly decreasing according to progress of learning, and that the penalty ratios of FALCON_ER are almost the same as those of the rule-based agent. The penalty ratios of FALCON_ER are smaller than those of FALCON_AP, and the action choice based on the expected reward is shown to be effective.

4 CONCLUSIONS

In this paper, we proposed the module that learns action prediction of other agents with 2-channel ART network and implemented a modified FALCON, namely FALCON_AP, to be able to register the output of the action prediction module. Through the experiments using Hearts, FALCON_AP can learn optimal action rules. Moreover, we implemented FALCON_ER that can choose an action on the basis of the expected reward. Through the experiments, we showed that the performance of FALCON_ER is almost the same as that of the rule-based agent.

ACKNOWLEDGEMENTS

This research is in part supported by Hiroshima City University Grant for Special Academic Research (General).

REFERENCES

CST (2010). IEICE Concurrent System Technology, http://www.ieice.org/ cst/compe10/.

Fujita, H. (2004). A reinforcement learning scheme for a partially-observable multi-agent system. Master's thesis, Graduate school of Information Science, Nara Institue of Science amd Technology.

G. A. Carpenter, S. G. and Rosen, D. (1991). Fuzzy ART: fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, Vol.4(No.6):pp.759–771.

Hamagami, T. and Hirata, H. (2003). An adjustment of the number of states on Q-learning segmenting state space adaptively. *Proc. of the IEEE International Conference on Systems, Man, and Cybernetics*, pages 3062–3067.

Hearts (2012). Hearts for GNOME, http://www.jejik.com/gnome-hearts/.

H.Ueda, T.Naraki, Y. K. K. and T.Miyahara (2008). State space segmentation for acquisition of agent behavior. *Web Intelligence and Agent Systems: An International Journal*, IOS Press, Vol.6(No.4):pp.373–385.

RoboCup (2012). The RoboCup Federation, http://www.robocup.org/.

Tan, A. H. (2004). FALCON: A fusion architecture for learning, cognition, and navigation. *Proc. of International Joint Conference on Neural Networks*, pages 3297–3302.

Tan, A. H. and Xiao, D. (2005). Self-organizing cognitive agents and reinforcement learning in multi-agent environment. *Proc. of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 351–357.