Behavior Acquisition based on Multi-Module Learning System in Multi-Agent Environment

Yasutake Takahashi[1], Kazuhiro Edazawa[2], and Minoru Asada[1]

[1] Emergent Robotics Area, Dept. of Adaptive Machine Systems, Graduate School of Engineering, Osaka University, Yamadaoka 2-1, Suita, Osaka 565-0871, Japan,
{yasutake,asada}@ams.eng.osaka-u.ac.jp,
[2] eda@er.ams.eng.osaka-u.ac.jp

Abstract. The conventional reinforcement learning approaches have difficulties to handle the policy alternation of the opponents because it may cause dynamic changes of state transition probabilities of which stability is necessary for the learning to converge. This paper presents a method of multi-module reinforcement learning in a multiagent environment, by which the learning agent can adapt itself to the policy changes of the opponents. We show a preliminary result of a simple soccer situation in the context of RoboCup.

1 Introduction

There have been an increasing number of approaches to robot behavior acquisition based on the reinforcement learning methods. The conventional approaches need an assumption that the environment is almost fixed or changing slowly so that the learning agent can regard the state transition probabilities are consistent during its learning. Therefore, it seems difficult to apply the reinforcement learning method to a multiagent system because a policy alteration of the other agents may occur, which dynamically changes the state transition probabilities from the viewpoint of the learning agent. RoboCup provides such a typical one, that is, a highly dynamic, hostile environment, in which an agent has to obtain purposive behaviors.

There are a number of papers on reinforcement learning system in a multiagent environment. Asada et al. [1] proposed a method which estimates the state vectors representing the relationship between the learner's behavior and those of other agents in the environment using a technique from system identification, then the reinforcement learning based on the estimated state vectors is applied to obtain the optimal behavior policy. However, this method requires re-learning or adjustment of learning agent's policy whenever the other agents change their policies, even if they switch their policies back which the learning agent has already adjusted before. This problem happens because one learning module can maintain only one policy.

A multiple learning module approach would provide one solution for this problem. If we can assign multiple learning modules to different situations in which each module can regard the state transition probabilities are consistent, then the system would provide reasonable performance. There are a number of works on the multi-learning module systems.

Singh [2, 3] has proposed compositional Q-learning in which an agent learns multiple sequential decision tasks with multi learning modules. Each module learns its own elemental task while the system has a gating module for the sequential task, and this module learns to select one of the elemental task modules. Takahashi and Asada [4] proposed a method by which a hierarchical structure for behavior learning is self-organized. The modules in the lower networks are organized as experts to move to different categories of sensor value regions and learn lower level behaviors using motor commands. In the meantime, the modules in the higher networks are organized as experts which learn higher level behaviors using lower modules. Each module assigns its own goal state by itself. However, there is no such measure to identify the situation that the agent can change modules corresponding to the current situation.

Sutton [5] has proposed DYNA-architecture which integrate world model learning and execution-time planning. Singh [6] has proposed a method of learning a hierarchy of models of the DYNA-architectures. The world model is not for the identification of the situations, but only for improving the scalability of reinforcement learning algorithms.

Doya et al. [7] have proposed MOdular Selection and Identification for Control (MOSAIC), which is a modular reinforcement learning architecture for non-linear, non-stationary control tasks. The basic idea is to decompose a complex task into multiple domains in space and time based on the predictability of the environmental dynamics. Each module has a state prediction model and a reinforcement learning controller. The models have limited capabilities of state prediction as linear predictors, therefore the multiple prediction models are required for the non-linear task. A domain is specified as a region in which one linear predictor can estimate sensor outputs based on its own prediction capability. The responsibility signal is defined by a function of the prediction errors, and the signals of the modules define the outputs of the reinforcement learning controllers. Haruno et al. [8, 9] have proposed another implementation of MOSAIC based on multiple modules of forward and inverse models.

In this paper, we propose a method by which multiple modules are assigned to different situations and learn purposive behaviors for the specified situations as results of the other agent's behaviors. We show a preliminary result of a simple soccer situation in the context of RoboCup.

2 A Basic Idea and An Assumption

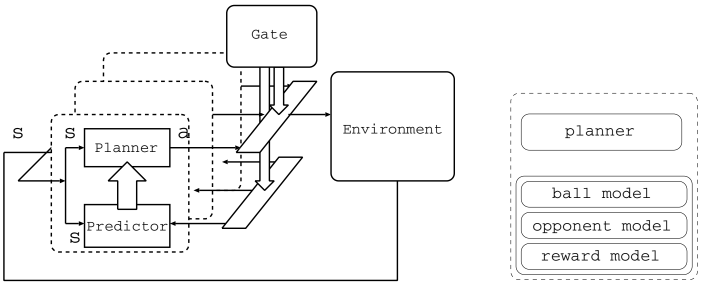
The basic idea is that the learning agent could assign one reinforcement learning module to each situation if it can distinguish a number of situations in which the state transition probabilities are consistent. We introduce a multiple learning module approach to realize this idea. A module consists of a learning component which models the world and an execution-time planning one. The whole system will follow these procedure simultaneously.

- find a model which represents the best estimation among the modules,
- update the model, and
- calculate action values to accomplish a given task based on dynamic programming (DP).

As a preliminary experiment, we prepare a case of ball chasing behavior with collision avoidance in the context of RoboCup. The problem here is to find the model which can most accurately describe the opponent's behavior from the view point of the learning agent. It may take a time to distinguish the situation, then, we put an assumption.

- The policy of the opponent might change match by match but is fixed during one match.

3 A Multi-Module Learning System



(a) system

(b) module

Fig. 1. A multi-module learning system and an architecture of a module

Fig.1(a) shows a basic architecture of the proposed system, that is, a multi-module reinforcement learning one. Each module has a forward model (predictor) which represents the state transition model and a behavior learner (policy planner) which estimates the state-action value function based on the forward model in the reinforcement learning manner. This idea of combination of a forward model and a reinforcement learning system is similar to the H-DYNA architecture [6] or MOSAIC [7–9]. In other words, we extend such architectures to an application of behavior acquisition in the multi-agent environment.

The system selects one module which has the best estimation of the state transition sequence by activating a gate signal corresponding to a module and by deactivating the goal signals of other modules, and the selected module sends action commands based on its policy.

3.1 Predictor

In this experiment, the agent recognizes a ball, a goal, and the opponent in the environment. The state space of the planner consists of features of all objects in order to calculate state values (discounted sum of the reward received over time) for each state and action pair. However, it is impractical to maintain a full size state transition model for real robot applications because the size of state-action space becomes easily huge and it is really rare to experience all state transitions within the reasonable learning time.

In general, the motion of the ball depends on the goal and the opponent because there are interactions between the ball, the goal, and the opponent. However, the proportion of the interaction time is much shorter than that of non-interaction time. Therefore, we assume that the ball motion is independent from the goal and the opponent. Further, we assume that the opponent motion from the viewpoint of the agent seems independent from the ball and the goal positions and to depend on only the learning agent's behavior even if the opponent's decision may depend on the ball and/or the goal positions. If the system has maintain the forward models of the ball, the goal, and the opponent separately, the each model can be much more compact and it is easy to experience most state transition within reasonable learning time.

Fig.1 (b) shows an architecture of one module in our system. As mentioned above, the module has three forward models for the ball, the goal, and the opponent. We estimate the state transition probability $\hat{\mathcal{P}}^a_{ss'}$ for the triplet of state s, action a, and next state s' using the following equation:

$$\hat{\mathcal{P}}^a_{ss'} = \hat{\mathcal{P}}^a_{{}^bs{}^bs'} \cdot \hat{\mathcal{P}}^a_{{}^gs{}^gs'} \cdot \hat{\mathcal{P}}^a_{{}^os{}^os'} \ , \tag{1}$$

where a state $s \in S$ is a combination of three states in the ball state space ${}^bs \in {}^bS$, the goal state space ${}^gs \in {}^gS$, and the opponent state space ${}^os \in {}^oS$. The system has not only the state transition model but also a reward model $\hat{\mathcal{R}}^a_{ss'}$.

We simply store all experiences (state-action-next state sequences) to estimate these models. According to the assumption mentioned in **2**, we share the state transition models of the ball and the goal and the reward model among the modules, and each module has its own opponent model. This leads to further compact model representation.

3.2 Planner

Now we have the estimated state transition probabilities $\hat{\mathcal{P}}^a_{ss'}$ and the expected rewards $\hat{\mathcal{R}}^a_{ss'}$, then, an approximated state-action value function $Q(s,a)$ for a state action pair $(s,\ a)$ is given by

$$Q(s,a) = \sum_{s'} \hat{\mathcal{P}}^a_{ss'} \left[\hat{\mathcal{R}}^a_{ss'} + \gamma \max_{a'} Q(s',a') \right] \ , \tag{2}$$

where $\hat{\mathcal{P}}^a_{ss'}$ and $\hat{\mathcal{R}}^a_{ss'}$ are the state-transition probabilities and expected rewards, respectively, and the γ is the discount rate.

3.3 Gating Signals

The basic idea of gating signals is similar to Tani and Nolfi's work [10] and the MOSAIC architecture [7–9]. The gating signal of the module becomes larger if the module performs better state transition prediction during a certain period, else smaller. We assume that the module which performs best state transition prediction has the best policy against the current situation because the planner of the module is based on the model which describes the situation best. In our proposed architecture, the gating signal is used for the gating the action outputs from modules. We calculate the gating signals g_i of the module i as follows:

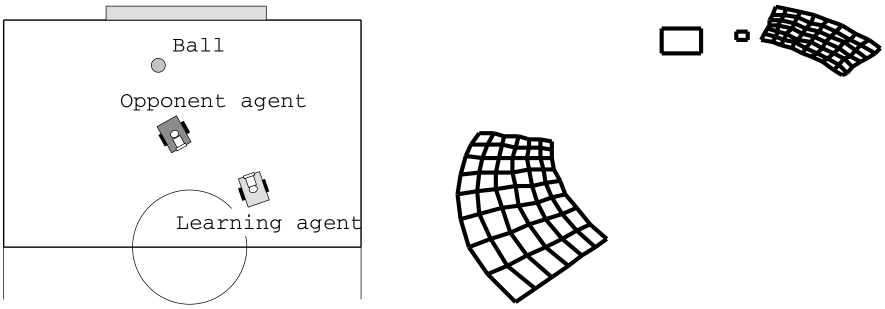
$$g_i = \prod_{t=-T+1}^{0} \frac{e^{\lambda p_i^t}}{\sum_j e^{\lambda p_j^t}}$$

where p_i is the occurrence probability of the state transition from the previous $(t-1)$ state to the current (t) one according to the model i, and the λ is a scaling factor.

4 Experiments

We have studied the preliminary experiments so far. The task of the learning agent is to catch the ball while it avoids the collision with an opponent.

4.1 Setting

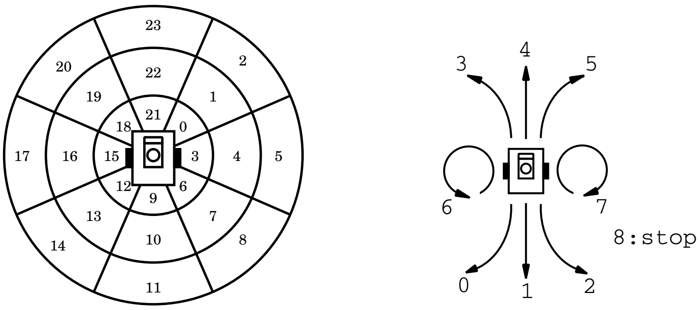


(a) Top view of the field

(b) Captured image of the camera with a omni-directional mirror

Fig. 2. Simulation Environment

We apply the proposed system to a mobile robot which participates in the RoboCup middle size league. The robot has an omni-directional camera system.



(a) State space

(b) Action space

Fig. 3. State-action space

A simple color image processing is applied to detect an ball area and an opponent one in the image in real-time (every 33ms). The driving mechanism is a PWS (Power Wheeled System); the vehicle is fitted with two differential wheels. The wheels are driven independently by separated DC motors, and two extra free wheels ensure the static stability. Figure 2 (a) shows one of situations in which the learning agent encounters and Figure 2 (b) shows the simulated captured image of the camera with the omni-directional mirror mounted on the robots. The larger box indicates the opponent and the smaller one indicates the ball.

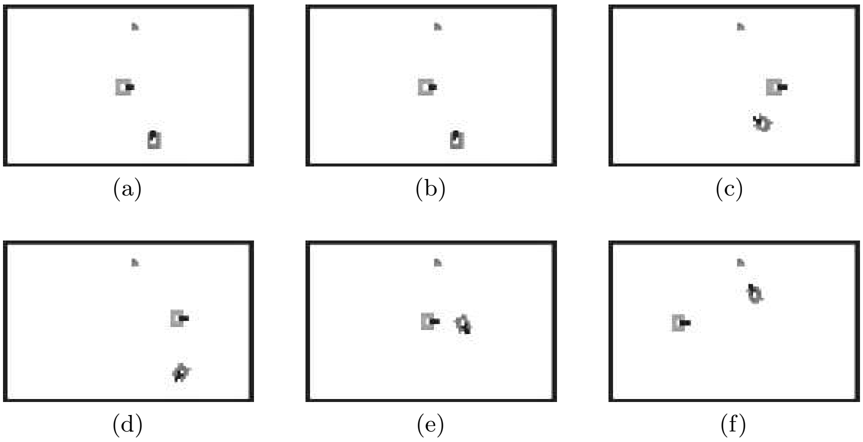
The state space is constructed in terms of the centroids of the ball and the opponent on the image (Figure 3 (a)). The action space is constructed in terms of two torque values to be sent to two motors corresponding to two wheels (Figure 3 (b)). These parameters of the robot system are unknown to the robot, and it tries to estimate the mapping from sensory information to appropriate motor commands by the method.

The opponent has a number of behaviors such as "stop", "move left", and "move right", and switches them randomly after a fixed period. The learning agent has models to those behaviors of the opponents. The learning agent behaves randomly while it gathers the data of the ball and the opponent image positions and builds up the models of them.

4.2 Simulation Result

Table 1. Comparison of the success rates between the agent with multi-module system and one with one-module system

system	success rate
multi-module	61 %
one-module	50 %


(a) (b) (c)

(d) (e) (f)

Fig. 4. A sequence of a chasing behavior

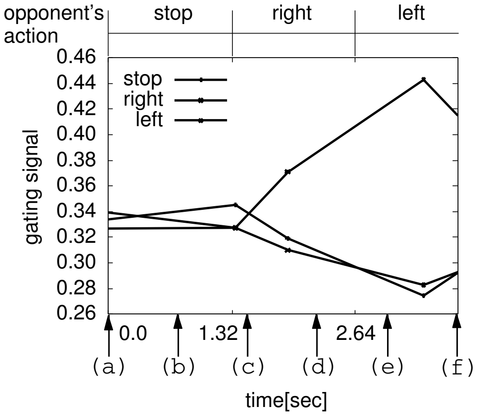


Fig. 5. A sequence of gating signal while the agent executes its learned policy

We have applied the method to a learning agent and compared the other agent which has only one learning modules. Table 1 shows the success rate of these two system after the learning. The success indicates that the learning agent successfully catches the ball with collision avoidance while the opponent moves randomly. The success rate indicates the number of successes in the one hundred trials. The multi-module system shows better performance than the one-module system. Figure 4 shows an example sequence of the behavior when the agent executes its learned policy and the opponent behaves randomly after a fixed period. Figure 5 shows the sequence of the gating signal (the opponent's behavior estimation) during the behavior. The arrows and alphabet indexes at the bottom correspond to the indexes of the figure 4. The agent seems to fail to estimate the opponent's behavior at the beginning and end periods, however, it accomplishes the given task. This mean that even if the agent fails to estimates the other agent's behavior, there is no problem in some situations where the learning agents policy does not depends on the other agent's behavior. For example, the opponent's behavior does not depend on the agent's behavior when the ball is near and the opponent is far from the agent. In such a case, the agent does not have to estimate the other's behaviors correctly.

5 Conclusion

In this paper, we proposed a method by which multiple modules are assigned to different situations and learn purposive behaviors for the specified situations as results of the other agent's behaviors. We have shown a preliminary result of a simple soccer situation in the context of RoboCup.

References

1. M. Asada, E. Uchibe, and K. Hosoda. Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development. *Artificial Intelligence*, 110:275–292, 1999.
2. Satinder Pal Singh. Transfer of learning by composing solutions of elemental sequential tasks. *Machine Learning*, 8:323–339, 1992.
3. Satinder P. Singh. The effeicient learnig of multiple task sequences. In *Neural Information Processing Systems 4*, pages 251–258, 1992.
4. Y. Takahashi and M. Asada. Vision-guided behavior acquisition of a mobile robot by multi-layered reinforcement learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 395–402, 2000.
5. Richard S. Sutton. Integrated modeling and control based on reinforcement learning and dynamic programming. *Advances in Neural Information Processing Systems 3*, pages 471–478, 1991.
6. Satinder P. Singh. Reinforcement learning with a hierarchy of abstract models. In *National Conference on Artificial Intelligence*, pages 202–207, 1992.
7. Kenji Doya, Kazuyuki Samejima, Ken ichi Katagiri, and Mitsuo Kawato. Multiple model-based reinforcement learning. Technical report, Kawato Dynamic Brain Project Technical Report, KDB-TR-08, Japan Science and Technology Corporation, June 2000.
8. Masahiko Haruno, Daniel M. Wolpert, and Mitsuo Kawato. Multiple paired forward-inverse models for human motor learning and control. *Advances in Neural Information Processing Systems*, 11:31–37, 1999. MIT Press, Cambridge, Massachusetts.
9. Masahiko Haruno, Daniel M. Wolpert, and Mitsuo Kawato. Mosaic model for sensorimotor learning and control. *Neural Computation*, 13:2201–2220, 2001.
10. Jun Tani and Stefano Nolfi. Self-organization of modules and their hierarchy in robot learning problems: A dynamical systems approach. Technical report, Technical Report: SCSL-TR-97-008, 1997.