2008 International Symposium on Nonlinear Theory and its Applications
NOLTA'08, Budapest, Hungary, September 7-10, 2008

A Profit Sharing Reinforcement Learning Method Using a Memory-Based Dynamic Reinforcement Function

Masaaki Usui†, Hidehiro Nakano† and Arata Miyauchi†

†Musashi Institute of Technology
1-28-1 Tamazutumi, Setagaya-ku Tokyo, 158-8557 Japan
Email: usui@ic.cs.musashi-tech.ac.jp, nakano@ic.cs.musashi-tech.ac.jp, miyauchi@ic.cs.musashi-tech.ac.jp

Abstract—This paper proposes a new reinforcement learning method which decides dynamic reinforcement function. The conventional method has problems in deciding dynamic reinforcement function. In the proposed method, it is decided based on a memory of rules which an agent selects and executes. The proposed method provides better learning performances than the conventional SDPS. We present some numerical simulation results for static and dynamic maze environments.

1. Introduction

Reinforcement learning (RL) is the framework of learning methods to adapt unknown environments through trial-and-error. There have been proposed many methods of RL. These methods are classified into exploration-oriented methods represented by Q-learning [1] and exploitation-oriented methods represented by Profit Sharing (PS) [2]-[5]. The PS can realize efficient learning not only for Marcov Decision Process (MDP) environments but also for non-MDP environments such as dynamic environments [4][5]. In this paper, we focus on the PS as a basic method of RL. The PS uses a reinforcement function when reward obtained from target environments is distributed. Generally, a geometrical decreasing function is used as the reinforcement function in order to realize learning with a rationality. Since reward decreases geometrically, it is difficult to apply the PS to large scale environments. In our previous works, we have proposed Dynamic PS (DPS) and Simplified Dynamic PS (SPDS) which use dynamic reinforcement functions referring learning progress [6]. These methods can realize more efficient learning and can be applied to larger scale environments than the conventional PS. However, there exist the cases where the conventional DPS and SDPS can not decide appropriate reinforcement functions in some environments.

In this paper, we show problems in deciding dynamic reinforcement function by using the conventional SDPS, and propose the new method to improve these problems. In the proposed SDPS, the dynamic reinforcement function is decided based on a memory of rules which an agent selects and executes. The proposed SDPS provides better learning performances than the conventional SDPS. Some numerical simulation results for static and dynamic maze environments are presented.

2. Simplified Dynamic Profit Sharing (SDPS)

In this section, we explain a basic algorithm of Simplified Dynamic Profit Sharing (SDPS) [6]. First, a learning agent recognizes a state in an environment. Next, the agent selects and executes an action in the state. Then a pair of the state and action is memorized as a rule. The agent repeats in this manner until it reaches an objective state. Here, an effective rule is defined as a rule that can contribute to achievement of an objective state, and an ineffective rule is defined as a rule that can not contribute to achievement of the objective state. If the agent reaches an objective state, it distributes reward to memorized rules based on a reinforcement function. Compared with the conventional Profit Sharing (PS) [2]-[5], the SDPS is a method improved in the reinforcement function. SDPS can improve learning efficiency and can fast solve various tasks. SDPS uses the following reinforcement function.

$$f_i = \frac{1}{S(i)} f_{i+1} \tag{1}$$

where f_i is the i-th reward and $S(i)$ is decreasing ratio. In SDPS, $S(i)$ is decided by Equations (2) and (3).

$$S(i) = \frac{1 - P_{ine}(i)}{P_{eff}(i)} + 1 \quad , P\varepsilon \leq P_{ine}(i) \tag{2}$$

$$S(i) = \frac{1 - P_{ine}(i)}{P_{eff}(i)} \quad , P\varepsilon > P_{ine}(i) \tag{3}$$

where $P_{eff}(i)$ is action selection probability[1] of effective rule in the i-th state, $P_{ine}(i)$ is action selection probability of ineffective rule in the i-th state and $P\varepsilon$ is a threshold parameter. The decreasing ratio $S(i)$ is calculated in each state. If the learning of the i-th state is insufficient, $S(i)$ is calculated as a large value based on Equation (2). Then the learning with a rationality such that ineffective rules are not reinforced is possible. If the learning of the i-th state is sufficient, $S(i)$ is calculated as a small value based on Equation (3). Then, a large amount of reward is propagated to each state and the learning speed can be accelerated.

The threshold parameter $P\varepsilon$ controls the learning speed and rationality, and has trade-off between them. If $P\varepsilon$ decreases, learning speed becomes slower and rationality be-

[1] In SDPS, an action selector is assumed to be a soft-max action selection rule.

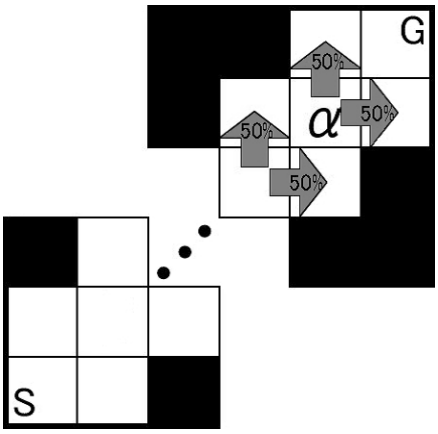

Figure 1: Problem for the convennsional SDPS

comes higher. Conversely, if $P\varepsilon$ increases, learning speed becomes faster and rationality becomes lower.

3. Problems

In the conventional SDPS, $P_{eff}(i)$ and $P_{ine}(i)$ are decided as follows.

- $P_{eff}(i)$: the largest action selection probability in the i-th state.
- $P_{ine}(i)$: the second largest action selection probability in the i-th state. This is based on an assumpiton of the worst case in the learning

The SDPS can be applied to various environments with a high learning efficiency, but a problem might occur in some environments. If plural effective rules exist in an environment and their dominance does not exist, their rules are reinforced evenly. Then, $S(i)$ decided by Equations (2) and (3) does not decrease sufficiently. Figure 1 shows a simple maze environment as an example. The learning agent searches a route to an objective state (G) from an initial state (S).

In the figure, effective rules in a state α are UP and RIGHT, and their dominance does not exist. Then, these two rules are reinforced evenly, and decreasing ratio in the states does not decrease sufficiently in the conventional SDPS: rules from the initial state to the state α are not reinforced sufficiently. Also, in dynamic environments, effective and ineffective rules might change as an environment changes into another environment. In the conventional SDPS, appropriate decreasing ratio to relearn the new environment can not be obtained. These problems should be improved so that SDPS is applied to real environments.

4. Proposed Method

In the proposed method, an effective rule memory is added to a learning agent. As shown in Figure 2, the agent memorizes finally selected rules in each state as effective

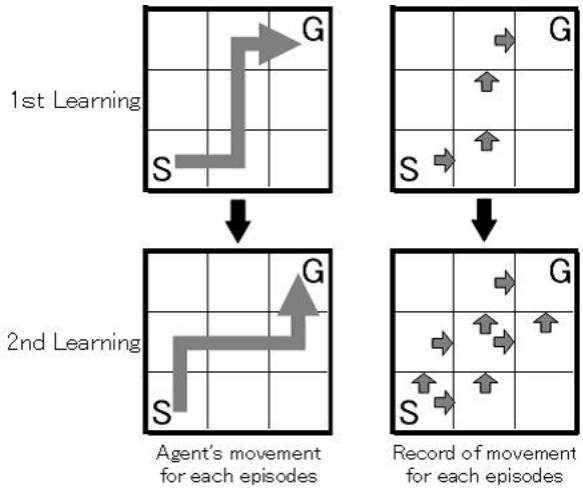

Figure 2: Effective rule memory

rules. The finally selected rules are obtaied from memorized rule sequence in each learning. In the proposed method, $P_{eff}(i)$ and $P_{ine}(i)$ are decided as follows using the effective rule memory.

- $P_{eff}(i)$: the largest action selection probability included in the effective rule memory.
- $P_{ine}(i)$: the largest action selection probability not included in the effective rule memory.

The effective rule memory is assumed to be empty in the begining of learning. In dynamic environments, effective rules and ineffective rules can change. If the agent does not forget them, approporeate relearning can not be realized. Therefore in the proposed method, the agent has forgetting scheme for effective rule memory. If the agent detects a change of the environment, effective rule memory is flushed. The change of the environment can be detected using effective rule memory. The agent recognizes the change of the environment when a state transition does not occur by selecting a rule included in effective rule memory.

5. Numerical Simulations

In order to verify effectivity of the proposed method, we perform numerical simulations for static and dynamic maze environments. Table 1 shows setting of the experiments.

Selectable	up, down, left, right
Selection Method	Roulette Selection
Trials	1,000
Learning Times	10,000
Gain Rewards	500,000
Initial Rewards	10,000,000
Minimum Steps	8

Table 1: Setting of numerical simulations

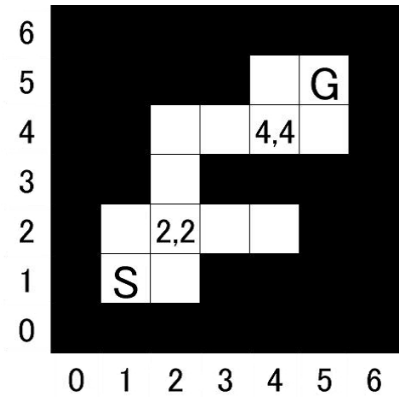

Figure 3: 5x5 static maze environment

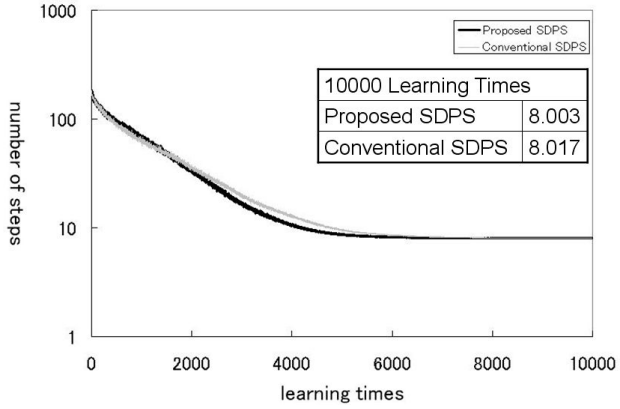

Figure 4: Learning curves for the static maze environment

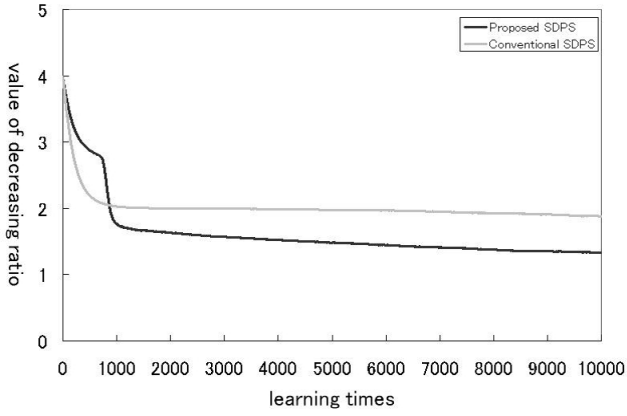

Figure 5: Transitions of decreasing ratio S at state (4,4)

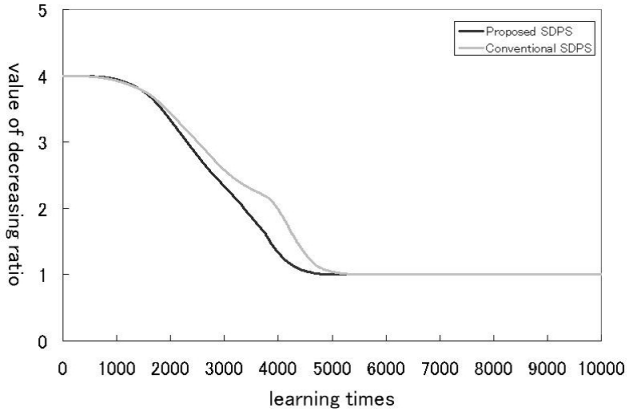

Figure 6: Transitions of decreasing ratio S at state (2,2)

5.1. Static Environment

Figure 3 shows 5x5 static maze environment having plural optimum solutions. Figure 4 shows learning curves for this environment by using the conventional and proposed SDPS. Horizontal-axis is learning times and vertical-axis is the number of steps from initial state(S) to goal state(G).

These learning curves are almost the same, but the learning curve for the proposed SDPS converges to the minimum steps faster. Figures 5 and 6 show transitions of decreasing ratio between conventional and proposed SDPS at each state. If the learning in each state progresses sufficiently, the decreasing ratio should converge to 1 in order to propagate a large amount of reward to each state. However, in the state (4,4), the decreasing ratio of the conventional SDPS converges to 2. Therefore, the reward from the initial state to this state decreases. On the other hand, the decreasing ratio of the proposed SDPS converges to 1. As shown in Figure 6, we can find that the learning speed of the proposed SDPS is faster than that of the conventional SDPS. Such a difference will be more prominent for larger scale environment.

5.2. Dynamic Environment

Next, we perform additional experiments for a dynamic maze environment as shown in Figure 7. Also we investigate transitions of decreasing ratio at each state. The environment changes only 2 states (3,4) and (4,3) at 2000 learning times. The agent should change effective and ineffective rules at the state (2,2), when environment changes. Figure 8 shows learning curves of each method. At 2000 learning times, the agent can not sufficiently respond to the changes. Also, a number of steps increase significantly.

Figures 9 and 10 show transitions of decreasing ratio at each state. At the state (4,4), effective and ineffective rules do not change. Therefore the transitions of decreasing ratio in Figure 9 is almost the same as those in Figure 5. However, at the state (2,2), effective and ineffective rules change at 2000 learning times. In this state, the decreasing ratio does not change in the conventional SDPS. Therefore, relearning speed of this method is slow as shown in Figure 8. In the proposed SDPS, appropriate decreasing ratio can be calculated if a change of the environment can be detected. Therefore the proposed SDPS can relearn new environment faster than the conventional SDPS.

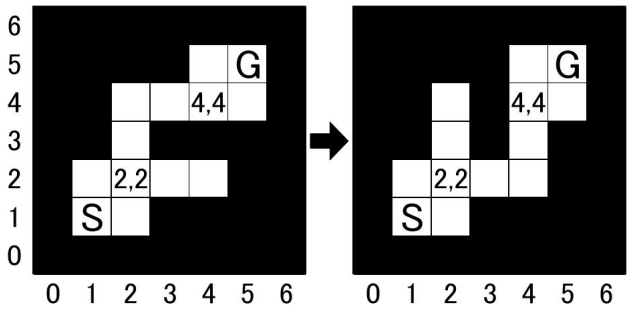

Figure 7: 5x5 dynamic maze environment

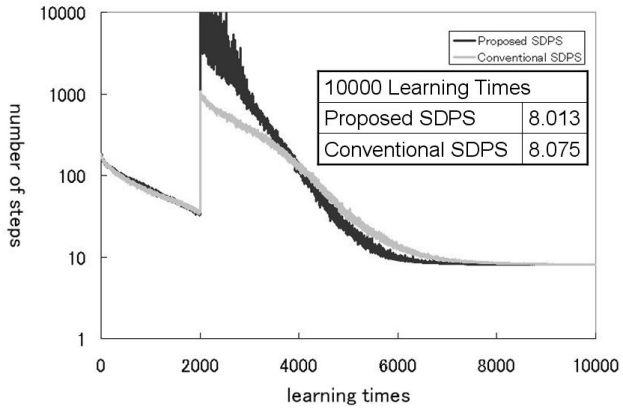

Figure 8: Learning curves for the dynamic maze environment

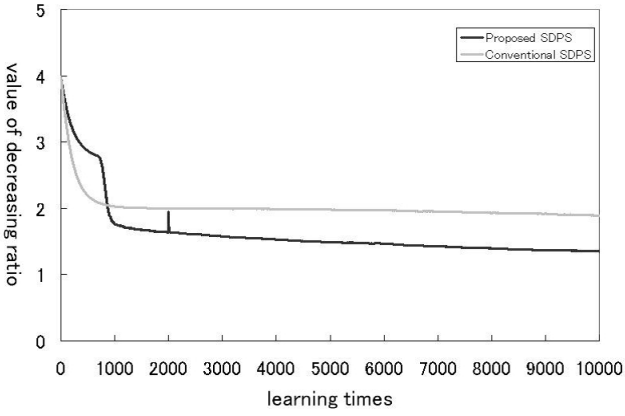

Figure 9: Transitions of decreasing ratio at (4,4)

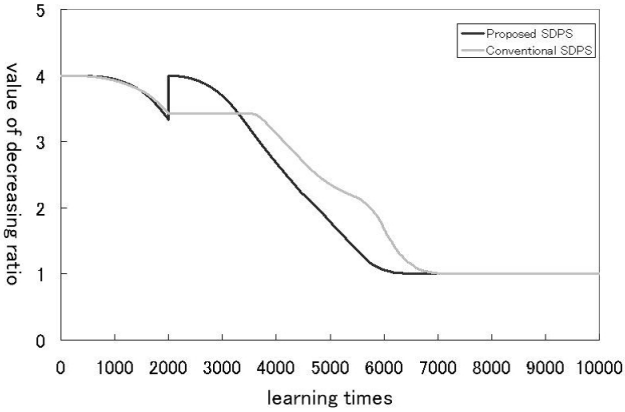

Figure 10: Transitions of decreasing ratio at (2,2)

6. Conclusion

We have studied effective reinforcement learning methods for an environment having plural optimum solutions, and a dynamic environment. An agent memorizes effective rules in order to calculate optimum decreasing ratio in each state. The proposed SDPS provides better learning performance than the conventional SDPS.

Future problems include development for larger scale environments with plural optimum solutions, analysis of the learning performances for the timing when environments change and improvement of action selection methods for dynamic environment.

References

[1] Watkins, C. J. H and Dayan, P. Technical note: Q-learning. *Machine Learning* Vol.8: 55-68, 1992.

[2] Grefenstette J. J. "Credit Assignment in Rule Discovery Systems Based on Genetic Algorithms," *Machine Learning* Vol.3, pages 225-245, 1988.

[3] K. Miyazaki, M. Yamashita and S. Kobayashi, "A theory of profit sharing in reinforcement learning," *J.JSAI*, vol.9, pp.580-587, 1994. (in Japanese)

[4] Arai S, Sycara K., "Credit assignment method for learning effective stochastic policies in uncertain domains," *Proc. Genetic and Evolutionary Computation Conference*, p 815-822, 2001.

[5] H.Nakano, S.Takada, S.Arai and A.Miyauchi, "An efficient reinforcement learning method for dynamic environments using short term adjustment," Proc. of 2005 Nonlinear Theory and its Applications, pp. 250-253, 2005.

[6] Y. Hasegawa, S. Takada, H. Nakano, S. Arai, and A. Miyauchi, "A Reinforcement Learning Method Using a Dynamic Reinforcement Function Based on Action Selection Probability," *System and Computers in Japan*, Vol.38, No.7, 2007.