

Proposed Particle-Filtering Method for Reinforcement Learning

Akira Notsu, Katsuhiro Honda and Hidetomo Ichihashi
Osaka Prefecture University
1-1 Gakuencho, Nakaku, Sakai, Osaka 599-8531, JAPAN
Email: {notsu, honda, ichi}@cs.osakafu-u.ac.jp

Abstract—We propose a novel action-search particle-filtering algorithm for reinforcement learning processes. This algorithm is designed to perform search domain reduction and heuristic space segmentation. In this method, each action space is divided into several new segments using particles. Appropriate search domain reduction can minimize learning time and enable the recognition of the evolutionary process of learning. In a numerical experiment, the proposed filtering method is applied to a single-pendulum simulation in order to demonstrate the adaptability of this simulation model.

Index Terms—reinforcement learning, particle-filter, space segmentation.

I. INTRODUCTION

This paper proposes a novel particle-filtering algorithm for reinforcement learning that periodically divides the agent environment and action search space during the learning process. The standard reinforcement learning algorithm [1], [2] requires well-suited segmentation before learning. In the proposed method, each state is searched by several particles, and distances between states are fixed according to the importance of each state. Herein, human-like space segmentation in actor-critic learning, which we investigated previously [3], [4], is further refined.

In post-supervised design, clustering, which is done after many trial and error iterations, is a probabilistic method. A genetic algorithm representative of these schemes is called a learning classifier system [5], [6]. In this system, inputted data are encoded as binary rules that a genetic algorithm alters before selecting from them the best rules. However, inputted data can be encoded during the learning, as in the previously investigated method. Our proposed algorithm is applicable not only to reinforcement learning but also to social simulation, which requires each agent to behave rationally during the trial and error iterations. Such an approach provides clues for understanding culture and language.

II. REINFORCEMENT LEARNING

Reinforcement learning assumes that agents can visit only a finite number of states, and that when visiting a state, they will collect a numerical reward (which can be interpreted as a punishment in the case of a negative reward [7]). Each state has a different value associated with it. Given an initial state, subsequent states are reached according to actions. The value of a given state is defined as the average reward that can be collected by selecting actions available in that state.

A. Particle in state and action space

A high-performance agent usually needs appropriately segmented state set S and action set A because of agent state-space explosion. Generally, a genetic algorithm is used to search for appropriate segmentation in such individual double-loop learning. In particular, we use a particle swarm, which is a classification system in which a minimization of (2) under the following condition takes place:

$$\sum_{i=1}^c u_{ki} = 1, \quad (1)$$

where c is the number of clusters and u_{ki} is called a membership function.

$$L = \sum_{i=1}^c \sum_{k=1}^N u_{ki} D(x_k, v_i), \quad (2)$$

where N is the amount of data, c is the number of clusters, and $D(x_k, v_i)$ denotes the distance between data vector x_k and cluster center v_i ; the difference between expected utility V and observed utility V' in the learning process.

B. Actor-critic

The actor-critic method includes the critic, which maintains the state value estimate V , and the actor, which is responsible for choosing the appropriate actions at each state (Fig. 1).

Value function approaches attempt to find a policy that maximizes the return by maintaining a set of estimates of expected rewards based on a policy π . In this method, an agent attempts to estimate the expected reward $V(s)$:

$$V(s) = E[R|s_t, \pi] = r_t + \gamma E[R|s_{t+1}, \pi] \quad (3)$$

for an agent following π and taking an action in state s .

- step 1 Agent observes state s and actor selects action a according to its stochastic policy π .
- step 2 Critic receives reward and calculates the following TD-error.

$$\text{TD-error} = r_t + \gamma V(s_{t+1}) - V(s_t), \quad (4)$$

where γ is discount factor.

- step 3 Actor updates its stochastic policy π according to the TD-error.

(TD-error) > 0 Selected action a is favorable, so increase action a selection probability.

(TD-error) < 0 Selected action a is not favorable, so decrease action a selection probability.

step 4 Update value function (Fig. 2).

$$V(s_t) \leftarrow V(s_t) + \alpha \text{TD-error}, \quad (5)$$

where α is the learning rate.

step 5 Go to **step 1**.

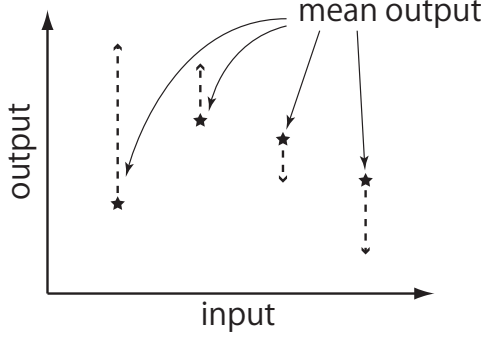


Fig. 1. Actor-critic

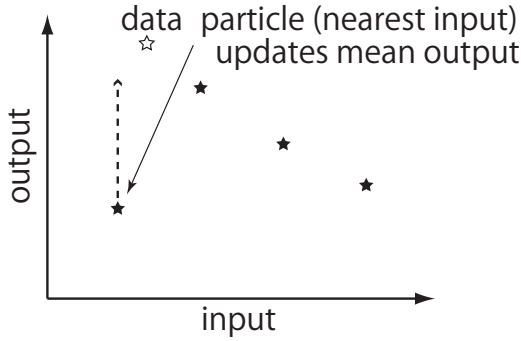


Fig. 2. Actor-critic

In this paper, we use a normal distribution as stochastic policy π (Fig. 3). Updating the stochastic policy is carried out by adjusting the particle's mean $\mathbf{a}$ and variance v of output (which is an agent's action).

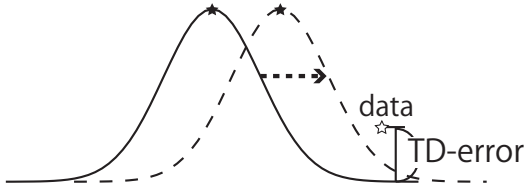


Fig. 3. Updating stochastic policy π

If (TD-error) > 0,

$$\mathbf{a} \leftarrow (1 - \alpha_a)\mathbf{a} + \alpha_a \text{vector}[\text{output}] \quad (6)$$

$$v \leftarrow (1 - \beta_1)v. \quad (7)$$

Else if (TD-error) < 0,

$$\mathbf{a} \leftarrow (1 + \alpha_a)\mathbf{a} - \alpha_a \text{vector}[\text{output}] \quad (8)$$

$$v \leftarrow (1 - \beta_2)v. \quad (9)$$

Here, α_a , β_1 , and β_2 are parameters.

C. Particle filter

The particle-filtering method is a model estimation technique based on Bayesian models in which the latent variables are connected in a Markov chain. A particle filter estimates the sequence of hidden parameters on the basis of only on the observed data.

The method assumes that x_k and the observations y_k can be modeled in the following form:

$x_0, x_1, \dots$ is a first-order Markov process such that

$$x_k | x_{k-1} \sim p_{x_k | x_{k-1}}(x | x_{k-1}), \quad (10)$$

where the initial distribution $p(x_0)$ is given.

Although each particle filter yields only a rough approximation, with enough particles, the estimate can be quite accurate.

The particle-filtering method generates a set of samples that approximate the filtering distribution $p(x_k | y_0, \dots, y_k)$. With P samples, expectations with respect to the filtering distribution are approximated by

$$\int f(x_k) p(x_k | y_0, \dots, y_k) dx_k \approx \frac{1}{P} \sum_{L=1}^P f(x_k^{(L)}), \quad (11)$$

and $f(\cdot)$ can provide all the moments of the distribution up to some degree of approximation.

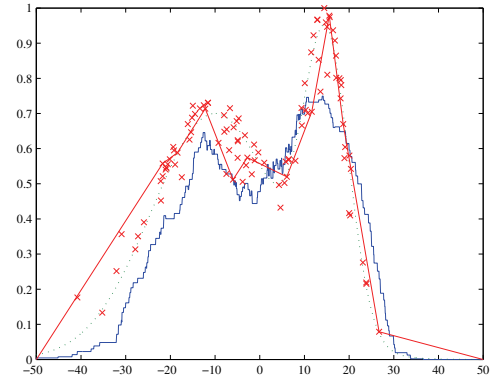


Fig. 4. Example of a particle filter

III. PARTICLES IN EACH ACTION SPACE

We propose particles in the action space. The proposed method using particles can search for the global solution (best output) and a local solution, which has the potential of being the global solution.

In each state, several particles have expected rewards according to expected utility V . If inputted data have a positive TD-error, the nearer particle is updated (shown in Fig. 5). This method decreases the probability of the local solution.

In addition, because an evenly segmented state is not always good for learning, we also propose adjusting the distance

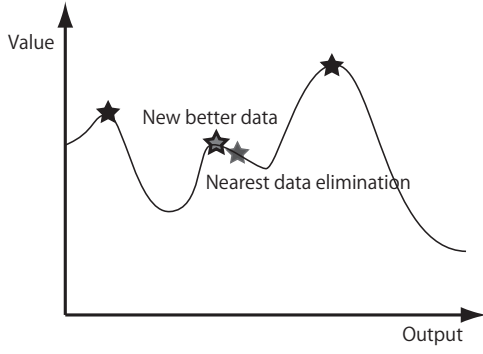


Fig. 5. Updating the output according TD-error

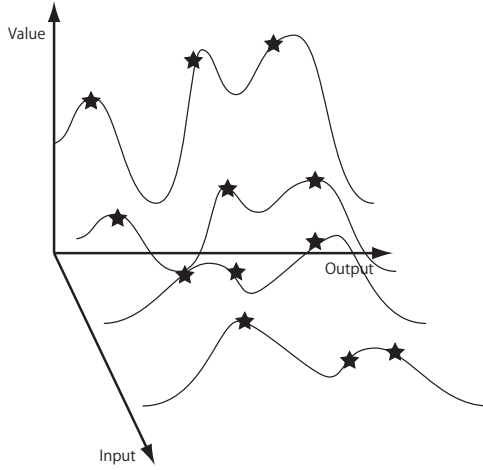


Fig. 6. Particles in the space

between state prototypes by setting the distance between successive best particles in the output-input space (in fig. 7). This is based on the fact that if the best output is quite different, near states should be searched closely, and otherwise do not need to be. In the process, the updated particle entrains the state particle.

Therefore, our proposed method requires that the input and output spaces be normalized.

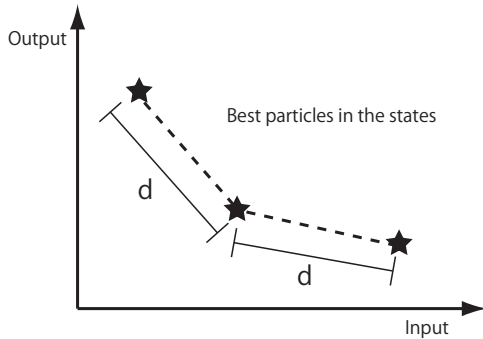


Fig. 7. Updating the input space distribution by distance

IV. SINGLE-PENDULUM SIMULATION

We simulated the single-pendulum standing task (fig. 8). This physical calculation is performed by a Runge-Kutta method.

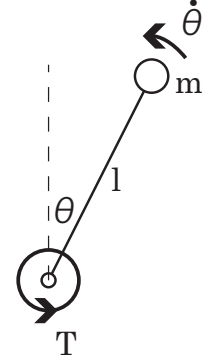


Fig. 8. Single pendulum

The following motion equation is simulated in increments of 0.05 seconds:

$$m\ddot{\theta} = -mgl \sin(\pi - \theta) + T + (-k\dot{\theta}) \quad (12)$$

where the parameters and variables are define as

- $m = 1$ (kg)
- $l = 1$ (m)
- $k = 0$
- $g = 9.8(m/s^2)$
- $-\pi < \theta \leq \pi$
- $-3\pi < \dot{\theta} \leq 3\pi$
- $-20 < \ddot{\theta} \leq 20$
- 1 episode = 5 seconds
- 1 learning cycle = 0.2 seconds
- $\theta = \pi, \dot{\theta} = \ddot{\theta} = 0$, in the initial state
- $R = (\pi - |\theta|)/\pi - T/40$
- $\alpha = 0.1$
- $\gamma = 0.9$
- variance $V_0 = \pi/2, V_{1000} = V_0 \cdot \beta^{1000} = 0.0001 \cdot V_0$ for constant β .

Figs. 9 to 11 show the results of several simulations. The vertical axes in these figures show the reward and the horizontal axes show episode times. In each figure, the left panel shows the 100-trial mean of the reward and the right panel shows the best reward. The shown results are for the normal method (blue), our proposed method in which couple particles are applied (green), and our proposed method along with adjusting the distance between state prototypes (red). The agent has 100 states in the learning process. In the simulations shown in Fig. 9, each state is segmented evenly. As shown, the proposed method is not always good for learning; however, it does not inhibit the learning process.

In Figs. 10 (over-segmented position space) and 11 (over-segmented velocity space), the state is not segmented appropriately. As shown, the proposed method leads to a better reward history in terms of speed and value.

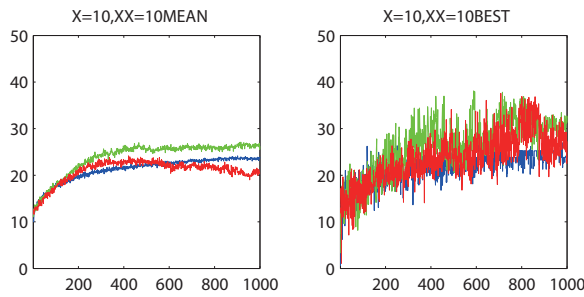


Fig. 9. Results in better segmentation by the particles
X=100, XX=1MEAN X=100, XX=1BEST

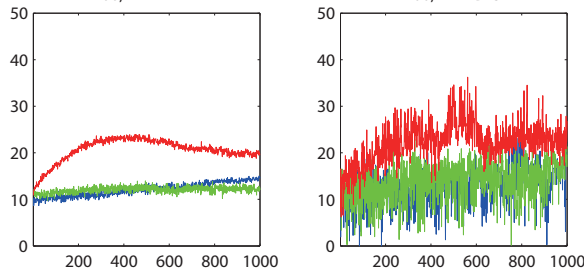


Fig. 10. Results in worse segmentation by the particles
X=1, XX=100MEAN X=1, XX=100BEST

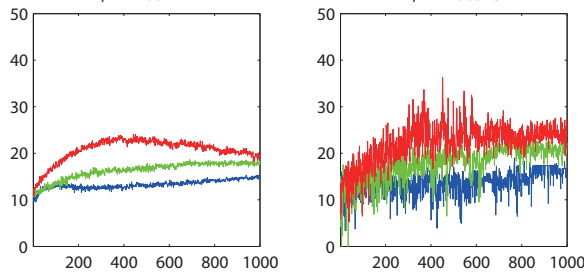


Fig. 11. Results in worse segmentation by the particles

Our simple segmentation method, especially in the state space, worked very well. Further adjustments in the parameters of the proposed method have the potential of further improving the results.

V. CONCLUSION

We proposed a novel particle-filtering approach to create a novel reinforcement learning algorithm that segments the agent's environment and action. The proposed filtering method is applied to a single-pendulum simulation in order to demonstrate the adaptation ability of the model.

ACKNOWLEDGMENT

This work was supported in part by the Ministry of Education, Culture, Science and Technology, Japan under Grant-in-Aid for Scientific Research No. 21700242 and funded by the Ministry of Internal Affairs and Communications, Strategic Information and Communications R&D Promotion Programme.

REFERENCES

- [1] T. Jaakkola, M. Jordan, and S. P. Singh, On the Convergence of Stochastic Iterative Dynamic Programming Algorithms, *Neural Computation*, vol. 6, pp. 341–362, 1992.

- [2] C. J. C. H. Watkins and P. Dayan, Technical Note: Q-Learning, *Machine Learning*, vol. 8, pp. 56–68, 1992.
- [3] A. Notsu, H. Ichihashi, K. Honda, State and Action Space Segmentation Algorithm in Q-learning, *Proc. of 2008 International Joint Conference on Neural Networks*, pp. 2385–2390, 2008.
- [4] A. Notsu, H. Wada, K. Honda, and H. Ichihashi, Cell Division Approach for Search Space in Reinforcement Learning, *International Journal of Computer Science and Network Security*, vol. 8, 6, pp. 18–21, 2008.
- [5] J. H. Holland and J. Reitman, *Cognitive Systems Based on Adaptive Algorithms*, in D. A. Waterman and F. Hayes-Roth (Eds.), *Pattern Directed Inference Systems*, Academic Press, 1978.
- [6] J. H. Holland, K. J. Holyoak, R. E. Nisbett, and P. R. Thagard, *Induction*, The MIT Press, 1986.
- [7] R. S. Sutton, *Learning to Predict by Method of Temporal Differences*, *Machine Learning*, vol. 3, 1, pp. 9–44, 1988.