Adaptive co-construction of state and action spaces in reinforcement learning

Masato Nagayoshi[a], Hajime Murao[b], and Hisashi Tamaki[c]

[a] Niigata College of Nursing, 240, Shinnan, Joetsu 943-0147, Japan
nagayosi@niigata-cn.ac.jp

[b] Faculty of Cross-Cultural Studies, Kobe Univ. 1-2-1, Tsurukabuto, Nada-ku, Kobe 657-8501, Japan
murao@i.cla.kobe-u.ac.jp

[c] Graduate School of Engineering, Kobe University, Rokko-dai, Nada-ku, Kobe 657-8501, Japan
tamaki@al.cs.kobe-u.ac.jp

Abstract

Reinforcement Learning (RL) attracts much attention as a technique of realizing computational intelligence such as adaptive and autonomous decentralized systems. In general, however, it is not easy to put RL into practical use. This difficulty includes a problem of designing suitable state and action spaces of an agent.

Until now, we have proposed an adaptive state space construction method which is called "state space filter" and an adaptive action space construction method which is called "switching RL", after the other space has been fixed. In this paper, we reconstitute these two construction methods as one method by treating the former method and the latter method as the combined method for mimicking infants' perceptual development in which perceptual differentiation progresses as infants become older and more experienced, and infants' motor development in which gross motor skills develop before fine motor skills respectively. Then the proposed method is based on introducing and referring to the "entropy". Further, a computational experiment was conducted by using a so-called "path planning problem" with continuous state and action spaces. As a result, the validity of the proposed method has been confirmed.

1 Introduction

Engineers and researchers are paying more attention to reinforcement learning (RL)[1] as a key technique of realizing autonomous systems. In general, however, it is not easy to put RL into practical use. Such issues as satisfying the requirement of learning speed, resolving the perceptual aliasing problem, and designing reasonable state and action spaces of an agent, etc. must be resolved. Our approach mainly deals with the problem of designing state and action spaces. By designing suitable state and action spaces adaptively, it can be expected that the other two problems will be resolved simultaneously. Here, the problem of designing state and action spaces involves the following two requirements: (i) to keep the characteristics (or structure) of an original search space as much as possible in order to seek strategies that lie close to the optimal, and (ii) to reduce the search space as much as possible in order to expedite the learning process. These requirements are, in general, in conflict.

Until now, we have proposed an adaptive state space construction method which is called "state space filter[2]" and an adaptive action space construction method which is called "switching learning system[3]" , after the other space has been fixed. In this paper, we reconstitute these two construction methods as one method by treating the former method and the latter method as the combined method for mimicking infants' perceptual and motor developments respectively. The proposed method is to construct state and action spaces adaptively by introducing and referring to the "entropy" as indexes of both necessity for division of the state space in the state and sufficiency for the number of learning opportunities in the state. Further, a computational experiment was conducted by using a so-called "path planning problem" with continuous state and action spaces.

2 Typical RL Methods

2.1 Q-learning

Q-learning works by calculating the Quality of a state-action combination, namely Q-value, that gives the expected utility of performing a given action in a given state. By performing an action $a \in \boldsymbol{\mathcal{A}}_{\mathrm{Q}}$, where $\boldsymbol{\mathcal{A}}_{\mathrm{Q}} \subset \boldsymbol{\mathcal{A}}$ is the set of available actions in Q-learning and $\boldsymbol{\mathcal{A}}$ is the action space of the agent, the agent can move from state to state. Each state provides the agent a reward r.

The Q-value is updated according to the following formula, when the agent is provided the reward :

$$Q(s(t\text{-}1), a(t\text{-}1)) \qquad Q(s(t\text{-}1), a(t\text{-}1))$$

$$+\alpha_{\mathrm{Q}}\{r(t\text{-}1) + \quad \max_{b\in\boldsymbol{\mathcal{A}}_{\mathrm{Q}}} Q(s(t),b) - Q(s(t\text{-}1),a(t\text{-}1))\}(1)$$

where $Q(s(t\text{-}1), a(t\text{-}1))$ is the Q-value for the state and the action at the time step t-1, $\alpha_{\mathrm{Q}} \in [0,1]$ is the learning rate of Q-learning, $\quad \in [0,1]$ is the discount factor.

The agent selects an action according to the stochastic policy, $\quad (a|s)$, which based on the Q-value. $\quad (a|s)$ specifies probabilities for taking each action a in each state s. Boltzmann selection, which is one of the typical action-selection methods, is used in this research. Therefore, the policy $\quad (a|s)$ is calculated as follows:

$$(a|s) = \frac{\exp(Q(s,a)/\tau)}{\sum_{b\in\boldsymbol{\mathcal{A}}} \exp(Q(s,b)/\tau)} \tag{2}$$

where τ is a positive parameter.

2.2 Actor-Critic

Actor-Critic methods have a separate memory structure to explicitly represent the policy independent of the value function. The policy structure is called "Actor", which selects actions, and the estimated value function is called "Critic", which criticizes the actions made by the Actor. The Critic is a state-value function. After each action selection, the Critic evaluates the new state to determine whether things have gone better or worse than expected. That evaluation is TD-error:

$$(t\text{-}1) = r(t\text{-}1) + \quad V(s(t)) - V(s(t\text{-}1)) \tag{3}$$

where $V(s)$ is the state Value.

Then, $V(s(t\text{-}1))$ is updated according to Eq. (4) in the Critic, based on this $\quad (t\text{-}1)$. In parallel, it is updated for the stochastic policy, $\quad (a|s)$, in the Actor.

$$V(s(t\text{-}1)) \qquad V(s(t\text{-}1)) + \alpha_{\mathrm{C}} \quad (t\text{-}1) \tag{4}$$

where $\alpha_{\mathrm{C}} \in [0,1]$ is the learning rate of the Critic.

It is typical for the normal distribution to be used, shown in Eq. (5), as the stochastic policy in the Actor, when Actor-Critic is applied to a continuous action space.In this case, both the mean the mean $\mu(s)$ and the standard error of the mean $\sigma(s)$ about the normal distribution are calculated using TD-error $\quad (t\text{-}1)$ in the Actor, as Eq. (6),(7).

$$(a|s) = \frac{1}{\sigma(s)\sqrt{2}} \exp\left(\frac{-(a-\mu(s))^2}{2\sigma(s)^2}\right) \tag{5}$$

$$\mu(s(t\text{-}1)) \qquad \mu(s(t\text{-}1)) + \alpha_{\mu} \quad (t\text{-}1)(a(t\text{-}1) - \mu(s(t\text{-}1))) \tag{6}$$

$$\begin{aligned}\sigma(s(t\text{-}1)) &\qquad \sigma(s(t\text{-}1)) \\ &+\alpha_{\sigma} \quad (t\text{-}1)((a(t\text{-}1) - \mu(s(t\text{-}1)))^2 - \sigma(s(t\text{-}1))^2)\end{aligned} \tag{7}$$

where $\alpha_{\mu} \in [0,1], \alpha_{\sigma} \in [0,1]$ are the learning rate of the mean and the standard error of the mean respectively. Here, if Eq. (7) is used directly, the standard error could be 0 or a negative value. So, it is necessary for the setting of the standard error to be creative to specify the range, etc.

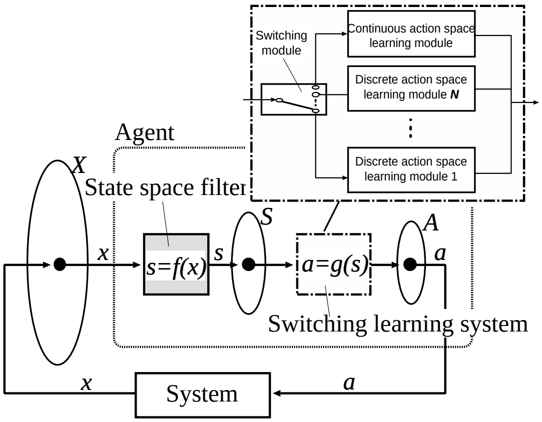
Switching module
Continuous action space learning module
Discrete action space learning module N
Discrete action space learning module 1
Agent
X
State space filter
S
A
x
s=f(x)
s
a=g(s)
a
Switching learning system
x
System
a

Figure 1: Proposed developing RL model.

3 Developing RL

3.1 Outline of a Computational Model

In this section, we propose developing RL model to mimic processes of infants' perceptual and motor developments. The proposed model is constructed by "state space filter[2]" to mimic a process of perceptual development in which perceptual differentiation progresses as infants become older and more experienced and "switching learning system[3]" to mimic a process of motor development in which gross motor skills develop before fine motor skills, as shown in Fig. 1.

This model mimics the process of perceptual development by differentiating the state space gradually from the undifferentiated state space. In parallel, this model mimics the process of motor development by switching discrete action space learning modules (hereafter called "DA module") from more coarse-grained DA module to more fine-grained DA module, and finally switching to a continuous action space learning module (hereafter called "CA module").

3.2 State and Action Spaces Construction Method

3.2.1 Basic Idea

A variety of methods to acquire the state space filter and to switch learning module can be considered. In this paper, we propose a method based on introducing and referring to the "entropy", which is defined on action selection probability distributions in a state, and the number of learning opportunities in the state. It is expected that the proposed method (i) is able to learn in parallel the state space filter and the switching learning system, (ii) does not required specific RL methods for the learning module.

The entropy of action selection probability distributions using Boltzmann selection in a state, $H_{\rm D}(s)$, is defined by

$$H_{\rm D}(s) = -(1/\log|\boldsymbol{\mathcal{A}}_{\rm D}|) \sum_{a\in\boldsymbol{\mathcal{A}}_{\rm D}} \quad (a|s)\log \quad (a|s) \qquad (8)$$

where $\boldsymbol{\mathcal{A}}_{\rm D}$ is the action space and $|\boldsymbol{\mathcal{A}}_{\rm D}|$ is the number of available actions of the DA module.

The state space filter is adjusted and the learning module is switched by treating this entropy $H(s)$ as an index of necessity of division for an inner state s and the action space. In parallel, the learning module is switched by treating this entropy $H(s)$ as an index of sufficiency for the number of learning opportunities in the state.

If the entropy does not get smaller despite being the learning module learned a sufficient number of opportunities in the inner state, then the state space filter is adjusted by dividing the inner state and the learning module is switched to more fine-grained one. In contrast, if the entropy get small regardless of the number of learning opportunities, the learning module is switched to the CA module due to the number of learning opportunities being sufficient.

In this paper, Q-learning and Actor-Critic are applied to the DA module and the CA module respectively. The learning module is switched in the order of Q-learning with an action space divided evenly into $n, 2n, \cdots, 2^{(N-1)}n$, finally ending with Actor-Critic, where N is a number of DA modules.

3.2.2 Adjustment of State Space Filter

If $L(s) > \ _L$ and $H(s) > \ _H$, where $L(s)$ is the number of learning opportunities in s, $\ _L$ is a threshold value of the number of learning opportunities, $\ _H$ is a threshold value of the entropy, and $\ _{\rm L}$ is set at a sufficiently big number, then the state space filter is adjusted by dividing a range of the input state mapped to the inner state s into 2 parts for each dimension, and mapping each part to a different inner state respectively. Simultaneously, the learning module is switched. Through this operation, a size of the inner state space after divided increases by $(2^M - 1)$, where M is a number of dimension. Also note that the values of the new 2^M inner states are the value of the inner state before divided.

In addition, after the learning module is switched to the CA module, if $L(s) > \ _L$, then the state space filter is adjusted by dividing the inner state to be more fine-grained.

3.2.3 Switching of Learning Module

If $H(s) > \ _H$, then the learning module is switched to the CA module due to the number of learning opportunities being sufficient. In the procedure to switch controllers, the result of Q-learning is succeeded by Actor-Critic. The following procedure is conducted : i) the state value of the Critic,$V(s)$, is initialized by

$$V(s) = \sum_{a\in\boldsymbol{\mathcal{A}}_{\rm Q}} \quad (a|s)\cdot Q(s,a) \qquad (9)$$

ii) the normal probability distribution used by the Actor is calculated by

$$\mu(s) = \arg\max_{a\in\boldsymbol{\mathcal{A}}_{\rm Q}} Q(s,a), \qquad (10)$$

$$\sigma(s) = |A_{\rm Q}(\arg\max_{a\in\boldsymbol{\mathcal{A}}_{\rm Q}} Q(s,a))|/6 \qquad (11)$$

where $|A_{\rm Q}(i)|$ is a range of the action space which represents action i of Q-learning.

If $L(s) > \ _L$ and $H(s) > \ _H$, then the learning module is switched to more fine-grained DA module, and finally ending with to the CA module. Simultaneously, the state space filter is adjusted.

Q-values of actions newly added a_i at this time are set according to the following formula : $Q(s,i) = \max_{j\in i-1,i+1} Q(s,j)$ where action $i-1$ and $i+1$ are adjacent to action i. This formula is set in consideration of a more efficient search as well as the idea of the optimistic initial values.

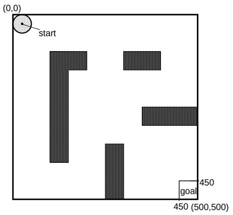

Figure 2: Path planning problem.

4 Computational Example

4.1 Path Planning Problem

The proposed method is applied to a so-called "path planning problem" where an agent is navigated from a start point to a goal area in a continuous space as shown in Fig. 2. Here, the agent has a circular shape (diameter = 50[mm]), and the continuous space is 500[mm] × 500[mm] bounded by the external wall with internal walls as shown in black. The agent can observe the center position of the agent: $(x_{\rm A}, y_{\rm A})$ as the input, and move 25[mm] in a direction, i.e., decide the direction: $\ _{\rm A}$ as the output.

The positive reinforcement signal $r_t = 10$ (reward) is given to the agent only when the center of the agent arrives at the goal area and the reinforcement signal $r_t = 0$ at any other steps. The period from when the agent is located at the start point to when the agent is given a reward, labeled as 1 episode, is repeated.

4.2 Comparison to Adaptive Methods

We have confirmed that a combined method of the state space filter and the switching learning system (hereafter called method "FS") demonstrates better performance than three Q-learning methods with the action space divided evenly into $4, 8$ and 16 ,and two Actor-Critic methods with the state space divided evenly into 10×10 and 40×40 in this task.

In this section, method FS is compared with two methods using the switching learning system with the state space divided evenly into 10×10 and 40×40 spaces (hereafter called method "S10" and "S40" respectively), Actor-Critic using the state space filter and Q-learning with the action space divided evenly into 4 spaces using the state space filter (hereafter called method "FAC" and "FQ4" respectively). Here, the initial state space filter is designed that divides the state space evenly into 10×10 spaces.

Then, the entropy of a continuous action space in a state for method FAS, $H_{\mathrm{C}}(s)$, is defined by

$$H_{\mathrm{C}}(s) = -\int_{-\infty}^{\infty} \quad (a|s)\log \quad (a|s)da. \tag{12}$$

By substituting the Eq. (5) into this formula, $H_{\mathrm{C}}(s) = \log(\sqrt{2\ \ e}\sigma)$. In method FAS, if $H_{\mathrm{C}}(s) < \ _{\mathrm{H_C}}$, then the state space filter is adjusted.

All initial values and the range of $\sigma(x)$ are set at and $[0.001, 2\ \ ]$ respectively, all initial means are set to randomize within a range of $[-\ \ ,\ \ ]$ for Actor-Critic. Then, all initial state values and Q-values are set at 5.0 as the optimistic initial values[1] for Actor-Critic and Q-learning respectively. Here, the initial values and the maximum limit of $\sigma(x)$ are set so that $\pm 1\sigma$ and the maximum limit become the size of the action space: $2\ \ $. Further, the adjustment of the state space filter is assumed until the third attempt in all inner states because it is impossible to evaluate sufficiency for division of the state space.

Computer experiments have been done with parameters as shown in Table 1. Here, $_{\mathrm{H_D}}$ was set referring to about 0.312 : the maximal value of the entropy when the highest selection probability for one action is 0.9, $_{\mathrm{H_C}}$ was set referring to about 0.335 : the entropy when the standard error σ is $\ /6$, $_{\mathrm{L}}$ was set in consideration of the enough big number.

The number of average steps required to accomplish the task was observed during learning over 20 simulations with various methods as described in Fig. 3.

Table 1: Parameters for experiments

Parameter	Value	Parameter	Value
$\alpha_{\mathrm{Q}}, \alpha_{\mathrm{C}}, \alpha_{\mu}, \alpha_{\sigma}$	0.1		0.9
$_{\mathrm{H_D}}$, $_{\mathrm{H_C}}$	0.3	τ	0.1
$_{\mathrm{L}}$	1000		

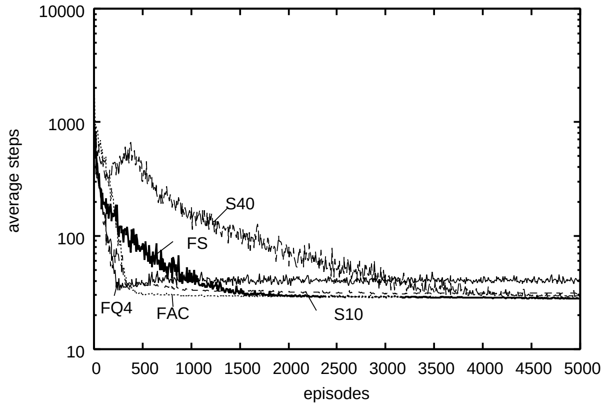

Figure 3: Required steps.

Learning speed and obtained control rule : It can be confirmed from Fig. 3 that, 1) method FS have worse performances than method FQ4, FAC and S10 , but better performances than method S40 with regard to the learning speed, 2) method FS has good performance as well as method FAC and S40 with regard to the obtained control rule,

Therefore, we have confirmed that method FAC and method FS, in that order, demonstrate better performance than any other method on the path planning problem with the continuous state and action spaces.

5 Conclusion

In order to design the suitable state and action spaces adaptively, we propose, in this paper, the developing RL model, and state and action spaces construction method referring to the "entropy". Then, through the computational experiment, we have confirmed that the combined method of the state space filter and Actor-Critic, and the combined method of the state space filter and the switching the learning system, in that order, demonstrate better performance than any other method on the path planning problem with the continuous state and action space.

Our future projects include to apply more complicated problems and real world problems, etc.

Acknowledgment
This work was supported in part by a Grant-in-Aid for Young Scientists (B), (No. 21700258), from MEXT, Japan.

References

[1] R.S. Sutton and A.G. Barto: Reinforcement Learning, A Bradford Book, MIT Press (1998).

[2] M. Nagayoshi, H. Murao, and H. Tamaki: A State Space Filter for Reinforcement Learning, *Proc. AROB 11th'06*, 615-618(GS1-3) (2006).

[3] M. Nagayoshi, H. Murao and H. Tamaki: A Reinforcement Learning with Switching Controllers for Continuous Action Space, Proc. the 15th International Symposium on Artificial Life and Robotics 2010, 236-239 (2010).