New Generation Computing, 24(2006)325-350
Ohmsha, Ltd. and Springer

NEW GENERATION COMPUTING

Tutorial Series on Brain-Inspired Computing

Part 4: Reinforcement Learning: Machine Learning and Natural Learning

Shin ISHII and Wako YOSHIDA
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara 630-0192, Japan
{ishii,wako-y}@is.naist.jp

Received 29 October 2005
Revised manuscript received 28 February 2006

Abstract The theory of reinforcement learning (RL) was originally motivated by animal learning of sequential behavior, but has been developed and extended in the field of machine learning as an approach to Markov decision processes. Recently, a number of neuroscience studies have suggested a relationship between reward-related activities in the brain and functions necessary for RL. Regarding the history of RL, we introduce in this article the theory of RL and present two engineering applications. Then we discuss possible implementations in the brain.

Keywords: Reinforcement Learning, Temporal Difference, Actor-critic, Reward System, Dopamine.

§1 Introduction

When a rodent is placed in a box, called a Skinner box, the rodent receives food when it happens to press the lever attached to the box. By continually receiving food following lever presses, the rodent associates a cause, the lever press, with an effect, the food, and comes to be motivated to press the lever; that is, a *reinforcement* occurs. This situation, in which an animal's behavior is modified according to its outcome, is called 'the law of effect', and is considered to be the most primitive aspect of behavioral learning. Although this example was a simple association between a cause and an effect, the case of general motor controls is more complicated. Let us consider, for example, learning to snowboard. When a learner is successful in performing well, on one hand, this outcome is seen as cool and so the behavior seems to be rewarded. Tumbling on the snow, on the other hand, could be painful and thus this outcome could be viewed as a punishment. Intuitively, learning complicated motor controls

in snowboarding would proceed based on such rewards or punishments, but understanding the process requires going beyond simple learning by association. The classical Rescorla-Wagner (R-W) model[41] can explain simple instrumental conditioning exemplified by the rodent case, while it cannot explain sequential motor learning such as occurs in learning to snowboard.

Reinforcement learning (RL)[55] is a natural extension of the R-W model to enable description of sequential learning by means of the conventional delta rule. Over the last decade in the field of neuroscience, the theory of RL has been extended to allow for explanations of not only lower-order neural activities related to rewards[34,47,50] but also higher-order brain activities involved in decision making,[48] although their evidence is still controversial.[15,32] Reinforcement learning has also been recognized as an approach to Markov decision processes[20] which were studied in the 1950s in the field of control theory, and which subsequently have become a major research topic in machine learning. In the field of artificial intelligence, RL is now attracting particular attention, because adaptability to and optimal control in complicated, dynamic and even unknown environments could be described by the RL theory. Paying attention to such a history of RL, in the first part of this article, we introduce the theory of RL and present two engineering applications. In the second part, we discuss possible implementations in the brain.

§2 Theory of Reinforcement Learning

2.1 Value-based Reinforcement Learning

As in Fig. 1, we have a controller (or agent) and a system (or environment) to be controlled (or interacted with). At a discrete time t, the controller emits a control signal, often called an action, a_t, based on the system's state x_t. The controller's function from a state x_t to an action a_t is called a *policy*, π, and represented as $a_t = \pi(x_t)$ or $a_t \sim \pi(a_t|x_t)$ for a deterministic or stochastic policy, respectively. The system is assumed to be probabilistic and memory-less, i.e., Markov, so that the state transition probability for reaching a next state x' from a previous state x by an action a is given by $P(x'|x,a)$. A deterministic system is regarded as a special case of the probabilistic system. We assume that

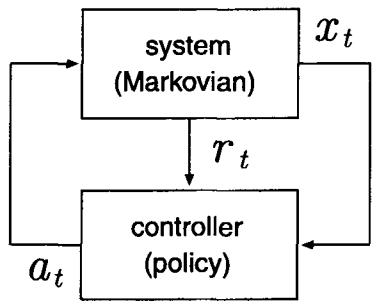

Fig. 1 The Problem Setting of Reinforcement Learning (Markov Decision Process)

when the system receives an action a_t at a state x_t, the controller receives a scalar response $r_t \equiv r(x_t, a_t)$, often called a *reward*, from the system, which represents the instantaneous goodness of the action a_t at the state x_t. We further define the *return*:

$$R_t \equiv \sum_{s=0}^{\infty} \gamma^s r_{t+s} = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots, \tag{1}$$

which represents a discounted summation of rewards to be received in the future starting from the state x_t at time t. Here, the discount factor γ is a meta-parameter to avoid the divergence of the summation. Using the above setting and definitions, the problem here is to obtain the 'optimal' policy π^* that maximizes the return from any state. This is an instance of a Markov decision process (MDP).

Although we deal in this section with deterministic policies for ease of explanation, a similar setting can be constructed for stochastic policies. Since the return (1) depends on the actual sequence of states and actions, we define its expectation with respect to possible sequences:

$$V^\pi(x) \equiv E\left[R_t | \, x_t = x\right] = Q^\pi(x, \pi(x)) \tag{2a}$$

$$\begin{aligned} Q^\pi(x,a) &\equiv E\left[R_t | \, x_t = x, a_t = a\right] \\ &= r(x,a) + \gamma \int_{x'} P(x'|x,a) V^\pi(x') dx'. \end{aligned} \tag{2b}$$

The function $V^\pi(x)$ represents the expected return when starting from a state x_t and employing a fixed policy π, and is called the *value function*. Similarly, the function $Q^\pi(x,a)$ represents the expected return when selecting an action a_t at the current state x_t and employing a policy π in subsequent states, and is called the action-value function or *Q-function*. Using these definitions, the above MDP is equivalent to the problem of obtaining the optimal policy π^* that maximizes the value function $V^\pi(x)$ for every state x.

Maximization of the value function (2a) with respect to policy π is a functional optimization, and seems difficult at first glance. Bellman's optimization principle[6] tells us, however, that this optimization problem is equivalent to obtaining the optimal value function that satisfies at every state x the following optimal Bellman equation:

$$V^*(x) = \max_a Q^*(x,a) \tag{3a}$$

$$Q^*(x,a) = r(x,a) + \gamma \int_{x'} P(x'|x,a) V^*(x') dx'. \tag{3b}$$

This is a function optimization problem for each state, and seemingly an easier problem than the functional optimization. Using the optimal value function, the optimal policy we seek is given by $\pi^*(x) \equiv \arg\max_a Q^*(x,a)$. When the state and action spaces are discrete, the optimization problem (3) can be solved efficiently

by dynamic programming (DP) methods. The famous RL algorithm called *Q-learning*[60] is a stochastic version, i.e., the stochastic approximation based on actual sampling, of such DP methods.

By putting $a_t = \pi(x_t)$ into (2), we see

$$V^\pi(x_t) \approx r(x_t, a_t) + \gamma V^\pi(x_{t+1}) \tag{4}$$

is approximately satisfied. Since we have ignored probabilistic nature of the state transition in the right-hand-side, the equality holds probabilistically. The difference between the right-hand-side and left-hand-side:

$$\delta_t \equiv r(x_t, a_t) - (V^\pi(x_t) - \gamma V^\pi(x_{t+1})) \tag{5}$$

is called the *temporal difference* (TD) error, and its expectation should approach zero as learning of the value function proceeds. Based on the conventional delta (Widrow-Hoff) rule, such learning can be performed as to make the TD error small:

$$V^\pi(x_t) := V^\pi(x_t) + \alpha_c \delta_t \tag{6}$$

is the famous *TD-learning.*[54] If the system is stationary, the learning coefficient α_c is often reduced as learning proceeds, so as to realize the stochastic approximation of the value function (2). Note that this TD-learning is to obtain the value function for a certain policy π, which is not necessarily optimal, and therefore, the policy improvement should be performed outside of the TD-learning (6). If V^π for the policy π is approximated by TD-learning, then the policy can be improved by using a good approximation to the value function; this is the *policy iteration* method. A more intelligent method performs policy learning by updating a specific utility function based on the TD error, concurrent with value learning. The *actor-critic* method,[4] the first RL method, implemented such a concurrent learning scheme, and is apparently much more efficient than simple policy iteration.

2.2 Partially Observable MDP

In the previous section, we assumed that the state variable x is completely observable and obeys a Markov process. In reality, however, it is often the case that only part of the real state is observable with the remainder forming an associated unobservable state variable. Therefore, we assume for this case that the real state variable of the system $s \in \mathcal{S}$ obeys an underlying Markov process $P(s'|s,a)$ but that this process is intrinsically unobservable. Instead the controller (or agent) can access an observation variable $x \in \mathcal{X}$ according to an observation process $P(x|s)$. The reward condition is defined similarly to that for the MDP in section 2.1. A partially observable MDP (POMDP)[23] is a problem of obtaining the optimal policy in such a partially observable environment. Since the controller cannot observe directly the system state s in a POMDP, an ideal controller has to estimate the system state s from the history of observations and to decide the action based on the estimation. When the controller estimates the system state s_t at time t based on the history of past observable variables

(observation and action), $H_t \equiv \{x_t, a_{t-1}, x_{t-1}, \ldots, a_1, x_1\}$, as a probability distribution $P(s_t|H_t)$, the distribution is called a *belief state* and represented as $b_t \in \mathcal{B}$. A belief state is a probability distribution of states s, $b_t \equiv P(s_t|H_t)$. If we know the Markov transition and observation processes, a belief state can be calculated according to the incremental Bayes formula:

$$b_{t+1}(s_{t+1}) = \frac{P(x_{t+1}|s_{t+1}) \sum_{s_t \in \mathcal{S}} P(s_{t+1}|s_t, a_t) b_t(s_t)}{P(x_{t+1}|a_t, H_t)}, \tag{7}$$

which represents the 'state transition' of belief states.

The objective of a POMDP controller is to obtain the optimal policy π^* that maximizes the value function for the system state, $V^\pi(s)$, for every state. Since the controller cannot directly observe the state s, however, this is almost impossible. Instead, we consider maximizing its expectation with respect to belief:

$$V^\pi(b_t) \equiv \sum_{s_t \in \mathcal{S}} b_t(s_t) V^\pi(s_t), \tag{8}$$

for every belief state. The optimization problem for MDPs was a functional optimization. Since the value function for POMDPs is itself a functional, a function of belief states, the optimization problem here is more difficult, and a solution often needs some approximations. Moreover, we must calculate summations over the system state $s \in \mathcal{S}$ as in equations (7) and (8), which becomes intractable as the number of possible states, $|\mathcal{S}|$, increases, even in a discrete-state space. It should be noted that the denominator in equation (7) is more concretely described as

$$P(x_{t+1}|a_t, H_t) \equiv \sum_{s_{t+1} \in \mathcal{S}} P(x_{t+1}|s_{t+1}) \sum_{s_t \in \mathcal{S}} P(s_{t+1}|s_t, a_t) b_t(s_t), \tag{9}$$

and its calculation requires taking sums twice over states; this calculation may also be intractable.

Corresponding to the optimal Bellman equation (3) in MDPs, the optimal value function in POMDPs, $V^*(b)$, satisfies[51] the Bellman equation for belief states:

$$V^*(b) = \max_a Q^*(b, a) \tag{10a}$$

$$Q^*(b, a) = r(b, a) + \gamma \int_{b'} P(b'|b, a) V^*(b') db'. \tag{10b}$$

Because a belief state should be a sufficient statistic which adequately represents the information contained in the history of past observations, the stochastic process on belief states should be Markov, and hence $P(b'|b, a)$ is stationary. Since the process on belief states becomes an MDP, this POMDP formulation is called a *belief-state MDP*.

The objective in a belief-state MDP is to maximize the functional $Q^*(b, a)$ with respect to the variable a for each belief state b, as can be seen in (10a), but there are two difficulties:

1. The value function $V^*(b)$ and Q-function $Q^*(b,a)$ are both functionals, and it is difficult to represent them.
2. The update of belief states requires application of the incremental Bayes formula, equation (7), but this calculation involves summing over states, which is difficult.

Some methods, for example, the witness algorithm,[9] have been proposed for solution of belief-state MDPs based on exact belief states, but they can be applied only to problems whose state space is relatively small. Recently, various methods to approximate the value function of belief states or to approximately represent belief states have been studied extensively (e.g., Reference[8,36,40,44]).

2.3 Policy-based Reinforcement Learning

Value-based RL methods, such as Q-learning and TD-learning, were very popular several years ago, but some of those methods led to unstable learning, especially when function approximators were used, partly because a small change in the value function may result in a significant change in the optimal policy. On the other hand, policy-based methods like Williams' REINFORCE[61] attempted to improve policy based on observed sequences of rewards. Since the methods of the latter type do not use value functions, their learning, although stable, proceeded extremely slowly.[56] Very recently, however, some policy-based methods, which incorporate value learning have been proposed which result in both stable and efficient learning.

Let a policy π be stochastic and parameterized by a parameter θ as π^θ. By assuming the Markov system has a stationary distribution of states, $D^\theta(x)$, the stationary distribution of state-action pairs is given by $D^\theta(x,a) \equiv D^\theta(x)\pi^\theta(a|x)$. The expectation of reward with respect to the stationary distribution is then given[26,56] by

$$\rho(\theta) \equiv \int_{x,a} r(x,a) D^\theta(x,a) da dx \equiv \langle r(x,a) \rangle_\theta .$$

In the current and the next sections, an MDP is defined so as to obtain the parameter θ that maximizes the expected reward $\rho(\theta)$. Although this problem is slightly different from the maximization problem of reward accumulation in section 2.1, a similar discussion to the following may be made if we use the reward accumulation as the objective function.

Let the return be the sum of the differences between the reward and the expected reward, from a state x_t at time t over all future states:

$$\begin{aligned} R_t &\equiv \sum_{s=0}^{\infty} (r_{t+s} - \rho(\theta)) \\ &= (r_t - \rho(\theta)) + (r_{t+1} - \rho(\theta)) + (r_{t+2} - \rho(\theta)) + \ldots \end{aligned}$$

With this definition, the value function under a policy π^θ is given by

$$V^\theta(x) \equiv E\left[R_t \middle| x_t = x\right] = \int_a \pi^\theta(a|x) Q^\theta(x,a) da \qquad (11a)$$

$$Q^\theta(x,a) \equiv E\left[R_t \middle| x_t = x, a_t = a\right]$$
$$= (r(x,a) - \rho(\theta)) + \int_{x'} P(x'|x,a)V^\theta(x')dx'. \qquad (11b)$$

Note that the value function V^θ and the Q-function Q^θ here are different by $-\rho(\theta)$ from those in section 2.1.

In the above-defined MDP, the differential of the expected reward with respect to the parameter is given by

$$\frac{\partial \rho(\theta)}{\partial \theta} = \int_{x,a} D^\theta(x,a)Q^\theta(x,a)\psi^\theta(x,a)dadx$$
$$= \langle Q^\theta(x,a)\psi^\theta(x,a)\rangle_\theta, \qquad (12)$$

where $\psi^\theta(x,a)$ is a compatible function vector defined as

$$\psi^\theta(x,a) \equiv \frac{\partial \log \pi^\theta(a|x)}{\partial \theta} = \frac{1}{\pi^\theta(a|x)}\frac{\partial \pi^\theta(a|x)}{\partial \theta}. \qquad (13)$$

Since θ is usually a parameter vector, $\psi^\theta \equiv \{\psi_i^\theta\}$ is a set (vector) of compatible function elements. Equation (12) is called the *policy gradient theorem* (for a brief proof, see Appendix A), and it states that if the correlation between the Q-function and the compatible function, given by the right-hand-side of equation (12), can be estimated, the policy can be (locally) optimized by modifying it in that direction. This is a *policy gradient method*, which is different in concept from the value-based RL methods presented in section 2.1.

2.4 Policy-gradient Actor-critic Learning

In this section, an elegant policy gradient learning method, *policy-gradient actor-critic learning*,[26] is introduced. Let a policy π have an n-dimensional parameter θ. If we define an inner product $\langle\langle \cdot, \cdot \rangle\rangle_\theta$ of two functions, $q_1(x,a)$ and $q_2(x,a)$, by

$$\langle\langle q_1(x,a), q_2(x,a)\rangle\rangle_\theta \equiv \int_{x,a} D^\theta(x,a)q_1(x,a)q_2(x,a)dadx,$$

and a set of basis functions, $\Psi \equiv \{\psi_i^\theta | i = 1, \ldots, n\}$, given by equation (13) automatically from the policy's parameterization, then $Q^\theta(x,a)$ can be projected onto a function space spanned by Ψ as

$$\Pi_\Psi Q^\theta(x,a) = \arg\min_{\hat{Q}\in\{\Psi\}} \|Q^\theta - \hat{Q}\|_\theta = w^T\psi^\theta(x,a) \equiv Q^w(x,a), \qquad (14)$$

where $\|\cdot\|_\theta$ denotes the norm induced by the inner product above, and T is the transpose. Since $Q^w(x,a)$ is in the space spanned by Ψ, it is represented as a linear combination of the basis functions. From equations (12) and (14), we have

$$\frac{\partial \rho(\theta)}{\partial \theta} = \langle Q^\theta \psi^\theta\rangle_\theta = \langle \left(\Pi_\Psi Q^\theta\right)\psi^\theta\rangle_\theta = \langle Q^w\psi^\theta\rangle_\theta.$$

This leads to the fact that the policy gradient is not sensitive even if the Q-function is projected by Π_Ψ, indicating that the policy gradient can be estimated without any bias if we approximate the Q-function as a linear combination of the basis functions in Ψ. As an instance of TD-learning (equation (6)), then, parameter w of the Q-function (or critic) can be updated by

$$\Delta w = \alpha_c \delta_t \psi^\theta(x_t, a_t),$$

where δ_t is a TD error:

$$\delta_t \equiv r(x_t, a_t) - \hat{\rho} - (Q^w(x_t, a_t) - Q^w(x_{t+1}, a_{t+1})).$$

This TD error defined for the Q-function is different from that in equation (5). The TD-learning for the Q-function is sometimes called SARSA.[55] The parameter for the policy (or actor) can be adjusted so as to approximate equation (12) from actual samples:

$$\Delta\theta = \alpha_a Q^w(x_{t+1}, a_{t+1}) \psi^\theta(x_{t+1}, a_{t+1}). \tag{15}$$

This is policy-gradient actor-critic learning. Note that the update in equation (15) can be accelerated by using the eligibility trace, which encourages the propagation of rewards along sample sequences.[55] It has been proved that policy-gradient actor-critic learning converges under certain conditions such as employing appropriate scheduling of the learning coefficients, α_c and α_a.[26] The approximate Q-function in equation (14), $Q^w(x, a)$, is n-dimensional. However, since the dimensionality of the real Q-function (11b) is usually high, reflecting the nonlinearity of the target system, the convergence is very slow due to the variance stemming from the model difference in (14) if the dimensionality of the approximate Q-function is low. To avoid such variance, then, we may construct a richer set of basis functions, $\Phi \equiv \{\phi_i | i = 1, \ldots, m, \phi_i = \psi_i, i = 1, \ldots, n, m \geq n\}$ so that the Q-function is represented in that function space, i.e., $Q^{\tilde{w}}(x, a) = \tilde{w}^T \phi(x, a)$.

In the meantime, the linear weight vector w in equation (14) is given as a least mean square solution:

$$w = \arg\min_w \|Q^\theta(x, a) - Q^w(x, a)\|_\theta,$$

leading to

$$w = \langle Q^\theta(x, a) \psi^\theta(x, a) \rangle_\theta \langle \psi^\theta(x, a) \psi^\theta(x, a)^T \rangle_\theta^{-1}. \tag{16}$$

Here, the numerator is identical to the policy gradient, equation (12), and the denominator is a positive semi-definite matrix, so w is in the same direction as the policy gradient. Therefore, the actor's parameter θ can be modified by making the modification proportional to w, $\Delta\theta = \alpha_a w$, instead of equation (15).[24] Because the denominator is formally the Fisher information matrix of the parametric stochastic policy π^θ:

$$\mathcal{F}_{i,j} = \left\langle \frac{\partial \log \pi^\theta(a|x)}{\partial \theta_i} \frac{\partial \log \pi^\theta(a|x)}{\partial \theta_j} \right\rangle_\theta,$$

equation (16) is in the same form as the natural gradient;[2] this actor-critic learning is then called *natural actor-critic*.[24,37] By using the second-order differential, learning may be expected to be accelerated, especially in the region where the policy gradient is small.

§3 Applications to Artificial Intelligence

3.1 RL for a Multi-agent Card Game

In order to understand possible mechanisms underlying communication in terms of RL, multi-agent games are well-defined testbeds. Acquisition of optimal controls through trial and error, by coping with unobservability and non-stationarity in an environment, is similar to the establishment of communication with unfamiliar companions. To perform effective communication, prediction of the 'internal state' of the companion is an important process. In the setting of multi-agent games, this process corresponds to the estimation of a hidden state and/or an internal strategy of the opponent agent.

We apply here an RL scheme to an automatic strategy acquisition problem for the multi-agent (four players) card game "Hearts"[22] (game rules are described in Appendix B). Because cards held by the opponent players cannot be observed by a learning agent, each opponent agent has an internal state. An optimal control problem in this environment is formulated as a POMDP, defined in section 2.2, and then our task, namely, to learn how to play the card game Hearts, is a typical and realistic POMDP problem. We present here a model-based RL approach to this difficult POMDP problem.

Let $M^i (i = 1, 2, 3)$ denote the i-th opponent agent. We assume that agent M^i probabilistically determines its action a_t^i based on its own observation x_t^i from a real state s_t^i, at his t-th playing turn. Under this assumption, the state transition between the t-th play and the $(t+1)$-th play of the learning agent, i.e., $s_t^0 \xrightarrow{a_t} s_t^1 \xrightarrow{a_t^1} s_t^2 \xrightarrow{a_t^2} s_t^3 \xrightarrow{a_t^3} s_t^4$, is given by

$$P(s_{t+1}|s_t, a_t) = \sum_{s_t^1, s_t^2, s_t^3} \sum_{a_t^1, a_t^2, a_t^3} \prod_{j=0}^{3} P(s_t^{j+1}|s_t^j, a_t^j) \prod_{i=1}^{3} \sum_{x_t^i} P(a_t^i|x_t^i) P(x_t^i|s_t^i), \tag{17}$$

where $s_t^0 = s_t$, $a_t^0 = a_t$, $s_t^4 = s_{t+1}$ and $x_t^0 = x_t$. x_t or x_t^i is an observation, which is partial, from s_t or s_t^i for the learning agent or an opponent agent M^i, respectively.

Due to the deterministic game process of Hearts, there are two facts:

- The new state s_t^{i+1} reached from a previous state s_t^i by an action a_t^i, is uniquely determined. Namely, $P(s_t^{i+1}|s_t^i, a_t^i)$ is 1 for a certain state and 0 for the other states.

- The observation, x_t or x_t^i, is uniquely determined at state s_t or s_t^i. Namely, $P(x_t|s_t)$ or $P(x_t^i|s_t^i)$ is 1 for a certain observation state and 0 for the other observation states.

Since state s_t is not observable for the learning agent, it should be estimated somehow, for example, as a belief state, using the history of the current game, $H_t \equiv \{(x_t, -, a_t^{1,2,3}), (x_{t-1}, a_{t-1}, a_{t-1}^{1,2,3}), \ldots, (x_1, a_1, a_1^{1,2,3})\}$.

Because the state space of this game, defined by the distribution of 52 cards, is huge, solving exactly the belief-state MDP, defined by equation (10), is almost impossible. Therefore, we here use the following approximate Bellman equation, instead of the exact belief-state Bellman equation (10):

$$V(x) = \max_a Q(x,a) \tag{18a}$$

$$Q(x,a) \approx r(x,a) + \int_{x' \in \mathcal{X}} P(x'|H,a)V(x')dx'. \tag{18b}$$

Since we use value functions for observation states, which are no longer functionals, this Bellman equation is just an approximation. Also that the value function defined for observation states suffers from large approximation variance, due to the problem called 'perceptual aliasing'. However, solving this problem is much easier than solving the belief-state MDP (10). It should be noted that if we replace $P(x'|H,a)$ by $P(x'|x,a)$, equation (18) becomes a naive MDP approximation of the original POMDP, by ignoring the unobservable part of the system state s, and hence may lead to a very poor solution. After solving the approximate Bellman equation (18), the policy is simply given by $\pi(x) = \arg\max_a Q(x,a)$.

Since the calculation of the transition probability for the observation state, $P(x'|H,a)$, requires taking sums (or integrating) twice as can be seen in equation (9), which is intractable in our case, we need further approximations. First, using the above-mentioned deterministic characteristics of the game and the assumed probabilistic nature of the opponent agents, we obtain

$$\begin{aligned}
&P(x_{t+1}|H_t, a_t) \\
&= \sum_{s_{t+1} \in \mathcal{S}_{t+1}} P(x_{t+1}|s_{t+1}) \sum_{s_t \in \mathcal{S}_t} P(s_{t+1}|s_t, a_t) b_t(s_t) \\
&= \sum_{s_{t+1} \in \mathcal{S}_{t+1}^-(x_{t+1})} \sum_{s_t \in \mathcal{S}_t} \sum_{d_t \in \mathcal{D}_t} P(s_{t+1}|s_t, d_t, a_t) P(d_t|s_t, a_t) b_t(s_t) \\
&= \sum_{d_t \in \mathcal{D}_t^-(x_{t+1}, H_t)} \sum_{s_t \in \mathcal{S}_t} P(d_t|s_t, a_t) b_t(s_t) \\
&= \sum_{d_t \in \mathcal{D}_t^-(x_{t+1}, H_t)} \sum_{s_t \in \mathcal{S}_t} b_t(s_t) \prod_{i=1}^{3} P(a_t^i|x_t^i, a_t),
\end{aligned} \tag{19}$$

where $d_t \equiv (a_t^1, a_t^2, a_t^3)$. $S_{t+1}^-(x_{t+1})$ is the set of system states s_{t+1} whose observation becomes x_{t+1} at time $t+1$, D_t is the set of allowable actions by the opponents in their t-th turn, and $D_t^-(x_{t+1}, H_t)$ the set of actions by the opponents which lead to an observation x_{t+1} at time $t+1$. Next, we use the following mean-field like approximation:

$$\sum_{s_t \in S} b_t(s_t) P(d_t|s_t, a_t) \approx P(d_t|\langle s_t|b_t\rangle_{b_t}, a_t).$$

By applying this approximation, equation (19) becomes

$$P(x_{t+1}|a_t, H_t) \approx \sum_{d_t \in D_t^-(x_{t+1}, H_t)} \prod_{i=1}^{3} P(a_t^i|\langle x_t^i|a_t, H_t\rangle_{b_t}, a_t), \qquad (20)$$

where $\langle x_t^i|a_t, H_t\rangle_{b_t} \equiv \sum_{x_t^i} x_t^i P(x_t^i|a_t, H_t)$. This approximation replaces the expectation of state transition with respect to belief states by a single state transition on an expected state with respect to belief states, and successfully removes one of the summations in equation (9).

It has been assumed that each opponent agent determines its action a_t^i with probability $P(a_t^i|x_t^i, a_t)$. However, this action selection probability and the real observation state x_t^i are unknown for the learning agent and they should be estimated in some way. Therefore, the learning agent assumes that the action selection process can be approximated by a stochastic process that is dependent on the mean estimated observation $\langle x_t^i|a_t, H_t\rangle_{b_t}$. Since the mean estimated observation incorporates the history of the current game H_t (or b_t) and the game knowledge, it provides essential information about the belief state b_t. Therefore, the stochastic process dependent on a discrete but unobservable observation state can be approximated by a stochastic process dependent on an analog (mean) and estimated observation state.

The strategy of an opponent agent M^i is represented and learned by using a function approximator. For a game finished in the past, an observation state and an action taken by each opponent agent at that state can be reproduced by replaying the game from the end to the start. To train the function approximator for M^i, the input and the target output are given by $\langle x_t^i|a_t, H_t\rangle_{b_t}$ and the action a_t^i actually taken by agent M^i at that turn, respectively.

Although equation (20) still requires a summation, corresponding to a traversal of a game tree, this difficulty is coped with by employing an appropriate algorithm to cut unnecessary branches in the game tree.

We carried out computer simulation experiments using one learning agent based on our RL method and three rule-based opponent agents. The rule-based agent has more than 50 rules so that it is an "experienced" level player of the game Hearts. The penalty ratio was 0.41 when an agent who only took out permitted cards at random from its hand challenged the three rule-based agents. The penalty ratio is the ratio of penalty points acquired by the learning agent

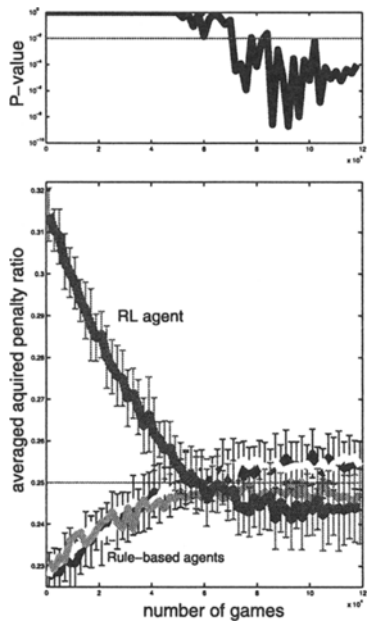

Fig. 2 A computer simulation result using one learning agent trained by our RL method and three rule-based agents. (bottom) Abscissa denotes the number of training games, and ordinate denotes the penalty ratio acquired by each agent, which is smoothed by using 2,000 games just before that number of training games. We executed twenty learning runs, each consisting of 120,000 training games, and each line in the figure represents the average over the twenty runs. For the proposed RL agent, the error bars denote the standard deviation over the twenty runs. (top) P-values of the statistical t test. The null hypothesis is "the proposed RL agent is equal in strength to the rule-based agents", and the alternative hypothesis is "the proposed RL agent is stronger than the rule-based agents". The statistical test was done independently at each point on the abscissa.

to the total penalty points of the four agents. That is, a random agent acquired about 2.1 times as many penalty points as the rule-based agents on average. Figure 2 shows the learning curve of an agent trained by our RL method when it challenged the three rule-based agents. This learning curve is an average over twenty learning runs, each of which consisted of 120,000 training games. After about 80,000 games playing with the three rule-based agents, our RL agent came to acquire a smaller penalty ratio than the rule-based agents. Namely, the RL agent became stronger than the rule-based agents. By observing the results of the twenty learning runs, we found that the automatic strategy acquisition could be achieved in a stable fashion by our RL method.

Figure 3 shows the result when two learning agents trained by our RL method and two rule-based agents played with each other. After about 50,000 training games, both learning agents became stronger than the rule-based agents.

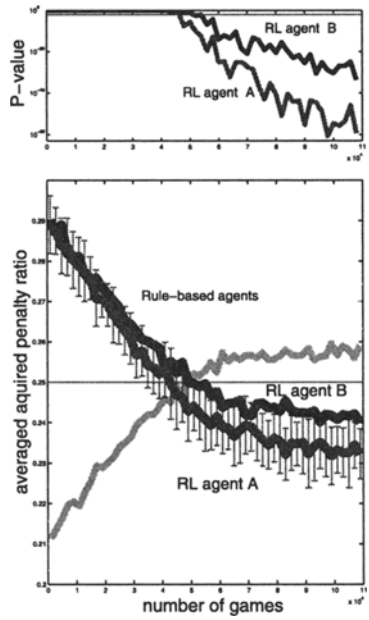

Fig. 3 A computer simulation result when two learning agents trained by our RL method and two rule-based agents played against each other. The meanings of the axes are the same as those in Fig. 2. In this simulation, the sitting positions of the four agents were fixed throughout the training run. This is the reason why the RL agent A got stronger than the RL agent B.

3.2 RL for Biped Locomotion

Next, we[33] applied the natural actor-critic RL method to the automatic control problem of a biped robot simulator formerly proposed by Taga et al..[57] The biped robot simulator is composed of five connected links, as depicted in Fig. 4(a). The motion of the links is restricted in the sagittal plane. Control torque is applied to each of the six joints, as $\tau_1, \ldots, \tau_6$. Intuitively, it is not easy to control this unstable dynamical system by RL, based on trial and error,

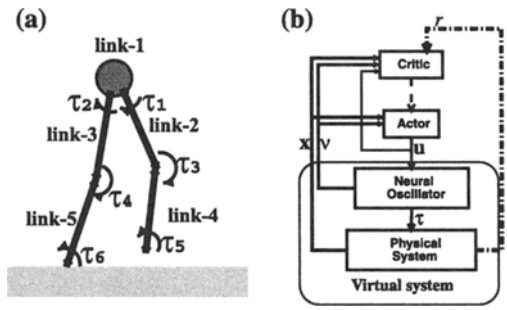

Fig. 4 **(a)** The Configuration of the Biped-robot Simulator **(b)** CPG-actor-critic Architecture

without any restriction on the controller; searching the space of possible controllers for an appropriate one suffers from the "curse of dimensionality". Taga et al.[57] implemented a neural oscillator network called a central pattern generator (CPG) network[18] to control this dynamical system, which was effective in restricting the controller to a subspace, in which the controller emitted only oscillatory control signals. Referring to this existing study, the problem here is to train the CPG controller by the RL method. We developed two strategies for the implementation.

- A naive RL cannot be directly applied to training of the CPG controller. Since the CPG controller is a recurrent neural network possessing its own dynamics, it is not possible to produce any control signal at a time instance, which makes it hard to use the CPG as an RL controller. To avoid this problem, we divided the CPG controller into two parts, the basic CPG and the actor, where the actor had no mutual connections and hence was a feed-forward neural network. This architecture is called a *CPG-actor-critic* model (see Fig. 4(b)).[31,46]
- In order to effectively search the controller space for an appropriate one, 'exploration' for the optimal policy is necessary, while in order to maximize the reward, 'exploitation' of the current (semi-)optimal policy is required.[13,55] Although 'meta'-control of exploration and exploitation is an important issue, naive control of the policy's randomness leads to instability of the RL process. To overcome this problem, we introduced a behavior policy, which could differ from the policy itself, to explore the controller space, and the estimation of the policy gradient was carried out using an importance sampling technique.[33,38,49] The meta-control of exploration and exploitation[21] was performed by controlling the variation of behavior policies. This new method achieved an efficient search (exploration) and the stabilization of an RL process simultaneously and hence was beneficial for accelerating the RL process.

We examined whether our method was able to produce a CPG controller, which enabled the biped robot to walk stably. Before learning, the parameters of the actor and the critic were set to small random values, which resulted in the robot soon falling down as can be seen in Fig. 5(a). Figure 6 shows the learning curve; the horizontal axis denotes the number of episodes, and the vertical axes denote (a) the average reward per step and (b) the episode duration before five seconds had elapsed or the robot fell down. After about 1,200 training episodes, equivalent to a total training time in the physical world of 100 minutes, the robot was less likely to fall down before five seconds had elapsed, and the average reward had increased substantially. Figure 5(b) shows biped locomotion by the robot after 3,000 learning episodes. When we conducted 100 times of 1 min test walking, the actor after 3,000 learning episodes (Fig. 5(b)) completed 99 times, while the actor before training (Fig. 5(a)) was unsuccessful in any of the 100 times.

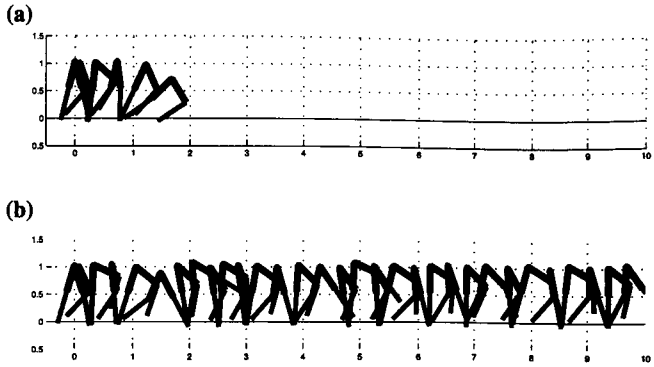

Fig. 5 **(a)** The Movement of the Biped Simulator before Learning **(b)** The Locomotion by the Biped Simulator after 3,000 Learning Episodes

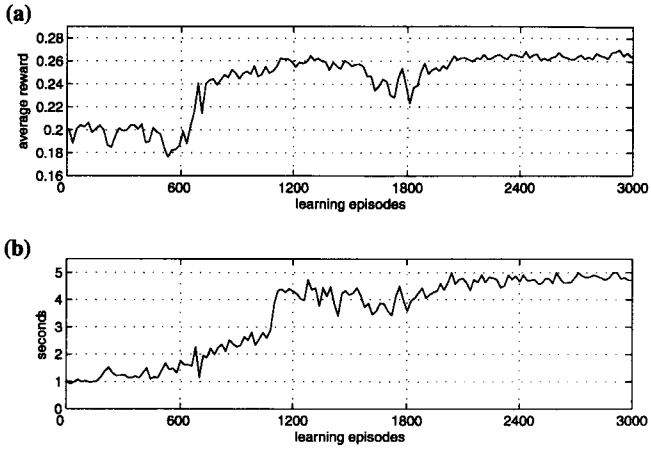

Fig. 6 The learning curves, where the abscissa denotes the number of learning episodes. **(a)** The average reward per step in an episode **(b)** The episode length in an episode, which is defined as the duration (in seconds) before five seconds elapsed or the robot fell down

§4 Reinforcement Learning in the Brain

4.1 Dopamine Represents Reward in Brain

RL is a learning scheme to acquire an action sequence, which maximizes the sum of rewards accumulated over the future, and it has been reported that the reward (reinforcer) is also represented in the brain. In 1954, Olds and Milner[35] first identified the brain region responsible for processing a reward, which is defined, in operant psychology, as a stimulus that increases the probability of operant or instrumental responses that precede or are contingent upon the stimulus.[43,52] They applied direct electrical stimulation to the limbic system of rats when the rats happened to press a lever, and found that the rats became very

willing to lever-press. This phenomenon, called 'intracranial self-stimulation', has been found in a variety of animals including humans. It has also been found that the reward function is implemented in several places in the brain centering around the limbic system, involving connected regions such as the ventral tegmental area (VTA), amygdala, nucleus accumbens (NA) and prefrontal cortex (PFC). Since the blocking of dopamine (DA) from the VTA to the NA suppresses the self-stimulation,[28] the activity of DA neurons has been thought to be strongly related to the reward system.

Schultz and his colleagues[47] suggested that the activity of DA neurons represents an information analogous to prediction error of a delayed reward. They recorded DA neurons of monkeys performing a simple conditioning task in which the monkeys received a drop of appetizing fruit juice as a reward by touching a lever after the appearance of a light stimulus. Before training and during the early stage of training in which the conditioning was not established sufficiently, DA neurons responded strongly with phasic activations just after the time of reward delivery. As the learning progressed by training, however, DA neurons showed their phasic activation to the light stimulus but not to the reward delivery. In addition, after learning, if a reward was not delivered accompanied by the conditioned stimulus (light), the activity of DA neurons was depressed exactly at the time when the reward would have occurred. Computational theories from machine learning have helped guide our understanding of these reward prediction error signals. The model of TD learning has been particularly effective in describing these kinds of signals and placing them in a functional context.[53] These changes in DA activity functionally resemble TD error,[5,29] which is positive when a reward predictor (the light stimulus in the Schultz's study) is presented, but zero when the reward is given. Furthermore, an electrophysiological study of monkeys suggested that the activity of DA neurons changes in proportion to the probability of reward delivery.[15] This study was of simple conditioning between a stimulus and a reward as in Schultz's study, but delivery of the reward was determined probabilistically. When the reward was given with 100% probability, DA neurons after learning responded to the stimulus but not to the reward. As the reward probability decreased, the response to the stimulus decreased while that to the reward increased; this result can be predicted from the theory of TD learning. However, they also found that some DA neurons show sustained activities, which were the largest when the uncertainty of reinforcers (rewards) was the highest; these sustained activities cannot be simply explained by the TD model.

Recently, Nakahara et al.[32] suggested that the prediction error signals represented by DA responses account for context - information from the past - that can be used to improve reward prediction. They used a conditioning task in which monkeys received a reward at the rate of once every four trials (with 25% reward probability), but the 'conditional' probability of a reward in a trial changed depending on the history of reward delivery in previous trials, namely, on the number of non-rewarded trials that occurred consecutively just prior to the trial (post-reward trial number: PRN). In the early learning phase,

the DA responses were the same as PRN varied, while in the later phase, they showed significant dependence on PRN; the activities decreased steadily with the increase of PRN, which corresponded to an increase in the conditional probability of reward delivery. The result from this study cannot be explained by the simple TD model, implying the possibility that DA neurons encode the reward prediction error by including contextual information (memory).

4.2 Reward-related Activation in Striatum and RL Models

The responses related to reward prediction have also been observed in the striatum of the basal ganglia (BG)[16] which is the main target area of innervations of DA neurons.

Shidara et al.[50] recorded ventral striatum neurons of monkeys performing a bar-release task with visual cue stimuli. In this task, there were three cue stimuli with different brightness but only the brightest stimulus was rewarded; monkeys received a reward only when they completed the task for the brightest stimulus but not for the other stimuli. When these three stimuli were presented at random, the striatal neurons were selectively activated for the reward-related stimulus and not for the others. However, when the visual stimulus was made to gradually approach the rewarded case, namely, success at a non-rewarded trial led to a brighter stimulus at the next trial, the cue-related neuronal activity was dependent on the brightness. These results indicate that neurons in the ventral striatum may be involved in scheduling for reward approach, and this reminds us of the value function in RL.

Another monkey study using a task with multiple action options indicated that the striatum is related to action-dependent reward prediction.[25] The monkeys were trained to perform a memory-guided saccade task in which a cue stimulus was set at the left, right, top, or bottom of a central fixed position for two different reward conditions; an all-directions-rewarded (ADR) condition and a one-direction-rewarded (1DR) condition. By recording from the dorsal striatum (caudate), a neuron was found which was activated only for the rewarded direction under the 1DR condition despite being activated for all directions under the ADR condition. This neuron may represent both action itself and reward prediction depending on the action, and from the viewpoint of RL, this function is similar to the Q-function or utility function (actor) which represents the preference of each action.

These results showing the strong relationship between the neural activities in the BG and functions necessary for RL led to some computational models of the BG which realize RL. The striatum can be clearly segregated into two partial structures whose cellular compositions and target areas are anatomically different,[17] and they receive inputs from different areas in the PFC. One is the *striosome* which projects to the substantia nigra pars compacta (SNc), and the other is called the *matrix* projecting to the globus pallidus (GP) and the substantia nigra pars reticulata (SNr). Barto[5] suggested that the BG possibly realizes an actor-critic, which is composed of two learning elements as described in section 2.1: an actor, which represents a policy or a utility function for the

action selection, and a critic which predicts the amount of future reward. In his model, the striosome represents the value function as a critic, and the matrix corresponds to an actor. The DA neurons in the SNc compute the TD error δ_t, equation (5), and learning in the BG is executed based on this error signal.

On the other hand, Doya[12] hypothesized that the striatum generally represents prediction or expectation of reward models, and the striosome and matrix represent the value function $V(s)$ for state s and the Q-function $Q(s,a)$ for a pair of state s and action a, respectively. The TD error calculated in the SNc propagates to the BG as the activity of DA neurons to regulate the plasticity of a cortico-striatum network calculating the two sorts of value functions. An action is then probabilistically selected in the GP-SNr network based on the evaluation for possible actions within the matrix, i.e., the Q-function.

In the actor-critic model proposed by Barto,[5] two elements with different properties, a value function and a policy (an action controller), were allocated to the patch structure within the same nucleus, whereas in Doya's Q-learning hypothesis, both different structures in the striatum represent value functions with different state/action dependence; they do not represent action options but instead carry out the related reward predictions. A recent human functional magnetic resonance imaging (fMRI) study using an instrumental conditioning task[34] observed that the dorsal and ventral striatum showed activities similar to actor and critic, respectively, which would support the actor-critic hypothesis. On the other hand, a recording study of a monkey's striatum during an RL task[45] supported the Q-learning hypothesis. In this study, monkeys were trained to perform a free-choice task with two types of actions in asymmetrically rewarded conditions, and there were striatal neurons which coded a pair of action and reward values, i.e., acted like a Q-function.

4.3 Prefrontal Cortex and Environmental Model

TD learning by itself is a kind of 'model-free' RL method in which the environmental dynamics, i.e., the Markov state transition $P(x'|x,a)$, is implicitly used in value learning, and hence policy learning does not use any model of the dynamics explicitly. This method requires a large number of trials to achieve stochastic approximation, but can be implemented by a relatively simple network and therefore exhibits a nice correspondence to Hebbian-like learning. In contrast, 'model-based' methods[21,30,55] try to directly model the environmental dynamics and select an action based on prediction of environmental changes using the model. Compared to model-free alternatives, the algorithms of the model-based methods are complicated but learning can be performed using a small number of trials by utilizing the environmental model effectively. Recently, some studies of animals including humans reported that a model of the external world, such as their own bodies or surrounding environments, may be maintained as 'internal states' in the PFC, which constitutes a structure of parallel loops with the striatum in the BG.[1] Based on these findings, it has been suggested that these two behavioral learning schemes may be used in the brain as the situation demands: the striatum and the PFC circuit implement model-

free and model-based RL, respectively, and the DA afferents control between them.[14]

The dorsolateral prefrontal cortex (DLPF) in the PFC has long attracted attention to its working memory function, i.e., the active maintenance of necessary information for state-action transformation (action selection). Some experimental results, which have revealed that DLPF neurons predict the quality and the quantity of future reward,[27,59] imply that DLPF neurons could be a pivotal area for maintaining an evaluation of the environment, which would correspond to the value function in RL. In addition, Barraclough,[3] conducted an electrophysiological experiment with monkeys and showed that DLPF neurons are also involved in updating the behavioral strategy. In their experiment, monkeys performed optimal decision-making in a multi-agent zero-sum game against a computer agent. They showed that the monkeys' performances could be explained by an RL algorithm and found that DLPF neurons were activated as a function of the history of their own actions and obtained rewards. They then suggested that the DLPF represents and updates the value function (Q-function) by integrating information of actions and rewards.

A physiological recording study using monkeys performing a delayed motor task investigated movement-related neuronal activities in the DLPF,[19] thus the DLPF is hypothesized to construct automata, i.e., cascade networks representing transitions of states, in order to successively achieve a behavioral goal.[58] This hypothesis was supported by a human fMRI study showing that DLPF activity is related not to information maintained in working memory itself, but rather to preparation of action sequences based on maintained information[39] and to the processing of sequence memories.[11] Since behavioral planning requires predicting environmental changes induced by one's own actions, we may speculate that environmental models in RL, which predict state changes, are expressed in the DLPF.[21] Therefore, we investigated our hypothesis in a human fMRI study using a simple RL task, and found that the DLPF, in particular the anterior region, was activated in the RL condition in which subjects were required to manipulate multiple candidates of maintained response sequences until they obtained the correct sequence.[62] This result showed that the anterior DLPF possibly plays a role in temporal processing such as the mental simulation of environmental models. In addition, the DLPF is known to have an important role in maintaining and representing contextual information such as behavioral goals and rules,[7,10] and to be a central engine of decision-making.

The DLPF forms a strong mutual connection with DA neurons and information stored in the DLPF may be controlled by projections from DA neurons,[11] i.e., tonic activities via D1 receptors realize stabilization and maintenance of working memory, and phasic activities via D2 receptors realize updating and learning of information. Since DA neurons are also known to control synaptic plasticity from the PFC to the striatum, they may be involved in learning of action evaluation in the BG based on prediction about environmental changes represented in the PFC.

Accordingly, the PFC and the BG form parallel and hierarchical networks,

and it is possible that DA neurons control them by using reward information received from the external world, thus adaptive learning in a dynamic environment and robust yet flexible decision-making in such an environment could be achieved. The RL theory has provided various computational models for reward-related neural activities and brain functions, though plausible implementations need further investigation.

§5 Concluding Remark

The most intriguing point of RL, regardless of whether it is model-free or model-based, is to realize prediction of future reward in terms of Bellman-like self-consistent equations. Whether this prediction is really implemented in the brain or not is still controversial. Such a self-referential structure, if exists, has many implications; RL may be a basis not only for decision making in a complicated world containing multiple agents, but also for recognition of the 'self' in such a world. The theory of RL will thus grow by incorporating concepts and knowledge from various research fields, such as machine learning, control theory, artificial intelligence, economics, neuroscience, and psychology.

References

1) Alexander, G.E., Crutcher, M.D. and DeLong, M.R., "Basal Ganglia-thalamocortical Circuits: Parallel Substrates for Motor, Oculomotor, "Prefrontal" and "Limbic" Functions," *Progress in Brain Research*, *85*, pp. 119-146, 1990.

2) Amari, S., "Natural Gradient Works Efficiently in Learning," *Neural Computation*, *10*, *2*, pp. 251-276, 1998.

3) Barraclough, D.J., Conroy, M.L. and Lee, D., "Prefrontal Cortex and Decision Making in a Mixed-strategy Game," *Nature Neuroscience*, *7*, pp. 404-410, 2004.

4) Barto, A.G., Sutton, R.S. and Anderson, C.W., "Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems," *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-13*, pp. 834-846, 1983.

5) Barto, A.G., "Adaptive Critics And The Basal Ganglia," in *Models of Information Processing in the Basal Ganglia*, pp. 215-232, MIT Press, Cambridge, MA, 1994.

6) Bellman, R.E., *Dynamic Programming*, Princeton University Press, Princeton, 1957.

7) Braver, T.S. and Barch, D.M., "A Theory of Cognitive Control, Aging Cognition, and Neuromodulation," *Neuroscience and Biobehavioral Reviews*, *26*, *7*, pp. 809-817, 2002.

8) Brafman, R.I., "A Heuristic Variable Grid Solution for POMDPs," in *Fourteenth National Conference on Artificial Intelligence*, *AAAI-9*, pp. 33-42, 1997.

9) Cassandra, A.R., Kaelbling, L.P. and Littman, M.L., "Acting Optimally in Partially Observable Stochastic Domains," in *Twelfth National Conference on Artificial Intelligence*, *AAAI-94*, pp. 1023-1028, 1994.

10) Cohen, J.D., Perlstein, W.M., Braver, T.S., Nystrom, L E., Noll, D.C., Jonides, J. and Smith, E.E., "Temporal Dynamics of Brain Activation During a Working Memory Task," *Nature*, *386*, pp. 604-608, 1997.

11) Cohen, J.D., Braver, T.S. and Brown, J.W., "Computational Perspectives on Dopamine Function in Prefrontal Cortex," *Current Opinion in Neurobiology*, *12*, 2, pp. 223-229, 2002.

12) Doya, K., "Complementary Roles of Basal Ganglia and Cerebellum in Learning and Motor Control," *Current Opinion in Neurobiology*, *10*, 6, pp. 732-739, 2000.

13) Doya, K., "Computational Model of Neuromodulation," *Neural Networks*, *15*, *4-6*, pp. 475-477, 2002.

14) Daw, N.D., Niv, Y. and Dayan, P., "Uncertainty-based Competition between Prefrontal and Dorsolateral Striatal Systems for Behavioral Control," *Nature Neuroscience*, *8*, pp. 1704-1711, 2005.

15) Fiorillo, C.D., Tobler, P.N. and Schultz, W. , "Discrete Coding of Reward Probability and Uncertainty by Dopamine Neurons," *Science*, *299*, pp. 1898-1902, 2003.

16) Gerfen, C.R., Herkenham, M. and Thibault, J., "The Neostriatal Mosaic. II. Patch and Matrix Directed Ddesostriatal Dopaminergic and Nondopaminergic Systems," *The Journal of Neuroscience*, *7*, pp. 3915-3934, 1987.

17) Graybiel, A.M., "Neurotransmitters and Neuromodulators in the Basal Ganglia," *Trends in Neurosciences*, *13*, pp. 244-254, 1990.

18) Grillner, S., Wallen, P., Brodin, L. and Lansner, A. "Neural Network Generating Locomotor Behavior in Lamprey," *Annual Review of Neuroscience*, *14*, pp. 169-199, 1991.

19) Hoshi, E., Shima, K. and Tanji, J., "Neuronal Activity in the Primate Prefrontal Cortex in the Process of Motor Selection Based on Two Behavioral Rules," *Journal of Neurophysiology*, *83*, pp. 2355-2373, 2000.

20) Howard, R.A., *Dynamic Programming and Markov Processes*, MIT Press, Cambridge, MA, 1960.

21) Ishii, S., Yoshida, W. and Yoshimoto, J., "Control of Exploitation-exploration Meta-parameter in Reinforcement Learning," *Neural Networks*, *15*, pp. 665-687, 2002.

22) Ishii, S., Fujita, H., Mitsutake, M., Yamazaki, T., Matsuda, J. and Matsuno, Y., "A Reinforcement Learning Scheme for a Partially-observable Multi-agent Game," *Machine Learning*, *59*, pp. 31-54, 2005.

23) Kaelbling, L.P., Littman, M. and Cassandra, A., "Planning and Acting in Partially Observable Stochastic Domains," *Artificial Intelligence*, *101*, pp. 99-134, 1998.

24) Kakade, S., "A Natural Policy Gradient," in *Advances in Neural Information Processing Systems 14*, pp. 1531-1538, 2001.

25) Kawagoe, R., Takikawa, Y. and Hikosaka, O., "Expectation of Reward Modulates Cognitive Signals in the Basal Ganglia," *Nature Neuroscience*, *1*, 5, pp. 411-416, 1998.

26) Konda, V.R. and Tsitsiklis, J.N., "Actor-critic Algorithms," *SIAM Journal on Control and Optimization*, *42*, pp. 1143-1146, 2003.

27) Leon, M.I. and Shadlen, M.N., "Effect of Expected Reward Magnitude on the Response of Neurons in the Dorsolateral Prefrontal Cortex of the Macaque," *Neuron*, *24*, pp. 415-425, 1999.

28) Mogenson, G.J., Takigawa, M., Robertson, A. and Wu, M., "Self-stimulation of the Nucleus Accumbens and Ventral Tegmental Area of Tsai Attenuated by Microinjections of Spiroperidol into the Nucleus Accumbens," *Brain Research*, *171*, 2, pp. 247-259, 1979.

29) Montague, P.R., Dayan, P. and Sejnowski, T.J., "A Framework for Mesencephalic Dopamine Systems Based on Predictive Hebbian Learning," *The Journal of Neuroscience*, *16*, pp. 1936-1947, 1996.

30) Moore, A.W. and Atkeson, C.G., "Prioritized Sweeping: Reinforcement Learning with Less Data and Less Real Time," *Machine Learning*, *13*, pp. 103-130, 1993.

31) Mori, T., Nakamura, Y., Sato, M. and Ishii, S., "Reinforcement Learning for CPG-driven Biped Robot," in *The Nineteenth National Conference on Artificial Intelligence*, *AAAI-04*, pp. 623-630, 2004.

32) Nakahara, H., Itoh, H., Kawagoe, R., Takikawa, Y. and Hikosaka, O., "Dopamine Neurons Can Represent Context-dependent Prediction Error," *Neuron*, *41*, pp. 269-280, 2004.

33) Nakamura, Y., Mori, T., Tokita, Y., Shibata, T. and Ishii, S., "Off-policy Natural Policy Gradient Method for a Biped Walking Using a CPG Controller," *Journal of Robotics and Mechatronics*, *17*, *6*, pp. 636-644, 2005.

34) O'Doherty, J., Dayan, P., Schultz, J., Deichmann, R., Friston, K. and Dolan, R.J., "Dissociable Roles of Ventral and Dorsal Striatum in Instrumental Conditioning," *Science*, *304*, pp. 452-454, 2004.

35) Olds, J. and Milner, P., "Positive Reinforcement Produced by Electrical Stimulation of Septal Area and Other Regions of Rat Brain," *Journal of Computational Physiological Psychology*, *47*, pp. 19-27, 1954.

36) Parr, R. and Russell, S., "Approximating Optimal Policies for Partially Observable Stochastic Domains," in *Proceedings of International Joint Conference on Artificial Intelligence*, *IJCAI-95*, pp. 1088-1094, 1995.

37) Peters, J., Vijayakumar, S. and Schaal, S., "Reinforcement learning for humanoid robotics," in *Third IEEE International Conference on Humanoid Robotics*, 2003.

38) Precup, D., Sutton, R.S. and Dasgupra, S., "Off-policy Temporal-difference Learning with Function Approximation," in *Proceedings of the 18th International Conference on Machine Learning*, *ICML*, pp. 417-424, 2001.

39) Pochon, J.B., Levy, R., Poline, J.B., Crozier, S., Lehericy, S., Pillon, B., Deweer, B., Le Bihan, D. and Dubois, B., "The Role of Dorsolateral Prefrontal Cortex in the Preparation of Forthcoming Actions: An fMRI Study," *Cerebral Cortex*, *11*, pp. 260-266, 2001.

40) Poupart, P. and Boutilier, C., "Value-directed Compression of POMDPs," in *Advances in Neural Information Processing Systems 15*, pp. 1579-1586, 2003.

41) Rescorla, R.A. and Wagner, A.R., "A Theory of Pavlovian Conditioning: Variations in the Effectiveness of Reinforcement and Nonreinforcement," in *Classical Conditioning II: Current Research and Theory*, pp. 64-99, New York, NY: Appleton, 1972.

42) Reynolds, J.N., Hyland, B.I. and Wickens, J.R., "A Cellular Mechanism of Reward-related Learning," *Nature*, *413*, pp. 67-70, 2001.

43) Robbins, T.W. and Everitt, B.J., "Neurobehavioural Mechanisms of Reward and Motivation," *Current Opinion in Neurobiology*, *6*, *2*, pp. 228-236, 1996.

44) Rodoriguez, A., Parr, R. and Koller, D., "Reinforcement Learning Using Approximate Belief State," in *Advances in Neural Information Processing Systems 12*, pp. 1036-1042, 2002.

45) Samejima, K., Ueda, Y., Doya, K. and Kimura, M., "Representation of Action-specific Reward Values in The Striatum," *Science*, *310*, pp. 1337-1340, 2005.

46) Sato, M., Nakamura, Y. and Ishii, S., "Reinforcement Learning for Biped Locomotion, in *Neural Networks - ICANN 2002*, *LNCS2415*, pp. 777-782, Springer-Verlag, Berlin, 2002.

47) Schultz, W., Dayan, P. and Montague, R.P., "A Neural Substrate of Prediction and Reward," *Science*, *275*, pp. 1593-1599, 1997.

48) Seymour, B., O'Doherty, J.P., Dayan, P., Koltzenburg, M., Jones, A.K., Dolan, R.J., Friston, K.J. and Frackowiak, R.S., "Temporal Difference Models Describe Higher-order Learning in Humans," *Nature*, *429*, pp. 664-667, 2004.

49) Shelton, C.R., "Policy Improvement for POMDPs Using Normalized Importance Sampling," in *Proceedings of the Seventeenth International Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 496-503, 2001.

50) Shidara, M., Aigner, T.G. and Richmond, B.J., "Neuronal Signals in the Monkey Ventral Striatum Related to Progress Through a Predictable Series of Trials," *The Journal of Neuroscience*, *18*, *7*, pp. 2613-2625, 1998.

51) Smallwood, R.D. and Sondik, E.J., "The Optimal Control of Partially Observable Markov Decision Processes Over a Finite Horizon," *Operations Research*, *21*, 1071-1088, 1973.

52) Stolerman, I., "Drugs of Abuse: Behavioural Principles, Methods and Terms," *Trends in Pharmacological Sciences*, *13*, *5*, pp. 170-176, 1992.

53) Sutton, R.S. and Barto, A.G., "Towards a Modern Theory of Adaptive Networks: Expectation and Prediction," *Psychological Review*, *88*, pp. 135-170, 1981.

54) Sutton, R.S., "Learning to Predict by the Method of Temporal Differences," *Machine Learning*, *3*, pp. 9-44, 1988.

55) Sutton, R.S. and Barto, A.G., *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.

56) Sutton, R.S., McAllester, D., Singh, S. and Manour, Y., "Policy Gradient Method for Reinforcement Learning with Function Approximation," in *Advances in Neural Information Processing Systems 12*, pp. 1057-1063, 2000.

57) Taga, G., Yamaguchi, Y. and Shimizu, H., "Self-organized Control in Bipedal Locomotion by Neural Oscillators in Unpredictable Environment," *Biological Cybernetics*, *65*, pp. 147-159, 1991.

58) Tanji, J. and Hoshi, E., "Behavioral Planning in the Prefrontal Cortex," *Current Opinion in Neurobiology*, *11*, pp. 164-170, 2001.

59) Watanabe, M., "Reward expectancy in primate prefrontal neurons," *Nature*, *382*, pp. 629-632, 1996.

60) Watkins, C.J.C.H. and Dayan, P., "Q-learning," *Machine Learning*, *8(3/4)*, pp. 279-292, 1992.

61) Williams, R., "Simple Statistical Gradient Following Algorithms for Connectionist Reinforcement Learning," *Machine Learning*, *8*, pp. 229-256, 1992.

62) Yoshida, W. and Ishii, S., "Model-based Reinforcement Learning: A Computational Model and an fMRI Study," *Neurocomputing*, *63C*, pp. 253-269, 2005.

Appendix

§A1

A differential of equation (11a) with respect to the parameter θ becomes

$$\begin{aligned}\frac{\partial V^\theta(x)}{\partial\theta} &= \int_a \left[\frac{\partial\pi^\theta(a|x)}{\partial\theta}Q^\theta(x,a) + \pi^\theta(a|x)\frac{\partial}{\partial\theta}Q^\theta(x,a)\right] da \\ &= \int_a \left[\frac{\partial\pi^\theta(a|x)}{\partial\theta}Q^\theta(x,a) + \pi^\theta(a|x)\right. \\ &\qquad \left.\left[-\frac{\partial\rho(\theta)}{\partial\theta} + \int_{x'} P(x'|x,a)\frac{\partial V^\theta(x')}{\partial\theta}dx'\right]\right] da,\end{aligned}$$

where we have used equation (11b) in this derivation. From this equation, we obtain

$$\begin{aligned}\frac{\partial\rho(\theta)}{\partial\theta} &= \int_a \left[\frac{\partial\pi^\theta(a|x)}{\partial\theta}Q^\theta(x,a) + \pi^\theta(a|x)\int_{x'} P(x'|x,a)\right. \\ &\qquad \left.\frac{\partial V^\theta(x')}{\partial\theta}dx'\right] da - \frac{\partial V^\theta(x)}{\partial\theta}.\end{aligned}$$

By taking the expectation with respect to the stationary distribution of states, $D^\theta(x)$, we obtain

$$\begin{aligned}\int_x D^\theta(x)\frac{\partial\rho(\theta)}{\partial\theta}dx &= \int_x D^\theta(x)\int_a \frac{\partial\pi^\theta(a|x)}{\partial\theta}Q^\theta(x,a)dadx \\ &\quad + \int_{x,a} D^\theta(x,a)\int_{x'} P(x'|x,a)\frac{\partial V^\theta(x')}{\partial\theta}dx'dadx \\ &\quad - \int_x D^\theta(x)\frac{\partial V^\theta(x)}{\partial\theta}dx \\ &= \int_x D^\theta(x)\int_a \frac{\partial\pi^\theta(a|x)}{\partial\theta}Q^\theta(x,a)dadx,\end{aligned}$$

where we have used the fact that the second and third right-hand-side terms are equal from the invariability of any integral over the stationary state distribution of the Markov process. Since the integral on the left-hand-side equals unity, equation (12) has been shown.

§A2

The game Hearts is played by four players and uses the ordinary 52-card

deck. There are four suits, spades (♠), hearts (♡), diamonds (♢), and clubs (♣), and there is an order of strength within each suit (i.e., A, K, Q, ..., 2). There is no strength order among the suits. Cards are distributed to the four players so that each player has in his/her hand 13 cards at the beginning of the game. Thereafter, according to the rules below, each player plays a card clock-wisely in order. When each of the four players has played a card, it is called a trick. The first card played in a trick is called the leading card and the player who plays the leading card is called the leading player. A single game ends when 13 tricks are carried out.

- Except for the first trick, the winner of the current trick becomes the leading player of the subsequent trick. In the first trick, ♣2 is the leading card, denoting that the player holding this card is the leading player.
- Each player must play a card of the same suit as the leading card. If a player does not have a card of the same suit as the leading card, he/she can play any card. When a heart is in such a case played for the first time in a single game, the play is called "breaking hearts". Until the breaking hearts occurs, the leading player may not play a heart. If the leading player has only hearts, it is an exceptional case and the player may lead with a heart.
- After a trick, the player that has played the strongest card of the same suit as the leading card becomes the winner of that trick.
- Each heart equals a one-point penalty and the ♠Q equals a 13-points penalty. The winner of a trick receives all of the penalty points of the cards played in the trick.

According to the rules above, a single game is played, and at the end of a single game, the score of each player is determined as the sum of the received points. The lower the score, the better.

Shin Ishii, Ph.D.: He is a professor of Graduate School of Information Science at Nara Institute of Science and Technology. He received his B.E. in 1986, M.E. in 1988, and Ph.D. in 1987 from University of Tokyo. His current research interests are computational neuroscience, systems neurobiology and statistical learning theory.

Wako Yoshida, Ph.D.: She is a researcher of Graduate School of Information Science at Nara Institute of Science and Technology. She received her B.A. in 1998 from Kobe College, M.E. in 2000 and Ph.D. in 2003 both from Nara Institute of Science and Technology. Her research interest includes theoretical and experimental approach to human's decision-making process through learning, memory and communication.