Available online at www.sciencedirect.com
SCIENCE DIRECT®
Neural Networks 17 (2004) 935–952
Neural Networks
www.elsevier.com/locate/neunet

Reliability of internal prediction/estimation and its application. I. Adaptive action selection reflecting reliability of value function

Yutaka Sakaguchi*, Mitsuo Takano[1]

Graduate School of Information Systems, University of Electro-Communications, 1-5-1, Chofugaoka, Chofu, Tokyo 182-8585, Japan

Received 2 November 2001; revised 13 May 2004; accepted 13 May 2004

Abstract

This article proposes an adaptive action-selection method for a model-free reinforcement learning system, based on the concept of the 'reliability of internal prediction/estimation'. This concept is realized using an internal variable, called the Reliability Index (RI), which estimates the accuracy of the internal estimator. We define this index for a value function of a temporal difference learning system and substitute it for the temperature parameter of the Boltzmann action-selection rule. Accordingly, the weight of exploratory actions adaptively changes depending on the uncertainty of the prediction. We use this idea for tabular and weighted-sum type value functions. Moreover, we use the RI to adjust the learning coefficient in addition to the temperature parameter, meaning that the reliability becomes a general basis for meta-learning. Numerical experiments were performed to examine the behavior of the proposed method. The RI-based Q-learning system demonstrated its features when the adaptive learning coefficient and large RI-discount rate (which indicate how the RI values of future states are reflected in the RI value of the current state) were introduced. Statistical tests confirmed that the algorithm spent more time exploring in the initial phase of learning, but accelerated learning from the midpoint of learning. It is also shown that the proposed method does not work well with the actor-critic models. The limitations of the proposed method and its relationship to relevant research are discussed.

Keywords: Internal prediction; Reliability; Model-free reinforcement learning; TD learning; Discount rate; Exploration–exploitation balance; Temperature parameter; Meta-learning

1. Introduction

Internal prediction, or estimation of the future, is the most essential step in deciding what actions an agent should take in an environment, because it foretells the result of a given action without taking the action. If the results of actions can be accurately predicted, the agent can achieve maximum performance by choosing the action that will bring the best outcome (i.e. 'greedy policy').

However, this assumption is violated if the internal prediction is incorrect. This situation is inevitably observed in an on-line learning system, which builds an internal predictor or estimator while working in an actual environment. The same holds true when the characteristics of the environment vary over time. In these cases, we cannot expect predictions to always be accurate; consequently, action-selection that relies totally on internal prediction is no longer effective. Instead, it may be meaningful to place more weight on exploratory actions to acquire more knowledge of the environment.

Therefore, an essential problem in an on-line reinforcement learning system is how to balance greedy action-selection, which utilizes internal prediction, and exploratory action-selection to learn about the environment. This problem is broadly known as the 'exploration–exploitation balance', and it has long been discussed in fields involving adaptive systems, from optimal control to machine learning (Dayan & Sejnowski, 1996; Fe'ldbaum, 1965; Sutton, 1990; Sutton & Barto, 1998; Thrun & Möller, 1992). Although an optimal strategy can be derived in a theoretical manner for some simple problems (Witten, 1976), no general practical solution is applicable to larger problems, and various heuristics have been proposed for given problems. This article discusses a heuristic, but rather general, method,

* Corresponding author. Tel.: +81-424-43-5646; fax: +81-424-43-5681.
E-mail addresses: sakaguchi@is.uec.ac.jp (Y. Sakaguchi), takano@tut.ac.jp (M. Takano).
[1] Present address: Department of Electronics, Faculty of Engineering, Tokyo University of Technology, Hachioji, Tokyo, Japan.

0893-6080/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2004.05.004

Nomenclature

a	action
s	state
$P(a)$	probability of selecting action a
$Q(s,a)$	action value function
$V(s)$	state value function
$R(s), R(s,a)$	reliability index
R_1	lowest limit of reliability index
$\boldsymbol{f}(s)$	feature vector
$\boldsymbol{w}(a), \boldsymbol{w}_V, \boldsymbol{w}_q(a)$:	weight vector
$\boldsymbol{R}_w$	reliability index for weight vector
δ	TD error
T	temperature parameter
$\alpha, \alpha_0, \alpha_1, \beta, \beta_0, \beta_1, \alpha_R$	learning coefficient
γ	discount rate for reward
γ_R	discount rate for reliability index
η	parameter for action-selection

focusing on a temporal difference (TD) learning system, based on the concept of 'reliability'.

An essential question that has been discussed in the literature on exploration–exploitation problems is that of which state an agent prioritizes when exploring the state space. Various ideas have been proposed for this problem. For example, 'exploration bonus' (Dayan & Sejnowski, 1996; Sutton, 1990) places additional weight on states that the agent has not visited recently. In 'prioritized sweeping' (Moore & Atkeson, 1993), the system puts the present state into the priority queue when the change in the state transition probability exceeds a given threshold. Including algorithms in the literature of artificial intelligence (Brafman & Tennenholtz, 2000; Kearn & Singh, 1998), most conventional studies have been based on model-based learning systems, that is, the systems included a state transition matrix and a reward matrix. These studies proposed guidelines for exploration that minimized the discrepancy between the actual environment and its model. In other words, they attempted to select the best states to visit in order to detect errors in the model.

By contrast, our paper focuses on model-free reinforcement learning systems, such as the TD learning system (Sutton, 1988). In a model-free system, the system does not construct an environmental model, but learns value functions directly. Although the lack of an explicit model ultimately has a number of limitations, such systems are attractive nevertheless, because of their simplicity.

The key to model-free systems is how to estimate value functions correctly; consequently, the value function plays an essential role in this paper. Another feature of our study is that we aim to adjust the randomness of action selection (i.e. the weight of exploration), instead of choosing states to visit. To this end, we try to estimate the current reliability of the value function, and to change the action-selection policy according to its reliability.

The fundamental idea is to change the randomness of action selection as learning progresses. In general, the agent needs to explore the environment more when it knows little about the characteristics of the environment; once it has adapted to the environment sufficiently, it can then draw on its knowledge. This implies that as learning proceeds less weight should be placed on exploratory actions. In fact, this idea has been discussed in the literature on reinforcement learning. Before going into our method in detail, let us examine the methods that are typically used to balance exploration and exploitation.

One is the epsilon-greedy method. In this method, an agent chooses actions based on the greedy policy in most trials, but sometimes takes exploratory actions, where the ratio of exploratory actions is designated by the parameter ε. Another method is the so-called Boltzmann action-selection rule, where actions are chosen in a probabilistic manner according to the probability

$$P(a) = \frac{\exp(Q(a)/T)}{\sum_{a'} \exp(Q(a')/T)}, \tag{1}$$

where $Q(a)$ gives a value for action a, and T is a positive constant called the 'temperature parameter'. Using this rule, the agent is more likely to choose actions with higher Q-values. The temperature parameter adjusts the exploration–exploitation balance; action-selection becomes more random with larger T, and is almost deterministic at the limit $T \to 0$.

In most demonstrations of reinforcement learning, these parameters are determined (by trial and error) so that the agent performs in the manner desired. If we want to reduce the weight of exploratory actions as learning proceeds, we can reduce the parameter values according to the number of learning steps taken. This operation, called 'annealing', is a simple way to adjust the balance according to progress in learning.

Although annealing is attractive because of its simplicity, it has at least one significant problem, which is that the schedule of annealing must be determined a priori. As nobody can predict the speed of learning, annealing is likely to be adversely affected when there is a mismatch of the learning speed and the annealing schedule. Another crucial deficit of annealing is that it cannot cope with a temporal change in the environment: annealing is no longer meaningful if the characteristics of the environment change in an unexpected manner.

Therefore, in order to change the action-selection policy according to progress in learning, we have to estimate the accuracy of the value function, and change the randomness parameters (ε and T) adaptively according to this estimation. This study proposes such a faithfully adaptive action-selection method.

The essence of the proposed algorithm is the concept of the 'reliability of internal prediction/estimation'. Reliability represents our '*confidence*' with our own prediction or knowledge. We originally applied this concept to explain changes in human motor behavior during visuo-motor adaptation (Sakaguchi, 1996; Sakaguchi, Akashi, & Takano, 2001; Sakaguchi & Nakano, 1992). In those papers, we illustrated how a computational model based on this concept well simulated the hand movements of human subjects adapting to visually distorted environments. The present study generalizes and re-formulates this idea in order to apply it to the TD learning system (Sakaguchi & Takano, 2001), and examines how the resultant algorithm behaves in comparison with conventional algorithms.

In order to implement the agent's subjective measure of the accuracy of predictions in a computational algorithm, we define an internal variable, called the Reliability Index (RI), which gives an expected value for the difference between the actual result and that predicted. In this study, we define this index for the value function of a TD learning system. Concrete algorithms are given for two types of value function: tabular-type functions, which give a value to every pair of state and action, and weighted-sum-type functions, which calculate a value that is a linear summation of the features. In a separate paper (Sakaguchi & Takano, under review), we discuss the method's application to modular learning.

Adaptive change in the temperature parameter can be regarded as meta-learning (see Doya, 2002 for review), a brain mechanism that is used to regulate the global parameters (i.e. the action-selection randomness, learning coefficient and discount rate) and structure of a learning system. Since people adjust the weight placed on exploration dynamically in daily life, it is plausible that adjusting action-selection randomness is one of the essential functions of the human brain. In addition to exploration weight, this paper addresses the possibility of applying the RI to automatically adjust another global parameter, the learning coefficient. Moreover, other researchers have applied the reliability concept to adjust the discount rate (Ogawa, Namiki, & Ishikawa, 2002). Therefore, the reliability concept provides automatic control of three global parameters and is meaningful for explaining the mechanism of meta-learning in the human brain.

The behavior of the proposed method is demonstrated using numerical experiments. In the first experiment, we examine the learning performance of the proposed algorithm with problems of various size, and compare its performance with those of typical conventional algorithms (i.e. epsilon-greedy and the Boltzmann action-selection rule). In the second experiment, we deal with the time-variant grid-world problem to illustrate adaptation ability of the proposed algorithm. The third experiment, which deals with the 'acrobot' problem, examines the behavior of the algorithm with weighted-sum-type value functions. Finally, we discuss the limitations of the proposed method and its relationship to relevant approaches.

2. Reliability of internal prediction/estimation and its application to value function

This section explains the concept of 'reliability' and how to apply this concept to a TD learning system.

As mentioned above, changing the exploration–exploitation balance according to the accuracy of internal prediction/estimation is a good strategy that an agent can use to efficiently adapt to a new environment. The problems are how to estimate its accuracy, and how to reflect this in action-selection.

The accuracy of prediction is objectively determined by the difference between a prediction and the outcome (this difference will be referred to as the 'predictive error', below. Note that in the TD learning system, this corresponds to the TD error, because it represents the discrepancy between the expected and actual reward sums). However, this information is available only after performing an action. In order to predict the accuracy of action selection, the agent must keep an estimator of predictive error as an internal variable.

The concept of reliability is one possible internal estimator. In simplest terms, the RI can be defined as an internal variable that estimates the expected predictive error; it is updated after every action by comparing itself to the actual error. The RI decreases when the predicted error is smaller than the RI, that is, when the prediction is more accurate than expected, and increases when the opposite is true. In a stationary environment, the RI is expected to converge on a certain value that depends on the statistical properties of the environment and the representation ability. The RI changes continually in a time-varying environment as the degree of predictive error varies over time.

Now, we consider defining the reliability of the value function, i.e. an internal estimate of the 'reward sum'. For the state-value function $V(s)$, we simply define an index $R(s)$. For the action-value function $Q(s,a)$, on the other hand, we can think of two manners, action-based reliability $R(s,a)$ and state-based reliability $R(s)$. The former is defined separately for each pair of state and action (that is, one $R(s,a)$ for each $Q(s,a)$) while the latter is defined commonly for different actions. This point will be discussed later.

The predictive error of 'reward sum' is given by the difference between the value function and the true reward sum and, in a TD learning system, this information is provided by TD error. Accordingly, the RI is updated by comparing the TD error and the RI itself. The RI decreases when the (absolute value of) TD error is smaller than

the RI, and increases when the opposite is true. Using these definitions, the RI is expected to become close to the standard deviation of the TD error. If the environment is stationary, this implies that the RI can be an estimator of the standard deviation of the reward sum.

In this sense, the proposed method appears to be an attempt to use the variance in a reinforcement learning system. The combination of the mean and variance is a natural expansion of typical reinforcement learning, which uses only the expected reward sum, and other researchers have discussed this (Sato, Mimura, & Kobayashi, 2001; White, 1988; Williams, 1992).

However, our focus is not on estimation of the variance parameter and its use, but on flexible change of the action-selection policy for a time-variant environment, as will be discussed in Section 5.1. (Note that Williams (1992) suggested using the variance parameter of a Gaussian unit to control the degree of exploration.) As we stated above, the RI represents the *self-confidence* of the agent, and depends on both the environment and the agent itself. Our aim is to examine how this '*psychologically-inspired*' algorithm works in some typical reinforcement learning problems.

3. Algorithm

The proposed algorithm consists of three parts: (1) adaptive action selection, (2) updating the value-function and RI, and (3) adjusting the learning coefficient. The following subsections explain each part.

3.1. Adaptive action selection

The core of the proposed method is to adjust the ratio of exploratory actions according to the RI. To implement this idea, we modified the Boltzmann rule by substituting the RI for the temperature parameter (T). Accordingly, the probability of selecting action a at state s is given by

$$P(a;s) = \frac{\exp(\eta Q(s,a)/R(s))}{\sum_{a'} \exp(\eta Q(s,a')/R(s))}, \tag{2}$$

where $Q(s,a)$ is the value function for action a in state s, $R(s)$ is the RI defined for state s, and η is a positive constant.

This rule is like the original Boltzmann rule, in that the agent is more likely to select actions that give larger Q-values. However, it definitely differs, in that the randomness of action selection varies dynamically according to the RI. The agent becomes more exploratory with larger R, and more deterministic with smaller R. Unlike annealing, which reduces the randomness uniformly, the proposed method makes action selection more or less probabilistic according to the amount of TD error. The agent's behavior becomes more deterministic as TD error decreases with learning. Conversely, if TD error increases due to an environmental change, the agent's behavior becomes exploratory in order to adapt to the new environment.

From another point of view, this method does not use the genuine Q-value for action selection, but uses the Q-value normalized by the RI (that is, the effect of the Q-value changes according to the amount of RI). This in turn brings subtle difference in the role of exploration parameters (i.e. T and η); η is operated to the normalized dimensionless value $Q(s,a)/R(s)$ (because $R(s)$ has the same dimension as the reward and Q-value), whilst T is operated to the Q-value itself. Thus, we can deal with the exploration parameter η independent of the fluctuation of Q-values and of progress in learning. Note that the exploration parameter η still has to be appropriately selected (see Section 3.5 for a guideline for this selection).

3.2. Definition of reliability index

Before going into the update rule of RI, we would like to explain the detailed definition of the RI.

In Section 2, we mentioned that two types of RI (i.e. state-based RI $R(s)$ and action-based RI $R(s,a)$) can be thought of for the action-value function $Q(s,a)$. Nevertheless, we adopted the state-based RI for the action selection rule (Eq. (2)). First, we explain the reasons for this. One is that action selection is performed with a state as a unit; thus, the randomness of action selection should be defined for a state, not for an action. The other reason is that if the index were defined separately for different actions, the index for a specific action might be very small compared to those for other actions. If this happened, the agent would always choose the specific action, because its normalized Q-value would be much larger than that of the others. Therefore, it is unreasonable to use action-based reliability for action selection.

Next, we consider whether $R(s)$ should be updated in common for all actions, or defined as the average of $R(s,a)$. We discuss this problem in detail, separately for Q-learning (Barto, Sutton, & Watkins, 1990; Watkins & Dayan, 1992), which utilizes the action-value function $Q(s,a)$, and for the actor-critic model (Barto, Sutton, & Anderson, 1983), which uses the state-value function $V(s)$.

In Q-learning, TD error is defined by

$$\text{TD error} = \text{reward} + \gamma \max_{a'} Q(s',a') - Q(s,a), \tag{3}$$

where s' is the state after action a. Note that this TD error is a function of s and a, meaning that it is calculated separately for different actions.

At the beginning of learning, the agent chooses different actions almost uniformly, and thus, TD error takes similar values for different actions. As the learning of Q-value proceeds, however, the agent comes to select specific actions; TD error decreases only for such selected actions. If we define $R(s,a)$ separately for different actions, thus, $R(s,a)$ would decrease only for the optimal

(and sub-optimal) actions a^*. Since commonly-updated $R(s)$ presumably behaves like $R(s, a^*)$ after sufficient learning, it is expected that commonly-updated $R(s)$ takes a smaller value than $R(s)$ defined as an average of $R(s, a)$. Accordingly, the learning would be faster with the commonly-updated $R(s)$, than with the average-based $R(s)$.

Based on this consideration, moreover, we can think of another definition of $R(s)$, that is, $R(s) = R(s, a^*)$. This can be implemented either by defining separate $R(s, a)$ and substituting $R(s, a^*)$ for $R(s)$ at every trial, or by updating a common $R(s)$ only when the agent chooses optimal action a^*. Because $R(s, a^*)$ is almost similar to (or smaller than) the common $R(s)$, it is to be expected that learning would be the same (or even accelerated) with this third method, compared to with the common RI method.[2]

In summary, we should define the RI for a state as the common RI for all actions, or as the RI for the optimal action (i.e. $R(s, a^*)$).

Now, we move on to the actor-critic model. In the actor-critic model, TD error is defined by

$$\text{TD error} = \text{reward} + V(s') - V(s). \tag{4}$$

In this case, the TD error is not defined separately for different actions. As a result, the TD error can change drastically, dependent on action selection because the next state (s') differs among different actions.

This is easily seen by comparing the following cases. Assume that the agent chooses action a_1, which leads to s_1, at state s_0 for most cases. Then, the TD error will converge on

$$\text{TD error}_1 = \text{reward} + V(s_1) - V(s_0). \tag{5}$$

If the agent chooses action a_2, which leads to s_2, as an exploratory action, then the TD error is

$$\text{TD error}_2 = \text{reward} + V(s_2) - V(s_0). \tag{6}$$

Accordingly, the absolute TD error suddenly increases if $V(s_1)$ and $V(s_2)$ are very different, and a common $R(s)$ is greatly affected by this sudden increase in the TD error.[3] In order to reduce such fluctuations caused by action selection, we had better define separate $R(s, a)$ for different action and calculate $R(s)$ as an average of $R(s, a)$, or as $R(s, a^*)$.

As for the actor-critic model, therefore, it is advisable to define the RI for a state as an average of $R(s, a)$ or as the RI for the optimal action.[4]

[2] We confirmed this by numerical experiments (data not shown).

[3] Fluctuation of the common $R(s)$ occurs also with Q-learning, but the amount of fluctuation is much smaller, thanks to separate $Q(s, a)$ for different actions.

[4] However, a result of numerical experiment showed that the learning performance was not remarkably different among these three methods (data not shown).

3.3. Update rule of the value function and its reliability index

Next, we explain the learning rule. The rules for updating value functions are the same as in the ordinary TD learning method. We will present the rules for tabular-type value functions and weighted-sum-type functions separately.

3.3.1. Tabular-type value functions

In Q-learning, Q-values are updated by

$$\Delta Q(s, a) = \alpha\delta, \tag{7}$$

$$\delta = r + \gamma \max_{a'} Q(s', a') - Q(s, a), \tag{8}$$

where α is a positive constant called the 'learning coefficient', δ is the TD error, γ is a constant called the 'discount rate', and r is the reward/punishment. In the actor-critic model, on the other hand, the value functions are updated by

$$\Delta V(s) = \alpha\delta, \tag{9}$$

$$\Delta q(s, a) = \beta\delta(1 - \pi(s, a)), \tag{10}$$

$$\delta = r + \gamma V(s') - V(s), \tag{11}$$

where α is a positive constant, and $\pi(s, a)$ is the probability of choosing action a in state s.

On the other hand, the RI is updated using the following equations for Q-learning and the actor-critic model, respectively

$$\text{Q-learning (common } R(s)\text{)}: \tag{12}$$

$$\Delta R^2(s) = \alpha_R(\delta^2 + \gamma_R R^2(s') - R^2(s)),$$

$$\text{actor-critic model}: \quad R(s) = \text{average}_a R(s, a), \tag{13a}$$

$$\Delta R^2(s, a) = \alpha_R(\delta^2 + \gamma_R R^2(s') - R^2(s, a)). \tag{13b}$$

Here, $R^2(s)$ is the square of $R(s)$. α_R is a positive constant that determines the rate of modification of the RI. $\gamma_R (0 \le \gamma_R \le 1)$ gives the magnitude of how the RI of the succeeding state affects the RI of the current state. Therefore, this update rule means that $R(s)$ increases or decreases depend on whether the sum of the TD error (i.e. the current error) and the RI of the succeeding state (i.e. the uncertainty of future error) is larger or smaller than $R(s)$. This is reasonable, because the reliability of a value function for a given state should depend on how reliable the value functions of the succeeding states are. In the special case when the uncertainty of the future state becomes zero (i.e. $R(s') = 0$), $R(s)$ indicates the variability of the TD error. Another special case is $\gamma_R = \gamma^2$, for which Sato et al. (2001) theoretically proved that $R^2(s)$ converges on the variance of the reward in a stationary environment. Below, we refer to γ_R as the 'RI discount rate,' because this term shows how the system discounts the uncertainty of a future state when estimating the uncertainty of the current state.

The initial value of $R(s)$ can be set arbitrarily as long as it is not too large with respect to the order of the value function. Actually, it had little effect on the result in the numerical experiments. Rather, it is helpful to define the minimum limit of the RI (denoted R_1, below). This prevents the RI from being zero, which is required for the calculation in Eq. (2). In general, R_1 should be sufficiently small with respect to the order of the value function.

3.3.2. Weighted-sum-type value functions

Although tabular-type value functions are easy to implement, in practice they cannot be applied to a complex learning system with a large number of states or continuous state variables. One of the common methods used for such cases is to realize a value function as a linear sum of feature values that represent the system states (Sutton & Barto, 1998). Concretely, an action value function and a state value function are given by

$$Q(s,a) = Q'(\boldsymbol{f}(s),a) = \boldsymbol{w}(a)\cdot\boldsymbol{f}(s), \tag{14a}$$

$$V(s) = V'(\boldsymbol{f}(s)) = \boldsymbol{w}\cdot\boldsymbol{f}(s), \tag{14b}$$

where $\boldsymbol{f}(s)$ is a K-dimensional feature vector (s represents the state), and $\boldsymbol{w}(a)$ or $\boldsymbol{w}$ is a weight vector.

In this case, the value function is updated using the following rule. For an algorithm using an action value function, such as Q-learning

$$\Delta\boldsymbol{w}(a) = \alpha\delta\boldsymbol{f}(s), \tag{15}$$

where α is a learning coefficient and δ is the TD error defined in Eq. (8). By contrast, for the actor-critic model

$$\Delta\boldsymbol{w}_V = \alpha\delta\boldsymbol{f}(s), \tag{16}$$

$$\Delta\boldsymbol{w}_q(a) = \alpha\delta(1-\pi(s,a))\boldsymbol{f}(s), \tag{17}$$

where $\boldsymbol{w}_V$ and $\boldsymbol{w}_q(a)$ are the weight vectors for the critic and actor, respectively, and δ is the TD error defined in Eq. (11).

When the value function is given by Eq. (14a) or (14b), the RI should be defined for a weight vector because the agent tries to estimate the weight vector. Theoretically, the RI for a weight vector should be given in a matrix, since the RI originally corresponds to a covariance matrix for random variables.[5] However, here we explain a simpler method, in which a RI is defined for each element of the weight vector, without considering correlation between different features. This is because a matrix-type RI requires much more memory ($K\times K$).

Now, we describe a concrete definition of the RI ($\boldsymbol{R}_w$) for the weight vector ($\boldsymbol{w}$) and its update rule. First, we deal with the case of Q-learning. In this case, the RI for a value function is given by

$$R^2(s,a) = \sum_{k=1,K} R^2_{wk}(a) f^2_k(s), \tag{18}$$

where $R_{wk}(a)$ and $f_k(s)$ are the kth components of $\boldsymbol{R}_w(a)$ and of $\boldsymbol{f}(s)$, respectively. Since the matrix-type RI was given by

$$R^2(s,a) = \sum_{j,k=1,K} \boldsymbol{R}^2_{wjk} f_j(s,a) f_k(s,a) = \boldsymbol{f}^{\mathrm{T}}(s)\boldsymbol{R}^2_w(a)\boldsymbol{f}(s). \tag{19}$$

($\boldsymbol{R}_{wjk}$ is the (j,k) element of $\boldsymbol{R}_w$), we can see that the simplified form in Eq. (18) ignores the cross-terms (i.e. the non-diagonal elements in R_w) in Eq. (19). Note that $R^2(s,a)$ is averaged over a or $R^2(s,a^*)$ is selected when it is used for action selection.

Conversely, the update rule for $R_{wk}(a)$ (or $\boldsymbol{R}_w(a)$) is given by

$$\begin{aligned}\Delta R^2_{wk}(a) &= \alpha_R(\delta^2 + \gamma^2_R R^2(s') - R^2(s,a)) \\ &\quad\times (R^2_{wk}(a) f^2_k(s)/R^2(s,a)) R^2_{wk}(a), \qquad (20a)\\ &= \alpha_R[(\delta^2 + \gamma^2_R R^2(s'))/R^2(s,a) - 1] f^2_k(s) R^4_{wk}(a). \qquad (20b)\end{aligned}$$

Eq. (20a) means that the error in the total RI ($\delta^2 + \gamma^2_R R^2(s') - R^2(s,a)$) is delivered to the RI of each feature component with a weight of $R^2_{wk}(a)f^2_k(s)/R^2(s,a)$. (Note that $R^2(s,a)$ is given by the sum of $R^2_{wk}(a)f^2_k(s)$, as in Eq. (18).) The validity of this rule can be understood in relation to the update rule for the matrix-type RI (equivalent to the measurement update rule of a linear Kalman filter) given by

$$\boldsymbol{R}^2_w(\mathrm{new}) = [\boldsymbol{R}^2_w(\mathrm{old})^{-1} + \boldsymbol{f}N^{-1}\boldsymbol{f}^{\mathrm{T}}]^{-1}, \tag{21}$$

where N is a matrix specifying the measurement noise, and s and a are eliminated for simplicity. Setting $N = \sigma^2\boldsymbol{I}$ and ignoring the non-diagonal components in this equation, we obtain an update rule for the kth diagonal term as

$$\begin{aligned}\boldsymbol{R}^2_{wkk}(\mathrm{new}) &= R^2_{wkk}(\mathrm{old})[1 + R^2_{wkk}(\mathrm{old}) f^2_k/\sigma^2]^{-1} \\ &\approx R^2_{wkk}(\mathrm{old})[1 - R^2_{wkk}(\mathrm{old}) f^2_k/\sigma^2] \qquad (22)\end{aligned}$$

(if the second term is sufficiently smaller than 1), or

$$\Delta R^2_{wkk} \approx -R^2_{wkk}\cdot R^2_{wkk} f^2_k/\sigma^2 = -(1/\sigma^2) f^2_k(s,a) R^4_{wk}. \tag{23}$$

This resembles the second term of Eq. (20b), if we regard $\alpha_R = (1/\sigma^2)$. By contrast, the first term of Eq. (20b) represents the increment of the variance based on the TD error, which may correspond to the increment of the variance from the prediction in the Kalman filter algorithm. If this term is zero, which means that both the TD error and the future RI are zero, the RI of the current state will decrease uniformly. Nevertheless, the RI is kept at a certain level while some TD error is observed.

[5] The original work on the reliability concept (Sakaguchi & Nakano, 1992) dealt with the matrix type of reliability. In the original paper, the weight vector and its covariance matrix were updated using the Kalman filter algorithm. The algorithm presented here approximates the update rule used for the covariance matrix of a Kalman filter.

The algorithm for the actor-critic model is almost the same. First, the RI for a value function is defined by

$$R^2(s,a) = \sum_{k=1,K} R^2_{Vk}(a) f^2_k(s), \tag{24}$$

where $R_{Vk}(a)$ is the kth element of the RI of the weight vector $\boldsymbol{w}_V(a)$. The index R_{Vk} (or $\boldsymbol{R}_V$) is updated using

$$\Delta R^2_{Vk}(a) = \alpha_R[(\delta^2 + \gamma^2_R R^2(s'))/R^2(s,a) - 1] f^2_k(s) R^4_{Vk}(a). \tag{25}$$

Again, $R^2(s,a)$ is averaged over a or $R^2(s,a^*)$ is chosen when it is used for action selection.

3.4. Adaptive adjustment of the learning coefficient

As described above, the RI presumably converges on the standard deviation of the reward sum (or TD error) after sufficient learning steps, in a stationary environment. This means that the RI and TD error assume comparable values after learning. This, in turn, implies that learning has not progressed well if the RI and TD error are very different.

If the TD error is much larger than the RI, that is, if the agent observes an unexpectedly large error, this suggests that the environment has changed drastically. In such cases, it is desirable to increase the learning coefficient so that the agent catches up with the environmental change more quickly.[6] If the TD error is much smaller than the RI, on the other hand, it means that the RI is meaninglessly large. It is desirable to increase the learning coefficient also in this case.

There are various ways of implementing this idea in an actual algorithm. The following rule gives one such possibility

$$\alpha = \min(\alpha_1, \alpha_0 \max(1, |\log \delta - \log R(s)|)), \tag{26a}$$

and

$$\alpha_R = \min(\alpha_{R1}, \alpha_{R0} \max(1, |\log \delta - \log R(s)|)), \tag{26b}$$

where α_0 and α_{R0} are the learning coefficient in ordinary trials, and α_1 and α_{R1} are the maximum limits of the respective learning coefficients. Many variations may exist, including more sophisticated ones.

3.5. Determining parameter η

We would like to discuss how to determine the arbitrary parameter η. Some suspect that there is no good way to determine η, as there are no guidelines for the temperature parameter. This is partially correct. If we wish to determine the optimal value of this parameter, we must search for it by trial and error.

However, we can approximate its value using the following guideline. After sufficient learning steps, the RI converges on the standard deviation of the value function or its lowest limit R_1 when the RI discount rate $\gamma_R = 0$. In this situation, the ratio R_1/η corresponds to the temperature parameter T in the original Boltzmann rule (see Eqs. (1) and (2)). Therefore, if we can give a desirable temperature in an asymptotic situation, we can determine the parameter η based on this relation. For example, imagine that the difference in the value functions for different actions is around 1. Then, the agent would select the action whose value function is the highest almost deterministically, if T is 0.01 or less (because $\exp(\Delta Q/I) = \exp(1/0.01) > 10^{43}$). In this case, the Boltzmann method results in greedy action-selection with $T = 0.01$ in an asymptotic situation. Therefore, we can set $\eta = 1$ when $R_1 = 0.01$ if we want the agent to perform in an equivalent manner in an asymptotic case.

This becomes complicated when γ_R is close to 1, because the RI of a current state is given by the sum of the RI values for succeeding states and the standard deviation of the value function. If the RI of the future state does not decrease enough, the RI of the current state barely reaches the minimum limit R_1, inevitably implying that the asymptotic performance was worse than designed.

Therefore, there is no guarantee that η determined using this guideline always gives good results. The behavior of the proposed algorithm following this guideline is shown in the numerical experiment and further discussed in Section 4.3.

3.6. Incompatibility with actor-critic models

Thus far, we have given the algorithm of the RI-based method in parallel for the Q-learning system and for the actor-critic model. Here, we would like to point out the possibility that the RI-based actor-critic model may not work well.

The most essential point is that the action selection in the actor-critic model is not directly linked to the state value function. The function used in the actor module ($q(s,a)$) is updated separately from the state-value function $V(s)$, and values of $q(s,a)$ are not necessarily comparable to the value function. This brings along a discrepancy that the reliability itself is determined based on $V(s)$ whilst the reliability operates on different quantity $q(s,a)$ in action selection. This discrepancy is inconsistent with the original simple philosophy of reliability and makes the behavior of the RI-based actor-critic model difficult to understand.

For example, the guideline for determining η discussed in the previous section cannot be applied to the actor-critic model. Because values of $q(s,a)$ are not necessarily comparable to the value function (generally, the range of $q(s,a)$ is larger than that of $Q(s,a)$ though its absolute value depends on the learning coefficient α), it is hard to know appropriate values of the temperature parameter, T, and thus, η.

[6] In a separate paper (Sakaguchi & Takano, in preparation), we discuss a modular learning network that switches modules when the observed error is much larger than the RI of the currently selected module.

Actually, this incompatibility gives harmful effects on the behavior of RI-based actor-critic models. Some examples will be illustrated in the numerical experiment, below.

4. Numerical experiments

4.1. Experiment 1: maze with walls

4.1.1. Problem

First, we examine the learning performance of the proposed algorithm with a common maze problem. We also discuss whether the size of a problem affects performance.

The structure of the maze problem is shown in Fig. 1. The agent's task is to find the best action-sequence for moving from the start to the goal in a 2D-maze or grid world. The agent receives a punishment of 1 for every step taken, and a punishment of 2 if it leaves the field. The agent tries to find a path that minimizes the total punishment along the path, where the optimal path agrees with the shortest path.

The maze size was $12 \times 4N$, and the minimum number of steps required for the solution was $8N + 7$. We examined the learning performance for the cases $N = 1, 2, 4, 8$, and 16.

4.1.2. Condition

We adopted Q-learning and the actor-critic model to solve this problem, and compared the performance of the conventional methods and the proposed method separately for each architecture. For conventional Q-learning, we examined both the ε-greedy and Boltzmann action selection methods; here, we deal mainly with the Boltzmann method. We updated $R(s)$ commonly for different actions.

As the procedure involved in each learning step was explained above, here, we summarize the parameter values used in the experiment. The learning coefficients for the value function were $\alpha_0 = 0.1$, $\alpha_1 = 0.5$, $\beta_0 = 0.01$, and $\beta_1 = 0.05$, and that for the reliability was $\alpha_R = 0.1$. The discount rate (γ) was 0.99. The lowest limit of the RI (R_1) and parameter η of the Q-learning system were set to 0.01 and 1, respectively, so that the asymptotic performance of the system was compatible with that of the Boltzmann method with $T = 0.01$ (Section 3.5). As for the actor-critic model, we set $R_1 = 0.01$ and $\eta = 10$. In order to see the effect of the RI discount rate (γ_R) (Section 3.3), we ran the experiments with various values of γ_R, although the following explanation deals mainly with the extreme cases $\gamma_R = 0$ and 0.99 ($= \gamma^2$).

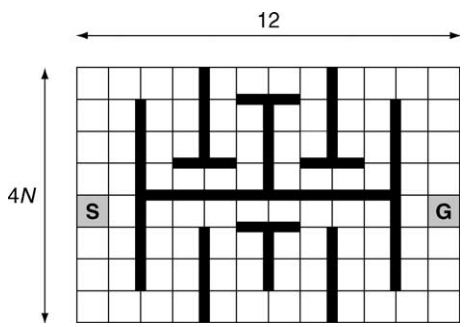

Fig. 1. A maze problem. The agent has to move from the start (S) to the goal (G) using the shortest path. The minimum number of steps to reach the goal is $8N + 7$.

To evaluate the learning performance, we first examined the learning curves for various methods, which helped us to understand the performance intuitively. Next, we ran statistical tests to compare the different methods. In the following, the performance of the Boltzmann method with temperature $T = 0.01$ (for Q-learning) or $T = 0.1$ (for the actor-critic model) is treated as the baseline for this comparison. We tried various temperatures in a preliminary examination, and found that these values marked close to the best performance. For the Q-learning system, the setting $T = 0.01$ is also favorable for comparing the conventional and proposed methods, because the parameters of the RI-based system were determined so that its asymptotic performance was compatible with this condition. We repeated the procedure 30 times using different pseudo-random numbers.

4.1.3. Results

Since the learning performance did not differ remarkably when the maze was small (i.e. $N = 1$ and 2), we discuss only the results for the larger mazes.

First, we show the learning curves of some representative cases, where a Q-learning agent solved a maze of size $N = 4$. Fig. 2 depicts the quartiles of punishment (i.e. the number of steps required for an episode) as a function of the number of episodes on a logarithmic scale, together with the best- and worst-case data. We used the average over 25 episodes to smooth the original bumpy curves. Data are shown for the following cases: the Boltzmann method with a fixed temperature $T = 0.01$ (i.e. the baseline condition) (Fig. 2a), the RI-based method with $\gamma_R = 0$ (Fig. 2b and c), and the RI-based method with $\gamma_R = 0.99$ (Fig. 2d and e). Fig. 2c and e show cases in which an adaptive learning coefficient was introduced.

The learning speed of the RI-based method with $\gamma_R = 0$ was almost the same as that of the baseline algorithm. Since the parameters were set so that the asymptotic randomness equaled the baseline method, this implies that the proposed algorithm behaved no better than the baseline performance when $\gamma_R = 0$. Conversely, learning was accelerated remarkably with the adaptive learning coefficient, as shown in Fig. 2c. The most notable difference is in the shape of the learning curve. With this option, the slope of the learning curve (on a logarithmic scale) remained almost constant until the end of learning while without this option, it became flatter as learning proceeded. As a result, the median punishment reached the minimal value much earlier with this option than without this option.

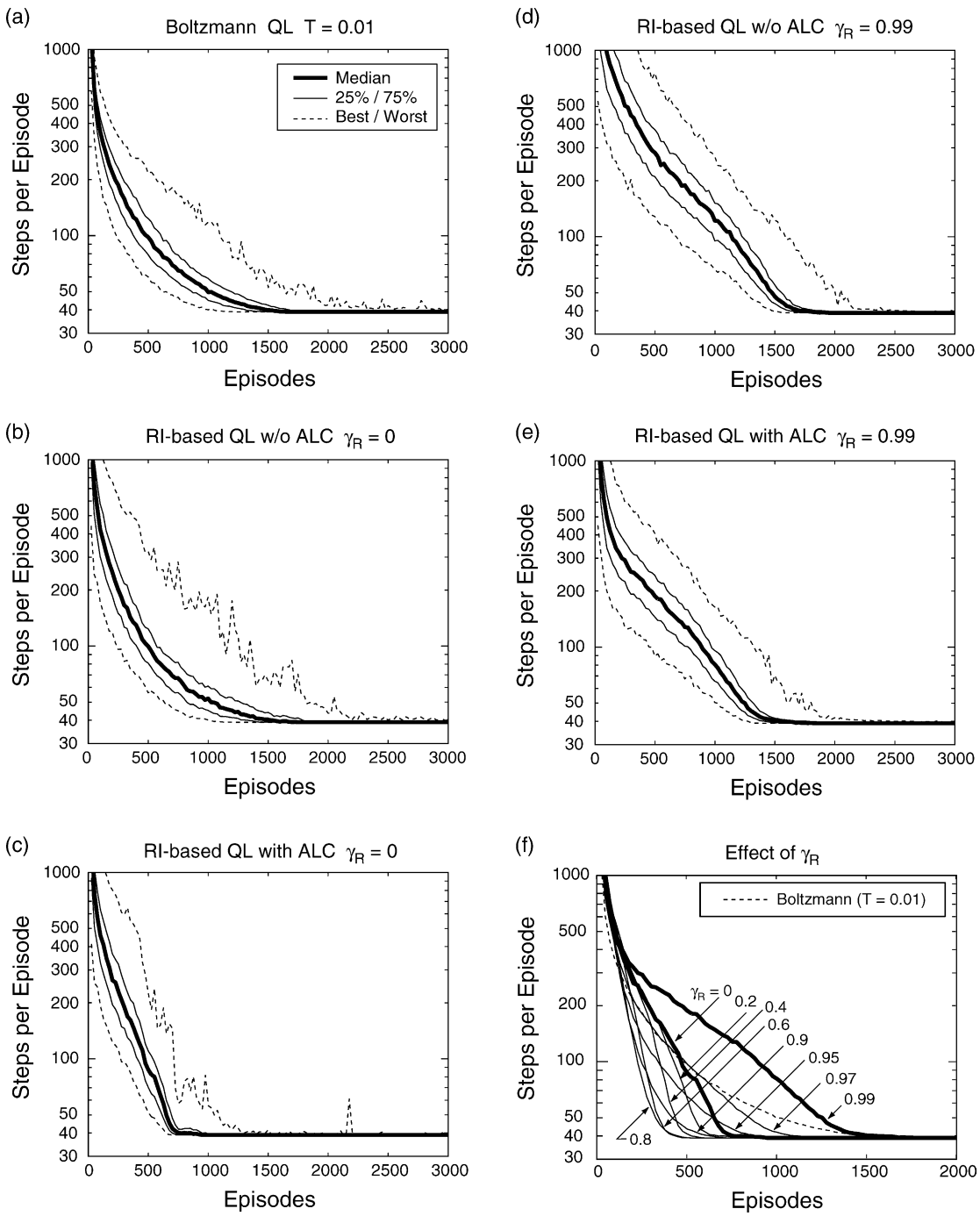

Fig. 2. Results of Experiment 1. The learning curves are drawn on a logarithmic scale for the case in which a Q-learning agent solves a maze of size $N = 4$. (a–e) show the quartiles of punishment in 30 experiments as a function of the number of episodes, together with the best- and worst-case data. The results are shown for the (a) Boltzmann method with $T = 0.01$, (b and c) the RI-based method with $\gamma_R = 0$, and (d and e) the RI-based method with $\gamma_R = 0.99$. (c and e) show the results with the adaptive learning coefficient (ALC). (f) shows the median data for the RI-based method with various values of γ_R. See the text for details.

On the other hand, when $\gamma_R = 0.99$, the learning speed was not as fast as in the above cases; the time at which the median punishment reached the asymptotic level was delayed. This is plausible, because with a large γ_R, the RI of a given state cannot decrease until the RI values of the succeeding states are reduced sufficiently. However, we should note that the slope of the learning curve remained steep until the end of learning, even without the adaptive learning coefficient. This implies that an agent with a large γ_R learns slowly at first, but catches up with the other agents beginning at the midway point. Again, learning was remarkably facilitated with the adaptive learning coefficient, as in Fig. 2e.

Some people may wonder how the performance is when the RI discount rate takes intermediate values. When no adaptive coefficient was introduced, learning gradually slowed down with larger γ_R (data not shown). Interestingly, however, the situation was different when the adaptive coefficient was introduced. Fig. 2f shows the median learning curves in this case, where the two thick lines indicate the results for two extreme cases ($\gamma_R = 0$ and 0.99), and the other thin lines indicate results for intermediate cases (the broken line is the result using the baseline Boltzmann method). When $\gamma_R < 0.8$, the learning speed improved with larger γ_R, although it deteriorated as γ_R approached 1. We have no explanation for this facilitation, but a balanced combination of a slow decrease in the RI (with a large γ_R) and fast learning of the value function (with a large α) might bring about this result. Here, we only report this curious phenomenon.

In summary, the above results led to three conclusions. First, the performance of the proposed method with $\gamma_R = 0$ is almost the same as with the conventional Boltzmann method. Second, the learning performance changes with the RI discount rate γ_R : when γ_R is large, learning is slower initially, but reaches the asymptote more rapidly, as compared to the case $\gamma_R = 0$. Third, adaptive adjustment of the learning coefficient works well.

To test these views, we ran statistical tests. We performed the Wilcoxon rank sum test for each bin of 50 episodes (the t-test was not adopted because the punishment did not obey a normal distribution). Fig. 3 shows the change in the p-values of the tests as a function of episode, where several methods were compared with the baseline Boltzmann method ($T = 0.01$). The left and right columns correspond to the results in cases $N = 8$ and 16, respectively. The p-value is shown on a logarithmic scale and the centerline corresponds to $p = 1$. The gray region indicates no significant difference (significance level 5%), and location of the curve above or below this region indicates whether the performance of a given method was significantly superior or inferior to the baseline method, respectively. Vertical dotted lines indicate the time when the median data of the baseline method reached the optimal level.

This figure clearly supports the above views. First, the performance of the RI-based algorithm with $\gamma_R = 0$ was comparable with, but never superior to, the baseline method (Fig. 3a). The shape of this curve resembles the curve for the Boltzmann method with a higher temperature parameter (i.e. $T = 0.1$), shown in Fig. 3c. This is reasonable, because in an equivalent manner the RI-based algorithm realizes a higher temperature parameter in the course of learning. By contrast, the performance was significantly improved when the adaptive learning coefficient was introduced. Performance surpassed the baseline method, and it remained superior until the end of learning.

Conversely, the RI-based method with $\gamma_R = 0.99$ took many more steps in the first phase of learning. Even with this slow start, it surpassed the baseline method by the midpoint of learning (Fig. 3b). This represents the features of the proposed method well, i.e. it reflects that the initial phase of learning involves much trial and error. Again, learning was facilitated using the adaptive learning coefficient.

Finally, we summarize the properties of other conditions that have not been commented on above. First, the behavior of the ε-greedy algorithm was comparable with that of the baseline algorithm initially, but its asymptotic performance was significantly worse, especially for larger problems (Fig. 3c). This is inevitable, because ε-greedy agents are compelled to make exploratory actions at a given rate and this forced exploration has an increasing effect as the problem size increases. Conversely, learning was slowed with a higher temperature parameter ($T = 0.1$) throughout the learning steps (Fig. 3c).

Actor-critic models (Fig. 3d) differed from the Q-learning system in several aspects. First, the asymptotic performance of the actor-critic model was not as good as that of Q-learning: the actor-critic agent could not reach the optimal solution in some experimental sessions, and its ratio increased with problem size (data not shown). In actuality, the median punishment did not reach the optimal value when $N = 8$ or 16 (which is why no vertical dotted lines are drawn in (d)). This tendency was common to the conventional and proposed methods. Second, unlike the Q-learning system, learning with the proposed method was faster initially, but slowed halfway through. In other words, the RI-based actor-critic model showed just the opposite features to the RI-based method. This confirms our suspicion that the RI-based method may not compatible with the actor-critic models.

A possible reason for this phenomenon is an indirect link between the values of $q(s,a)$ and the RI. The update rule for $q(s,a)$ (Eq. (10)) includes the term $1 - \pi(s,a)$, where $\pi(s,a)$ is the action selection probability. This term was originally introduced to regulate the increase in $q(s,a)$ when the corresponding $\pi(s,a)$ approaches 1; but if the probability is restrained by the large RI value, then $q(s,a)$ could unnecessarily increase, and as a result, action selection policy would be fixed precociously.

Therefore, the specificities of the actor-critic model stem from its architecture, in which the RI is defined for the critic's $V(s)$ but it affects action selection based on the actor's $q(s,a)$, as we discussed in Section 3.6.

In summary, the result of Experiment 1 shows that in a stationary environment, the proposed method with a large RI discount rate (γ_R) showed its characteristic feature (i.e. there was more trial and error in the initial phase and rapid learning from the midpoint on), especially with an adaptive learning coefficient. The next experiment examines the proposed method's ability to adapt to changing environments.

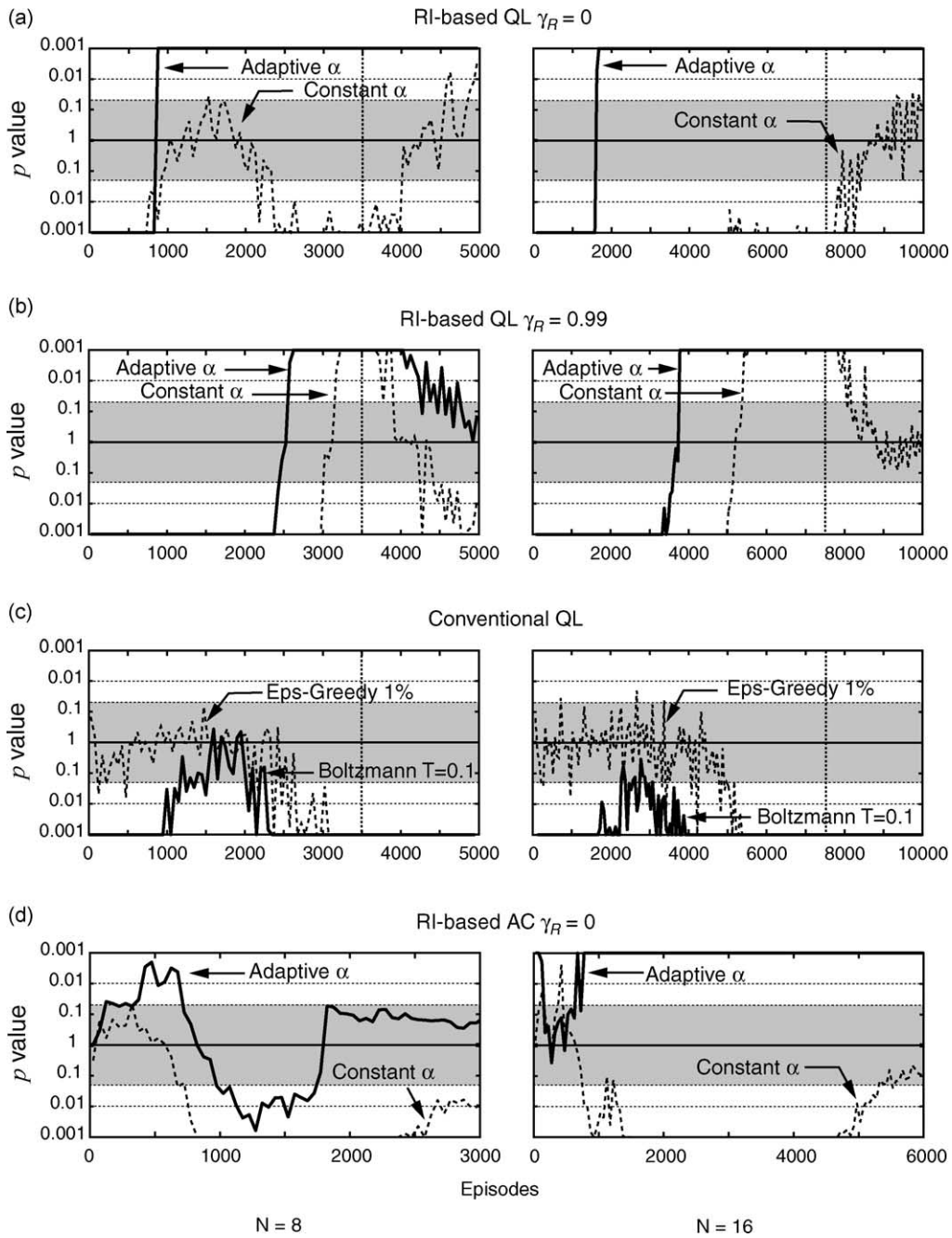

Fig. 3. Results of Experiment 1. The p-values of the Wilcoxon test are depicted as a function of the number of episodes. (a–c) show the results for the Q-learning system, where the target methods are compared to the Boltzmann method with $T = 0.01$ (baseline method): (a) the RI-based method with $\gamma_R = 0$, (b) the RI-based method with $\gamma_R = 0.99$, and (c) the ε-greedy method ($\varepsilon = 1\%$) and Boltzmann method with $T = 0.1$ (i.e. higher temperature). (d) shows the results for the RI-based actor-critic model with $\gamma_R = 0$, where the baseline is the case $T = 0.1$.

4.2. Experiment 2: Temporally variable grid world

4.2.1. Problem and conditions

In the second experiment, we examined the ability of different methods to adapt to a time-variant environment. To this end, we adopted a non-uniform 2D-grid world problem, where the punishment (or negative reward) given at a state-transition depends on the position in the world, and its spatial distribution varies temporally. The agent's task is to find the best action-sequence for moving from the start to the goal with minimum punishment. The punishment distributions were designed so that the total punishment along the optimal path was always 20, for any environment.

The environment changed every 5000 episodes, and one experimental session consisted of 50,000 episodes (i.e. 10 different environments). Since the minimum punishment was kept constant, the punishment in any episode should return to this value if the agent can completely adapt to

the new environment. Therefore, we compared the adaptation ability by examining how closely the punishment returned to the minimum value after every environmental change.

We ran the experiment using two types of environmental change, stepwise and gradual. In the former condition, the punishment distribution was switched abruptly every 5000 episodes, while in the latter condition, the punishment value changed gradually for 3000 episodes (referred to as the changing phase, below), and remained constant for the remaining 2000 episodes (the stationary phase). Note that the punishment along the optimal path occasionally deviated from 20 during the change in the latter condition. In the following, our explanation concentrates on the gradual condition, since the results in the stepwise condition were the same, but less remarkable than those for the gradual condition.

To avoid the results being dependent on a specific punishment distribution, we ran the experiment 50 times using different environment sequences (i.e. in all 500 different environments were used in the experiment) instead of using a fixed sequence, and averaged the learning curves over these sequences.

Here, we report only the results for Q-learning because the RI-based actor-critic model showed only inferior performance as in Experiment 1. The parameter values were the same as in Experiment 1.

4.2.2. Results

We start by comparing the learning curves for the Boltzmann and RI-based methods. The upper panel in Fig. 4a shows the temporal change in the punishment per episode for the Boltzmann ($T = 0.01$) and RI-based ($\gamma_R = 0.99$ with the adaptive coefficient) methods. Quartiles and best/worst data for 50 different sequences are shown on a linear scale. The gray regions indicate the changing phases. The lower panel shows the temporal change in the average entropy of action selection during an episode, which indicates the overall randomness of action selection. To flatten the curves, these values were calculated for bins 50 episodes wide.

First, we would like to discuss the change in entropy of action selection. Generally, the entropy increased at every environmental switch, and then fell back to almost zero, for both algorithms. Note that this cyclic change was also observed for the Boltzmann method with a fixed temperature, because the value function $Q(s, a)$ can take similar values for different actions (a) on route to adaptation. However, the entropy traced different curves with the two methods. First, the amplitude of the entropy change was much larger for the proposed method. Second, the entropy started to increase just after the environment switch for the proposed method, meaning that the agent successfully detected the environment change. Therefore, with the RI-based method, more explorations were performed at environmental changes, as expected.

Next, let us examine the learning performance. The learning curves showed similar patterns with both methods: they increased in the first half of the changing phase, decreased in the second half, and remained essentially constant during the stationary phase. Looking closely at the median curves, however, we see that the punishment in the stationary phase was smaller with the proposed method. It decreased below 25 with the RI-based method, while it remained between 25 and 30 with the conventional method. Moreover, the quartile (25%) curves always returned to the optimal value with the proposed method, but not with the conventional method.

The results for the other conditions are summarized in Fig. 4b, where only the median learning curves are shown. The left figure includes the data for the Boltzmann method with higher temperatures (i.e. $T = 0.1$ and 1) while the right figure shows the data for other RI discount rates (γ_R). The data shown here are for the cases with the adaptive learning coefficient, but the adaptation ability did not change significantly irrespective of this option, presumably because of the gradual environmental change (see the results of the statistical tests below).

The following findings are notable:

1. With the Boltzmann method, using higher temperatures improved the adaptation ability somewhat.
2. The adaptation ability of the proposed method with $\gamma_R = 0$ was comparable to that of the conventional method. This was true irrespective of whether the adaptive learning coefficient was used (data not shown), implying that the RI-based algorithm cannot demonstrate its adaptation ability with a small RI discount rate.
3. The adaptation ability improved with a higher RI discount rate. In this experiment, good performance was obtained for $\gamma_R > 0.9$. This was true both with and without the adaptive learning coefficient.

To compare the performance quantitatively, we ran statistical tests. As in Experiment 1, we took the Boltzmann method with $T = 0.01$ as the baseline method, and compared the other methods to it, using a Wilcoxon test. The results are shown in Fig. 5.

The test supported the above view. When $\gamma_R = 0.99$ (Fig. 5a), the performance of the proposed method was inferior in the changing phase (the gray regions), but superior in the stationary phase (the white regions), to that of the conventional method. This clearly shows that the proposed method made more explorations at environmental change, and performed better in the stationary environment. This feature was observed irrespective of introducing the adaptive learning coefficient; in both cases, significant superiority was observed in five out of nine environments (the first environment was excluded because it does not represent adaptation ability). In a supplementary experiment (data not shown), this superiority continued and was even

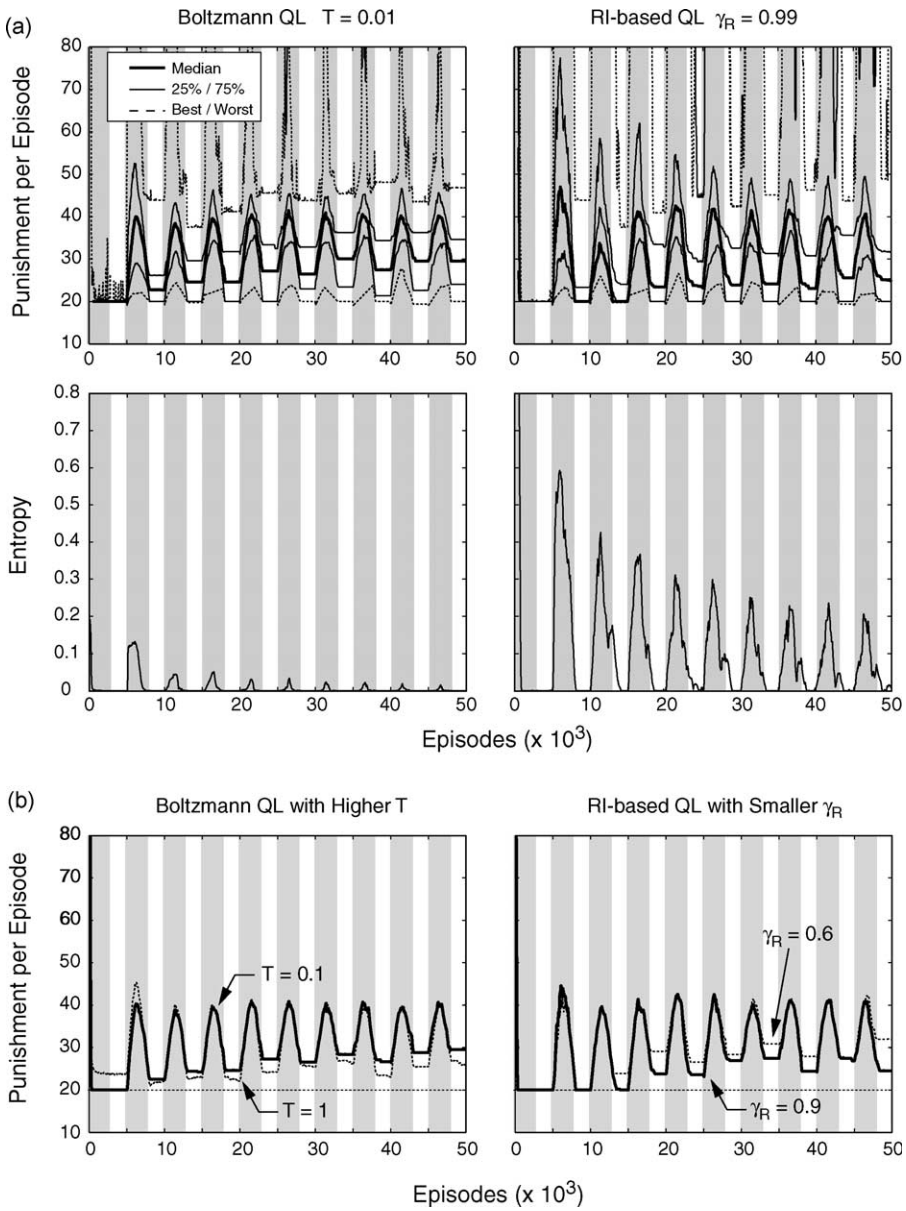

Fig. 4. Results of Experiment 2. (a) shows the learning curves and temporal changes in the action selection entropy for the Boltzmann method with $T = 0.01$ (left panel) and for the RI-based method with $\gamma_R = 0.99$ (right panel). The gray regions indicate the changing phase. The learning curves are given using quartiles and the best/worst data for punishments in an episode. (b) summarizes the results in the other conditions, where the learning curves are given using the median data only. See the text for details.

strengthened until at least 100,000 episodes (i.e. 19 environmental switches).

In the case $\gamma_R = 0$ (Fig. 5b), the performance of the proposed method was not significantly different from that of the conventional method. This confirms that the adaptation ability of the proposed method can be demonstrated with a large RI discount rate. This effect probably occurs because the RI of a given state is kept high until those of the succeeding states become sufficiently small, as pointed out above. As for the present problem, this effect reached significance for $\gamma_R > 0.9$ (Fig. 5c), and reached a plateau for $\gamma_R > 0.95$ (data not shown).

Summarizing Experiments 1 and 2, when the environment was stationary, learning was fastest with $\gamma_R = 0$. Using a large γ_R slowed initial learning, but improved the ability to adapt to environmental change. The best value for γ_R may depend on the problem; at present, we do not know how to decide the best value.

4.3. Experiment 3: acrobot

4.3.1. Problem and conditions

The third example is the famous 'acrobot' problem. In this problem, a robot has to raise the tip of its second link (or foot) above a certain level, utilizing the torque imposed at

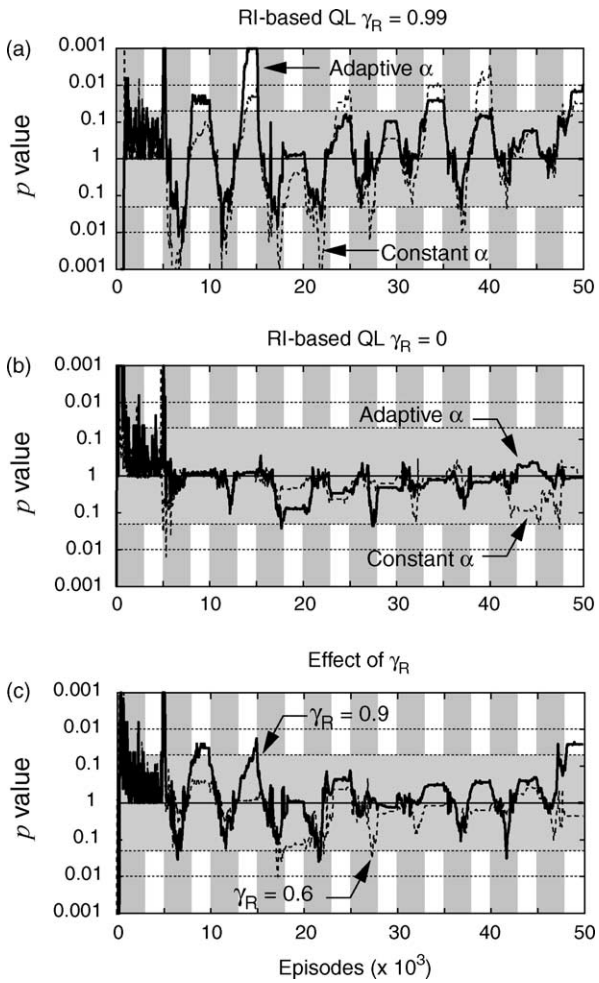

Fig. 5. Results of Experiment 2. The p-values of the Wilcoxon test are plotted as a function of the number of episodes. The RI-based method with several RI discount rates is compared with the Boltzmann method with $T = 0.01$ (baseline method): (a) $\gamma_R = 0.99$, (b) $\gamma_R = 0$, and (c) intermediate values of γ_R. The performance of the proposed method with a large RI discount γ_R was significantly better.

the second joint (Fig. 6). Since it is impossible to raise its body in one swing, the robot has to swing repeatedly, increasing the amplitude of the swing until its foot goes beyond the goal.

The robot's state is represented by a pair of joint angles, i.e. by two continuous variables. This means that a tabular-form value function cannot be used to solve this problem. Here, we used the tiling method of Sutton and Barto (Section 11.3, pp. 270–274, 1998) to represent the state, and applied the weighted-sum type of algorithm described in Section 3.3.2. The detailed settings of the dynamical equation, physical parameters, and tiling design are the same as described in Sutton and Barto (1998). As a result, we used a 25,028-dimensional feature vector with 48 tiles.

An experimental session continued until the robot achieved 2000 goals. To examine ability to adapt to environmental change, we considered the case in which the sensor configurations and robot's physical parameters

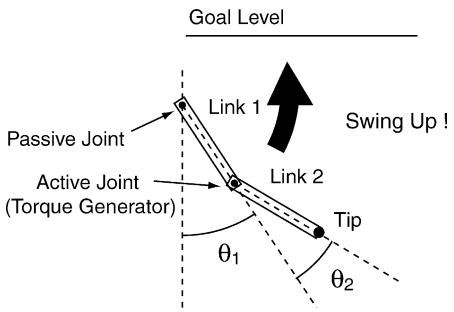

Fig. 6. Experiment 3: The acrobot problem. The 'acrobot' consisted of two links, and a torque generator at their joint. The task was to swing the tip of the second link above a given height.

(such as the length and weight of the links) changed every 500 goals (i.e. four different conditions). Only stepwise environmental change was tested, since a gradual change did not cause a continuous change in the feature vectors when using the tiling method. The experiment was repeated 50 times for statistical analysis. Unlike Experiment 2, an identical environment sequence was used repeatedly in this experiment. We report only the result with the Q-learning system.

For the acrobot problem, it was expected that the value function (i.e. the expected number of steps to the goal) would be more divergent among alternative actions than in the grid-world problem, because the future trajectory depends largely on the direction of the force in the current state. Considering the discussion in Section 3.5, this means that the optimal action selection in the asymptotic situation can be reached with a higher temperature. In actuality, a preliminary examination revealed that the performance of the fixed-temperature Boltzmann method was best with temperatures between 0.1 and 1 and deteriorated at lower temperatures: The robot sometimes could not reach the goal, even when $T = 0.1$, and it had trouble adapting to environmental changes with lower temperatures. Therefore, we chose the case $T = 0.5$ as the baseline condition.

As for the RI-based method, we set the lowest limit of the RI of each weight as $R_{1f} = 0.1$, which gave the RI of the value function of $R_1 \approx 0.7 (= (48 \times 0.1 \times 0.1)^{0.5})$, because we used 48 tiles). This setting seemed sufficient to avoid overflow when calculating the action selection probabilities. To compensate for a large R_1, we set $\eta = 10$, which was larger than in the previous experiments. Accordingly, the equivalent temperature was 0.07 ($= R_1/\eta$, see Section 3.5). Here, note that the fluctuation in the value function might increase with the tiling method, since the feature vector and true system-state do not have a one-to-one correspondence. As a result, we suspected that the RI of the value function did not reach the lowest limit defined above, unlike the previous experiments. This point will be discussed below.

4.3.2. Results

Learning curves and the Wilcoxon test were used to compare the different methods. We show only the results

using the adaptive learning coefficient, since the behavior of the proposed method without this option was comparable to that of the conventional method. We used average data for every bin 10 episodes wide when drawing these curves.

Fig. 7a–d shows the learning curves and the temporal change in the p-value of a statistical test. The learning curves are shown using quartiles (and the best and worst data) in Fig. 7a–c, while only the median is shown in Fig. 7d, where the results for two values of γ_R are depicted. Note that the optimal solution depended on the environment, because the physical parameters changed when the environment changed. For comparison, we show the median punishment of the baseline method when a fresh agent experienced 500 episodes in a given environment (single environment condition).

In general, the tendency was similar to that in the previous experiments, implying that with the Boltzmann method, the punishment changed with every environmental change, but converged on a certain level after about 300 episodes. The median data reached this performance

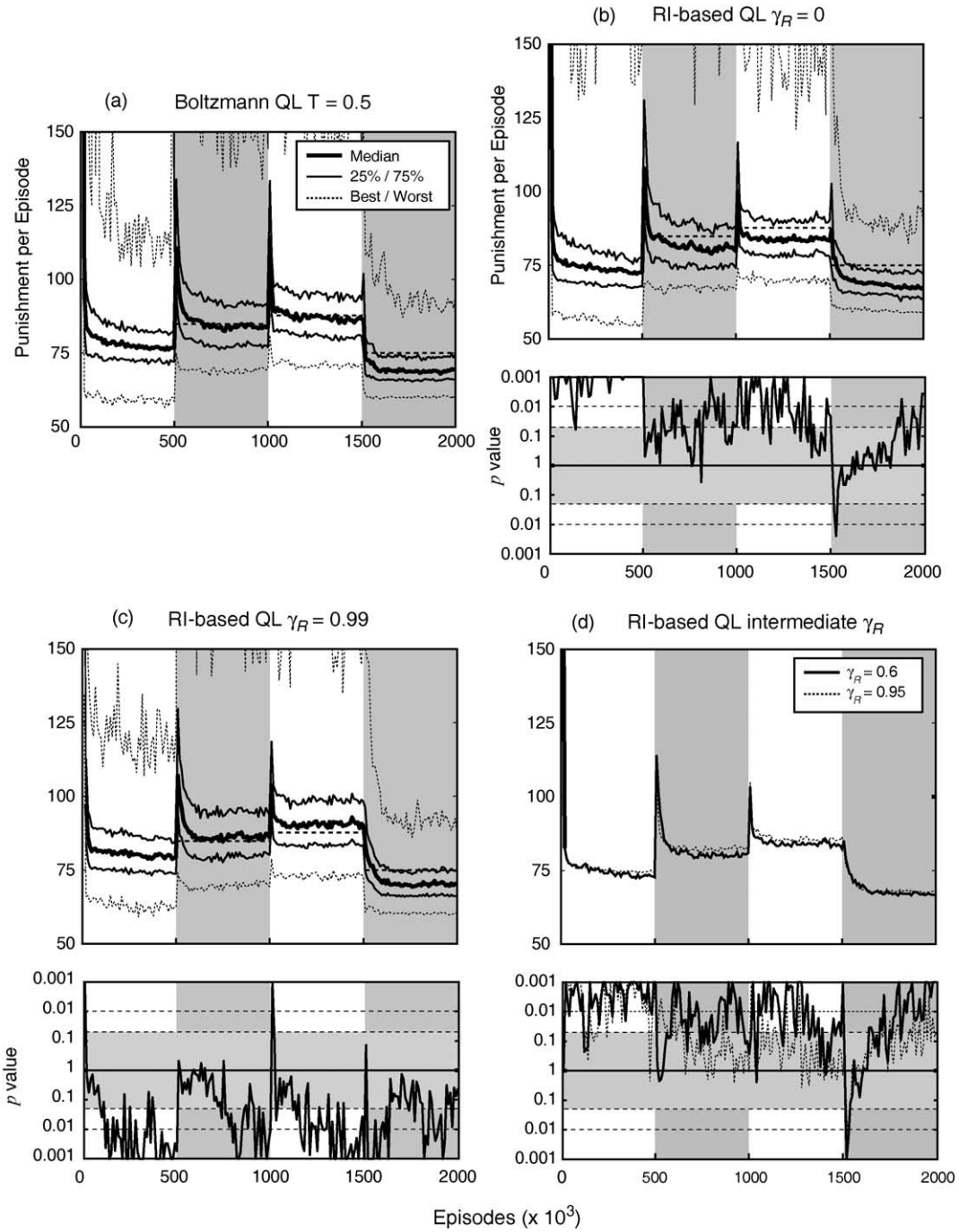

Fig. 7. Results of Experiment 3. The learning curves and relative performance are shown for four cases: (a) the Boltzmann method with $T = 0.5$, (b) the RI-based method with $\gamma_R = 0$, (c) the RI-based method with $\gamma_R = 0.99$, and (d) the RI-based method with intermediate values of γ_R.

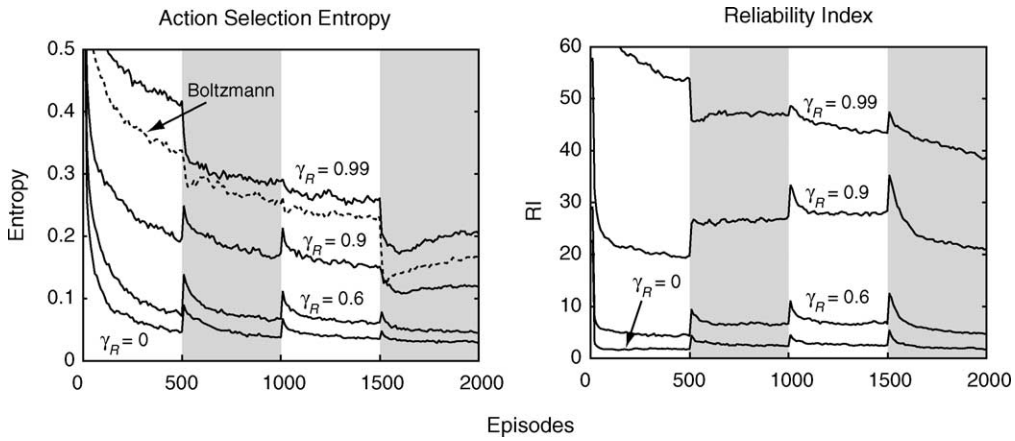

Fig. 8. Results of Experiment 3. Temporal changes in the action selection entropy and the RI are shown. See text for details.

in the single environment condition; consequently, the Boltzmann method successfully adapted to the environmental changes with this temperature.

The RI-based method performed better with $\gamma_R = 0$. This was clearly supported statistically; the p-value was consistently significant. One shortcoming is that the worst-case performance was inferior to the baseline method.

By contrast, the performance with $\gamma_R = 0.99$ was comparable or slightly inferior to that of the conventional method. The performance with intermediate values of γ_R was between these two extremes. The apparent performance changed gradually with larger γ_R, that is, the overall performance deteriorated, but the worst-case performance was improved. Balanced performance was obtained with $0.5 < \gamma_R < 0.9$. Some of this feature can be observed in (d), which shows the results for $\gamma_R = 0.6$ and 0.95.

In order to understand the reason for this tendency, we examined the temporal changes in the action selection entropy and the RI. Fig. 8 shows these data for the baseline and proposed methods with some values of γ_R. We see that both entropy and the RI depended greatly on γ_R. They changed in synchrony with the environmental changes when γ_R was small (i.e. $\gamma_R = 0$ and 0.6). That is, adaptive action selection worked well in these conditions. When $\gamma_R = 0.99$, however, their changes were not as sharp or closely linked to the environmental changes. The latter feature was also observed in the baseline method, consistent with the fact that the performance with $\gamma_R = 0.99$ was comparable to the baseline method. Considering that the Boltzmann method could not adapt to the environmental switches with lower temperatures, this implies that the adaptation ability of the baseline method was realized with this relatively large entropy.

Fig. 8 also shows that the minimum value of the RI did not reach the designed value ($R_1 = 0.7$) in any case. It remained between about 40 and 50 when $\gamma_R = 0.99$, and this is why learning did not proceed in this condition. This result is probably because the fluctuation in the value function remained large due to the tiling method, which prevented the RI from decreasing, and the effect was propagated to previous states when the RI discount rate was large. Therefore, it is not necessarily optimal to set $\gamma_R = 0.99$ ($= \gamma^2$) when there is large fluctuation in the value function.

In summary, the proposed method for weighted-sum type value functions worked as well as for tabular type value functions. In addition, this experiment demonstrated that using the largest RI discount rate (i.e. $\gamma_R = 0.99$) does not always work well if the value function fluctuates. As mentioned above, there are some optimal values of γ_R, which are dependent on the task and system representation. It seems difficult to determine these values of γ_R in advance based on definite guidelines. A possible solution to this problem is to control γ_R dynamically, depending on how learning proceeds. One idea may be to enlarge γ_R occasionally in order to temporally improve the adaptation ability. Further research is required to evaluate the characteristics of this option.

5. Discussion and concluding remarks

5.1. Limitations

As described in Section 1, a general practical solution for the exploration–exploitation problem does not exist. In other words, we cannot expect any ‘almighty’ method to have superior performance for all problems (Moore & Atkeson, 1993; Sutton & Barto, 1998). This, of course, is true of the proposed method, which has some crucial limitations.

The most marked limitation is that the adaptation ability of the proposed method is evident only when the agent can detect the environmental change. In other words, the proposed method does not cope with a change that cannot be observed in the current path of the agent. This shortcoming becomes more pronounced as learning proceeds, because the weight placed on exploratory actions diminishes as learning proceeds (i.e. the path becomes fixed).

This limitation can be easily demonstrated using the ‘shortcut maze’ problem (Sutton & Barto, 1998), which is

a 2D-grid world problem, in which a wall blocks the agent's movements, as in Experiment 1. The crucial feature of this problem is that part of the wall is removed after a certain number of learning steps and a shortcut appears. In order to find this new optimal path, the agent must monitor the wall for changes. That is, the agent has to make exploratory actions to check for environmental change. With the proposed algorithm, however, once sufficient learning steps have elapsed, the agent will not find a new optimal path because it places little weight on exploratory actions.

This deficit is inevitable, given the basic philosophy of the proposed algorithm. The proposed algorithm uses a conservative or optimistic strategy, in which the action selected by the agent changes if the agent suffers from an environmental change. The algorithm uses the greedy policy so long as the agent does not detect a change, and this is why the proposed method performs well after sufficient learning. To find a shortcut, a more adventurous strategy would have to be used. Within the present framework, this can be realized by increasing the RI globally and forcibly after a certain interval. However, this ultimately increases the use of inefficient exploratory actions, degrading the typical performance.

Therefore, efficient action selection after sufficient learning is realized at the cost of an inability to detect unobservable environmental change. Another meta-policy is required to solve this dilemma.

5.2. Relationship to other attempts

Many solutions have been proposed for the exploration–exploitation problem, because this is an essential problem to solve in an on-line reinforcement learning system. This section examines some attempts that are closely related to the proposed method.

One proposed solution is a reinforcement-learning algorithm that uses both expectation and variance of the value function (White, 1988 for review). In most TD learning algorithms, the agent uses only Q-values for action selection, although Q-values simply represent the expectation of the reward function. In some problems, however, especially in the investment field, the agent has to consider variance in the reward function in order to limit the risk of actions. Some recent algorithms solve this problem by estimating the variance of the reward function using on-line learning. The method proposed by Sato et al. (2001) is one such algorithm. They defined an internal variable (corresponding to the RI) that estimates the variance of the reward, and used it to determine the action. Moreover, they used the same update rule as Eq. (12) with $\gamma_R = \gamma^2$ to estimate the variance, and proved theoretically that with this update rule, their estimator converged on the true variance of the reward asymptotically.

Although the concrete procedures of these methods are quite similar, the ultimate purpose of utilizing the variance (or reliability) is quite different. The algorithms involving variance assume that the environment (or the problem) is stochastic, but its statistical property is time-invariant. Using this assumption, the algorithm tries to estimate the variance (i.e. a parameter of the statistical system) and use it for optimal action selection. In contrast, the proposed method does not assume that the environment is time-invariant. More importantly, it focuses mainly on transient processes involved in learning, such as adaptive change in action selection and acceleration of the learning process, rather than on the asymptotic performance after sufficient learning. Most variance-involved TD learning pays no attention to either acceleration of the learning process or meta-learning. In one exception, Williams (1992) suggested using the variance parameter of a Gaussian unit to control the degree of exploration, but he did not give any concrete method for this control.

In another closely related study, Yoshida and Ishii (2001; see also Ishii, Yoshida, & Yoshimoto, 2002) controlled the temperature parameter. They defined the 'confidence' of the state, and reflected it in the temperature parameter of the Boltzmann rule. Confidence means the variance of Q-values among actions. Their essential idea is as follows: If the Q-values are essentially the same among different actions, the state is not critical. To the contrary, if the Q-values differ largely among the actions, such a state is very important because action selection in this state has a large effect on the reward/punishment that the agent receives. Thus, the agent should change the weight placed on exploratory actions according to the variance of Q-values (i.e. confidence). Combining this concept and other techniques, such as an exploration bonus, they proposed an integrated algorithm that gives a good solution to the exploration–exploitation problem.

The most significant difference between our research and theirs is that their algorithm does not treat TD learning, but treats model-based reinforcement learning. Another difference is that the operations in our proposed method are based simply on the concept of reliability, while their algorithm consists of different operations, such as confidence and an exploration bonus. Nevertheless, the concepts involved in their algorithm seem quite similar to our research.

5.3. Concluding remarks

We proposed an adaptive action-selection method based on the concept of reliability, which aims to control the entropy of action selection dynamically according to the uncertainty of the value function (i.e. RI). Numerical experiments illustrated that the proposed method improved the learning performance and ability to adapt to environmental change, with the help of adaptive adjustment of the learning coefficient. In addition, we pointed out that the nature of learning performance depended on the RI discount rate, that is, how much the agent weighted the uncertainty of the future state when estimating the uncertainty of the current state.

Although this article demonstrates the overall characteristics of the proposed method, through numerical experiments, it may have hidden deficits. We hope that its application to various problems will elucidate its characteristics. It is also desirable to evaluate its performance and limitations based on mathematical grounds.

Acknowledgements

This research was partly supported by 'Creating the Brain' project of CREST, JST (Japan). The authors would like to express great thanks to anonymous reviewers for their critical but helpful comments.

References

Barto, A. G., Sutton, R. S., & Anderson, C. W. (1983). Neuronlike elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, *13*, 835–846.

Barto, A. G., Sutton, R. S., & Watkins, C. J. C. H. (1990). Learning and sequential decision making. In M. Gabriel, & J. W. Moore (Eds.), *Learning and computational neuroscience: Foundation of adaptive networks*. Boston, MA: MIT Press.

Brafman, R. I., & Tennenholtz, M. (2000). A near optimal polynomial time algorithm for learning in certain classes of stochastic games. *Artificial Intelligence*, *121*, 31–47.

Dayan, P., & Sejnowski, T. J. (1996). Exploration bonuses and dual control. *Machine Learning*, *25*, 5–22.

Doya, K. (2002). Metalearning and neuromodulation. *Neural Networks*, *15*, 495–506.

Fe'ldbaum, A. A. (1965). *Optimal control systems*. San Diego, CA: Academic Press.

Ishii, S., Yoshida, W., & Yoshimoto, J. (2002). Control of exploration–exploitation meta-parameter in reinforcement learning. *Neural Networks*, 665–687.

Kearn, M., & Singh, S. (1998). Near-optimal reinforcement learning in polynomial time. *Proceedings of 15th International Conference on Machine Learning (ICML-98)*.

Moore, A. W., & Atkeson, C. G. (1993). Prioritized sweeping: Reinforcement learning with less data and less real time. *Machine Learning*, *13*, 103–130.

Ogawa, N., Namiki, A., & Ishikawa, M. (2002). Adjustment of discount rate using index for progress of learning. *Institute for Electronics, Information and Communication Engineers (IEICE) Technical Report NC2002-129* (in Japanese).

Sakaguchi, Y. (1996). Motor planning based on the reliability of internal model. *Transaction of Institute for Electronics, Information and Communication Engineers (IEICE)*, *J79-D-II*, 248–256 (in Japanese).

Sakaguchi, Y., Akashi, Y., & Takano, M. (2001). Visuo-motor adaptation to stepwise and gradual changes in the environment: Relationship between consciousness and adaptation. *Journal of Robotics and Mechatronics*, *13*, 601–613.

Sakaguchi, Y., & Nakano, K. (1992). Motor planning according to reliability of internal model. *Proceedings of International Joint Conference on Neural Networks (IJCNN-93)* (pp. 1321–1324).

Sakaguchi, Y., & Takano, M. (2001). An algorithm of reinforcement learning based on reliability of value function. *Proceedings of the 11th Annual Meeting of Japanese Neural Network Society (JNNS-2001).* (pp. 103–104) (in Japanese).

Sakaguchi, Y., & Takano, M. (under review). Reliability of internal prediction/estimation and its application. II. An incremental modular learning network.

Sato, M., Kimura, H., & Kobayashi, S. (2001). TD algorithm for the variance of return and mean-variance reinforcement learning. *Transactions of the Japanese Society for Artificial Intelligence*, *16*, 353–362. (in Japanese).

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, *3*, 9–44.

Sutton, R. S. (1990). Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. *Machine Learning, Proceedings of the Seventh International Conference*, 216–224.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning*. Boston, MA: MIT Press.

Thrun, S. G., & Möller, K. (1992). Active exploration in dynamic environments. In J. E. Moody, S. J. Hanson, & R. P. Lippman (Eds.), (*4*) (pp. 531–538). *Advances in neural information processing system*, Los Altos, CA: Morgan Kaufmann.

Watkins, C. J. C. H., & Dayan, P. (1992). Technical note: Q-learning. *Machine Learning*, *8*, 279–292.

White, D. J. (1988). Mean, variance, and probabilistic criteria in finite-Markov decision processes: A review. *Journal of Optimization Theory and Applications*, *56*, 1–29.

Williams, R. J. (1992). Simple statistical gradient-following algorithm for connectionist reinforcement learning. *Machine Learning*, *8*, 229–256.

Witten, I. H. (1976). The apparent conflict between estimation and control—A survey of the two-armed bandit problem. *Journal of Franklin Institute*, 161–189.

Yoshida, W., & Ishii, S. (2001). Control of exploration and exploitation in reinforcement learning. *Institute for Electronics, Information and Communication Engineers (IEICE) Technical Report NC2001-28* (in Japanese).