PAPER

A Nonlinear Approach to Robust Routing Based on Reinforcement Learning with State Space Compression and Adaptive Basis Construction

Hideki SATOH[†], ***Member***

SUMMARY A robust routing algorithm was developed based on reinforcement learning that uses (1) reward-weighted principal component analysis, which compresses the state space of a network with a large number of nodes and eliminates the adverse effects of various types of attacks or disturbance noises, (2) activity-oriented index allocation, which adaptively constructs a basis that is used for approximating routing probabilities, and (3) newly developed control space compression based on a potential model that reduces the control space for routing probabilities. This algorithm takes all the network states into account and reduces the adverse effects of disturbance noises. The algorithm thus works well, and the frequencies of causing routing loops and falling to a local optimum are reduced even if the routing information is disturbed.

key words: *robust routing, reinforcement learning, multivariate analysis, function approximation*

1. Introduction

Advances in communications technologies have led not only to the Internet but also to various types of networks such as sensor networks [1] and ad hoc networks [2]. To enable effective use of these networks, various dynamic routing algorithms have been developed [1], [2], [3]. However, dynamic routing problems have not been solved completely for these networks. This is because they are essentially equivalent to a large-scale nonlinear control problem for a time-varying system that has a great many state variables and control inputs. That is, not only traffic but also the network structure frequently changes, and the routing problems have a high-dimensional state space and a high-dimensional control space because the network has a large number of nodes. As a result, it is difficult to construct an effective mathematical model of the network, and thus it is almost impossible to obtain the optimum solution within the allowable time. Secure and robust routing is also a difficult problem because of the growing number of attacks for which countermeasures have not yet been implemented [1].

Reinforcement learning [4] has received much attention because it can handle various types of environments without using a mathematical model, and many routing algorithms based on reinforcement learning have been developed [5], [6], [7]. However, all possible state spaces and control spaces are not handled in these algorithms because it is very difficult using reinforcement learning to handle environments with a high-dimensional state space and a high-dimensional control space. Thus, the effectiveness of these algorithms is limited, and they frequently cause routing loops, which makes it difficult to find the optimum route. Moreover, most of these algorithms cannot take the adverse effects of various types of attacks or disturbance noises into account.

A robust routing algorithm based on reinforcement learning that uses three methods has been developed to solve these problems. The first method is state space compression based on reward-weighted principal component analysis [8], which is used to extract the principal elements from a large number of state variables and compress the state space. The second method is adaptive basis construction based on activity-oriented index allocation [9], which is used to update the orthonormal basis and to approximate a control function. These two methods are effective for reinforcement learning with a high-dimensional state space environment. Their use in combination was investigated, and reinforcement learning using them is described in Sect. 2. The third method is newly developed control space compression based on a potential model described in Sect. 3. It is used to reduce the control space for routing probabilities and to reduce the frequencies of causing routing loops and falling to a local optimum. The application of these three methods to routing problems using a nonlinear approach is described in Sect. 3. Following a description of the routing algorithm in Sect. 4, it is shown that the algorithm can identify the approximate optimum route even if there are attacks that disturb the routing information.

2. Reinforcement Learning in High-Dimensional Continuous State Space

Two previously developed methods, state space compression [8] and adaptive basis construction [9], are effective in reinforcement learning for a high-dimensional state space environment. A structure of reinforcement

Manuscript received July 23, 2007.
Manuscript revised February 21, 2008.

[†]The author is with the Future University-Hakodate, Hakodate-shi, 041-8655 Japan.

learning that includes both methods was investigated in detail. The structure is described and its efficiency is clarified in this section.

2.1 Reinforcement Learning

A system with reinforcement learning [4] is divided into two parts: agents and an environment. The former provide control functions, and the latter is the target system to be controlled. An agent observes the state and the reward from the environment, updates a control function accordingly, and outputs a new control input to the environment. The control function and control input are referred to as policy and action, respectively.

Consider a discrete-time, continuous-state, continuous-action environment. Let $\boldsymbol{s} \stackrel{\text{def}}{=} (s_1, \cdots, s_{d_\text{s}})^\text{t}$ be the state vector, $\boldsymbol{u} \stackrel{\text{def}}{=} (u_1, \cdots, u_{d_\text{u}})^\text{t}$ be the action vector, and superscript t denote transposition. An implementation of the actor-critic method [4], [8], which is used to perform reinforcement learning, is summarized in this section. The actor-critic method consists of two parts: an actor and a critic. Let $\boldsymbol{s}_t$ be the state vector at time t and $\boldsymbol{u}_t$ be the action vector at time t. The actor observes state $\boldsymbol{s}_t$ and decides on action $\boldsymbol{u}_t$, which is a sample value from the conditional Gaussian distribution with density $p(\boldsymbol{u}_t|\boldsymbol{s}_t)$ defined by

$$p(\boldsymbol{u}|\boldsymbol{s}) \cong \prod_{d=1}^{d_\text{u}} p_d(u_d|\boldsymbol{s}), \tag{1}$$

where $p_d(u_d|\boldsymbol{s})$ is a one-dimensional conditional Gaussian density function with mean $\mu_d(\boldsymbol{s})$ and standard deviation $\sigma_d(\boldsymbol{s})$.

The critic observes reward r_t and updates $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ so that discount return R_t, defined by

$$R_t \stackrel{\text{def}}{=} \sum_{t'=0}^{\infty} \nu_\text{DR}^{\ t'} r_{t+t'+1},$$

is maximized, where ν_DR is the discount rate ($0 \le \nu_\text{DR} \le 1$). The values of $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ are initially set so that $\boldsymbol{u}_t$ is distributed across the definition domain. Thus, the critic can learn the relationships among r_t, $\boldsymbol{s}_t$, and $\boldsymbol{u}_t$. As the critic progresses in the learning, $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ are updated so that R_t becomes larger. For $1 \le d \le d_\text{u}$, $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ are expressed using the following linear function approximations:

$$\mu_d(\boldsymbol{s}) \stackrel{\text{def}}{=} \sum_{i=0}^{N} \xi_{di}\phi_i(\boldsymbol{s}), \tag{2}$$

$$\sigma_d(\boldsymbol{s}) \stackrel{\text{def}}{=} h_\text{limit}(\sum_{i=0}^{N} \eta_{di}\phi_i(\boldsymbol{s})), \tag{3}$$

where $\{\phi_i(\boldsymbol{s})\}$ is a basis and $h_\text{limit}(\cdot)$ is a monotone increasing function such that $0 \le h_\text{limit}(x) < \infty$ for $-\infty \le x \le \infty$. The update of $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ can be done by adjusting parameters $\xi_{d0}, \cdots, \xi_{dN}$, and $\eta_{d0}, \cdots, \eta_{dN}$ using temporal-difference (TD) learning and the steepest descent method.

When the dimension of $\boldsymbol{s}_t$ is high, N, the degree of expansion of $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$, is too large to perform reinforcement learning. This problem, which is the so-called "curse of dimensionality," is the most serious problem of function approximation and reinforcement learning for a high-dimensional state space.

2.2 State Space Compression Based on Reward-Weighted Principal Component Analysis

Let us consider an environment with a high-dimensional state space. In such a case, it is difficult to perform reinforcement learning, as explained in Sect. 2.1. A state space compression method based on reward-weighted principal component analysis (RWPCA) [8], which was developed to solve such problems, is summarized in this section.

Let $\boldsymbol{x}_t \stackrel{\text{def}}{=} (x_{1;t}, \cdots, x_{d_\text{x};t})^\text{t}$ be the state vector of the environment at time t. The most simple solution is to perform principal component analysis (PCA) [10], which can extract the principal components from $\boldsymbol{x}_t$. However, it is not always true that elements in $\boldsymbol{x}_t$ with large dynamics contribute to increasing the reward. Some elements may be disturbance noises or attacks, which adversely affect the system [1].

To eliminate such adverse effects from $\boldsymbol{x}_t$ and to construct principal axes that contribute to increasing the reward, the resolution of $x_{d;t}$, res_d, is introduced, which represents the importance of $x_{d;t}$ with respect to increasing the reward. Consider the following multiple regression equations [10]:

$$x_{d;t+1} = \beta_{d0} + \sum_{d'=1}^{d_\text{u}} \beta_{\text{u}dd'} u_{d';t} + \sum_{d'=1}^{d_\text{x}} \beta_{\text{x}dd'} x_{d';t}, \tag{4}$$

$$r_{t+1} = b_0 + \sum_{d=1}^{d_\text{u}} b_{\text{u}d} u_{d;t} + \sum_{d=1}^{d_\text{x}} b_{\text{x}d} x_{d;t}, \tag{5}$$

where b_0, $b_{\text{u}d}$, $b_{\text{x}d}$, β_{d0}, $\beta_{\text{u}dd'}$, and $\beta_{\text{x}dd'}$ are the partial regression coefficients. Because it is clear that $|b_{\text{x}d}|$ is proportional to the effect of $x_{d;t}$ on the reward and that $|\beta_{\text{u}dd'}|$ is proportional to the controllability of $x_{d;t+1}$ by $u_{d';t}$, res_d is set so that res_d is proportional to $|b_{\text{x}d}|$ and $|\beta_{\text{u}dd'}|$.

Let $\boldsymbol{x}_{\text{R}\,t}$ ($\stackrel{\text{def}}{=} (res_1 x_{1;t}, \cdots, res_{d_\text{x}} x_{d_\text{x};t})^\text{t}$) be the state vector weighted by the resolution and $\boldsymbol{e}_{\text{R}\,d}$ be the dth eigen vector obtained by PCA with respect to $\boldsymbol{x}_{\text{R}\,t}$. Using principal axes matrix $M_\text{R} \stackrel{\text{def}}{=} [\boldsymbol{e}_{\text{R}\,1}, \cdots, \boldsymbol{e}_{\text{R}\,d_\text{x}}]$, the reward-weighted principal axes matrix M_RW can be defined by

$$M_\text{RW} \stackrel{\text{def}}{=} [\text{diag}[res_d] M_\text{R}]_1^{d_\text{s}}, \tag{6}$$

where $d_\text{s} \le d_\text{x}$ and $[X]_1^{d_\text{s}}$ denotes the matrix that comprises the 1st- to d_sth-column vector in matrix X. State

$\boldsymbol{s}_t$ used by reinforcement learning is set as

$$\boldsymbol{s}_t = M_{\mathrm{RW}}{}^{\mathrm{t}}\boldsymbol{x}_t. \tag{7}$$

Because M_{RW} is composed of d_{s} eigen vectors that affect the reward significantly, we not only can compress the d_{x}-dimensional state, $\boldsymbol{x}_t$, to the d_{s}-dimensional state, $\boldsymbol{s}_t$, but also eliminate the adverse effects of disturbance noises from $\boldsymbol{x}_t$ [8].

2.3 Adaptive Basis Construction Based on Activity-Oriented Index Allocation

Let $\{K(\boldsymbol{s},\boldsymbol{k})\}$ be a multi-dimensional orthonormal basis and $\boldsymbol{k} \stackrel{\mathrm{def}}{=} (k_1, \cdots, k_{d_{\mathrm{s}}})^{\mathrm{t}}$; $\boldsymbol{k}$ is referred to as the index vector. Let $\phi_i(\boldsymbol{s})$ be defined as

$$\phi_i(\boldsymbol{s}) \stackrel{\mathrm{def}}{=} K(\boldsymbol{s},\boldsymbol{k}), \tag{8}$$

and basis $\{\phi_i(\boldsymbol{s})\}$ be a subset of $\{K(\boldsymbol{s},\boldsymbol{k})\}$, where i is referred to as the index of the basis [9]. Let $\boldsymbol{\mathcal{D}}_{\mathbf{k}}$ denote a set of $\boldsymbol{k}$ that is used in Eq. (8). When $\mathcal{D}_{\mathrm{k}d} \stackrel{\mathrm{def}}{=} \{0, 1, \cdots, N_d\}$ and $\boldsymbol{\mathcal{D}}_{\mathbf{k}}$ is given by the Cartesian product as $\boldsymbol{\mathcal{D}}_{\mathbf{k}} = \mathcal{D}_{\mathrm{k}1} \times \mathcal{D}_{\mathrm{k}2} \times, \cdots, \times \mathcal{D}_{\mathrm{k}d_{\mathrm{s}}}$, the relationship between $\boldsymbol{k}$ and i can be obtained using

$$i = \sum_{d=1}^{d_{\mathrm{s}}} k_d \prod_{d'=d+1}^{d_{\mathrm{s}}} N_{d'}, \tag{9}$$

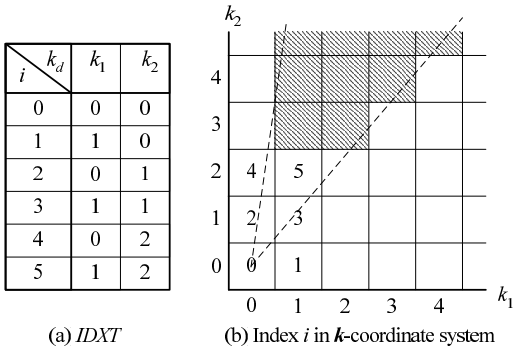
and N is given by

$$N = \prod_{d=1}^{d_{\mathrm{s}}} (N_d + 1) - 1. \tag{10}$$

It is clear from Eq. (10) that N geometrically increases with respect to d_{s}. Thus, if $\boldsymbol{\mathcal{D}}_{\mathbf{k}}$ is given by the Cartesian product of $\mathcal{D}_{\mathrm{k}d}$ and d_{s} is large, N is too large to perform reinforcement learning. If N is small, we may not obtain the required accuracy. This is the curse of dimensionality.

Adaptive basis construction is another solution to this problem. Various methods have been developed for adaptively constructing a basis function, most of which use the radial basis function (RBF) [4] and adjust the parameters of RBF [11]. However, orthonormal bases are superior to non-orthogonal bases such as RBF from the viewpoint of the trade-off between N and the approximation error [12]. Thus, the activity-oriented index allocation method (AIA) [9] is used here; it adaptively constructs an orthonormal basis in accordance with the changes in the environment so that reinforcement learning works well for a given N.

Let us introduce an index table, *IDXT*, defined by an $(N+1) \times d_{\mathrm{s}}$ matrix to express the relationship between i and $\boldsymbol{k}$. An example *IDXT*, when $\boldsymbol{\mathcal{D}}_{\mathbf{k}}$ is given by the Cartesian product, $d_{\mathrm{s}} = 2$, $N_1 = 1$, and $N_2 = 2$, is shown in Fig. 1-a. Consider $k_1, \cdots, k_{d_{\mathrm{s}}}$ to be the coordinates with respect to a rectangular coordinate system,



i \ k_d	k_1	k_2
0	0	0
1	1	0
2	0	1
3	1	1
4	0	2
5	1	2

(a) *IDXT*

(b) Index i in $\boldsymbol{k}$-coordinate system

Fig. 1 Index table *IDXT* and index i in $\boldsymbol{k}$-coordinate system.

which is referred to as the $\boldsymbol{k}$-coordinate system. Index i in Fig. 1-a is expressed in the $\boldsymbol{k}$-coordinate system, as shown in Fig. 1-b.

Although the accuracies of Eqs. (2) and (3) increase as N increases, we cannot use an arbitrarily large value of N. Therefore, when N and $\{K(\boldsymbol{s},\boldsymbol{k})\}$ are given, it is necessary to identify the elements in $\{K(\boldsymbol{s},\boldsymbol{k})\}$ that affect the accuracy. From this viewpoint, the importance of an element in $\{K(\boldsymbol{s},\boldsymbol{k})\}$ is proportional to $\|\boldsymbol{\xi}\|_i$ defined by

$$\|\boldsymbol{\xi}\|_i \stackrel{\mathrm{def}}{=} \sum_{d=1}^{d_{\mathrm{u}}} \xi_{di}^2, \tag{11}$$

and AIA updates *IDXT* so that $\|\boldsymbol{\xi}\|_i$ for $0 \le i \le N$ are as large as possible. Let $\|\boldsymbol{\xi}\|_{i_0}$ be the smallest one, $\|\boldsymbol{\xi}\|_{i_1}$ be the largest one, the index vector corresponding i_0 be $\boldsymbol{k}_0$, and the index vector corresponding i_1 be $\boldsymbol{k}_1$. AIA replaces the elements in the i_0th row of *IDXT*, $\boldsymbol{k}_0$, with $\boldsymbol{k}_{0\mathrm{new}}$. Here, $\boldsymbol{k}_{0\mathrm{new}}$ is the index vector closest to $\boldsymbol{k}_1$ in the search space, which is a conical space spreading in the $\boldsymbol{k}_1$ direction in the $\boldsymbol{k}$-coordinate system. Note that the index vectors that are already in *IDXT* are eliminated from the search space. An example of the search space when $i_1 = 5$ is shown by the shaded area in Fig. 1-b. An example of *IDXT* after updating when $i_0 = 4$ and $i_1 = 5$ is shown in Fig. 2-a. The shaded area in Fig. 2-a is the updated index vector, $\boldsymbol{k}_{0\mathrm{new}}$. Index i_0 in the $\boldsymbol{k}$-coordinate system after updating is shown by the shaded area in Fig. 2-b.

With repeated updating of *IDXT*, basis $\{\phi_i(\boldsymbol{s})\}$ changes so that it contains significant elements in $\{K(\boldsymbol{s},\boldsymbol{k})\}$. Therefore, the accuracies of the function approximations in Eqs. (2) and (3) increase as the critic progresses in the learning.

2.4 Actor-Critic Method with State Space Compression and Adaptive Basis Construction

Because state space compression based on RWPCA and adaptive basis construction based on AIA are effective [8], [9], applying both methods to actor-critic methods should be more effective. An actor-critic method with RWPCA and AIA is presented here. Figure 3

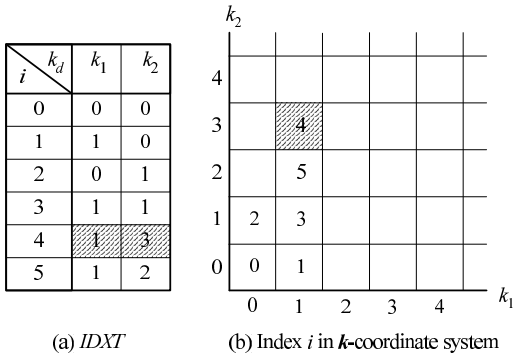


Fig. 2 Index table *IDXT* and index i in $\boldsymbol{k}$-coordinate system after updating.

shows its structure. Although it is strikingly similar to neural networks in its structure, there are many differences.

- The role of each stage is clear. The first stage performs RWPCA (unsupervised learning) using Eq. (7) to compress the state of the environment ($\boldsymbol{x}_t$) to a reward-weighted principal component vector ($\boldsymbol{s}_t$) and to eliminate the adverse effects of disturbance noises. The second stage maps the principal component vector to the feature vector $((\phi_0(\boldsymbol{s}_t), \cdots, \phi_N(\boldsymbol{s}_t))^{\mathrm{t}})$ using basis $\{\phi_i(\boldsymbol{s})\}$ constructed by AIA, as described in Sect. 2.3. The third and fourth stages compose the actor-critic method described in Sect. 2.1. This is like the role sharing between reinforcement learning and unsupervised learning in our brain [13].
- The structure is based on unsupervised learning and reinforcement learning, which can also perform supervised learning. Thus, it can cope with a wider variety of problems than neural networks based on supervised or unsupervised learning.
- The outputs from the second stage are orthogonal and are not redundant while those in neural networks are not orthogonal.
- The three methods with different aims (RWPCA in the first stage, AIA in the the second stage, and the actor-critic method in the third stage) complement each other. Moreover, they work consistently because not only the actor-critic method but also the other methods take the immediate reward into account.

This actor-critic method with RWPCA and AIA was applied to routing problems, and its performance was evaluated (Sect. 4).

3. Nonlinear Approach to Routing Problems

Network routing problems are generally high-dimensional and nonlinear. That is, the dimension of the state is at least the number of nodes and that of the control input is at least the sum of the output links from the nodes. Thus, it is almost impossible to take all the state space and control space into account. As a result,

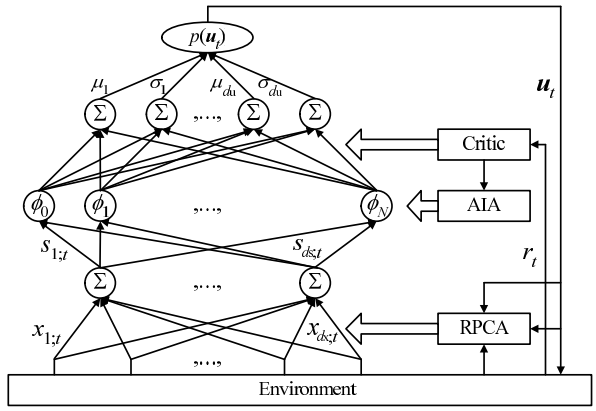


Fig. 3 Structure of actor-critic method with state space compression and adaptive basis construction.

routing algorithms frequently cause routing loops, and the optimum route cannot always be found.

A potential model and a nonlinear approach are thus proposed to solve these problems. The former is used to reduce the control space. The latter is used to apply the potential model and the actor-critic method with RWPCA and AIA, which can reduce the state space, to network routing problems.

3.1 Network Model and Routing Problems

Consider a packet network with a destination node and N_{SN} source nodes. Nodes 1 to N_{SN} are the source nodes, and the $(N_{\mathrm{SN}}+1)$th-node is the destination node. They are distributed in a two-dimensional field like sensor networks [1] or ad hoc networks [2]. $N_{\mathrm{AN}\ell}$ nodes in the neighborhood of the ℓth node, which are referred to as the adjacent nodes of the ℓth node, are connected to the ℓth node. Let $n_{\ell j}$ be the node number of the jth adjacent node of the ℓth node. Each node has a buffer that can store b packets, the link that connects two nodes is two-way, and the channel capacity of each way is C.

To simplify performance evaluation, the following example network is used. The source nodes are arranged in a rectangular lattice at regular intervals in the (X, Y)-plane, where $0 \le X \le X_{\max}$, $0 \le Y \le Y_{\max}$, and the (X, Y)-plane denotes a field in which the source and destination nodes are allocated. Let $(X_{\mathrm{D}}, Y_{\mathrm{D}})$ be the coordinates of the destination node and $(X_{\mathrm{S}\ell}, Y_{\mathrm{S}\ell})$ be the coordinates of the ℓth source node. The destination node is located at $(X_{\mathrm{D}}, Y_{\mathrm{D}}) = (X_{\max}/2, 0)$ and has two links. Thus, the maximum packet arrival rate at the destination node is $2C$. This is a simple model of a sensor network [1]. A network structure with $N_{\mathrm{SN}} = 16$ is shown in Fig. 4. For the performance evaluation in Sect. 4, four regions (S_1, S_2, S_3, and S_4), into which the (X, Y)-plane is divided by the lines $X = X_{\max}/2$ and $Y = Y_{\max}/2$, are introduced, as shown in Fig. 4.

The ℓth source node generates packets at rate λ_ℓ. Let $p_{\ell m}$ be the routing probability $(0 \le p_{\ell m} \le 1)$ from

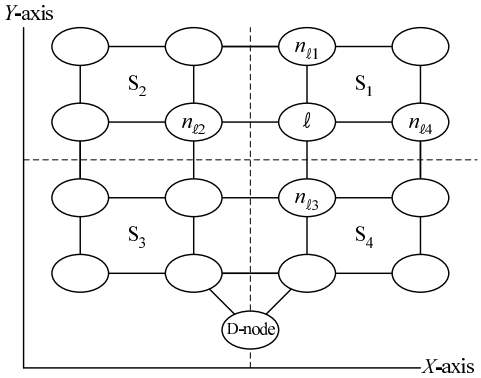


Fig. 4 Example network structure ($N_{\mathrm{SN}} = 16$).

the ℓth to the mth node. Packets generated and received at the ℓth source node are stored once in the buffer and sent to the adjacent nodes on the basis of $p_{\ell m}$. Let q_ℓ be the queue length of the buffer in the ℓth node and e_ℓ be the packet loss rate at the ℓth node. In packet networks, the quality of communication (delay time and error ratio) becomes worse as q_ℓ and e_ℓ increase. Thus, $\sum q_\ell$ and $\sum e_\ell$ should be sufficiently small even if $\sum \lambda_\ell$ is large. However, q_ℓ and e_ℓ increase with $\sum \lambda_\ell$ while they are affected by $p_{\ell m}$. Thus, the aim of the routing problem is to take this trade-off into account and to determine $p_{\ell m}$ so that $\sum q_\ell$ and $\sum e_\ell$ are as small as possible. The environment of the network routing problems considered in this paper is the packet network described above. Let $\lambda_{\ell;t}$, $q_{\ell;t}$, and $e_{\ell;t}$ be the values of λ_ℓ, q_ℓ, and e_ℓ at time t, respectively. Because $\lambda_{\ell;t}$, $q_{\ell;t}$, and $e_{\ell;t}$ are dependent on each other in packet networks, the reward for routing problems is defined using only $e_{\ell;t}$:

$$r_t \stackrel{\mathrm{def}}{=} -\sum_{\ell=1}^{N_{\mathrm{SN}}} e_{\ell;t}^2. \tag{12}$$

3.2 Potential Model for Routing Problems

The dimension of the control space spanned by all the routing probabilities is $\sum_{\ell=1}^{N_{\mathrm{SN}}} N_{\mathrm{AN}\ell}$. Thus, a routing algorithm has to search for a solution in a high-dimensional control space when N_{SN}, the number of nodes in the network, is large. As a result, the solution almost always falls to a local optimum.

The following potential function is presented for reducing the control space and solving this problem.

$$\begin{aligned}\mathrm{pot}(X, Y, X_{\mathrm{B}}, Y_{\mathrm{B}}) = \exp(-c((X-X_{\mathrm{D}})^2 + (Y-Y_{\mathrm{D}})^2)) \\ - \exp(-c((X-X_{\mathrm{B}})^2 + (Y-Y_{\mathrm{B}})^2)),\end{aligned} \tag{13}$$

where c is a constant (set to 0.1 here) and $(X_{\mathrm{B}}, Y_{\mathrm{B}})$ is the coordinate that gives the minimum value of the potential function. The potential function for $(X_{\mathrm{B}}, Y_{\mathrm{B}}) = (1.0, 8.0)$ and $(X_{\mathrm{D}}, Y_{\mathrm{D}}) = (5.0, 0.0)$ is shown in Fig. 5 as an example. Routing probability $p_{\ell m}$ for $m \in \{\ell, n_{\ell 1}, n_{\ell 2}, \cdots, n_{\ell N_{\mathrm{AN}\ell}}\}$ is given by

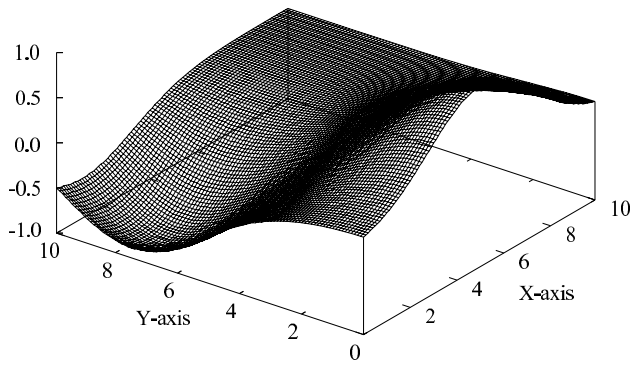


Fig. 5 Example potential function.

$$p_{\ell m} = \begin{cases} \dfrac{\max(\mathrm{pot}_m - \mathrm{pot}_\ell, 0)}{\Delta \mathrm{pot}_\ell} & \text{if } \Delta \mathrm{pot}_\ell > 0 \\ 1 \quad \text{if} \quad \Delta \mathrm{pot}_\ell = 0 \quad \text{and} \quad m = \ell \\ 0 \quad \text{if} \quad \Delta \mathrm{pot}_\ell = 0 \quad \text{and} \quad m \neq \ell, \end{cases} \tag{14}$$

where $\Delta \mathrm{pot}_\ell \stackrel{\mathrm{def}}{=} \sum_{j=1}^{N_{\mathrm{AN}\ell}} \max(pot_{n_{\ell j}} - pot_\ell, 0)$ and $\mathrm{pot}_\ell \stackrel{\mathrm{def}}{=} \mathrm{pot}(X_{\mathrm{S}\ell}, Y_{\mathrm{S}\ell}, X_{\mathrm{B}}, Y_{\mathrm{B}})$.

The potential function denotes relative routing probabilities. This means that packets are transmitted from a node with a lower potential to one with a higher potential. On the basis of the potential function in Eq. (5), the routing probabilities $p_{\ell m}$ for all nodes are determined by setting only two variables $(X_{\mathrm{B}}, Y_{\mathrm{B}})$. Thus, by applying Eqs. (13) and (14) to routing problems and setting control input $\boldsymbol{u}$ to $(X_{\mathrm{B}}, Y_{\mathrm{B}})^{\mathrm{t}}$, we can reduce the dimension of the control space from $\sum_{\ell=1}^{N_{\mathrm{SN}}} N_{\mathrm{AN}\ell}$ to 2.

3.3 Nonlinear Approach to Robust Routing

Consider a time-invariant environment in which $\lambda_{\ell;t}$ for $\forall \ell$ is a constant. In this environment, $q_{\ell;t}$ and $e_{\ell;t}$ converge to constants if the routing probabilities are fixed. Thus, the routing problem is reduced to the problem of maximizing $\lim_{t\to\infty} r_t$ with respect to the routing matrix:

$$\max_P \lim_{t\to\infty} r_t, \tag{15}$$

where P is the $N_{\mathrm{SN}} \times (N_{\mathrm{SN}} + 1)$-routing matrix with elements $p_{\ell m}$.

Next let us consider an environment in which the $\lambda_{\ell;t}$ vary with time. Let $\boldsymbol{x}_t$ be a state variable comprised of some or all of $\lambda_{\ell;t}$, $q_{\ell;t}$, and $e_{\ell;t}$, and let the routing matrix be the control input. The dynamic routing problem is thus reduced to a nonlinear control problem:

$$\begin{cases} \max\limits_{P_t} R_t \\ \boldsymbol{x}_{t+1} = f_{\mathrm{net}}(\boldsymbol{x}_t, P_t), \end{cases} \tag{16}$$

where R_t is the discount return defined by Eq. (2), P_t is the routing matrix at time t, and $f_{\mathrm{net}}(\cdot)$ denotes the

environment.

The dimension of the control space in Eqs. (15) and (16) is $\sum_{\ell=1}^{N_{\rm SN}} N_{{\rm AN}\ell}$, which is too large to obtain the optimum route, as mentioned above. By applying Eq. (14) to Eqs. (15) and (16) and setting control input $\boldsymbol{u}$ to $(X_{\rm B}, Y_{\rm B})^{\rm t}$, we obtain the following equations:

$$\max_{\boldsymbol{u}} \lim_{t\to\infty} r_t, \tag{17}$$

$$\begin{cases} \max_{\boldsymbol{u}_t} R_t \\ \boldsymbol{x}_{t+1} = f_{\rm net}(\boldsymbol{x}_t, \boldsymbol{u}_t). \end{cases} \tag{18}$$

Using these equations, we can search for the optimum route in the two-dimensional control space. Although the dimension of the state space in these equations remains high, we can use the actor-critic method with RWPCA and AIA, which works well in a high-dimensional state space, to solve them. The performance of the routing algorithms with the potential model and the actor-critic method with RWPCA and AIA is presented in the next section.

4. Performance Evaluation

4.1 Static Routing for Time-Invariant Environment

Three routing strategies for a time-invariant environment were examined using the network model described in Sect. 3.1 to evaluate the effect of the potential model presented in Sect. 3.2.

- Steepest descent method for routing matrix: In a time-invariant environment, it is not necessary to use reinforcement learning because the problem can be solved, as shown in Eq. (15), by searching for the value of the routing matrix that provides the maximum value of $\lim_{t\to\infty} r_t$. Thus, a simple nonlinear programming technique can be applied to the problem. In this paper, the steepest descent method is used because it is used in the actor-critic method to update ξ_{di} and η_{di} in Eqs. (2) and (3).
- Steepest descent method for potential model: Equation(17), which is based on the potential model described in Sect. 3.2, is solved using the steepest descent method.
- Thorough search method for routing matrix: For $p_{\ell m} \in \{0, 1\}$, the optimum solution for Eq. (15) is obtained by conducting a thorough search for all combinations of $p_{\ell m}$.

Let $\lambda_{\rm all} \stackrel{\rm def}{=} \sum_{\ell=1}^{N_{\rm SN}} \lambda_\ell$ be the total packet generation rate and λ_ℓ be constant. Set λ_ℓ of the source nodes in $\rm S_1$ and $\rm S_4$ to $1.9 \times \lambda_{\rm all}/N_{\rm SN}$, λ_ℓ of the source nodes in $\rm S_2$ and $\rm S_3$ to $0.1 \times \lambda_{\rm all}/N_{\rm SN}$, $\lambda_{\rm all}$ to 3.6, C to 2, and b to 32. If $p_{\ell m}$ are appropriately determined, it is clear that most packets generated at the nodes in $\rm S_1$ are transmitted through $\rm S_2$ and $\rm S_3$ and that the packets generated at the nodes in $\rm S_4$ are transmitted to the destination node without passing through other regions. It is also clear that all packets arrive at the destination node without any loss because $\sum_{\ell=1}^{N_{\rm SN}} \lambda_\ell = \lambda_{\rm all} = 3.6 \le 2C$.

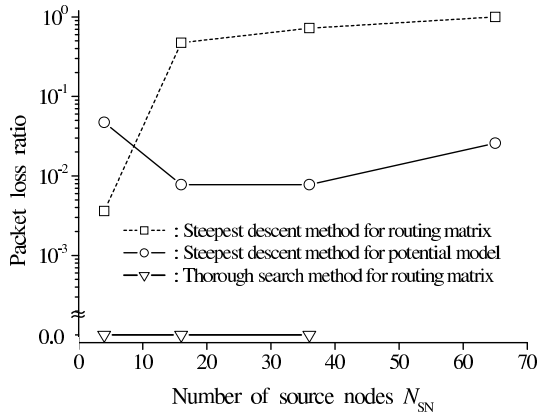


Fig. 6 Effect of potential model on static routing.

The packet loss ratios obtained using the three strategies for a time-invariant environment are shown in Fig. 6. The thorough search method provided the optimum solution although it worked only for $N_{\rm SN} \le 36$ because of the calculation cost. The steepest descent method for the routing matrix worked well (i.e., the packet loss ratio was sufficiently small) for $N_{\rm SN} = 4$, but the solutions for $N_{\rm SN} \ge 16$ were not optimum. This is because the dimension of the control space is too high when $N_{\rm SN}$ is large, so the method causes routing loops and falls to a local optimum. While the packet loss ratio of the steepest descent method for the potential model was not optimum for $N_{\rm SN} = 4$, those for $N_{\rm SN} \ge 16$ were sufficiently small. This is because the dimension of its solution control space remains two independent of $N_{\rm SN}$. Thus, the potential model is effective for solving the routing problem except when $N_{\rm SN}$ is small.

4.2 Dynamic Routing for Time-Varying Environment

Consider a time-varying environment in which two regions are randomly selected from $\rm S_1$, $\rm S_2$, $\rm S_3$, and $\rm S_4$ at periodic intervals of T_λ. The $\lambda_{\ell;t}$ of the nodes in the selected regions are set to $1.9 \times \lambda_{\rm all}/N_{\rm SN}$, the $\lambda_{\ell;t}$ of the nodes in the other two regions are set to $0.1 \times \lambda_{\rm all}/N_{\rm SN}$, $\lambda_{\rm all}$ is set to 3.6, C to 2, b to 32, and T_λ to 100. Although $\lambda_{\ell;t}$ varies at intervals of T_λ, P_t can be determined without any packet loss because $\sum_{\ell=1}^{N_{\rm SN}} \lambda_{\ell;t} = \lambda_{\rm all} = 3.6 \le 2C$ in the same manner as in the time-invariant environment. To evaluate the robustness of routing algorithms, a disturbance noise was introduced at a node adjacent to the destination node. The noise was a random variable from 0 to buffer length b and changed with period $T_{\rm noise}$. It prevented the other nodes from observing the queue length of the adjacent node.

Because the potential model is effective, as shown in Sect. 4.1, the effect of RWPCA and that of AIA were evaluated using the following three routing strate-

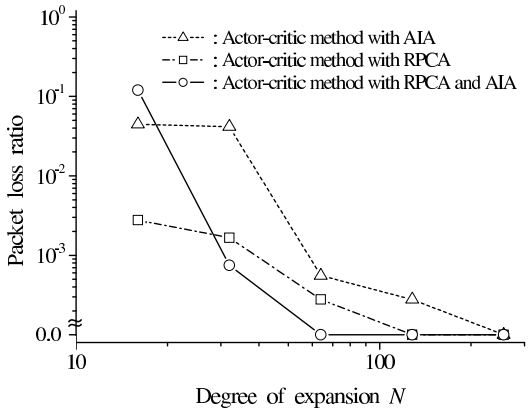


Fig. 7 Effect of RWPCA and AIA on packet loss ratio ($N_{\mathrm{SN}} = 16$, $T_{\mathrm{noise}} = 8$).

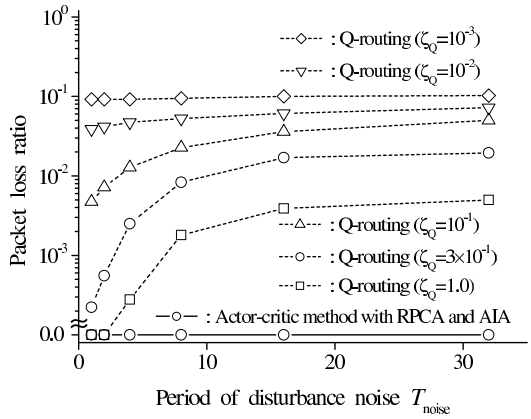


Fig. 8 Effect of disturbance noise on packet loss ratio ($N_{\mathrm{SN}} = 16$, $N = 256$).

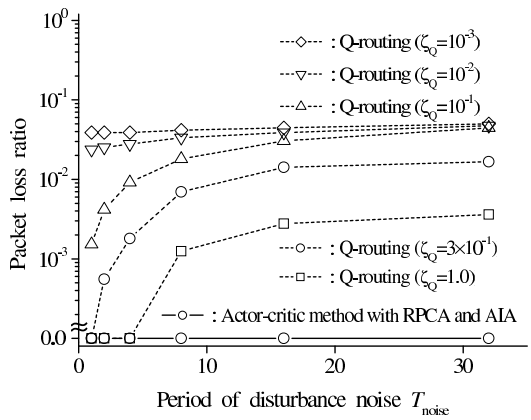


Fig. 9 Effect of disturbance noise on packet loss ratio for different traffic pattern ($N_{\mathrm{SN}} = 16$, $N = 256$).

gies based on the actor-critic method. They observe $\boldsymbol{x}_t \stackrel{\mathrm{def}}{=} (\lambda_{1;t}, q_{1;t}, \cdots, \lambda_{N_{\mathrm{SN}};t}, q_{N_{\mathrm{SN}};t})^{\mathrm{t}}$, and routing problems are solved using Eq. (18), which is based on the potential model. Routing matrix P_t is determined by Eq. (14). Q-routing [5] was also evaluated to isolate the effect of RWPCA and AIA.

- Actor-critic method with RWPCA and AIA: As shown in Sect. 2.4, RWPCA transforms $\boldsymbol{x}_t$ to $\boldsymbol{s}_t$ using Eq. (7) and basis $\{\phi_i(\boldsymbol{s})\}$ is constructed by AIA. Control input $\boldsymbol{u}$ is determined using the actor-critic method.
- Actor-critic method with RWPCA [8]: This strategy does not use AIA. Basis $\{\phi_i(\boldsymbol{s})\}$ is constructed on the basis of $\boldsymbol{\mathcal{D}}_{\mathbf{k}}$ given by the Cartesian product as $\boldsymbol{\mathcal{D}}_{\mathbf{k}} = \mathcal{D}_{\mathrm{k}1} \times \mathcal{D}_{\mathrm{k}2} \times, \cdots, \times \mathcal{D}_{\mathrm{k}d_{\mathrm{s}}}$, and N_d is determined on the basis of the standard deviation in $\boldsymbol{s}_t$.
- Actor-critic method with AIA [9]: This strategy does not use RWPCA and thus sets $\boldsymbol{x}_t$ to $\boldsymbol{s}_t$.
- Q-routing: Q-routing [5] is the most well-known routing method based on reinforcement learning. It assigns a Q-value to each adjacent node on the basis of Q-learning [4] and selects the route using the Q-value. The Q-value of each node is updated with step size ζ_{Q} on the basis of the queue length of the adjacent nodes.

The three routing algorithms based on the actor-critic method were evaluated for $N_{\mathrm{SN}} = 16$, $T_{\mathrm{noise}} = 8$, and various values of N in an environment with disturbance noise. As shown in Fig. 7, the packet loss ratios obtained were sufficiently small when N was 256, and those obtained by the actor-critic method with RWPCA and AIA were the smallest except when N was 16. These results show that the adverse effect of disturbance noise can be effectively eliminated by using both RWPCA and AIA except when $N = 16$ compared with using only RWPCA or AIA. This is because the states of all the nodes are taken into account, RWPCA and AIA complement each other, and they work consistently by taking the immediate reward into account.

In an environment with disturbance noise, the packet loss ratio was evaluated for $N_{\mathrm{SN}} = 16$, $N = 256$, and various values of T_{noise} to investigate the robustness of Q-routing and the effectiveness of the actor-critic method with RWPCA and AIA. Figure 8 shows that the packet loss ratios obtained by the actor-critic method with RWPCA and AIA were sufficiently small regardless of T_{noise} although the packet loss ratios of Q-routing increased as T_{noise} increased. Figure 9 shows the packet loss ratios evaluated for a traffic pattern different than that used for Fig. 8: $N_{\mathrm{SN}}/2$ source nodes were randomly selected, their $\lambda_{\ell;t}$ were set to $1.9 \times \lambda_{\mathrm{all}}/N_{\mathrm{SN}}$, and the $\lambda_{\ell;t}$ of the other source nodes were set to $0.1 \times \lambda_{\mathrm{all}}/N_{\mathrm{SN}}$. Figure 9 shows that the actor-critic method with RWPCA and AIA was robust in the same manner as in Fig. 8. The reason Q-routing was affected by the disturbance noise is that Q-routing directly uses the queue length, which was disturbed by the noise. In contrast, the actor-critic method with RWPCA and AIA eliminates the effect of noise from the state by using RWPCA and AIA. We can thus conclude that the actor-critic method with RWPCA and AIA is robust and able to cope with various traffic patterns.

Figure 10 shows the packet loss ratio for $N = 256$ and various values of N_{SN} when a disturbance noise was not introduced. The packet loss ratios were sufficiently

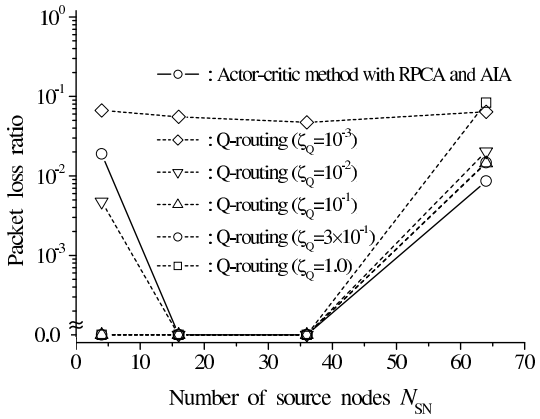


Fig. 10 Effect of number of source nodes on packet loss ratio ($N = 256$).

small when $16 \leq N_{\mathrm{SN}} \leq 36$ except for Q-routing with $\zeta_{\mathrm{Q}} = 10^{-3}$. However, when N_{SN} was large, the packet loss ratios of both algorithms were higher; that is, it is difficult to keep the packet loss ratio small when N_{SN} is large, apparently because the packets are stored in more nodes when N_{SN} is large, making it more difficult to control them. Although the packet loss ratio can be reduced by increasing buffer length b of each node, this is problematic because it causes a longer delay time.

The potential function in Eq. (13) is very simple and has only one maximum value and one minimum value. This is because it was made for a network that has only one destination node and packet traffic that is not complicated. Thus, it cannot express the complicated packet traffic that arises in a network when N_{SN} is large. This problem could be solved by improving the potential function, which remains for a future study.

5. Conclusion

Routing problems were formulated as nonlinear control problems, and the high-dimensional control space of the problem was reduced to a two-dimensional control space by using newly developed control space compression based on a potential model. The state space of the problem was compressed using reward-weighted principal component analysis, and a basis for approximating a control function was adaptively constructed using activity-oriented index allocation. A routing algorithm based on these methods and the actor-critic method can efficiently search for an approximate optimum route even if the routing information is disturbed.

References

[1] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," Ad Hoc Networks Journal, vol. 1, pp. 293–315, Sept. 2003.

[2] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," Proc. 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking, pp. 85–97, 1998.

[3] C. Huitema, Routing in the Internet, Prentice Hall, USA, 1995.

[4] R. S. Sutton and A. G. Barto, Reinforcement Learning, MIT Press, USA, 1998.

[5] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: a reinforcement learning approach," Advances in Neural Information Processing Systems, vol. 6, pp. 671–678, 1993.

[6] S. Kumar and R. Miikkulainen, "Confidence based dual reinforcement Q-routing: an adaptive on-line routing algorithm," Proc. 16th International Joint Conference on Artificial Intelligence, pp. 231–238, 1999.

[7] Y. Zhang, J. Liu, and F. Zhao, "Information-directed routing in sensor networks using real-time reinforcement learning," Combinatorial Optimization in Communication Networks, Chapter 10, Springer, pp. 259-288, 2006.

[8] H. Satoh, "A state space compression method based on multivariate analysis for reinforcement learning in high-dimensional continuous state spaces," IEICE Trans. Fundamentals, vol. E89-A, no. 8, pp. 2181–2191, Aug. 2006.

[9] H. Satoh, "Feature space construction and function approximation for reinforcement learning," Tech. Rept. IEICE NLP2006-41, pp. 43–48, July 2006.

[10] R. A. Johnson and D. W. Wichern, Applied Multivariate Statistical Analysis, 5th ed, Pearson Education, Prentice Hall, USA, 2001.

[11] I. Menache, S. Mannor, and N.Shimkin, "Basis function adaptation in temporal difference reinforcement learning," Annals of Operations Research, vol. 134, no. 1, pp. 215–238, Feb. 2005.

[12] H. Satoh, "Reinforcement learning for continuous stochastic actions – An approximation of probability density function by orthogonal wave function expansion –," IEICE Trans. Fundamentals, vol. E89-A, no. 8, pp. 2173–2180, Aug. 2006.

[13] K. Doya, "Complementary roles of basal ganglia and cerebellum in learning and motor control," Current Opinion in Neurobiology, vol. 10, no. 6, pp. 732–739, Dec. 2000.