PAPER

Autonomous Decentralized Control for Indirectly Controlling System Performance Variable of Large-Scale and Wide-Area Networks

Yusuke SAKUMOTO[†a)], **Masaki AIDA**[†b)], ***and*** **Hideyuki SHIMONISHI**[††c)], ***Members***

SUMMARY In this paper, we propose a novel Autonomous Decentralized Control (ADC) scheme for indirectly controlling a system performance variable of large-scale and wide-area networks. In a large-scale and wide-area network, since it is impractical for any one node to gather full information of the entire network, network control must be realized by inter-node collaboration using information local to each node. Several critical network problems (e.g., resource allocation) are often formulated by a system performance variable that is an amount to quantify system state. We solve such problems by designing an autonomous node action that indirectly controls, via the Markov Chain Monte Carlo method, the probability distribution of a system performance variable by using only local information. Analyses based on statistical mechanics confirm the effectiveness of the proposed node action. Moreover, the proposal is used to implement traffic-aware virtual machine placement control with load balancing in a data center network. Simulations confirm that it can control the system performance variable and is robust against system fluctuations. A comparison against a centralized control scheme verifies the superiority of the proposal.

key words: *large-scale and wide-area network, autonomous-decentralized mechanism, data center network, virtual machine placement problem*

1. Introduction

Networks have become one of society's basic infrastructures and so require very high reliability. To ensure high network reliability, the usage of centralized network controls should be minimized because it is weak against disasters. As an alternative to centralized control, Autonomous Decentralized Control (ADC) is being actively discussed in [1]–[4]. In ADC, each node has the same capability to collaborate with other nodes for controlling the entire network. Hence, if some nodes suddenly fail, the network keeps running. ADC has higher reliability than centralized control, and thus expectations for its use in future networks are high.

One of the challenges in implementing an ADC scheme is to design a node action that allows us to indirectly control large-scale and wide-area networks (e.g., the data center network of Google [5]). In ADC, each node gathers local information of the network and takes action to control its state. In a large-scale and wide-area network, the information gathering by a node is limited in the area immediately surrounding the node [3]. This is because the time spent on gathering the information must be short since the information changes frequently in a large-scale and wide-area network. Hence, in a large-scale and wide network, it is impractical for any one node to gather full information of the entire network. The problem is how to design a truly effective node action using only local information.

In [2], [3], an ADC node action using only local information was proposed for spatially structuring large-scale and wide-area networks. The resulting spatial structure can be used for network clustering and layering. However, those studies did not address other important problems (e.g., resource allocation in networks), which are often formulated by a system performance variable (e.g., the total throughput of a network that is an amount to quantify system state including all node states. This paper remedies that omission.

How can a node action that uses only local information indirectly control a system performance variable? Given the ability to acquire the entire system state, we could deterministically optimize the desired system performance variable with an optimization tool. However, given the impracticality of acquiring sufficient state information, such optimization is deemed to be impossible. Hence, each node should behave stochastically to the limited information so as to improve system performance. The system performance variable also stochastically fluctuates as a result of the stochastic behavior of nodes, and so each node should indirectly control the frequency distribution of the system performance variable by using its local information.

In a statistical system, each node (e.g., particle) behaves stochastically, and state quantities (e.g., energy) follow a probability distribution. By using a node action designed by Markov Chain Monte Carlo (MCMC) [6], [7], the probability distribution of a state quantity can be indirectly controlled. Thus MCMC should be useful for indirectly controlling a system performance variable of even extremely large networks.

We use MCMC to design an ADC node action for indirectly controlling a system performance variable in a large-scale and wide-area network. It is usually thought to be impossible to analyze the global properties associated with the control of a large-scale and wide-area network. Our solution is to conduct analyses based on statistical mechanics. We show that our ADC node action can (a) indirectly control

Manuscript received June 17, 2015.

[†]The authors are with Tokyo Metropolitan University, Hino-shi, 191-0065 Japan.
[††]The author is with the NEC Corporation, Kawasaki-shi, 211-8666 Japan.
a) E-mail: sakumoto@tmu.ac.jp
b) E-mail: aida@tmu.ac.jp
c) E-mail: h-shimonishi@cd.jp.nec.com

DOI: 10.1587/transcom.E98.B.2248

a system performance variable, and (b) adjust the strengths of two different controls (i.e., node concentration and node distribution) with a single control parameter. We introduced a MCMC-based ADC scheme in [8]. However, no use was made of the concept of statistical mechanics, and the global characteristics of the proposal were not sufficiently analyzed in [8]. Note that a preliminary version of this paper was presented in [9]. The main innovation of this paper over [9] lies in its advanced investigation of (a) robustness against system fluctuations, and (b) its effectiveness compared with centralized control.

We apply our ADC proposal to realize traffic-aware virtual machine (VM) placement control with load balancing in a data center network (DCN). In [8], we also described a similar VM placement function, but the global property of our ADC proposal was not utilized. Thus the control functionality described in this paper is more sophisticated than that in [8]. Simulation experiments confirm the validity of our analyses based on statistical mechanics, and the effectiveness of our ADC proposal.

This paper is organized as follows. Section 2 details our ADC proposal for indirectly controlling a system performance variable, and explains its global property. Section 3 uses our ADC proposal to realize traffic-aware VM placement control with load balancing in a DCN. Section 4 details the experiments conducted for investigating the performance of our ADC proposal. Finally, in Sect. 6, we conclude this paper and discuss future works.

2. Autonomous Decentralized Control for Indirectly Controlling a System Performance Variable

2.1 System Model

We introduce a system model that includes control of a system performance variable depending on system state. This system model has N_S states and N nodes. Node i's state is denoted by $x_i \in \{1, ..., N_S\}$. Each node is able to select one from common selections among all nodes, so N_S is a common variable. System state $\boldsymbol{X}$ is given by all node states $(x_1, x_2, ..., x_N)$. The set of nodes included in $\boldsymbol{X}$ is time invariant. Let Ω be the space of system state $\boldsymbol{X}$. Let ϕ_k be the set of nodes with state k (i.e., $\phi_k = \{i | x_i = k, 1 \le i \le N\}$). Permissible node state transitions are described by a given state transition graph. Let a_k be the set of states to which nodes with state k can transit on the state transition graph. a_k is time invariant. Figure 1 shows an example of the system model with $N_S = 4$. For instance, a node with state 1 can transit to state 2 or 3 in this example.

Let $M(\boldsymbol{X})$ be the system performance variable indirectly controlled by our ADC proposal. We define $M(\boldsymbol{X})$ as

$$M(\boldsymbol{X}) = \sum_{i=1}^{N} \sum_{x_j \in \chi_i} m_{i\,j}(x_i, x_j), \tag{1}$$

where χ_i is the set of nodes interacting with node i in the network, and $m_{i\,j}(x_i, x_j)$ is the performance variable depending

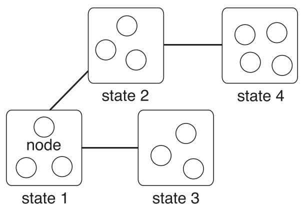


Fig. 1 An example of the system model.

on node states x_i and x_j with conditions $m_{i\,i}(x_i, x_i) = 0$ and $m_{i\,j}(x_i, x_j) = m_{j\,i}(x_j, x_i)$. The formulation given by $M(\boldsymbol{X})$ is used in finding optimal weighted configuration in a network [10]. In this paper, we define that smaller $m_{i\,j}(x_i, x_j)$ is better. We believe our ADC proposal can be extended to handle performance variables depending on more than three states, but this is not addressed hereafter. Our ADC proposal targets the problem formulated as system performance variable M given by Eq. (1).

2.2 Node Action

We design a node action of our ADC proposal that allows indirect control of the probability distribution of system performance variable $M(\boldsymbol{X})$ on the basis of MCMC. Since each node has only local information, it should behave stochastically to the limited information. Hence, we derive the probability of a node state transition under stochastic behavior. We first consider the condition determining the probability of system state transition $\boldsymbol{X} \to \boldsymbol{X}'$ on the basis of MCMC, and then derive the probability of node i's state transition $x_i \to x_i'$ according to the condition.

According to MCMC, $\boldsymbol{X}$ in a Markov chain generated by state transition probability $P(\boldsymbol{X}'|\boldsymbol{X})$ follows a stationary distribution $P(\boldsymbol{X})$ if $P(\boldsymbol{X}'|\boldsymbol{X})$ satisfies the condition

$$P(\boldsymbol{X}'|\boldsymbol{X})\,P(\boldsymbol{X}) = P(\boldsymbol{X}|\boldsymbol{X}')\,P(\boldsymbol{X}'), \tag{2}$$

and the ergodic condition, that is, the probabilities of Markov chains with arbitrary length more than a certain value moving from one system state to any of the other system states are larger than 0.

Next, we derive probability $T_i(x_i \to x_i')$ of node i's state transition $x_i \to x_i'$ according to the condition (2). We give $\boldsymbol{X}' = (x_1', ..., x_N')$ where $x_i \neq x_i'$ and $x_j = x_j'$, and the stationary distribution $P(\boldsymbol{X})$ by exponential form (i.e., $P(\boldsymbol{X}) = Ae^{-\lambda\,M(\boldsymbol{X})}$ where $\lambda > 0$). By using $P(\boldsymbol{X}) = Ae^{-\lambda\,M(\boldsymbol{X})}$, Eq. (2) is rewritten by

$$\begin{aligned}
\frac{P(\boldsymbol{X}'|\boldsymbol{X})}{P(\boldsymbol{X}|\boldsymbol{X}')} &= e^{-\lambda\,(M(\boldsymbol{X}')-M(\boldsymbol{X}))} \\
\frac{T_i(x_i \to x_i')}{T_i(x_i' \to x_i)} &= e^{-\lambda\,\sum_{j\in\chi_i}(m_{i\,j}(x_i',x_j)-m_{i\,j}(x_i,x_j))} \\
&= e^{-\lambda\,\Delta M_i(x_i\to x_i')} \\
&= \frac{e^{-\alpha\,\lambda\,\Delta M_i(x_i\to x_i')}}{e^{(1-\alpha)\,\lambda\,\Delta M_i(x_i\to x_i')}} \\
&= \frac{\frac{1}{|a_{x_i}|}e^{-\alpha\,\lambda\,\Delta M_i(x_i\to x_i')}}{\frac{1}{|a_{x_i}|}e^{-(1-\alpha)\,\lambda\,\Delta M_i(x_i'\to x_i)}}.
\end{aligned} \tag{3}$$

In the process of deriving Eq. (3), we conventionally replace 2λ with λ, introduce parameter α $(0 < \alpha < 0.5)$, and use the relation $\Delta M_i(x_i \to x'_i) = -\Delta M_i(x'_i \to x_i)$.

We finally obtain node state transition probability $T_i(x_i \to x'_i)$, as follows

$$T_i(x_i \to x'_i) = \begin{cases} \dfrac{1}{|a_{x_i}|} e^{-\alpha \lambda \Delta M_i(x_i \to x'_i)} \\ \qquad \text{if } \Delta M_i(x_i \to x'_i) < 0 \\ \dfrac{1}{|a_{x_i}|} e^{-(1-\alpha)\lambda \Delta M_i(x_i \to x'_i)} \\ \qquad\qquad \text{otherwise} \end{cases}. \quad (4)$$

According to Eq. (4), as λ approaches 0, our ADC proposal more closely resembles simple random control with $T_i(x_i \to x'_i) = 1/|a_{x_i}|$. We assume that each node uses a common value among all nodes as the value of λ. By using Eq. (4), each node can change its state autonomously because $T_i(x_i \to x'_i)$ depends on $\Delta M_i(x_i \to x'_i)$ instead of M. To derive such a node state transition probability on the basis of MCMC, the following requirements are needed; (a) M is defined by linear sum of m_{ij}'s, and (b) stationary distribution $P(\boldsymbol{X})$ in designing by MCMC is given by the exponential form.

$T_i(x_i \to x'_i)$ for $\alpha = 0$ is the same as the state transition probability used in the M-H algorithm [7]. The reason why we use $\alpha > 0$ is simple; in a large-scale network, information used in a control is frequently changed with the external environment of the network. If we use $\alpha = 0$ for controlling a network, needless state transitions would occur so we set $\alpha > 0$. Although $T_i(x_i \to x'_i)$ must have the ergodic condition to control the probability distribution, this is achieved if all states are connected in the state transition graph.

To use our ADC proposal, nodes have to fulfill the following restrictions; (a) nodes can gather local information to calculate $\Delta M_i(x_i \to x'_i)$, (b) nodes can calculate Eq. (4), and (c) nodes can change its state according to probability $T_i(x_i \to x'_i)$ given by Eq. (4).

2.3 Global Property

2.3.1 Controllability of Probability Distribution of System Performance Variable

To show that the designed node action can indirectly control the probability distribution of system performance variable M, we derive probability distribution $P(M)$ of system performance variable M with node state transition probability $T_i(x_i \to x'_i)$. To obtain $T_i(x_i \to x'_i)$ in the previous section, we give $P(\boldsymbol{X})$ with the exponential form $Ae^{-\lambda M(\boldsymbol{X})}$. With this form, system states with the same value of system performance variable M occur with the same probability, and cannot be distinguished. Hence, by summing probabilities $P(X)$ with the same $M(\boldsymbol{X})$, $P(M)$ is given by

$$P(M) = \frac{G(M)\, e^{-\lambda M}}{\sum_{Y \in \Omega_M} G(Y)\, e^{-\lambda Y}} = \frac{1}{Z} G(M)\, e^{-\lambda M}, \quad (5)$$

where $G(M)$ is the number of system states with system performance variable M; it is called *system state density distribution*. Z is the normalization constant for a given λ, and Ω_M is the set of all possible M. Z represents a kind of moment-generating function of M. According to Eq. (5), we can confirm that the node action can indirectly control the probability distribution of performance system variable M. If $\lambda = 0$, $P(M)$ is simply proportional to $G(M)$. As λ increases, $P(M)$ is shifted by $e^{-\lambda M}$. Therefore, the node action can indirectly control $P(M)$ for a given λ.

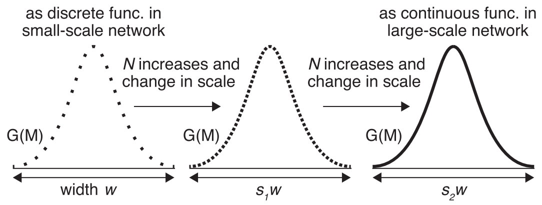
The indirect control of M by our ADC node action is achieved when the system state must become steady state where M follows Boltzmann distribution. The time required for becoming the steady state is determined by (a) the execution count of the node action required for becoming the steady state, and (b) the computation time of the node action. Since the execution count depends on several factors (e.g., N, $G(M)$, and λ), we do not estimate its exact value. However, the execution count is not unrealistically too large. This is explained as follows. The steady state is transited slowly, so the execution count to adapt to next steady state would be small. The computation time of the node action highly depends on the computational complexity required for calculating Eq. (4). In the node action, each node calculates Eq. (4), $|a_k|$ times. Since the computational complexity of Eq. (4) is given by the order of $|\chi_i|$, the computational complexity of the node action is given by $|a_k|\,|\chi_i|$. Since actual networks are sparse, $|\chi_i|$ is too small if the network is large scale. We can set $|a_k|$ to small value while a_k fulfills the condition that all states are connected in the state transition graph. Hence, the computational complexity of the node action is not too unrealistically large, so the node action has high scalability for N.

In statistical mechanics, the probability distribution given by Eq. (5) is called *Boltzmann distribution*, which is a well-understood distribution. Statistical mechanics is often used to analyze the global properties of a thermodynamic system around the most frequent point M^* of Boltzmann distribution. Such an analysis is valid if N is sufficiently large because the property of states around M^* becomes more dominant as N increases, see Sect. 2.3.2. In what follows, through analysis based on statistical mechanics, we will clarify the global property of our ADC proposal in a large-scale network.

We first introduce $F(M) := M - 1/\lambda \log G(M)$, which is called *Helmholtz free energy* in statistical mechanics. By substituting $F(M)$ into Eq. (5), $P(M)$ is deformed by

$$\begin{aligned} P(M) &= \frac{1}{Z} e^{\log G(M) - \lambda M} = \frac{1}{Z} e^{-\lambda (M - \frac{1}{\lambda} \log G(M))} \\ &= \frac{1}{Z} e^{-\lambda F(M)}. \end{aligned} \quad (6)$$

$F(M)$ is an important state quantity for understanding a thermodynamic system because the most frequent point, M^*, of Boltzmann distribution can be derived by solving $dF(M)/dM = 0$ where it is assumed that $G(M)$ can be modeled as a function of class C^2. This assumption would be



as discrete func. in small-scale network
N increases and change in scale
as continuous func. in large-scale network
N increases and change in scale
G(M)
G(M)
G(M)
width w
s1w
s2w

Fig. 2 Convergence of $G(M)$ from a discrete function to a continuous function ($1 \ll s_1 \ll s_2$).

valid in a large-scale system because discrete intervals between pairs of consecutive M in Ω_M are very smaller than the width of $P(M)$, see Fig. 2. By solving $dF(M)/dM = 0$, M^* satisfies the condition

$$\frac{\partial}{\partial M} \log G(M)\Big|_{M=M^*} = \lambda. \tag{7}$$

To analyze the global property of an ADC scheme around M^*, we convert $F(M)$ to a Taylor series at M^*. The Taylor series of $F(M)$ at M^* is given by

$$F(M) = F(M^*) - \sum_{k=2}^{\infty} \frac{1}{k!\,\lambda} \frac{d^k}{dM^k} \log G(M)\Big|_{M=M^*} \epsilon_M^k, \tag{8}$$

where $\epsilon_M := M - M^*$. Because M is generally a monotonically increasing function of N, the derivatives of $\log G(M)$ for $k \geq 3$ decrease more quickly than that for $k = 2$ as N increases. Hence, in a large-scale network, $F(M)$ can be approximated by

$$F(M) \simeq F(M^*) - \frac{1}{2\,\lambda} \frac{d^2}{dM^2} \log G(M)\Big|_{M=M^*} \epsilon_M^2. \tag{9}$$

Let μ and σ be the average and the standard variance of M, respectively. By substituting Eq. (9) into Z, μ and σ are given by

$$\mu = -\frac{d}{d\lambda} \log Z \simeq M^*, \tag{10}$$

$$\sigma^2 = \frac{d^2}{d\lambda^2} \log Z \simeq -\frac{dM^*}{d\lambda} = C_M. \tag{11}$$

By substituting Eq. (9) into $G(M) = e^{\lambda(M-F(M))}$, $G(M)$ is approximately given by

$$G(M) \simeq \frac{|\hat{\Omega}|}{\sqrt{2\pi C_M}} e^{-\frac{(M-M^*-\lambda C_M)^2}{2C_M}}, \tag{12}$$

where $|\hat{\Omega}| = \sum_{M\in\Omega_M} G(M) = Ze^{\lambda(M^*+\lambda C_M/2)}$. $\hat{\Omega}$ is the subset of Ω. If λ is large, a Markov chain of M is limited in the subset of system state space Ω because state transitions with $\Delta M_i > 0$ rarely occur by our ADC proposal. Hence, as λ increases, Ω shrinks to $\hat{\Omega}$. Since $|\hat{\Omega}|$ decreases as λ increases, Z should exponentially decrease against the exponential increase in $e^{\lambda(M^*+\lambda C_M/2)}$.

We denote the average and the standard variance of M at $\lambda = 0$ by μ_0 and σ_0, respectively. Since $P(M) \propto G(M)$ when $\lambda = 0$, μ_0 and σ_0 are given by

$$\mu_0 = \sum_{M\in\Omega_M} M \frac{G(M)}{|\hat{\Omega}|} \simeq M^* + \lambda C_M, \tag{13}$$

$$\sigma_0^2 = \sum_{M\in\Omega_M} \left(M^2 \frac{G(M)}{|\hat{\Omega}|} \right) - \mu_0^2 \simeq C_M. \tag{14}$$

Equation (13) also shows that the our ADC proposal can control system performance variable M by parameter λ of the node action since $M^* = \mu_0 - \lambda C_M$ and $C_M > 0$. Our analyses based on statistical mechanics also confirm the effectiveness of the node action of our ADC proposal. However, to obtain this conclusion, we assumed that the network has numerous nodes. Hence, if N is small, the conclusion may be not valid. In Sect. 4, we will confirm that the conclusion is valid when using relatively-small N.

2.3.2 Variation of System Performance Variable M

In our ADC proposal, system performance variable M behaves stochastically due to the probabilistic node action. The variation of M is related to system stability and quality. Hence, we should understand how M fluctuates when the system uses our ADC proposal.

In this section, we discuss the coefficient of variation of M instead of the variance of M because the scale of the variation of M should increase as the average of M increases. In the previous section, we showed that the average and the variance of M are approximately given by M^* and $\sqrt{C_M}$, respectively. According to these results, the coefficient of variation of M is given by $\sqrt{C_M}/M^*$. According to Eq. (1), M^* and $\sqrt{C_M}$ simultaneously increase as N increases. Since $C_M = -dM^*/d\lambda$, the scale for M^* exceeds that for $\sqrt{C_M}$. Hence, $\sqrt{C_M}/M^*$ decreases as N increases. Therefore, in a large-scale network with our ADC proposal, the variation of M should be insignificant.

This variation in the property of our ADC proposal is also observed in a thermodynamic system. A thermodynamic system is a large-scale system, and state quantities (e.g., energy) in the thermodynamic system remain roughly constant with a value that corresponds to its most frequent value. Hence, in a thermodynamic system, the property for the most frequent value is dominant.

2.3.3 Adjustability of Strengths of Node Concentration and Node Distribution

Another notable feature of our ADC proposal is that single parameter λ controls the strengths of both node concentration and node distribution. Node concentration encourages strongly interacting nodes to take the same state. On the contrary, node distribution encourages each state to be assigned to the same number of nodes.

For controlling the node concentration and the node distribution, we use the following performance variable $m_{ij}(x_i, x_j)$, which is given by

$$m_{ij}(x_i, x_j) = r_{ij} d_{ij}(x_i, x_j), \tag{15}$$

where r_{ij} is the interaction force between nodes i and j, and

$d(x_i, x_j)$ is a function of the distance between states x_i and x_j. If $x_i = x_j$, $d(x_i, x_j)$ returns the smallest positive value than that for $x_i \neq x_j$. When λ is large, a small M frequently occurs in our ADC proposal. To realize a small M, m_{ij} must be small. Hence, if λ is large, nodes with a large r_{ij} prefer to take the minimum $d(x_i, x_j)$ (i.e., $x_i = x_j$). Therefore, by changing λ, our ADC proposal can control the strength of the node concentration.

How does our ADC proposal control the strengths of node concentration and node distribution, simultaneously? Since weak node concentration corresponds to strong node distribution, our ADC proposal can adjust strengths of both node concentration and node distribution by a single parameter λ. In what follows, we explain the correspondence between node concentration and node distribution in a discussion of the global property of our ADC proposal around M^*.

In the previous section, we explained that our ADC proposal encourages system states to take small $F(M)$. Hence, we also discuss the correspondence between node concentration and node distribution from the viewpoint of $F(M)$. By substituting Eq. (12) to $F(M) := M - 1/\lambda \log G(M)$, $F(M)$ is approximately given by

$$F(M) \simeq \frac{M^* + \mu_0}{2} - \frac{1}{\lambda}\left\{\log\left(\frac{|\hat{\Omega}|}{\sqrt{2\pi C_M}}\right) - \frac{(M - M^*)^2}{2C_M}\right\}. \quad (16)$$

To discuss the average behavior of our ADC proposal, we derive the average of $F(M)$. From the above approximated $F(M)$, the average μ_F of $F(M)$ is approximately given by

$$\mu_F \simeq \frac{M^* + \mu_0}{2} + \frac{1}{\lambda}\log\left(\frac{\sqrt{2\pi e\, C_M}}{|\hat{\Omega}|}\right). \quad (17)$$

System states with small $F(M)$ readily occur, but there is a trade-off between the first term and second term's logarithm in Eq. (17) since M^* and $1/|\hat{\Omega}|$ cannot become small, simultaneously. If M^* is small, node concentration is favored, and node movement is limited to a few states. Hence, when M^* is small, $1/|\hat{\Omega}|$ is large. The balance realized between first term and second term's logarithm is determined by λ. If λ is large, the first term is small (i.e., the node concentration is strong) since the weight $1/\lambda$ of the second term is relatively small. On the contrary, if λ is small, the second term's logarithm is small (i.e., $|\hat{\Omega}|$ is large). A large $|\hat{\Omega}|$ corresponds to strong node distribution, see Appendix. Therefore, by changing λ, our ADC proposal can adjust strengths of both node concentration and node distribution.

2.4 Multi-Timescale Control Scheme

A control scheme that can support a large-scale and wide-area network must satisfy what requirements? First, since the environment surrounding such a network will change frequently, the scheme should be highly responsive. That is, it should quickly respond to environmental fluctuations. System managers want to control the entire network as well as small-scale networks, so the scheme should also offer controllability, that is the ability to manage the entire network according to the system manager's goals (e.g., adjusting a measurement to a given target value). To realize responsiveness and controllability, we consider a multi-timescale scheme based on the control framework [3].

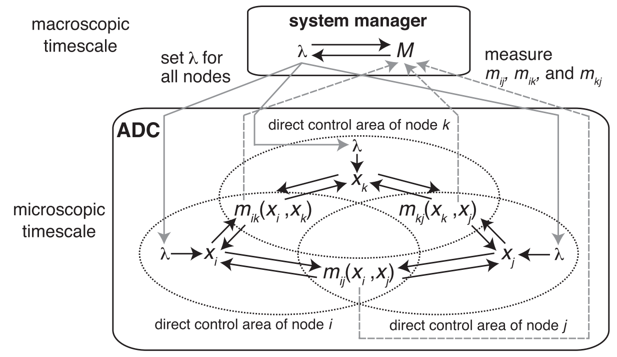


Fig. 3 Multi-timescale control scheme.

The multi-timescale scheme for network control is shown in Fig. 3. In the scheme, multiple sub-schemes work at microscopic and macroscopic timescales. At a microscopic timescale, each node uses our ADC scheme. Thus each node gathers local information, and then adjusts its state according to the data. Hence, the multi-timescale scheme can realize responsiveness. At a macroscopic timescale, the entire network is controlled by a macroscopic sub-scheme set by the system manager. The macroscopic sub-scheme imposes the system manager's goals on all nodes through control parameter λ of our ADC scheme. By using Eq. (4) and λ, node i directly adjusts its state x_i and $m_{ij}(x_i, x_j)$ where $j \in \chi_i$. In Fig. 3, nodes i, j, and k perform direct control. Through the direct control of $m_{ij}(x_i, x_j)$ by nodes, system performance variable M is indirectly controlled. The macroscopic sub-scheme measures the average values of $m_{ij}(x_i, x_j)$ for all pairs of (i, j) on a macroscopic timescale, and calculates the average of M. The system manager confirms whether the average of M suits the intended goal. If it does not, the system manager reconfigures λ using an existing network management method.

The most important challenge for realizing the multi-timescale scheme is how to design an ADC scheme to be able to offer controllability to the system manager of the networks. The main contribution of this paper lies in solving this challenge. To solve the challenge, we apply MCMC to design an ADC scheme in this paper. In the proposed ADC scheme, nodes indirectly control system performance variable M in accordance with λ. Hence, by using ADC based on MCMC, the system manager can adjust the average of M through λ.

3. Application to Traffic-Aware Virtual Machine Placement with Load Balancing

We apply our ADC proposal to traffic-aware placement of VMs in a DCN. In a DCN, the distribution of traffic rates between VMs is usually uneven [11]. Hence, for better network performance, a DCN controller should perform traffic-aware VM placement in which VMs communicating with a high traffic rate should be sited in neighboring of physical machines (PM). However, such traffic-aware VM placement would lead to concentrating VM loads on a few PMs, and thus degrade their computing performance. Hence, load balancing between PMs should be performed together with traffic-aware VM placement. Since strengths of both traffic-aware placement and load balancing should be adjusted according to a given system design policy, we introduce an ADC that adjusts strengths of both traffic-aware placement and load balancing by using the node action designed in Sect. 2. Our ADC proposal implements traffic-aware placement control in an autonomous decentralized manner, and so has higher reliability than the scheme of [11].

3.1 Formulate VM Placement Problem

We formulate the VM placement problem on the basis of the system model explained in Sect. 2.1. Nodes and states in the system model correspond to VMs and PMs, respectively. Node concentration and node distribution correspond to traffic-aware placement and load balancing, respectively.

As system performance variable $M(\boldsymbol{X})$, we use the traffic-cost product sum, which is also used in [11]. According to [11], $r_{i\,j}$ and $d(x_i\, x_j)$ in Eq. (15) are the traffic rate between VMs i and j, and the communication cost (e.g., the number of hops in the shortest path) between PMs x_i and x_j, respectively.

Each VM collects its traffic rates to communicating VMs and communication cost to adjacency PMs in the state transition graph, and calculates ΔM_i and transition probability T_i to control its own migration.

As the metric for the load balancing, we use PM load variance $\mathrm{Var}[\boldsymbol{\rho}]$ ($\boldsymbol{\rho} = (\rho_1, ..., \rho_{N_S})$), which is defined by

$$\mathrm{Var}[\boldsymbol{\rho}] = \frac{1}{N_S}\sum_{l=1} (\rho_l - \mathrm{E}[\boldsymbol{\rho}])^2, \qquad (18)$$

where $\rho_l = \sum_{i\in\phi_l} \rho_i^{(\mathrm{VM})}$ and $\rho_i^{(\mathrm{VM})}$ is VM i's load. ρ_T is the PM load average, which is given by

$$\mathrm{E}[\boldsymbol{\rho}] = \frac{1}{N_S}\sum_{l=1}^{N_S} \rho_l. \qquad (19)$$

4. Experiment

4.1 Experiment Model

The performance of our ADC proposal depends on $G(M)$ because $G(M)$ determines the global property of our ADC proposal. According to Eq. (12), $G(M)$ strongly depends on $\mu_0 = M^* + \lambda C_M$ and $\sigma_0 = \sqrt{C_M}$. Specifically, σ_0 depends on σ_T and σ_D, which are the standard variances of traffic rates and communication costs, respectively. σ_D is set when designing the network topology of a DCN. Hence, the network topology design is related to achieving the desired performance of our ADC proposal. Therefore, we investigate the relation between network topology and the performance of our ADC proposal.

Different network topologies have different path redundancies and installation costs (i.e., number of network links). The network topology used in a DCN is determined by considering the trade-off between path redundancy and installation costs. This paper examines three topologies (tree, fat-tree, and full-mesh) with different levels of path redundancy. It investigates how the trade-off relates to the performance of our ADC proposal.

Figures 4(a) through (c) show tree topology, fat-tree topology, and full-mesh topology, respectively. Tree topology and fat-tree topology are the same as used in [11]. To give the state transition graph in the system model, we separate the set of PMs into 4 groups, and allow VM migration only between PMs in the state transition graph shown in Fig. 4(d). Figure 4(d) omits some edges for ease of viewing.

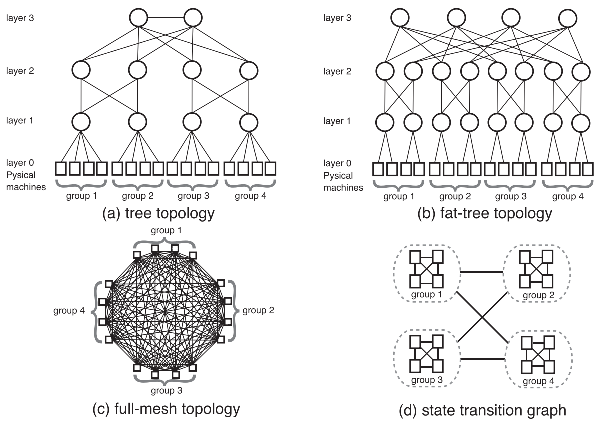


Fig. 4 Network topologies and state transition graph.

Table 1 Topological feature quantities (i.e., μ_T, σ_T, μ_D, and σ_D).

	μ_T	σ_T	μ_D	σ_D
Tree	114.9	31.6	0.44	0.19
Fat-tree	114.9	31.6	0.44	0.15
Full-mesh	114.9	31.6	0.44	0.11

Table 1 shows statistics (i.e., μ_T, σ_T, μ_D, and σ_D) for the three topologies. We define the communication cost between PMs as the sum of link costs on the shortest path between PMs. As an exception, we set the internal communication cost within a PM to the smallest positive value. When the two communicating VMs i and j are assigned in the same PM, $d(x_i, x_j)$ is the smallest communication cost. We set the link cost of each network topology in such a way that all network topologies have the same average communication cost. The differences in the standard deviations of communication costs, σ_D's, among the network topologies affect $G(M)$ and the performance of our ADC proposal.

Regardless of network topology, we use the following

Table 2 Parameter configuration.

number of states (PMs), N_S	16
number of VMs, N	160
average number of high traffic VMs, N_H	10
internal cost in a PM	0.0001 [s]
link cost of tree topology	0.1[s]
link cost of fat-tree topology	0.085[s]
link cost of full-mesh topology	0.47[s]
high traffic rate T_H	10 [Mbit/s]
low traffic rate T_L	0.1 [Mbit/s]
VM load $\rho^{(VM)}$	1
control parameter α	0.1
simulation time	1,000,000 [s]

rule for generating traffic rates. Each VM communicates with a randomly-chosen N_H (average) VMs with high traffic rate T_H, and other VMs with low traffic rate T_L. The same rule for generating traffic rates allows us to focus on the impact of network topology on the performance of our ADC proposal.

At the start of each simulation run, we place N VMs in a randomly-chosen PM. At each simulation time unit, a VM uses the node action to determine if it should migrate to another PM. We perform 200 simulations for different random number sequences under the same parameter setting, and calculate the average and confidence interval for the same parameter setting.

The other parameters are set to the values shown in Table 2. We examine a small-scale system with $N = 160$ due to the computational limits. However, since the statistical approximation used in Sect. 2 works more effective as N increases, in principle, the results obtained in this simulation would also be valid for large-scale networks. However, for practical reasons, we should clarify how scale our ADC proposal is effective. We will confirm that our ADC proposal is effective for the network with $N = 160$. This confirmation would indicate the effectiveness of our ADC proposal for the networks with $N \geq 160$.

4.2 Controllability of Probability Distribution of System Performance Variable M

First, we visually confirm that our ADC proposal can control probability distribution $P(M)$ from the results simulation experiments.

Figures 5(a) through (c) plot probability distribution $P(M)$ when we use tree topology, fat-tree topology, and full-mesh topology. Section 2.3.1 explained that the average of system performance variable, M^*, decreases as λ increases in our ADC proposal. According to these figures, the simulation results do not contradict the explanation in Sect. 2.3.1, and the sensitivity of λ on M^* depends on network topology. In what follows, we investigate the difference among the network topologies.

4.3 Effectiveness of Traffic-Aware VM Placement Control with Load Balancing

Figures 6 and 7 plot averages of traffic-cost product sum M and PM load variance Var[$\boldsymbol{\rho}$] for different values of control parameter λ, respectively. Our ADC proposal more closely resembles simple random control as λ approaches 0. Hence, the results for a small λ are representative of the results for randomly placing VMs. From these figures, our ADC proposal can adjust strengths of both traffic-aware VM placement and load balancing by changing single parameter λ. The averages of M for fat-tree and full-mesh topologies drastically decrease at a certain λ; we call this phenomenon *phase transition*. The phase transition is caused by the shrinkage of the system state space Ω to $\hat{\Omega}$. Figure 8 plots the average of the number of PMs having VMs, $\hat{N}_S$. Since $|\hat{\Omega}|$ can be estimated by $\hat{N}_S^N$, decreasing $|\hat{\Omega}|$ (i.e., shrinkage of Ω) is confirmed by decreasing $\hat{N}_S$. Figure 8 shows that $\hat{N}_S$ for fat-tree and full-mesh topologies more strongly decreases at a certain λ. This phenomenon corre-

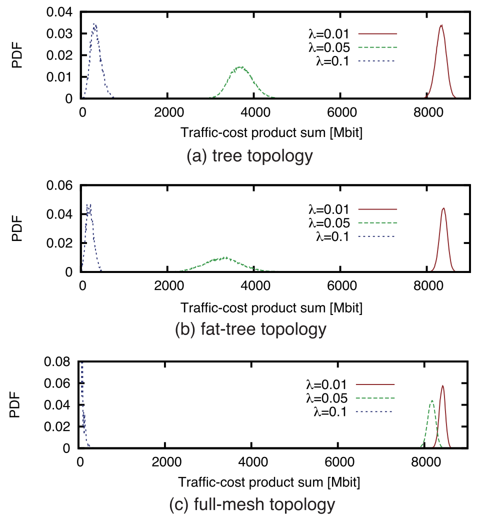


Fig. 5 Probability distribution $P(M)$.

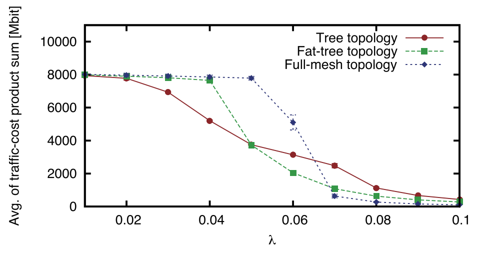


Fig. 6 Average of traffic-cost product sum M.

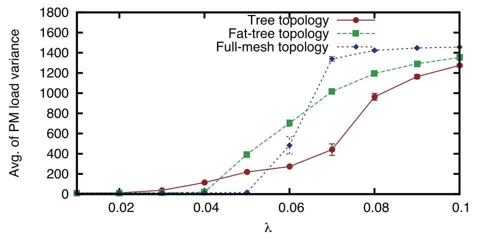


Fig. 7 Average of PM load variance Var[$\boldsymbol{\rho}$].

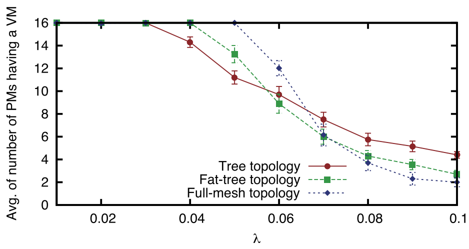


Fig. 8 Average of the number of PMs having VMs, N_S.

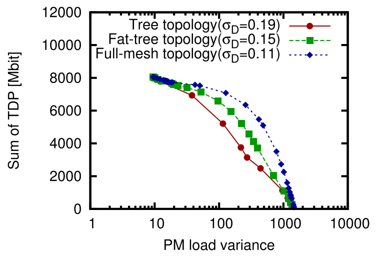


Fig. 9 Traffic-cost product sum vs. PM load variance.

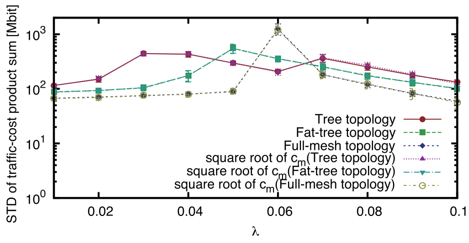


Fig. 10 Standard deviation (STD) of traffic-cost product sum M.

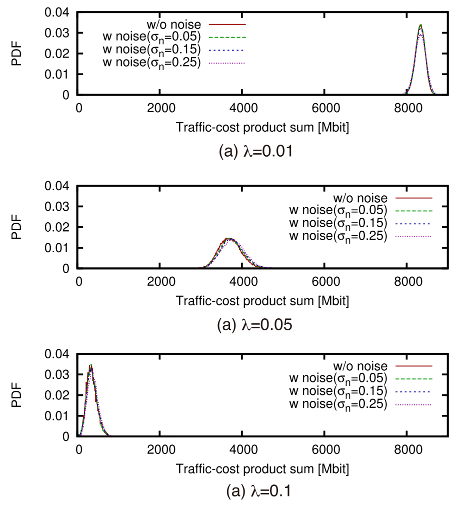


Fig. 11 Probability distribution $P(M)$ for different fluctuation strengths σ_n.

sponds to the above-mentioned phase transition.

For a more detailed discussion, we investigate the average of traffic-cost product sum M vs. the average of PM load variance Var[$\boldsymbol{\rho}$], see Fig. 9. In this figure, confidence intervals are not shown because we repeat the simulation over 200 times until they are sufficiently small. Since small M and Var[$\boldsymbol{\rho}$] are desired, a line closer to the origin is higher efficient in term of the trade-off between traffic-aware VM placement and load balancing. From Fig. 9, the trade-off improves with larger values of the standard deviation of communication cost, σ_D.

Therefore, our ADC proposal is highly efficient if the network topology has large standard variance σ_D (i.e., low path redundancy).

4.4 Variance of System Performance Variable M

We next investigate this variance since it impacts system stability. Figure 10 plots the standard deviation of M obtained from a simulation.

The standard deviation of a network topology with a smaller standard variance of communication cost, σ_D, is larger, except for a few results (i.e., the results with $\lambda = 0.05$ and 0.06). This can be explained as follow. First, the standard deviation of $P(M)$ equals that of $G(M)$. In the experiment setting, the standard variance σ_D mainly affects the standard deviation of $G(M)$. Since the shrinkage of system state space $|\Omega|$ shown in Fig. 8 also affects it, the inverted relationship occurs in $\lambda = 0.05$ and 0.06. However, the inverted relationship is a fairly minor result.

Therefore, the network topology with our ADC proposal and a small standard variance σ_D (i.e., high path redundancy) has good stability. However, since high path redundancy does not provide a good trade-off between traffic-aware VM placement and load balancing, the network topology used in a DCN should be selected according to the performance desired.

In Fig. 10, to confirm the validity of the approximation in Section 2, we also plot $\sqrt{C_m}$, which is calculated from

$$\frac{dM^*}{d\lambda} \simeq \frac{M^*(\lambda + \epsilon_\lambda) - M^*(\lambda - \epsilon_\lambda)}{2\,\epsilon_\lambda}, \tag{20}$$

where $\epsilon_\lambda = 0.001$. In Fig. 10, the curves of $\sqrt{C_M}$ overlap with those of the standard deviation of M. Figure 10 allows us to confirm the validity of the approximation since the standard deviations of M coincide with C_M.

4.5 Robustness against Fluctuation in the System

Our ADC proposal will have the robustness against fluctuation in a system since we design its probabilistic node action to absorb the fluctuation. Hence, we will confirm here the robustness of our ADC proposal against fluctuations in traffic rates and VM loads.

To investigate the robustness, we generate noises around initial value init$_{\text{val}}$ for traffic rates r_{ij} and VM loads $\rho_i^{(\text{VM})}$. The noises follow normal distribution $N(0, \text{init}_{\text{val}}\sigma_n)$ where σ_n is fluctuation strength.

Figures 11(a) and (c) plot probability distribution $P(M)$ with $\lambda = 0.01$, 0.05 and 0.1 for different σ_n values when using the tree topology. Although we discuss here only with tree topology, we also have confirmed the validity for other topologies and other λ. From this figure, our ADC proposal maintains the probability distribution regardless of fluctuation strength σ_n. Hence, our ADC proposal has the robustness against fluctuation.

4.6 Centralized Control versus Our ADC Proposal

Finally, we compare our control against a centralized control, which collects information from the whole system and then sets the system state so as to optimize performance. Our comparison objective is to clarify the side effect of our ADC proposal. In ADC, each node controls its state by using only local information. On the contrary, in centralized control, the management node controls the system state by using full information of the entire network. Hence, in static situation where information is time invariant, centralized control would be higher performance than the proposed ADC. In this section, we investigate how the proposed ADC is lower performance than centralized control.

We examine just centralized control with clustering to simply investigate the performance difference. The centralized control first tries to assign VMs with high traffic rates to the same cluster, and clusters to PMs such that each PM has an equal number of VMs. As the number of clusters increases, it becomes easy to perform load balancing but it becomes difficult to perform traffic-aware VM placement. Hence, by changing the number of clusters, N_C, the centralized control can adjust the balance between traffic-aware VM placement and load balancing.

We summarize below the procedures of the centralized control with clustering.

1. Input the target number of clusters, N_C.
2. Collect traffic rates for all pairs of VMs.
3. Make initial clusters for each VM. Cluster i corresponds to VM i, and the edge weight w_{ij} between clusters i and j is given by r_{ij}. If $r_{ij} = 0$, the edge between clusters i and j does not exist.
4. Select the edge with maximum weight. We assume that clusters i and j are connected by the maximum edge.
5. Merge clusters i and j to cluster k, and set edge weights w_{kl} between the merged cluster k and other clusters l by $w_{kl} = (w_{il} + w_{jl})/2$.
6. Repeat 4) through 5) until the number of clusters becomes N_C.
7. Assign the N_C clusters to PMs. Namely, we assign the x-th cluster to the y-th PM where $y = (x-1 \bmod N_S)+1$.

Figure 12 plots an example of the procedures of the centralized control with clustering. In this figure, circles and dashed ellipses correspond to VMs (i.e., VM a, b, c, d, and e) and clusters, respectively. We do not draw links that have weight 0.

Figure 13 shows the average of traffic-cost product sum M^* vs. the average of PM load variance Var[$\boldsymbol{\rho}$] for our ADC proposal and the centralized control when using the tree topology. From this figure, if M^* is small, our ADC proposal has almost the same performance as the centralized control.

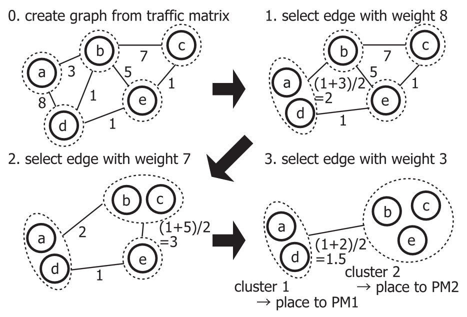

0. create graph from traffic matrix
1. select edge with weight 8
2. select edge with weight 7
3. select edge with weight 3
(1+3)/2 =2
(1+5)/2 =3
(1+2)/2 =1.5
cluster 1 → place to PM1
cluster 2 → place to PM2

Fig. 12 An example of the procedures of centralized control with clustering ($N_C = 2$).

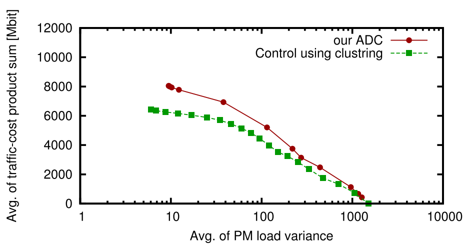


Fig. 13 Traffic-cost product sum vs. PM load variance of our ADC proposal and the centralized control.

This is because that our control yields, in effect, simple random VM placement, as traffic concentration is weak.

The performance difference between our ADC proposal and the centralized control would become small in a large-scale network since the load balancing with simple random VM placement works more efficiently as the number of nodes increases. However, in dynamic environment, the performance of centralized control is suspect because of the limitation of the gathering information by a node. Therefore, we should compare the performances in a large-scale system with state information fluctuation in a future work.

5. Related Work

There are many centralized control schemes [11]–[14]. The centralized VM placement control in [11] performs traffic concentration in a DCN by using the traffic rates between VMs and network costs between PMs. The centralized resource management control of [14] assigns jobs to servers in a data center according to job size and server capacity. These centralized controls need to gather information from the whole system, and so they cover much smaller-scale networks than our targets.

Several ADCs have been proposed in [10], [15]–[20]. In [19], [20], ADC schemes are designed on the basis of biologically inspired engineering. Unlike such a bio-inspired ADC schemes, our ADC proposal is able to offer the controllability to the system manager of large-scale and wide-area networks. In [15], decentralized control is realized by decomposing the overall control problem into a set of smaller subproblems. This scheme is scalable, but its application is limited to decomposable problems. The VM placement problem explained in this paper cannot be decomposed into a set of subproblems, and so we need another autonomous decentralized scheme of [15] for solving the VM placement problem. In [10], [16]–[18], the ADCs based on Markov approximation are proposed. The Markov approximation is a method that approximates an optimization problem as a stochastically-optimized Markov chain. This Markov chain is generated by using MCMC, and so these controls using the Markov approximation are similar to our ADC proposal. However, we focus on controlling a system performance variable, while they control system state. By controlling the system performance variable, it becomes easy to not only understand the system global properties, but also implement global control.

6. Conclusion and Future Work

In this paper, we introduced an ADC scheme for indirectly controlling a system performance variable in large-scale and wide-area networks. Our ADC proposal uses a node action based on MCMC, and was investigated by analyses based on statistical mechanics. The results showed that the node action of our ADC proposal can (a) indirectly control a system performance variable, and (b) adjust the strengths of two different controls (i.e., node concentration and node distribution) with a single control parameter.

We applied our ADC proposal to design a traffic-aware VM placement control with load balancing in a DCN. This control can adjust strengths of both traffic-aware VM placement and load balancing by using a single control parameter. Through simulation experiments, we clarified that (a) our placement control is effective several network topologies, and (b) the network topology of a DCN should be selected according to the performance requirements of the DCN manager, because, in ADC, network topologies with low path redundancy are better in term of efficiency but worse in terms of stability. Moreover, we clarified the robustness of our ADC proposal against system fluctuations, and compared the performance of our ADC proposal to that of a centralized control scheme using clustering.

Our future work is divided into two parts. In the first part, we are planning to develop VM placement control for a realistic environment as well as a realistic DCN model. This paper focused on simple VM placement control, and so we need to refine the designed VM placement control to make it practical. After refining the VM placement control, we will implement the VM placement control in a DCN, and investigate the performance of our ADC proposal in a real environment. In particular, we will clarify the computation time of our ADC proposal. In the second part, we will investigate the effectiveness of our ADC proposal under several conditions. In particular, we will clarify the effectiveness of our ADC proposal compared with centralized control in a dynamical environment. In the comparison, we should consider the effect of the time spent on gathering the information explained in Sect. 1. Then, we will design an adaptive control of λ by each node in a dynamic environment, and clarify its effectiveness. Moreover, we will investigate the robustness against the fluctuations in the numbers of states and nodes according to practical situation.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Number 24700072 and 15K15985.

References

[1] V.A. Cicirello and S.F. Smith, "Ant colony control for autonomous decentralized shop floor routing," Proc. 5th International Symposium on Autonomous Decentralized Systems, pp.383–390, March 2001.
[2] C. Takano, M. Aida, M. Murata, and M. Imase, "Proposal for autonomous decentralized structure formation based on local interaction and back-diffusion potential," IEICE Trans. Commun., vol.E95-B, no.5, pp.1529–1538, May 2012.
[3] M. Aida, "Using a renormalization group to create ideal hierarchical network architecture with time scale dependency," IEICE Trans. Commun., vol.E95-B, no.5, pp.1488–1500, May 2012.
[4] K. Leibnitz, N. Wakamiya, and M. Murata, "Biologically inspired self-adaptive multi-path routing in overlay networks," Communications of the ACM, vol.49, no.3, pp.62–67, March 2006.
[5] "Google data centers." http://www.google.com/about/datacenters/inside/index.html
[6] W.R. Gilks, S. Richardson, and D.J. Spiegelhalter, Markov chain Monte Carlo in practice, vol.2, Springer, 1996.
[7] W.K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," Biometrika, vol.57, no.1, pp.97–109, 1970.
[8] Y. Sakumoto, M. Aida, and H. Shimonishi, "Autonomous decentralized mechanisms for generating global order in large-scale system: Using metropolis-hast ings algorithm and applying to virtual machine placement," Proc. 9th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT 2012), pp.1–6, Nov. 2012.
[9] Y. Sakumoto, M. Aida, and H. Shimonishi, "An autonomous decentralized control for indirectly controlling system performance variable in large-scale and wide-area network," Proc. 2014 16th International Telecommunications Network Strategy and Planning Symposium (Networks), pp.1–7, Sept. 2014.
[10] M. Chen, S.C. Liew, Z. Shao, and C. Kai, "Markov approximation for combinatorial network optimization," 2010 Proc. IEEE INFOCOM, pp.1–9, 2010.
[11] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," 2010 Proc. IEEE INFOCOM, pp.1–9, 2010.
[12] C. Hyser, B. Mckee, R. Gardner, and B.J. Watson, "Autonomic virtual machine placement in the data center," Technical Report of Hewlett Packard Laboratories (HPL-2007-189), June 2007.
[13] M. Tarighi, S.A. Motamedi, and S. Sharifian, "A new model for virtual machine migration in virtualized cluster server based on fuzzy decision making," J. Telecommunications, vol.1, no.1, pp.901–907, Feb. 2010.
[14] H. Xu and B. Li, "Anchor: A versatile and efficient framework for resource management in the cloud," IEEE Trans. Parallel Distrib. Syst., vol.24, no.6, pp.1066–1076, June 2013.
[15] R. Wang, D.M. Kusic, and N. Kandasamy, "A distributed control framework for performance management of virtualized computing environments," Proc. 7th International Conference on Autonomic Computing — ICAC'10, pp.89–98, June 2010.
[16] S. Zhang, Z. Shao, and M. Chen, "Optimal distributed P2P streaming under node degree bounds," Proc. The 18th IEEE International

Conference on Network Protocols, pp.253–262, Oct. 2010.

[17] J.W. Jiang, T. Lan, S. Ha, M. Chen, and M. Chiang, "Joint VM placement and routing for data center traffic engineering," 2012 Proc. IEEE INFOCOM, pp.2876–2880, March 2012.

[18] Z. Shao, X. Jin, W. Jiang, M. Chen, and M. Chiang, "Intra-data-center traffic engineering with ensemble routing," 2013 Proc. IEEE INFOCOM, pp.2148–2156, April 2013.

[19] G. Neglia and G. Reina, "Evaluating activator-inhibitor mechanisms for sensors coordination," 2007 2nd Bio-Inspired Models of Network, Information and Computing Systems, pp.129–133, 2007.

[20] E. Sakhaee, K. Leibnitz, N. Wakamiya, and M. Murata, "Bio-inspired layered clustering scheme for self-adaptive control in wireless sensor networks," Proc. 2009 2nd International Symposium on Applied Sciences in Biomedical and Communication Technologies, pp.1–6, Nov. 2009.

Appendix: Correspondence between Large $|\hat{\Omega}|$ and Strong Node Distribution

We have used the correspondence between large $|\hat{\Omega}|$ and strong node distribution in Sect. 2 to show that our ADC proposal can adjust the strengths of both node concentration and node distribution.

To make the discussion explicit, we define $|\hat{\Omega}|$ and the variable for node distribution as follows.

- We approximate $|\hat{\Omega}|$ by $\hat{N}_S^N$ where $\hat{N}_S (1 \leq \hat{N}_S \leq N_S)$ is the number of states assigning nodes. When node concentration is strongly emphasized, node transition is limited to a few states (i.e., $\hat{N}_S$ states). Hence, the change in $\hat{N}_S$ represents the change in $|\hat{\Omega}|$.
- We use variance $\mathrm{Var}[\boldsymbol{\rho}]$ as a variable for node distribution where $\boldsymbol{\rho} = (\rho_1, ..., \rho_{N_S})$ and ρ_k is the number of nodes with state k. If all ρ_k's for each state are equal, node distribution is ideal. Hence, small $\mathrm{Var}[\boldsymbol{\rho}]$ values are better for node distribution.

With the above definition, we confirm the correspondence between large $\hat{N}_S$ and small $\mathrm{Var}[\boldsymbol{\rho}]$ to show the correspondence between large $|\hat{\Omega}|$ and strong node distribution.

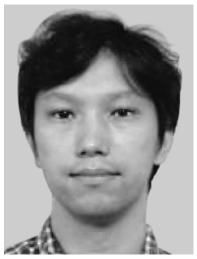
When $\hat{N}_S$ increases to $\hat{N}'_S$ $(\hat{N}_S < \hat{N}'_S)$, $|\hat{\Omega}|$ increases from $\hat{N}_S^N$ to $\hat{N}'^N_S$. If the probability property (i.e., average and variance) of node loads is the same, ρ_k decreases to $\hat{N}_S/\hat{N}'_S\,\rho_k$ because the number of nodes with a state decreases to $\hat{N}_S/\hat{N}'_S$. Since $\mathrm{Var}[a\,\boldsymbol{X}] = a^2\mathrm{Var}[\boldsymbol{X}]$, $\mathrm{Var}[\boldsymbol{\rho}]$ decreases to

$$\frac{\hat{N}_S^2}{\hat{N}'^2_S}\mathrm{Var}[\boldsymbol{\rho}]. \tag{A$\cdot$1}$$

From the above discussion, we can confirm the correspondence between large $\hat{N}_S$ and small $\mathrm{Var}[\boldsymbol{\rho}]$ (i.e., strong node distribution).

Yusuke Sakumoto received M.E. and Ph.D. degrees in the Information and Computer Sciences from Osaka University in 2008 and 2010, respectively. He is currently an assistant professor at Graduate School of System Design, Tokyo Metropolitan University, Japan. His research work is in the area of autonomous decentralized control for large-scale systems. He is a member of IEEE, IPSJ and IEICE.

Masaki Aida received the B.S. degree in Physics and M.S. degree in Atomic Physics from St. Paul's University, Tokyo, Japan, in 1987 and 1989, respectively, and received the Ph.D. in Telecommunications Engineering from the University of Tokyo, Japan, in 1999. After joining NTT Laboratories in April 1989, he has been engaged in research on traffic issues in computer communication networks. From April 2005 to March 2007, he was an Associate Professor at the Faculty of System Design, Tokyo Metropolitan University. He has been a Professor of the Graduate School of System Design, Tokyo Metropolitan University since April 2007. Prof. Aida is a member of the IEEE, IEICE, and the Operations Research Society of Japan.



Hideyuki Shimonishi received M.E. and Ph.D. degrees from the Graduate School of Engineering Science, Osaka University, Osaka, Japan, in 1996 and 2002. He joined NEC Corporation in 1996 and has been engaged in research on traffic management in high-speed networks, switch and router architectures, and traffic control protocols. As a visiting scholar in the Computer Science Department at the University of California at Los Angeles, he studied next-generation transport protocols. He now works in Knowledge Discovery Research Laboratories at NEC Corp, engaged in researches on networking technologies including SDN, OpenFlow and NFV for carrier, data center and enterprise networks. He is a member of the IEICE.