Chapter 4
Yuragi-Based Virtual Network Control

Yuki Koizumi

Abstract Reconfiguring a virtual network, which consists of a set of virtual links and routers, on top of a physical network is a promising approach to accommodate time-varying traffic. However, optimization-based approaches are generally incapable of quickly adapting to time-varying traffic due to their computational complexity. For swift adaptation to changes in traffic patterns and network failures, this chapter presents *Yuragi-based virtual network control*. This control is based on *attractor selection*, which models behavior whereby biological systems adapt to unknown changes in their surrounding environments. A biological system driven by attractor selection adapts to environmental changes, selecting *attractors*, in which the condition of the system is well suited for a certain environment, by using noise, also referred to as *Yuragi*. There are two main challenges in achieving virtual network control based on attractor selection. The first involves determining how to map the behavior of a biological system to virtual network control, while the second involves designing attractors for robust virtual network control. This chapter summarizes previous studies that solved these two research problems.

4.1 Introduction

The emergence of new application layer services, such as video on demand and user-generated content delivery, results in fluctuations in network environments, such as network failures and traffic changes, as well as increased Internet traffic. The Internet plays an increasingly vital role as social infrastructure; therefore, the ability to withstand and recover from various changes in network environments has become a crucial requirement.

Y. Koizumi (✉)
Graduate School of Information Science and Technology, Osaka University, Suita, Osaka, Japan
e-mail: ykoizumi@ist.osaka-u.ac.jp

M. Murata, K. Leibnitz (eds.), *Fluctuation-Induced Network Control and Learning*,
https://doi.org/10.1007/978-981-33-4976-6_4

One promising approach to accommodate fluctuation in network environments is to construct a *virtual network*, which consists of a set of virtual links and routers, on top of a physical network according to the current network environment. Several network virtualization technologies, such as overlay, multi-protocol label switching (MPLS), and generalized MPLS (GMPLS) networks, have been proposed [5]. Among such network virtualization technologies, this chapter focuses on wavelength-division multiplexing (WDM) networks, which offer a flexible virtual network infrastructure by using wavelength routing. WDM networks are suitable for accommodating high Internet traffic owing to their high link capacity; therefore, much research has been devoted to developing methods for carrying Internet traffic over wavelength-routed WDM networks [1, 6, 10, 15, 18, 19, 23, 29, 30]. In a wavelength-routed WDM network, optical transport channels, referred to as lightpaths, are established between routers via optical cross-connects (OXCs), and a set of lightpaths and routers forms a virtual network. Virtual network control, which configures a virtual network on the basis of a given network environment, has been investigated in several studies [21, 24].

By reconfiguring virtual networks, wavelength-routed WDM networks offer a means of adapting to changing network environments. Existing virtual network control methods, which are based on the control paradigm developed in the area of engineering, primarily consider a particular set of scenarios of environmental changes and use an algorithm to perform predefined countermeasures to those changes. Although these methods guarantee optimal performance for their assumed environments, they cannot achieve expected performance in the case of unexpected environmental changes.

A remarkable example of adapting to various environmental changes can be observed in a biological system, which is studied in the field of life sciences [13]. This chapter focuses on *attractor selection*, which models the behavior of organisms when they adapt to unknown changes in their surrounding environments and recover their conditions. Kashiwagi et al. [14] proposed an attractor selection model for an Escherichia coli cell that adapts to changes in nutrient availability. Furusawa and Kaneko [8] introduced another attractor selection model for explaining the adaptability of a cell consisting of a gene regulatory network and a metabolic network. The fundamental principle of attractor selection is that a system adapts to environmental changes by selecting an *attractor* suitable for the current surrounding environment. This selection mechanism in attractor selection is based on deterministic behavior and stochastic behavior that are controlled by simple feedback on the system condition.

The selection mechanism in attractor selection is one of the main differences between attractor selection and heuristic or optimization approaches in the field of engineering. In contrast to engineering systems, biological systems do not rely on predefined algorithms. Instead, they exploit stochastic behavior, which is environmental noise from various sources, to adapt to environmental changes. This chapter refers to attractor selection as *Yuragi-based control*, where *Yuragi* is the Japanese term for noise. On the one hand, biological systems do not guarantee optimal performance, whereas engineering systems generally achieve optimal performance for assumed environments. On the other hand, biological systems are capable of adapting to unknown environmental changes thanks to their stochastic behavior, whereas engineering systems cannot handle environmental changes that are not accounted for by their predefined algorithms. We therefore adopt attractor selection as the key mechanism in our virtual network control method to achieve adaptability to various environmental changes.

There are two main challenges in achieving virtual network control based on attractor selection. The first challenge involves determining how to map attractor selection in a biological system to virtual network control in a wavelength-routed WDM network. The second challenge involves determining how to design attractors for adaptive virtual network control. Previous studies [16, 17] solved the first research problem. Focusing on the similarity between a wavelength-routed WDM network, which consists of a physical and a virtual network, and a cell, which consists of a gene regulatory and a metabolic reaction network, we developed a virtual network control method based on attractor selection found in a cell [8]. In addition, we incorporated the knowledge of the Hopfield neural network [12] into the proposed method to embed attractors into the deterministic behavior of the proposed virtual network control method, focusing on the similarity between attractor selection in a cell and the Hopfield neural network. Another study [22] solved the second research problem by proposing a method for designing diverse attractors to be stored in the attractor structure to handle various environmental changes. This chapter summarizes this series of studies.

The remainder of this chapter is organized as follows. Section 4.2 introduces the concept of attractor selection, while Sect. 4.3 proposes an adaptive virtual network control method based on attractor selection. Section 4.4 presents an algorithm for designing attractors for adaptive virtual network control, while Sect. 4.5 summarizes related approaches. Finally, Sect. 4.6 concludes the chapter.

4.2 Attractor Selection

This section describes attractor selection, which is the core mechanism in our adaptive virtual network control method. The original model for attractor selection was introduced in [8].

4.2.1 Concept of Attractor Selection

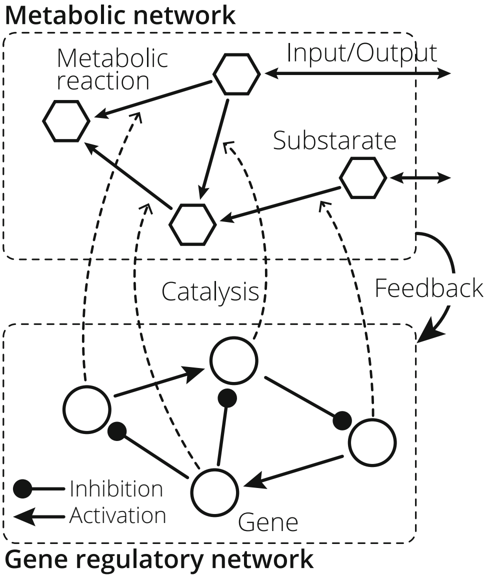
A dynamic system driven by attractor selection uses noise to adapt to environmental changes. In attractor selection, attractors are a part of the equilibrium points in the phase space in which the system condition is preferable. The underlying mechanism consists of deterministic and stochastic behaviors. When the current system condition is suitable for the environment, i.e., the system state is close to one of the attractors, the deterministic behavior drives the system to the attractor. When the current system condition is poor, however, the stochastic behavior dominates over the deterministic behavior. While the stochastic behavior is dominant in controlling the system, the system state fluctuates randomly due to noise, and the system searches for a new attractor. When the system condition recovers and the system state approaches an attractor, the deterministic behavior again controls the system. These two behaviors are controlled by simple feedback on the system condition. Therefore, attractor selection adapts to environmental changes by selecting attractors using the stochastic behavior, the deterministic behavior, and the simple feedback. In Sect. 4.2.2, we introduce attractor selection that models the behavior of the gene regulatory and the metabolic reaction networks of a cell.

4.2.2 Cell Model

Figure 4.1 presents a schematic diagram of the cell model used in [8]. It consists of two networks: the gene regulatory network in the dotted rectangle at the top of the figure and the metabolic reaction network in the dotted rectangle at the bottom of the figure.

Each gene in the gene regulatory network has protein expression levels, and deterministic and stochastic behaviors in each gene control the expression level. The deterministic behavior controls the expression level due to the effects of activation and inhibition from the other genes. In Fig. 4.1, the effects of activation and inhibi-

Fig. 4.1 Schematic diagram of a gene regulatory and a metabolic reaction network



tion are indicated by triangular-headed and circular-headed arrows, respectively. In the stochastic behavior, inherent noise randomly changes the expression levels.

In the metabolic reaction network, metabolic reactions consume various substrates and produce new substrates. Proteins in the corresponding genes catalyze these metabolic reactions. In Fig. 4.1, metabolic reactions are illustrated as fluxes of substrates, and catalyses of proteins are indicated by dashed arrows. Changes in the concentrations of metabolic substrates are caused by metabolic reactions and the transportation of substrates from outside the cell. Some nutrient substrates are supplied from the environment by diffusion through the cell membrane.

The dynamics of the metabolic reactions determines the growth rate. Some metabolic substrates are necessary for cellular growth; thus, the growth rate is determined as an increasing function of their concentrations. The gene regulatory network uses the growth rate as feedback on the condition of the metabolic reaction network and controls the deterministic and stochastic behaviors using the growth rate. If the metabolic reaction network is in poor condition and the growth rate is low, the stochastic behavior dominates over the deterministic behavior, triggering a search for a new attractor. During this phase, the expression levels are randomly changed by noise, and the gene regulatory network searches for a state that is suitable for the current environment. After the condition of the metabolic reaction network is recovered and the growth rate increases, the deterministic behavior again drives the gene regulatory network to a stable state. Section 4.2.3 describes the mathematical model of attractor selection in greater detail.

4.2.3 Mathematical Model of Attractor Selection

The internal state of a cell is represented by a set of protein expression levels of n genes $(x_1, x_2, \ldots, x_n)$, and concentrations of m metabolic substrates $(y_1, y_2, \ldots, y_m)$. The dynamics of the protein expression level of the ith gene, x_i, is described as

$$\frac{dx_i}{dt} = \alpha \left(\varsigma \left(\sum_j W_{ij} x_j - \theta \right) - x_i \right) + \eta. \tag{4.1}$$

The first and second terms on the right-hand side represent the deterministic behavior of gene i, while the third term represents the stochastic behavior. In the first term, the regulation of the protein expression level of gene i by other genes is indicated by a regulatory matrix W_{ij}, which takes 1, 0, or -1 corresponding to activation, no regulatory interaction, or inhibition of gene i by gene j, respectively. The rate of increase in the expression level is given by the sigmoidal regulation function, $\varsigma(z) = 1/(1 + e^{-\mu z})$, where $z = \sum W_{ij} x_j - \theta$ is the total regulatory input with threshold θ for increasing x_i, and μ denotes the gain parameter of the sigmoid function. The second term represents the rate of decrease in the expression level of gene i. This term signifies that the expression level decreases depending on the current expression level. The last term on the right-hand side of Eq. (4.1), η, represents molecular fluctuation, which is Gaussian white noise. Noise η is independent of the production and consumption terms, and its amplitude is constant. The change in expression level x_i is determined by the deterministic behavior, i.e., the first and second terms in Eq. (4.1), and the stochastic behavior η. The deterministic and stochastic behaviors are controlled by the growth rate α, which represents the condition of the metabolic reaction network.

In the metabolic reaction network, metabolic reactions, which are an internal influence, and the transportation of substrates from the outside of the cell, which is an external influence, determine the changes in the concentrations of metabolic substrates y_i. Proteins of the corresponding genes catalyze the metabolic reactions, and the expression level x_i determines the strength of the catalysis. A high x_i accelerates the metabolic reaction, while a low x_i suppresses it. In other words, the gene regulatory network controls the metabolic reaction network through catalysis.

Some of the metabolic substrates are necessary for cellular growth. The growth rate of the cell, α, is determined as an increasing function of the concentrations of these vital substrates. The gene regulatory network uses α as feedback on the condition on the metabolic reaction network and controls the deterministic and stochastic behaviors. The feedback value is referred to as *activity* in the framework of attractor selection. If the concentrations of the required substrates decrease due to changes in the concentrations of nutrient substrates outside the cell, α also decreases. By reducing α, the effects of the first and second terms in Eq. (4.1) on the dynamics of x_i decrease, and the effects of η increase. Thus, x_i fluctuates randomly, and the gene regulatory network searches for a new attractor. The fluctuation in x_i leads to changes in the rate of metabolic reactions via the protein catalysis. When the concentrations of the required substrates again increase, α also increases. Then, the first and second terms in Eq. (4.1) again dominate the dynamics of x_i over the stochastic behavior, and the system converges to the state of the attractor. Since we mainly use the gene regulatory network, we omit a detailed description of the metabolic reaction network from this chapter, and readers can refer to [8] for additional information. Section 4.3 describes the proposed virtual network control method based on the attractor selection model.

4.3 Virtual Network Control Based on Attractor Selection

In this section, we present a virtual network control method based on the attractor selection model of a gene regulatory network. We first introduce the network model used in this chapter and then describe our proposed method.

4.3.1 Virtual Network Control

A physical network consists of nodes having routes overlaying OXCs, with the nodes interconnected by optical fibers, as illustrated in Fig. 4.2a. Optical demultiplexers allow optical signals to be dropped at routers, and OXCs allow optical signals to pass through them. In such wavelength-routed WDM networks, nodes are connected with dedicated virtual circuits called lightpaths. Virtual network control configures lightpaths between routers via OXCs, and these lightpaths and routers form a virtual network, as illustrated in Fig. 4.2b. When lightpaths are configured in the WDM network, as illustrated in Fig. 4.2a, the virtual network in Fig. 4.2b is formed. The IP network uses a virtual network as its network infrastructure and transports IP traffic on the virtual network. By reconfiguring virtual networks, that is, by establishing lightpaths, wavelength-routed WDM networks offer the means to adapt to changing network environments. Determining how to reconfigure virtual networks is thus indispensable to develop an adaptive virtual network control method.

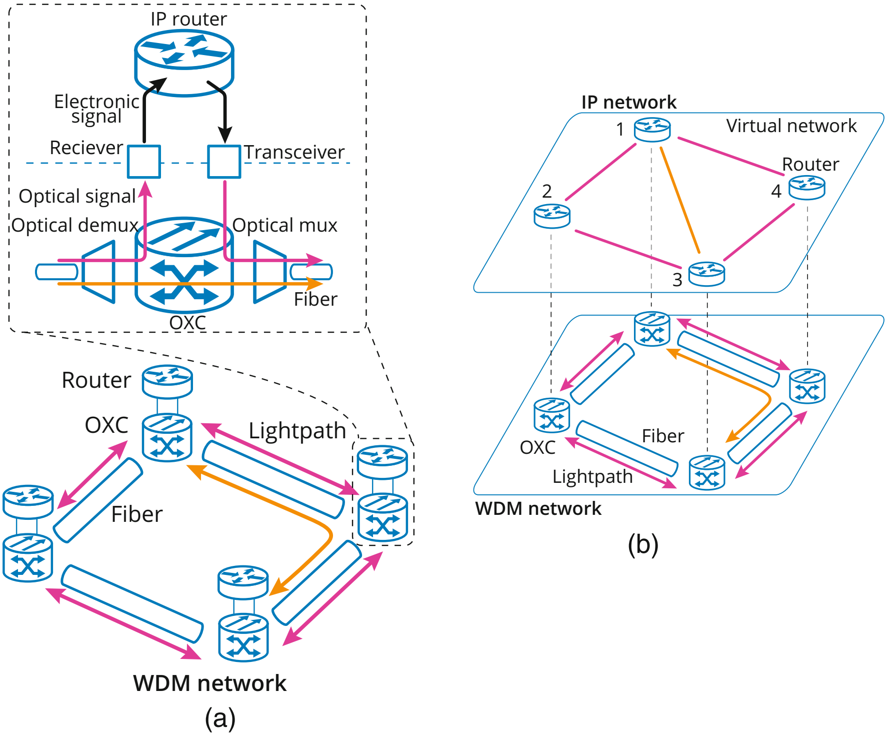


Fig. 4.2 Example of a wavelength-routed WDM network. (**a**) Physical view of the network. (**b**) Logical view of the network, where the lower layer represents a WDM network, which consists of OXCs and fibers, and the upper layer represents an IP network, which uses a virtual network constructed due to the wavelength-routing capability as network infrastructure

4.3.2 Overview of Virtual Network Control Based on Attractor Selection

In a cell, the gene regulatory network controls the metabolic reaction network, and the growth rate, which is the condition of the metabolic reaction network, is recovered when it is degraded due to changes in the environment. Similarly, the main objective of our virtual network control method is to recover the performance of the IP network by appropriately constructing virtual networks when the performance is degraded due to changes in the network environment. We therefore interpret the gene regulatory network as a WDM network and the metabolic reaction network as an IP network, as illustrated in Fig. 4.3. Using the stochastic behavior, our virtual network control method adapts to various changes in the network environment by selecting suitable attractors, which correspond to virtual networks in our method, for the current network environment, and the performance of the IP network can be recovered after it has degraded due to network failures.

A flowchart of the proposed virtual network control method is presented in Fig. 4.4. Our proposed approach works on the basis of periodic measurements of the IP network condition. One example of a condition measure is the load on links, which is the traffic volume on links. The link load is converted into activity, which is the value that controls the deterministic and stochastic behaviors. We describe the activity and the deterministic and stochastic behaviors in Sect. 4.3.3. The proposed method controls the deterministic and stochastic behaviors in the same way as attractor selection depending on the activity. The method constructs a new virtual network according to the system state of attractor selection, and the constructed virtual network is applied as the new infrastructure for the IP network. By sending

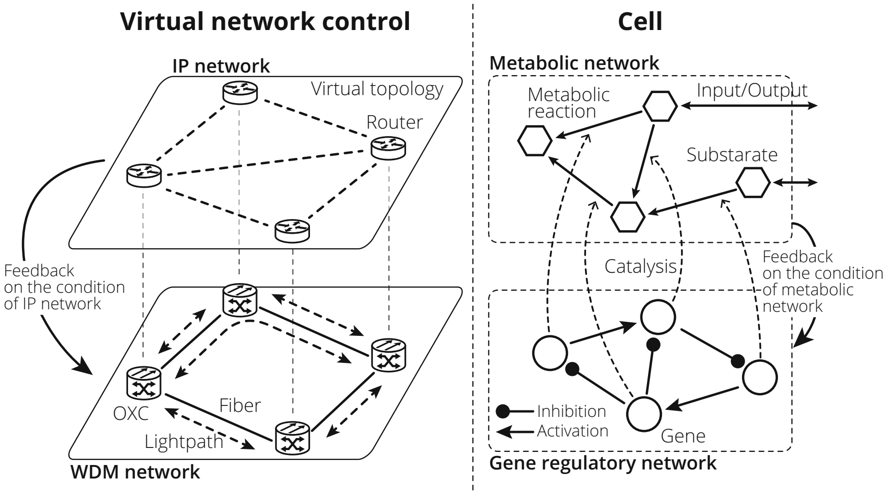


Fig. 4.3 Application of attractor selection to virtual network control

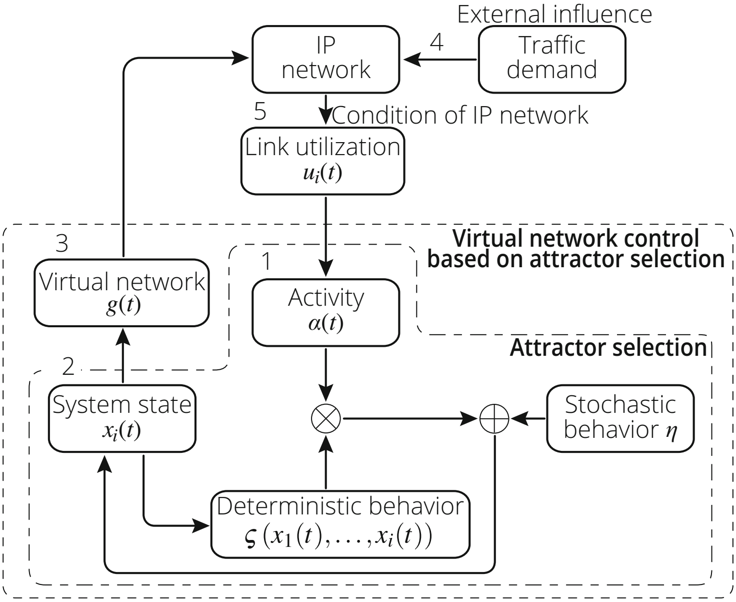

External influence
Traffic demand
4
IP network
5
Condition of IP network
Link utilization $u_i(t)$
Virtual network control based on attractor selection
3
Virtual network $g(t)$
1
Activity $\alpha(t)$
Attractor selection
2
System state $x_i(t)$
Stochastic behavior η
Deterministic behavior $\varsigma(x_1(t),\ldots,x_i(t))$

Fig. 4.4 Flowchart of virtual network control based on attractor selection

traffic over this new virtual network, the link load on the IP network is changed, and the method retrieves the link load to determine the IP network condition.

4.3.3 Dynamics of Virtual Network Control

We place genes on all candidates of possible lightpaths, where l_i denotes the ith lightpath. Each gene has its protein expression level x_i, and the l_i is controlled by x_i. The dynamics of x_i is described as

$$\frac{dx_i}{dt} = \alpha \left(\varsigma \left(\sum_j W_{ij} x_j \right) - x_i \right) + \eta, \tag{4.2}$$

where η represents white Gaussian noise, $\varsigma(z) = 1/(1 + \exp(-z))$ is the sigmoidal regulation function, and the activity α, which is equivalent to the growth rate in cells introduced in Sect. 4.2, represents the IP network condition. We define α below. We use the same formula as Eq. (4.1) to calculate x_i, which is used to determine whether l_i is established. In our method, we establish l_i if $x_i > 0.5$; otherwise, we do not establish l_i. Therefore, our method interprets x_i as the virtual network.

In a cell, α represents the condition of the metabolic reaction network, and the gene regulatory network seeks to optimize α. Our method uses the maximum link utilization on the IP network as a metric representing the condition of the IP network. To obtain the maximum link utilization, we collect the traffic volume on all links and select the maximum value. This information is quickly and directly retrieved using simple network management protocol (SNMP). The activity must be an increasing function of the condition of the target system (in this case, the IP network), as mentioned in Sect. 4.2. Note that any metric that indicates the condition of an IP network, such as average end-to-end delay, average link utilization, and throughput, can be used to define α. We employ maximum link utilization as the IP network condition because it is a major performance metric for virtual network control that has been used in many studies [9, 24]. We convert the maximum link utilization on the IP network, $u_{\max}$, into α as follows:

$$\alpha = \frac{1}{1 + \exp\left(\delta\left(u_{\max} - \zeta\right)\right)}, \tag{4.3}$$

where δ represents the gradient of this function, and the constant ζ is the threshold for α. If the maximum link utilization is greater than ζ, α rapidly approaches 0 due to the unsatisfactory condition of the IP network. Therefore, the dynamics of our virtual network control method is governed by noise and the search for a new attractor. When the maximum link utilization is less than ζ, we rapidly increase α to improve the maximum link utilization.

A smooth transition between the current virtual network and the newly calculated virtual network is another important problem in virtual network control, as discussed in [7, 31]. Our virtual network control method constructs a new virtual network on the basis of the current virtual network, and the difference between these two virtual networks is given by Eq. (4.2). A high degree of activity signifies that the current system state x_i is near the attractor, which is one of the equilibrium points in Eq. (4.2); therefore, the difference given by this equation is close to zero. Consequently, our virtual network control method makes small changes to the virtual network, enabling adaptation to changes in the network. When there is a low degree of activity due to the poor IP network condition, the stochastic behavior dominates over the deterministic behavior. Thus, x_i fluctuates randomly due to noise η to search for a new virtual network that has higher activity, i.e., the lower maximum link utilization. Our method makes large changes to the virtual network to efficiently discover a suitable virtual network from a multitude of possible virtual networks. In this way, the proposed method modifies virtual networks depending on the maximum link utilization in the IP network and adapts to changes in traffic demand. We have already demonstrated that this method achieves smooth transition between virtual networks in [16].

The proposed method constructs virtual networks according to x_i, and x_i converges to one of attractors, which are a part of the equilibrium points in the phase space; therefore, the definition of attractors is a challenging and essential aspect of our proposed method. Section 4.3.4 presents how to define attractors in the phase space.

4.3.4 Attractor Structure

The regulatory matrix W_{ij} in Eq. (4.2) is an important parameter since it determines the locations of attractors in the phase space, referred to as the *attractor structure*. Our method selects one of the attractors according to Eq. (4.2) and constructs a virtual network corresponding to the selected attractor. Hence, defining W_{ij} is a challenging research problem. To define arbitrary attractors in the phase space, we use knowledge about the Hopfield neural network, which has a similar structure to gene regulatory networks.

The dynamics of our proposed method is expressed by Eq. (4.2). From the perspective of dynamical systems, α is regarded as a constant value that determines the convergence speed, and noise η is Gaussian white noise with mean 0. These values do not affect the equilibrium points, i.e., attractors in our method, in the phase space. Therefore, the equilibrium points are determined by the following differential equation:

$$\frac{d}{dt}x_i = \varsigma\left(\sum_j W_{ij}x_j\right) - x_i.$$

This is the same formula as for a continuous Hopfield neural network [12]. We therefore use methods to use Hopfield neural networks as content-associative memory to store arbitrary attractors in the phase space [3, 25].

Suppose that we store a set of virtual networks $g_k \in G$ in the phase space defined by Eq. (4.2). Let $\boldsymbol{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, \ldots, x_i^{(k)})$ be the vector of the expression levels corresponding to virtual network g_k. To store $\boldsymbol{x}^{(k)}$ in the phase space, we adopt the method introduced in [3], which stores patterns in the phase space by orthogonalizing them. Due to space limitations, we omit a detailed description of this method, and readers refer to [3] for a more detailed description. We store m virtual networks, $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}$, in the phase space. Let $\boldsymbol{X}$ be a matrix of which rows are $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}$. The regulatory matrix $\boldsymbol{W} = \{W_{ij}\}$, of which attractors are $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}$, is defined as

$$\boldsymbol{W} = \boldsymbol{X}^{+}\boldsymbol{X}, \tag{4.4}$$

where $\boldsymbol{X}^{+}$ is the pseudo-inverse matrix of $\boldsymbol{X}$.

Although pattern orthogonalization results in the high stability of stored patterns [3], our method can also use more straightforward memorization approaches, such as Hebbian learning [11]. Using Hebbian learning, W_{ij} is defined as follows:

$$W_{ij} = \begin{cases} 0 & \text{if } i = j \\ \sum\limits_{g_s \in G} (2x_i^{(s)} - 1)(2x_j^{(s)} - 1) & \text{otherwise.} \end{cases} \tag{4.5}$$

Our method can use either Eq. (4.4) or (4.5) to define the attractor structure. We mainly use Eq. (4.4) to define the regulatory matrix, as Baram [3] reported that pattern orthogonalization results in higher memory capacity than Hebbian learning. In Sect. 4.4, we describe a method for dynamically reconfiguring the attractor structure to adapt to a dynamic network environment.

4.4 Attractor Structure Design

According to Eq. (4.2), the expression levels converge to one of the attractors stored in the attractor structure defined in Eq. (4.4). Convergence signifies that the proposed method constructs any of the stored virtual networks. One approach to adapting to various changes in the network environment is to store all possible virtual networks as attractors. This approach is, however, impossible due to the memory capacity limitations of the Hopfield network [3, 11]. We therefore must select candidates for virtual networks embedded in an attractor structure.

There are two approaches to overcome the limitation on the number of attractors: The first is to reconfigure the attractor structure dynamically in accordance with the current environment, while the second is to design diverse attractors to be stored in the attractor structure so that they cover various environmental changes. This section presents methods to design virtual networks embedded in an attractor structure as attractors. Section 4.4.1 formulates the problem of designing an attractor structure, while Sect. 4.4.2 proposes a method to reconfigure an attractor structure dynamically. Sections 4.4.3 and 4.4.4 describe other methods for designing diverse attractors.

4.4.1 Problem Formulation

Suppose that we have a physical network consisting of n nodes. Under the assumption that lightpaths can be established between any node pairs, the number of possible lightpaths is approximately n^2. In this case, the number of possible topologies of virtual networks, which is the size of the solution space of virtual network control, is 2^{n^2}. In contrast, the number of attractors stored in an attractor structure, of which dimension is n^2, is approximately 10% of n^2 [3]. The problem is thus to select $0.1n^2$ virtual networks from 2^{n^2} possibilities so that the following objective is satisfied.

The objective of selecting virtual networks is to achieve adaptability to various environmental changes; therefore, the attractor structure must have diverse virtual networks as attractors. The problem of designing attractors thus involves selecting diverse $0.1n^2$ virtual networks so that any of them can accommodate a variety of environmental changes.

4.4.2 Dynamic Reconfiguration of Attractor Structure

It is crucial to develop a method for increasing the number of attractors suitable for the current network environment in the phase space. By incorporating new attractors suitable for the current network environment into the attractor structure, we achieve adaptability to various environmental changes when only a limited number of attractors are stored in the attractor structure.

To achieve this, we update the regulatory matrix W_{ij} in the case that our proposed method finds a virtual network that is suitable for the current network environment and is not a member of G when α is low. Before describing this approach in detail, we present the control flow of our proposed method in Fig. 4.4. For simplicity, we use the terms used in Eqs. (4.2), (4.3), and (4.4) with time t, e.g., the expression level at time t is expressed as $x_i(t)$. In Step 1 of Fig. 4.4, we calculate α from the maximum link utilization at time t, $\alpha(t)$. In Step 2, we update $x_i(t)$ with Eq. (4.2). Next, we convert $x_i(t)$ into the virtual network $g(t)$ and provide it as the network infrastructure of the IP network in Step 3. In Step 4, traffic flows on this virtual network $g(t)$. In Step 5, the resulting flow of traffic determines link utilization $u_i(t)$ on lightpath l_i. Then, we calculate α at time $t+1$, $\alpha(t+1)$, from $u_i(t)$. $\alpha(t+1)$ indicates the quality of virtual network $g(t)$, which is converted from x_i. We can thus determine the quality of virtual network $g(t)$ for the current network environment by observing $\alpha(t+1)$.

We can determine whether the performance on virtual network $g(t)$ is sufficiently high if $\alpha(t+1) > A$, where A is a threshold value. We add $g(t)$ to the set of attractors G if $\alpha(t+1) > A$, $\alpha(t) \leq A$, and $g(t) \notin G$. Then, we update W_{ij} with Eq. (4.4). We add another condition $\alpha(t) \leq A$ to prevent unnecessary updates of W_{ij}. The expression levels $x_i(t)$ always fluctuate due to the constant noise term, η, in Eq. (4.2). Even if α is high and the deterministic behavior dominates the dynamics of x_i over the stochastic behavior, x_i changes slightly; thus, our method sometimes constructs a slightly different virtual network from the already constructed network. Without the condition $\alpha(t) \leq A$, this behavior generates many attractors similar to one of the stored attractors, resulting in a lack of diversity of attractors. Therefore, the condition $\alpha(t) \leq A$ is necessary, and the algorithm adds $g(t)$ as a new attractor to the attractor structure only if the stochastic behavior dominates the dynamics of x_i.

Finally, we describe the deletion of attractors. Because the memory capacity of the Hopfield network is limited, as mentioned above, we cannot keep all of the virtual networks added in the attractor structure. We must delete some of the stored attractors when we add virtual network $g(t)$ as a new attractor. For this purpose, we use the first-in first-out (FIFO) policy for managing attractors when a new attractor is added Although we can use a more sophisticated policy to manage attractors, our method achieves sufficient adaptability with the FIFO policy, as discussed in [17].

4.4.3 Design of Diverse Attractor Structures

This section presents an algorithm for selecting diverse attractors. This algorithm is summarized as follows:

1. Generate virtual networks with the isomorphic graph structure of a given virtual network.
2. Classify the generated virtual networks into groups on the basis of their characteristics.
3. Select representative virtual networks from each group.

The principle behind this algorithm is that virtual networks that are derived from a desirable virtual network have the same desirable characteristics. In our case, desirable virtual network signifies that a virtual network accommodates a given traffic demand with low link utilization. Let us assume that a virtual network g is designed to accommodate a given traffic demand matrix T with a sophisticated algorithm, such as an optimization-based algorithm. In the first step, the algorithm derives isomorphic virtual networks from g assuming that the generated virtual networks accommodate T', which is different from T. Next, the algorithm generates clusters of virtual networks with similar characteristics. Finally, the algorithm selects a representative virtual network from each group and embeds it in the attractor structure as an attractor. The steps of the algorithm are described below.

Step 1. Generation of Isomorphic Virtual Networks The algorithm first generates virtual networks that have an isomorphic graph structure of a given virtual network g. Such virtual networks are referred to as isomorphic virtual networks of g, hereinafter. Figure 4.5 presents an example of isomorphic virtual networks. Virtual network g_1 consists of five nodes N_0, $N_1, \ldots,$ and N_4, and g_2 and g_3 are isomorphic virtual networks of g_1. g_2 is generated by shifting N_0 of g_1 to N_1, N_1 to N_2, N_2 to N_3, N_3 to N_4, and N_4 to N_0. g_3 is also generated from g_1 in the same way. Virtual networks that do not satisfy the constraints on resources in the physical network, such as the number of ports on a router, are eliminated from the candidate pool. In this process, the algorithm generates at most $n!$ virtual networks from a given virtual network.

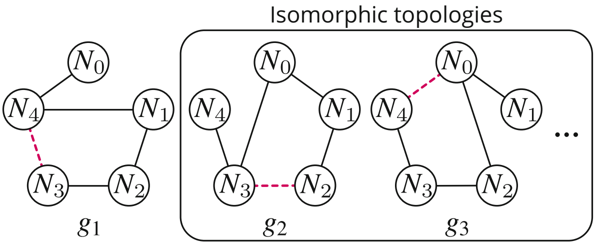


Fig. 4.5 Example of isomorphic virtual networks

Let us assume that a virtual network g_1 is designed to accommodate a given traffic demand matrix T_1 with a sophisticated algorithm, such as an optimization-based algorithm. The load on the link between N_3 and N_4, indicated by a dashed line in Fig. 4.5, is highest but is sufficiently low because g_1 is designed for T_1. Suppose that a traffic demand matrix T_2 is generated by randomly shuffling all the elements in T_1. In this case, at least one of the isomorphic virtual networks, such as g_2, is capable of accommodating T_2 because g_2 is derived from g_1 by shuffling the nodes in g_1. This implies that the isomorphic virtual networks of a well-designed virtual network are capable of accommodating changing traffic demand. In this study, the proposed algorithm uses I-MLTDA [2], which is a heuristic algorithm to design virtual networks for a given traffic demand matrix. However, the algorithm must further reduce the number of candidates for virtual networks because the number of the isomorphic virtual networks derived from g is still too high. Hereinafter, G denotes the set of isomorphic virtual networks, including g.

Step 2. Classify Virtual Networks To further reduce the number of virtual network candidates, the algorithm classifies the isomorphic virtual networks into groups on the basis of the characteristics of the networks and selects one network from each group as a representative virtual network. The algorithm uses edge betweenness centrality, which is the number of shortest paths that pass through a link as a characteristic. The principle behind the use of edge betweenness centrality is that a link with the highest edge betweenness centrality is likely to be a bottleneck link because a larger number of shortest paths travel this link than any other links. The algorithm categorizes virtual networks with the same bottleneck link, i.e., highest edge betweenness centrality link, into the same group, as illustrated in Fig. 4.6 and selects one representative virtual network from each group, thereby generating diverse virtual networks in terms of bottleneck links.

A formal definition of the virtual network classification is summarized as follows:

$$G_p = \{g_i \mid g_i \in G, C(g_i, l_p) = \max_q C(g_i, l_q)\}, \tag{4.6}$$

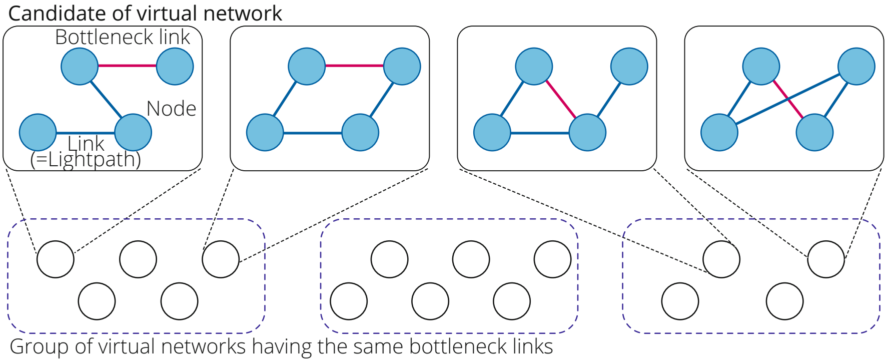

Candidate of virtual network
Bottleneck link
Node
Link
(=Lightpath)
Group of virtual networks having the same bottleneck links

Fig. 4.6 Classification of virtual network candidates

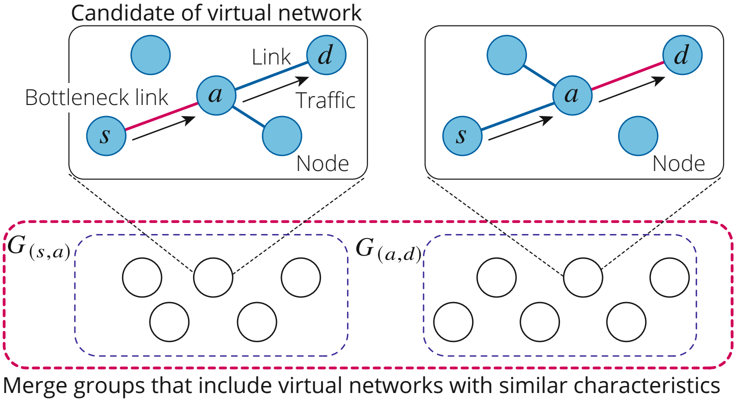

Candidate of virtual network
Link
Bottleneck link
Traffic
Node
Node
s
a
d
$G_{(s,a)}$
$G_{(a,d)}$
Merge groups that include virtual networks with similar characteristics

Fig. 4.7 Merging groups of similar virtual networks

where $p = (s, d)$ denotes a node pair of source node s and destination node d, l_p denotes a link (lightpath) established between node pair p, and $C(g_i, l_p)$ is the value of edge betweenness centrality of l_p in a virtual network g_i. Virtual networks in the group G_p are expected to contain the bottleneck link l_p.

The number of groups is still high compared to the requirement $0.1n^2$. Specifically, the number of groups is at most n^2 because the number of possible lightpaths is n^2. We must therefore reduce the number of groups further, and the algorithm consequently merges the virtual network groups according to the similarity of the location of bottleneck links. Figure 4.7 illustrates the condition of merging virtual network groups. Let us assume that a large amount of traffic flows from node s to d via a whose degree is low, and $l_{(s,a)}$ is the bottleneck of this virtual network. In this case, a large part of the traffic on link $l_{(s,a)}$ also flows on link $l_{(a,d)}$ because the degree of a is low. Thus, the load on $l_{(a,d)}$ is likely to be high. In this sense, virtual network candidates that belong to group $G_{(s,a)}$ and $G_{(a,d)}$ have similar characteristics. In the same way, virtual networks in $G_{(s,d)}$ also have similar characteristics. According to this observation, the algorithm merges the virtual network groups as

$$G_{(s,d)} \leftarrow G_{(s,a)} \cup G_{(a,d)} \cup G_{(s,d)}. \tag{4.7}$$

The algorithm selects three nodes, in this case, a, s, and d, in ascending order of their degree since the correlation of the traffic load on $l_{(s,a)}$ and $l_{(a,d)}$ is high in the case that the degree of a is low. Note that each group has many virtual networks, and the degree of a of the virtual networks differs depending on the network topologies. The algorithm thus selects the three nodes according to the degree of each node averaged over all the virtual networks in the group. The algorithm repeatedly merges virtual network groups, while the number of groups is greater than $0.1n^2$.

Step 3. Selection of Representative Virtual Networks Finally, the algorithm selects a virtual network whose maximum edge betweenness centrality is the lowest among other virtual networks in the group as the representative of each group. The rationale is that the maximum link utilization of a virtual network with low edge betweenness centrality is likely to also be low. The algorithm then embeds the selected virtual networks in the attractor structure.

4.4.4 Scalable Design of Attractor Structure by Graph Contraction

Although the proposed algorithm is able to design diverse attractors, as demonstrated in [22], it has scalability problems due to the high computational complexity of the generation process of isomorphic virtual networks. The number of isomorphic virtual networks for a given network is $n!$, and a commercial off-the-shelf computer, for instance, was able to compute the algorithm for physical networks with up to 10 nodes in a reasonable time in our experiment. This section thus extends the algorithm proposed in Sect. 4.4.3 to design attractors for large-scale physical networks.

The algorithm derives a graph minor of a given physical network to reduce the number of nodes in the physical network and performs the procedures described in Sect. 4.4.3 for the graph minor of the physical network. Instead of contracting edges, the algorithm divides the given physical network into clusters and places edges between the clusters, as illustrated in Fig. 4.8. The edge contraction procedure is outlined as follows:

Step 1 Divide a physical network into clusters.

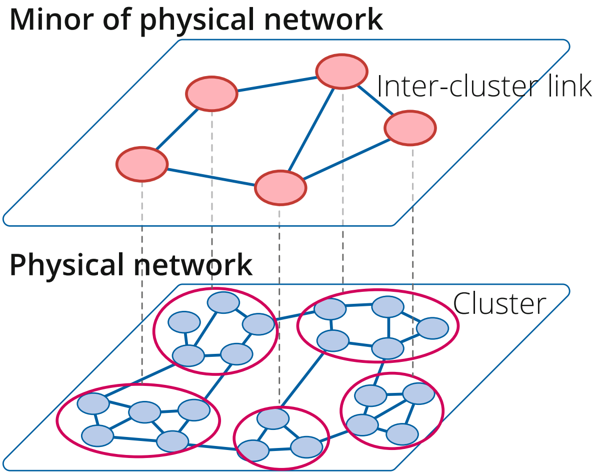


Fig. 4.8 Contraction of physical network topology

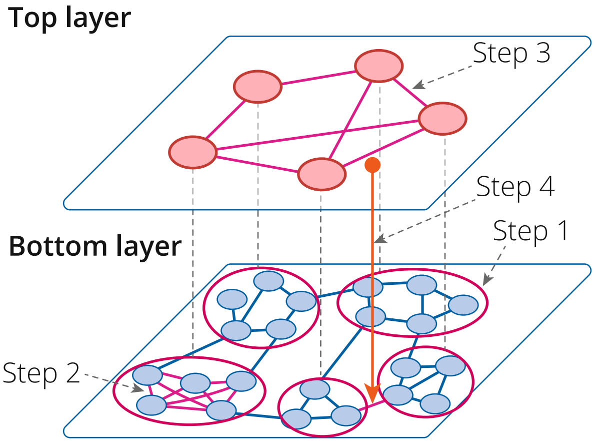


Fig. 4.9 Outline of hierarchical design of attractors

Step 2 Construct virtual network candidates in clusters at the bottom layer.
Step 3 Construct virtual network candidates at upper layers according to the procedures explained in Sect. 4.4.3.
Step 4 Connect lightpaths between clusters to nodes in the clusters.

Figure 4.9 illustrates this procedure. The remainder of this section explains the procedure step by step.

Step 1. Cluster Division of a Physical Network The algorithm divides a given physical network into c clusters. If the number of vertices in a cluster is greater than c, the algorithm divides the cluster into further small clusters recursively until the number of vertices in all clusters is equal to or less than c. This procedure implies that the algorithm creates a hierarchical structure for a given physical network, as illustrated in Fig. 4.9. An upper layer consists of clusters containing nodes in a lower layer.

Step 2. Construction of Virtual Networks Inside Clusters at the Bottom Layer The algorithm starts constructing virtual networks from the bottom layer. In the bottom layer, the algorithm constructs virtual networks possessing either a full-mesh or a star topology inside each cluster so that each topology in the clusters can adapt to changes in a cluster and maintain connectivity in the case of network failure.

Step 3. Construction of Virtual Networks in Upper Layers The algorithm then designs virtual networks in the upper layers according to the procedures described in Sect. 4.4.3. It should be noted that the algorithm does not need to merge virtual network groups. This step prepares at most $c(c-1)/2$ virtual network groups because the algorithm sets up lightpaths bidirectionally to reduce the number of possible virtual networks. The detailed procedure is summarized as follows:

Step 3–1: Construct a virtual network g using an existing heuristic method, i.e., I-MLTDA [2], and generate isomorphic virtual networks from g.

Step 3–2: Categorize the isomorphic virtual networks into at most $c(c-1)/2$ groups on the basis of edge betweenness centrality.
Step 3–3: Select the representative virtual network from each group.

Step 4. Establishment of Links Between Clusters The algorithm finally connects clusters by creating lightpaths. This procedure corresponds to mapping lightpaths in the upper layers to those in the bottom layer. The algorithm establishes lightpaths between clusters from the kth layer to the $(k+1)$th layer, i.e., from an upper layer to a lower layer. The probability of a lightpath $l^k_{i,j}$ being established between u and v ($u \in V^k_i$ and $v \in V^k_j$) is formulated as follows:

$$P_{u,v} = (k_u k_v)^{-1}, \tag{4.8}$$

where $l^k_{i,j}$ represents a lightpath bidirectionally established between C^k_i and C^k_j, C^k_x represents the xth cluster in the kth layer, k_u represents the number of lightpaths connected to node u (the degree of u), and V^k_x represents nodes belonging to C^k_x. Equation (4.8) aims at balancing the traffic load. Because a large amount of traffic is likely to travel via a node with a high degree, the algorithm connects nodes whose degrees are low.

The proposed algorithm described in Sects. 4.3 and 4.4 exhibits high adaptability to both changes in traffic and network failures. Readers can refer to [16, 17, 22] for detailed descriptions of the performance evaluation.

4.5 Related Work

Before concluding this chapter, we briefly introduce engineering-based approaches for recovering from network failures in wavelength-routed WDM networks. Such approaches can be classified into two categories: protection and restoration [32].

With protection approaches, a dedicated backup lightpath for each working lightpath is reserved for recovery from network failures at network design time. Protection approaches generally enable fast recovery from expected network failures [20, 26, 27]. Such approaches, however, cannot handle unexpected network failures because they exploit several assumptions or prior knowledge about network failures and predefine backup lightpaths at network design time. For instance, most protection approaches do not take into account a situation in which both working and backup lightpaths fail simultaneously. Though several protection approaches that enable recovery from dual-link failures have been proposed [28], they also exploit several assumptions or prior knowledge about network failures. Therefore, protection approaches are generally unable to handle unexpected network failures.

In contrast, restoration approaches dynamically reconfigure an alternative virtual network for lightpaths affected by a network failure when the failure occurs [4]. Restoration approaches discover spare resources in the network by collecting network information, such as surviving optical fibers and OXCs, to establish an alternative virtual network. Restoration approaches can maintain the connectivity of virtual networks if they are able to collect network information. However, if they are unable to collect network information changed by the failure, recovery from the failure is not possible.

Protection and restoration approaches, which are based on the control paradigm developed in the field of engineering, take into account a specific class of failures and are optimized to achieve fast and efficient recovery from the assumed failures. Hence, they may not be able to recover from unexpected network failures, such as multiple and series of network failures caused by a disaster. It is difficult for engineering approaches to adapt to various environmental changes as long as they use predefined algorithms. Therefore, the development of a virtual topology control method that adapts to various environmental changes in networks is indispensable.

4.6 Conclusion

This chapter presents a Yuragi-based virtual network control method, which is based on attractor selection found in a gene regulatory network, focusing on the similarity between gene regulatory networks and virtual network control. Unlike most engineering-based virtual network control methods, the proposed method does not rely on predefined algorithms. Instead, it exploits stochastic behavior in the case that the performance of a system is degraded, thereby adapting to unknown changes in the network environment. Since a system driven by attractor selection converges to one of the attractors embedded in its attractor structure, it is a challenge to design attractors to allow the system to adapt to various environmental changes. This chapter proposes two approaches for designing attractors.

Although attractor selection is applied to virtual network control in a wavelength-routed WDM network in this chapter, the proposed approach is applicable to other networks, such as virtualized networks and software-defined networks, which have recently attracted significant attention.

References

1. Arakawa, S., Murata, M., Miyahara, H.: Functional partitioning for multi-layer survivability in IP over WDM networks. IEICE Trans. Commun. **E83-B**(10), 2224–2233 (2000)
2. Banerjee, D., Mukherjee, B.: Wavelength-routed optical networks: Linear formulation, resource budgeting tradeoffs, and a reconfiguration study. IEEE/ACM Trans. Netw. **8**(5) (2000)
3. Baram, Y.: Orthogonal patterns in binary neural networks. Technical Memorandum 100060, NASA (1988)
4. Cheng, X., Shao, X., Wang, Y.: Multiple link failure recovery in survivable optical networks. Photon Netw. Commun. **14**(2), 159–164 (2007)
5. Chowdhury, N.M.M.K., Boutaba, R.: A survey of network virtualization. Comput. Netw. **54**, 862–876 (2010)

6. Comellas, J., Martinez, R., Prat, J., Sales, V., Junyent, G.: Integrated IP/WDM routing in GMPLS-based optical networks. IEEE Netw. Mag. **17**(2), 22–27 (2003)
7. Durán, R.J., Lorenzo, R.M., Merayo, N., de Miguel, I., Fernández, P., Aguado, J.C., Abril, E.J.: Efficient reconfiguration of logical topologies: Multiobjective design algorithm and adaptation policy. In: Proceedings of BROADNETS, pp. 544–551 (2008)
8. Furusawa, C., Kaneko, K.: A generic mechanism for adaptive growth rate regulation. PLoS Comput. Biol. **4**(1), e3 (2008)
9. Gençata, A., Mukherjee, B.: Virtual-topology adaptation for WDM mesh networks under dynamic traffic. IEEE/ACM Trans. Netw. **11**(2), 236–247 (2003)
10. Ghani, N., Dixit, S., Wang, T.S.: On IP-over-WDM integration. IEEE Commun. Mag. **38**(3), 72–84 (2000)
11. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proc. Natl. Acad. Sci. U. S. A. **79**(8), 2554–2558 (1982)
12. Hopfield, J.J.: Neurons with graded response have collective computational properties like those of two-state neurons. Proc. Natl. Acad. Sci. U. S. A. **81**, 3088–3092 (1984)
13. Kaneko, K.: Life: an introduction to complex systems biology. In: Understanding Complex Systems. Springer, New York (2006)
14. Kashiwagi, A., Urabe, I., Kaneko, K., Yomo, T.: Adaptive response of a gene network to environmental changes by fitness-induced attractor selection. PLoS ONE **1**(1), e49 (2006)
15. Kodialam, M., Lakshman, T.V.: Integrated dynamic IP and wavelength routing in IP over WDM networks. In: Proceedings of IEEE INFOCOM, pp. 358–366 (2001)
16. Koizumi, Y., Miyamura, T., Arakawa, S., Oki, E., Shiomoto, K., Murata, M.: Adaptive virtual network topology control based on attractor selection. IEEE/OSA J. Lightwave Technol. **28**(11), 1720–1731 (2010)
17. Koizumi, Y., Arakawa, S., Murata, M.: Nature-inspired computing and optimization: theory and applications. In: Adaptive Virtual Topology Control Based on Attractor Selection. Springer, Berlin (2017)
18. Lee, K., Shayman, M.A.: Rollout algorithms for logical topology design and traffic grooming in multihop WDM networks. In: Proceedings of IEEE Global Telecommunications Conference 2005 (GLOBECOM ’05), vol. 4 (2005)
19. Li, J., Mohan, G., Tien, E.C., Chua, K.C.: Dynamic routing with inaccurate link state information in integrated IP over WDM networks. Comput. Netw. **46**, 829–851 (2004)
20. Lin, T., Zhou, Z., Thulasiraman, K.: Logical topology survivability in IP-over-WDM networks: Survivable lightpath routing for maximum logical topology capacity and minimum spare capacity requirements. In: Proceedings of the International Workshop on the Design of Reliable Communication Networks (2011)
21. Mukherjee, B., Banerjee, D., Ramamurthy, S., Mukherjee, A.: Some principles for designing a wide-area WDM optical network. IEEE/ACM Trans. Netw. **4**(5), 684–696 (1996)
22. Ohba, T., Arakawa, S., Koizumi, Y., Murata, M.: Scalable design method of attractors in noise-induced virtual network topology control. IEEE/OSA J. Opt. Commun. Networking **7**, 851–863 (2015)
23. Rahman, Q., Sood, A., Aneja, Y., Bandyopadhyay, S., Jaekel, A.: Logical topology design for WDM networks using Tabu search. In: Distributed Computing and Networking. Lecture Notes in Computer Science, vol. 7129, pp. 424–427. Springer, Berlin (2012)
24. Ramaswami, R., Sivarajan, K.N.: Design of logical topologies for wavelength-routed optical networks. IEEE J. Sel. Areas Commun. **14**, 840–851 (1996)
25. Rojas, R.: Neural Networks: A Systematic Introduction. Springer, Berlin (1996)
26. Sahasrabuddhe, L., Ramamurthy, S., Mukherjee, B.: Fault management in IP-over-WDM networks: WDM protection versus IP restoration. IEEE J. Sel. Areas Commun. **20**(1), 21–33 (2002)
27. Shen, G., Grover, W.D.: Extending the p-cycle concept to path segment protection for span and node failure recovery. IEEE J. Sel. Areas Commun. **21**(8), 1306–1319 (2003)
28. Sivakumar, M., Sivalingam, K.M.: On surviving dual-link failures in path protected optical WDM mesh networks. Opt. Switch. Netw. **3**(2), 71–88 (2006)

29. Xin, Y., Rouskas, G.N., Perros, H.G.: On the physical and logical topology design of large-scale optical networks. IEEE/OSA J. Lightwave Technol. **21**(4), 904–915 (2003)
30. Ye, T., Zeng, Q., Su, Y., Leng, L., Wei, W., Zhang, Z., Guo, W., Jin, Y.: On-line integrated routing in dynamic multifiber IP/WDM networks. IEEE J. Sel. Areas Commun. **22**(9), 1681–1691 (2004)
31. Zhang, Y., Murata, M., Takagi, H., Ji, Y.: Traffic-based reconfiguration for logical topologies in large-scale WDM optical networks. J. Lightwave Technol. **23**, 1991–2000 (2005)
32. Zhou, D., Subramaniam, S.: Survivability in optical networks. IEEE Netw. **14**(6), 16–23 (2000)