Virtual Network Reconfiguration for Reducing Energy Consumption in Optical Data Centers

Yuya Tarutani, Yuichi Ohsita, and Masayuki Murata

***Abstract*—Energy consumption by data centers has become a serious problem, and measures for its reduction should be developed. Such measures should address not only the energy consumption of the servers, but also that of the network itself because the latter is responsible for a substantial portion of the total energy consumption. One approach to reducing the energy consumption of the network within a data center is to use optical circuit switches (OCSs) at the core of the data center, where electronic switches are connected to the OCSs. In such a network, a virtual network can be configured by setting the OCSs to connect different ports of the electronic packet switches. Thus, the energy consumption of the network can be reduced by configuring the virtual network to minimize the number of ports required by the electronic packet switches and powering down any unused ports. In this paper, we propose a method called VNR-DCN that immediately reconfigures the virtual network so as to reduce the energy consumption under the constraints on the bandwidth and delay between servers in data center networks based on optical communication paths. In VNR-DCN, we configure the virtual network to satisfy the requirements by setting the parameters of the topology, called GFB, instead of solving an optimization problem. In the evaluation, we show that a virtual network configured by VNR-DCN requires a small number of active ports. In addition, we also show the impact of virtual network configuration on energy consumption.**

***Index Terms*—Data Center; Energy Consumption; Virtual Network; Optical Network;**

I. Introduction

In recent years, online services such as cloud computing have become popular, and the amount of data processed by such online services is increasing steadily. To handle such large amounts of data, large data centers with hundreds of thousands of servers have been built. However, the energy consumption of a data center increases as its size increases. Thus, energy-efficient data centers have been discussed [1-3]. Although most previous research has focused on the energy consumption of servers [2] as well as on cooling [3], the contribution of the network itself to the total energy consumption of the data center can be significant, reaching 10–20% of the total energy consumption [4] and increasing as the size and speed of the network increase. Therefore, reducing the energy consumption of data center networks would be essential for constructing energy-efficient data centers [1-3].

Naturally, performance is of paramount importance in data centers. In a data center, servers communicate with each other to handle large amounts of data. Insufficient bandwidth and long delays between servers may lead to suboptimal communication between servers, thus degrading the performance of the entire data center. Therefore, the energy consumption of the network should be reduced without resulting in performance degradation due to insufficient bandwidth or long delays.

There have been many studies on the construction of data center network topologies (e.g., [5-16]). However, no single network topology provides sufficient bandwidth and low latency combined with low energy consumption for arbitrary traffic patterns. When traffic demand is high, a topology providing sufficient bandwidth (e.g., [5, 6]) is required, but such networks generally consume a lot of energy. On the other hand, when traffic demand is low, the network should consume only small amounts of energy, with bandwidth being of secondary importance (e.g., [7]). Because the load of data centers varies with time, the network topology should be reconfigured to suit the instantaneous traffic demand. Accordingly, a method allowing the topology to change dynamically by powering down the ports of switches has been proposed by Heller et al. [17]. In that method, the ports of switches are powered down if sufficient bandwidth can be provided without them. However, that approach powers down only a limited number of ports because it cannot change the network topology drastically, even if there exists a network topology requiring only a small number of ports and able to accommodate the instantaneous traffic demand. Therefore, a notable reduction in energy consumption can be achieved if the network topology can be changed flexibly based on changes in traffic demand.

A data center network architecture that allows for flexible reconfiguration has been proposed by Singla et al. [18]. In their network architecture, the core of the data center network is constructed of one or more *optical circuit switches (OCSs)*. Electronic switches called top-of-rack (ToR) switches are deployed in each server rack, and all servers in the server rack are connected to the ToR switch. ToR switches are connected to OCSs in the core of the data center network by connecting their ports to the OCSs. Then, a virtual network is configured by setting virtual links between the ports of ToR switches. In this network architecture, the virtual network can be reconfigured by adding or removing virtual links. Singla et al. also proposed a method for configuring the virtual network to achieve high throughput by connecting ToR switch pairs that handle large amounts of traffic. Such network topology, enabling flexible reconfiguration of the virtual network, could also be used to reduce energy consumption. In fact, one of the main goals of the current study is to develop a virtual network reconfiguration method based on minimizing the number of

Yuya Tarutani, Yuichi Ohsita, and Masayuki Murata are with the Graduate School of Information Science and Technology, Osaka University, Suita 565-0871, Japan(e-mail: y-tarutn, y-ohsita, murata@ist.osaka-u.ac.jp).

Manuscript received October 7, 2013;

active ports of ToR switches and powering down unused ports without causing performance degradation of the network. This is owing to the fact that the energy consumption of an OCS is much lower than that of a ToR switch.

A virtual network reconfiguration method aiming at reducing the energy consumption of the data center network should satisfy the following requirements.

Requirement 1: The virtual network reconfiguration method should configure a suitable network topology within a short time, even in a large data center. A large data center can include hundreds or more server racks. The optimization problem of constructing the virtual network requires a long calculation time for a large data center, because it requires to find the best combination of $N(N-1)$ binary variables, where N is the number of server racks. Thus, the heuristic method is required.

Requirement 2: Routing should be set up immediately after the virtual network reconfiguration. Even if the virtual network suitable to the current traffic is reconfigured, the new virtual links cannot be used and the performance of the network remains degraded until the routes are updated.

Requirement 3: The virtual network reconfiguration method should consider traffic changes. 100 new flows arrive every millisecond and the traffic pattern may change within a few seconds for the whole of the data center [19, 20], while the total amount of traffic changes gradually and is stable for 10 minutes [17]. The virtual network reconfiguration method should consider both kinds of the traffic changes.

In this paper, we propose *Virtual Network Reconfiguration for Data Center Networks (VNR-DCN)*, which satisfies the above requirements. The VNR-DCN obtains the suitable virtual network by setting the parameters of the *Generalized Flattened Butterfly (GFB)*, which is a new topology proposed in this paper where various types of the network topologies can be reproduced by setting only a small number of parameters, instead of solving the optimization problem. By setting a small number of parameters, we obtain the suitable virtual network topology in a short time.

We also propose a routing method for GFB where routes can be set up in a distributed manner from the GFB parameters, without exchanging any routing information. Thus, routes can be reconfigured immediately after the virtual network reconfiguration. In addition, the routing method for GFB can work with the load balancing methods, and handle the frequent changes in the traffic patterns.

The VNR-DCN sets the parameters of the GFB so as to accommodate the total amount of current traffic, considering the load balancing combined with the routing for GFB. By this approach, the changes of the total amount of traffic are handled by reconfiguring the virtual network, while the frequent changes in traffic pattern are handled by the load balancing over the virtual network without frequent reconfiguration of the virtual network. In this paper, we discuss an implementation of VNR-DCN and show that VNR-DCN works with existing technologies.

Through numerical simulation, we show that the VNR-DCN achieves lower energy consumption than an energy saving method using only electronic switches.

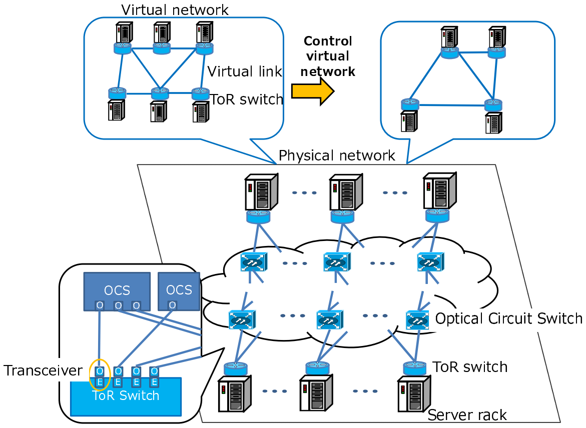


Fig. 1. Data Center Network Using OCSs and ToR switches

The rest of this paper is organized as follows. In Section II, we provide an overview of data center networks containing both OCSs and electronic switches. In Section III, we discuss a virtual network suitable for optical data center networks, and present GFB. In Section IV, we present VNR-DCN, which controls the virtual network by setting the GFB parameters. Also, we discuss the implementation and settings of VNR-DCN in current optical data center networks. In Section V, we evaluate VNR-DCN and clarify that it can ensure sufficiently high performance of communication between servers while achieving low energy consumption. In Section VI, we discuss the scalability of VNR-DCN with the increase of the number of servers or amount of traffic. Finally, Section VII provides a conclusion.

II. Virtual Networks in Optical Data Center Networks

An optical network architecture that allows for reconfiguration of the virtual network in a data center has been proposed in [18]. Following that proposition, in this section, we briefly introduce a data center network architecture that uses OCSs and electronic switches and describe how the energy consumption in such a datacenter network can be reduced.

A. Optical Data Center Network Architecture

Figure 1 shows a data center network architecture where the core of the data center network is constructed of OCSs. Each ToR switch is connected to all servers within the same server rack, as well as to one of the ports of OCSs in the core network through its ports. A *virtual link*, which is considered a directly connected high-capacity link for ToR switches, is established by configuring the OCSs. One approach using the virtual links is to establish the virtual links between all communicating ToR switches. However, this approach has difficulty in handling the traffic changes with a short period, because it requires to reconfigure the OCSs every time the communicating ToR switch pair changes.

Instead, we construct the virtual network, which is formed by the set of the ToR switches. The traffic between server racks is relayed over the virtual network; ToR switches may relay packets from one virtual link to another virtual link according to the final destinations of the packets. In this approach, the virtual network reconfiguration is not necessary even if the communicating ToR switch pairs change unless the amount of the traffic changes significantly. When the amount of the traffic changes and the current virtual network becomes no longer suitable, the virtual network is reconfigured by adding or deleting virtual links.

Singla et al. [18] also proposed a method for configuring virtual networks to achieve high throughput. In that method, virtual links are added between connected server racks that handle large amounts of traffic to maximize the throughput. Such networks are also useful for reducing energy consumption; a virtual network that consumes only a small amount of energy can be dynamically reconfigured to accommodate the instantaneous traffic demand. However, existing research has not considered virtual network reconfiguration aimed at minimizing energy consumption for data center networks. Then, we discuss such a virtual network configuration method in Section IV.

B. *Virtual Network Reconfiguration Method for Reducing Energy Consumption*

The energy consumption of OCSs is much lower than that of ToR switches. For example, the 192-port Glimmerglass OCS [21] consumes less than 85 W of power, while the 48-port Arista 7148SX switch [22] consumes 600 W. Thus, to reduce the energy consumption of the virtual network, the energy consumption of ToR switches should be considered. In this paper, we assume that the ports of ToR switches can be powered down to save energy. Thus, the energy consumption of the network can be reduced by minimizing the number of open ToR switch ports used in the virtual network, and powering down any unused ports if sufficiently large bandwidth and low delay can be ensured. The virtual network with the minimum required number of open ports can be obtained as a solution to an optimization problem. However, optimization problems require considerable time to be solved for large networks.

Therefore, in this paper, we propose a new method called *VNR-DCN*, which can be used to configure a suitable virtual network that achieves a sufficiently large bandwidth and low latency with a small number of open ports. In VNR-DCN, we configure a suitable virtual network by setting the parameters of the base topology called GFB instead of solving an optimization problem. The complexity of VNR-DCN in setting up the GFB parameters is $\mathcal{O}(N|K_{\mathrm{can}}|)$, where N is the number of ToR switches and K_{can} is the set of candidates for the number of layers in GFB (typically two or three, as shown in Subsection IV-B3). In the next section, we discuss the base topology used in our virtual network configuration, before proceeding to explain the method for setting the parameters of the base topology in Section IV.

III. Virtual Network Topologies Suitable for Optical Data Center Networks

In this section, we discuss the requirements on the base topology used in our virtual network reconfiguration method and investigate the properties of existing network topologies for data center networks. Then, we present GFB, whose parameters can be adjusted to construct various data center network topologies. GFB is an extension of FB [6]. Although FB was originally proposed as a topology for interconnection networks of multiprocessors, Abts et al. [4] found that data center networks constructed using FB provides sufficient bandwidth with lower energy consumption than ones constructed using the FatTree topology because the number of required switches is smaller than that in FatTree. However, FB requires a large number of links, while a topology with a small number of links is suitable for low traffic when considering energy consumption. In this paper, first we extend FB and introduce the layers and parameters indicating the number of links in each layer. By setting an appropriate number of links for each layer, we construct an energy-efficient network topology that uses only the number of links required to accommodate the instantaneous traffic demand.

A. *Properties of an Optimal Virtual Network for a Data Center*

An optimal virtual network for a data center has the following properties.

a) Low Energy Consumption: The energy consumption of the network is responsible for a non-negligible fraction of the total energy consumption in the data center, as mentioned above. In a data center network consisting of optical switches, most of the energy is consumed by ToR switches. Therefore, the number of active ToR switches should be minimized. That is, in an optimal network topology, the only active ToR switches are those connected to servers communicating with other servers. Moreover, the energy consumption of ToR switches can be reduced by powering down any unused ports. Thus, the energy consumption of a data center network can be minimized by constructing a virtual network using the smallest possible number of active ToR switch ports.

b) Large Bandwidth between Servers: In some applications, such as distributed file systems, large amounts of data are exchanged between servers. The bandwidth between servers is thus important for such applications. Therefore, the virtual network should provide sufficient bandwidth for communication between servers. To ensure sufficient bandwidth, virtual links should be added to accommodate the instantaneous traffic demand without congestion.

c) Short Delay between Servers: Data centers handle large amounts of data by using distributed computing frameworks [23, 24], where a large number of servers communicate with each other. If inter-server communication experiences long delays, it takes time to obtain the required data from other servers, which degrades the performance of the entire data center. Thus, the delay should be kept sufficiently low for the purposes of the particular data center.

However, delays in communication between servers are difficult to predict when constructing a virtual network because delays are affected by traffic load. Therefore, in this study we minimize the delay in communication between servers by constructing a virtual network with a small number of hops between servers.

B. Existing Data Center Network Topologies

Several data center network topologies have already been proposed. Although they apply to physical networks constructed of electronic switches and servers, they may be used as base topologies for virtual networks because the objective here is to change the topology by setting its parameters. In this subsection, we discuss how such existing data center network topologies can be used as a base for our virtual network reconfiguration method.

Al-Fares et al. proposed a topology construction method called *FatTree* by using switches with a small number of ports [5]. FatTree is a tree topology constructed of multiple roots and pods containing aggregation switches. Each pod is regarded as a switch having a large number of ports consisting of multiple switches having a small number of ports. Pods are constructed using the butterfly topology, where each switch uses half of its ports to connect it to switches close to root switches, and the other half to connect it to switches close to leaf switches. Leaf switches are connected to servers. The number of switches in FatTree depends on the depth of the tree and the number of ports on each switch. FatTree with a depth of k constructed of switches with n ports each includes $(2k-1)\frac{n}{2}^{k-1}$ switches. In FatTree, the number of links from a switch close to a leaf switch equals the number of links to a switch close to a root switch; this is true for each switch. That is, the total bandwidth from a switch to switches close to a root switch equals that from switches close to a leaf switch to that switch. Therefore, none of the switches become bottlenecks, and a sufficiently large bandwidth is provided between all servers. However, FatTree is not suitable for virtual networks configuration because switches, except for leaf switches, are not connected to servers. In other words, ToR switches that are not connected to servers must be powered on, which results in high energy consumption.

Kim et al. [6] proposed the FB data center network topology. FB is constructed by *flattening* the butterfly topology, where switches in each row of the butterfly topology are combined into a single switch. FB provides a sufficiently large bandwidth between all servers while reducing the energy consumption compared to FatTree [5]. In addition, all switches in FB are connected to servers. Thus, unlike FatTree, all ToR switches that are not connected to any working servers can be powered down if FB is constructed as a virtual network. However, FB requires switches with a large number of ports to construct a large data center network even if traffic demand is small. Thus, FB is not preferred when there is low traffic demand.

Guo et al. proposed a data center network topology called *DCell*, which is constructed from a small number of switches and servers with multiple ports [7]. DCell uses a recursively defined structure; the level-0 DCell is constructed by connecting one switch with n ports to n servers, and the level-k DCell is constructed by connecting servers belonging to different level-(k-1) DCells. By directly connecting server ports, the DCell topology reduces the number of switches required to construct a large data center network. However, in the optical data centers introduced in Section II, only ToR switches are connected to the OCSs, and virtual links are added between the ToR switches, and virtual links that connect servers directly cannot be added. Therefore, we can extend DCell by replacing a level-0 DCell with a switch. We call this topology *switch-based DCell*. Similar to DCell, switch-based DCell can be used to construct a large data center network by using switches with a small number of ports. That is, switch-based DCell achieves low energy consumption. However, switch-based DCell cannot provide a large bandwidth between all servers, because it has only one link between lower-level DCells.

TABLE I
COMPARISON OF EXISTING TOPOLOGIES

Topology	Energy consumption	Bandwidth
FatTree [5]	Very high	Sufficiently large
FB [6]	High	Sufficiently large
Switch-based DCell	Low	Small

Table I summarizes the properties of the data center network topologies outlined above. Switch-based DCell consumes only a small amount of energy and is suitable for situations where the traffic volume between servers is small. However, it may not provide sufficient bandwidth for applications where servers generate a large amount of traffic. Furthermore, FB can provide large bandwidth between servers but consumes a lot of energy. That is, the most suitable network topology depends on the traffic demand. Accordingly, we propose GFB, in which we can reproduce various topologies, including FB and switch-based DCell, by adjusting the GFB parameters. By using GFB, we can set the parameters so as to provide sufficient bandwidth (similarly to FB) when the traffic amount is large and to reduce the number of links to the appropriate minimum when the traffic amount is small.

C. GFB Topology

In this subsection, we explain GFB in detail. GFB is constructed hierarchically as shown Figure 2, where the upper-layer GFB is constructed by connecting multiple lower-layer GFBs. GFB has the following parameters.

- Number of layers: $K_{\max}$
- Number of links per switch used to construct layer-k GFB: L_k
- Number of layer-(k-1) GFBs used to construct layer-k GFB: N_k

We can construct various topologies, including FB and switch-based DCell, by adjusting these parameters. We can construct GFB as a physical network or a virtual network. In this paper, we propose GFB as the base topology used for our virtual network configuration method. Thus, in this subsection, we describe the generic structure of GFB and use GFB to construct a virtual network.

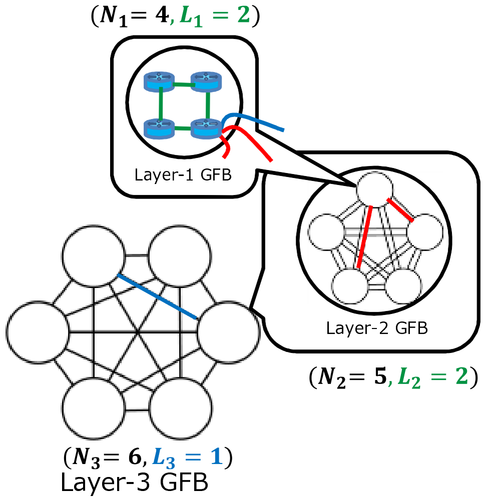


Fig. 2. GFB topology

These parameters are changed periodically so as to provide sufficient bandwidth for the performance of applications and to satisfy the requirements on the network set by the data center administrator. The interval for updating the parameters is not necessarily short, even though the traffic pattern can change rather frequently, e.g., every few seconds. This is because the total amount of traffic changes gradually, and frequent changes in traffic are absorbed by a load balancing technique in the routing method instead of by reconfiguring the virtual network. For example, in the evaluation discussed in Section V, we use a 10-min interval between reconfigurations, following Heller et al. [17].

In the rest of this subsection, we describe the steps for constructing a virtual network using GFB in Paragraph III-C1. Then, we explain the routing method for GFB, which uses the GFB parameters, in Paragraph III-C2. Finally, in Paragraph III-C3, we discuss the properties of GFB based on its parameters.

1) Steps to Construct GFB: GFB is constructed hierarchically by constructing each GFB layer in order from layer-1 GFB to layer-$K_{\max}$ GFB. Layer-k GFB is constructed by the following two steps.

Step I Construct connections between all layer-(k-1) GFBs.

Step II Select switches connected to the links between each two layer-(k-1) GFBs.

In these steps, we use the IDs assigned to GFBs in each layer. A switch can be identified by the ID of the GFB it belongs to. We denote the ID of layer-k GFB to which switch s belongs to as $D^{\mathrm{GFB}}_k(s)$. We also define $D^{\mathrm{sw}}(s)$ for the ID of switch s in layer-k GFB as

$$D^{\mathrm{sw}}(s) = \sum_{1 \le i \le K_{\max}} \left(D^{\mathrm{GFB}}_i(s) \prod_{j=1}^{i-1} N_j \right).$$

We provide details about Steps I and II in Paragraphs III-C1a and III-C1b, respectively.

a) Connections between layer-(k-1) GFBs: In Step I, we select connections between layer-(k-1) GFBs to construct layer-k GFB. We construct the connections between layer-(k-1) GFBs by the following steps.

Step I-1 Compute the number of links L^{GFB}_k necessary to connect one layer-(k-1) GFB to other layer-(k-1) GFBs by

$$L^{\mathrm{GFB}}_k = L_k \prod_{i=1}^{k-1} N_i. \tag{1}$$

Step I-2 If L^{GFB}_k is larger than (N_k-1), connect all layer-(k-1) GFBs. Otherwise, construct a ring topology by connecting GFBs with neighboring IDs.

Step I-3 Compute the number of residual links L'^{GFB}_k which can be used to connect one layer-(k-1) GFB to other layer-(k-1) GFBs. This number is given as

$$L'^{\mathrm{GFB}}_k = L^{\mathrm{GFB}}_k - \bar{L}^{\mathrm{GFB}}_k, \tag{2}$$

where $\bar{L}^{\mathrm{GFB}}_k$ is the number of links per layer-(k-1) GFB constructed at Step I-2.

Step I-4 Check whether layer-(k-1) GFBs have any residual links which can be used to connect layer-(k-1) GFBs. If there are residual links, connect GFB of ID $D^{\mathrm{GFB}}_{k-1}(a)$ to GFB of ID $D^{\mathrm{GFB}}_{k-1}(b)$ when the following equation is satisfied:

$$D^{\mathrm{GFB}}_{k-1}(b) = (D^{\mathrm{GFB}}_{k-1}(a) + \lceil p_k \rceil + C^{\mathrm{r}}_k \lfloor p_k \rfloor) \bmod N_k, \tag{3}$$

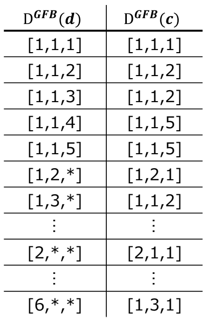
where C^{r}_k (C^{r}_k = 0,1,$\cdots$,L'^{GFB}_k-1) is an integer value that represents the number of residual links used for connections. Furthermore, p_k is a variable that represents the interval containing the IDs of layer-(k-1) GFBs connected to the same layer-(k-1) GFBs. p_k is obtained by

$$p_k = \frac{N_k}{L'^{\mathrm{GFB}}_k + 1}. \tag{4}$$

b) Selection of Switches for Connecting layer-(k-1) GFBs: In Step II, we select the switches used for connecting layer-(k-1) GFBs after constructing the connections between layer-(k-1) GFBs. Switch $D^{\mathrm{sw}}_{\mathrm{connect}}(s)$ included in GFB of ID $D^{\mathrm{GFB}}_{k-1}(a)$ is connected to GFB of ID $D^{\mathrm{GFB}}_{k-1}(b)$ when the following condition is satisfied:

$$D^{\mathrm{sw}}_{\mathrm{connect}}(s) = D^{\mathrm{GFB}}_{k-1}(b) + \left\lfloor \frac{C_k(s) n_{D^{\mathrm{GFB}}_{k-1}(a)}}{l_{(D^{\mathrm{GFB}}_{k-1}(a), D^{\mathrm{GFB}}_{k-1}(b))}} \right\rfloor$$

where $C_k(s)$ ($C_k(s)$=0,1,$\cdots$,L_k-1) is an integer value that represents that the number of links used for connections between GFB of ID $D^{\mathrm{GFB}}_{k-1}(a)$ and GFB of ID $D^{\mathrm{GFB}}_{k-1}(b)$, $n_{D^{\mathrm{GFB}}_{k-1}(a)}$ is the number of switches in GFB of ID $D^{\mathrm{GFB}}_{k-1}(a)$, and $l_{(D^{\mathrm{GFB}}_{k-1}(a), D^{\mathrm{GFB}}_{k-1}(b))}$ is the number of links to be constructed between GFBs of IDs $D^{\mathrm{GFB}}_{k-1}(a)$ and $D^{\mathrm{GFB}}_{k-1}(b)$. By connecting switches using the above condition, the intervals containing IDs of switches connected to the same GFB become constant, and we can avoid a large number of hops from any switch belonging to a GFB to other GFBs.



$D^{GFB}(d)$	$D^{GFB}(c)$
[1,1,1]	[1,1,1]
[1,1,2]	[1,1,2]
[1,1,3]	[1,1,2]
[1,1,4]	[1,1,5]
[1,1,5]	[1,1,5]
[1,2,*]	[1,2,1]
[1,3,*]	[1,1,2]
⋮	⋮
[2,*,*]	[2,1,1]
⋮	⋮
[6,*,*]	[1,3,1]

Fig. 3. Routing table for a ToR switch belonging to the GFB of ID [1,1,1]

2) GFB Routing: In GFB, routes are established based on GFB IDs. Below, we call routing for GFB *GFB routing*. In GFB routing, packets to a destination ToR switch belonging to a layer-k GFBs different from that of the source ToR switch, are sent via the following path.

Step R-1 Select the ToR switch that belongs to the same GFB as the source switch and is connected to the destination GFB. This ToR switch is obtained from the GFB parameters. If multiple ToR switches are connected to the destination GFB, select one of them randomly.

Step R-2 Encapsulate the packet so that its destination becomes the selected ToR switch, and then relay the packet.

Step R-3 The ToR switch with the ID specified in the encapsulated packet decapsulates the packet.

By repeating these steps, we can relay the packet to the destination ToR switch.

The above routing can be implemented as a routing table. Because GFB is constructed in a hierarchical manner, the routing table is also hierarchically aggregated. An example routing table is shown in Figure 3. In the figure, $D^{\mathrm{GFB}}(c)$ represents a destination address of an encapsulated packet for a packet to $D^{\mathrm{GFB}}(d)$.

The routing table can be set by the following steps. First, entries in the routing table for ToR switches in the same layer-1 GFB are set up by a shortest-hop-count routing algorithm, such as the Dijkstra algorithm. Because the number of ToR switches in each layer-1 GFB is small, this configuration takes only a short time. Then, the entries for the different layer-k (k=2,3,$\cdots$,$K_{\max}$) GFBs are set as follows: if the switch is connected to a ToR switch in the destination GFB, the entry is set to the ToR switch in the destination GFB. Otherwise, the entry is set to represent the encapsulation of the packet with the ID of a ToR switch connected to the destination GFB. The calculation time for the routing table in GFB routing is $\mathcal{O}(\sum_{1<=i<=K_{\max}} N_i)$, which is significantly shorter than $\mathcal{O}(N)$.

3) Properties of GFB: In GFB, the maximum number of hops or the number of paths passing through each link can be obtained from the GFB parameters as described below.

a) Maximum Number of Hops: The maximum number of hops H_k between switches in layer-k GFB is obtained by

$$H_k = (h_k + 1)H_{k-1} + h_k, \tag{5}$$

where h_k is the largest number of links between layer-(k-1) GFBs passed through by the traffic between layer-(k-1) GFBs. h_k is obtained by the following steps. If all layer-(k-1) GFBs are connected directly, then $h_k = 1$. In all other cases, we obtain h_k by computing the largest number of links between layer-(k-1) GFBs passed through by the traffic from the source layer-(k-1) GFB whose ID is 0, because all GFBs are equivalent. From the viewpoint of the source GFB, the topology constructed of layer-(k-1) GFBs is a ring topology where some shortcut links are added directly from the source GFB. To obtain h_k, we divide the set of GFBs which are not directly connected to the source GFB into groups so that the shortcut links from the source GFB become the border of the group. Here, m_j denotes the set of switches within the j-th group, and M denotes the set of all groups. Based on the steps required to construct the connections between layer-(k-1) GFBs, $|m_j|$ is obtained by

$$|m_j| \quad = \quad \begin{cases} \lceil p_k \rceil - 3, & \text{if } j = 1 \text{ or } |M|, \\ \lfloor p_k \rfloor - 2, & \text{otherwise.} \end{cases} \tag{6}$$

GFBs in each group form a ring topology. Thus, the maximum number of links passed through by traffic from the source GFB or from the GFB belonging to group m_j is obtained by $\left\lceil \frac{|m_j|+2}{2} \right\rceil$. Since at least one group includes GFBs for which the number of hops from the source GFB is the largest, h_k reaches a maximum of $\left\lceil \frac{|m_j|+2}{2} \right\rceil$ for all groups. That is,

$$h_k = \begin{cases} 1, & \text{if } L_k^{\mathrm{GFB}} \geq (N_k - 1), \\ \lceil \frac{p_k}{2} \rceil, & \text{if } L_k^{\mathrm{GFB}} < (N_k - 1) \text{ and } L_k^{'\mathrm{GFB}} \leq 1, \\ \lceil \frac{\lfloor p_k \rfloor + 2}{2} \rceil, & \text{otherwise.} \end{cases} \tag{7}$$

b) Number of Flows through a Link: The number of layer-(k-1) GFB source-destination pairs whose traffic passes through link l between layer-(k-1) GFBs (denoted as x_l^k) is obtained by calculating the number of flows passing through the link in an abstracted topology where layer-(k-1) GFB is regarded as a single node. Multiplying that number by the number of flows passing through layer-(k-1) GFBs, we obtain the number of flows passing through each link. Since all layer-k GFBs are equivalent, the number of flows between a pair of layer-(k-1) GFBs is independent of the GFB ID.

Thus, the number of flows X_l^k passing through link l between layer-(k-1) GFBs is obtained by

$$X_l^k = F_k x_l^k, \tag{8}$$

where F_k is the number of flows from a layer-(k-1) GFB to other layer-(k-1) GFBs. x_l^k and F_k can be obtained as follows. In an abstracted topology where lower-layer GFBs are regarded as single nodes, there are two types of links: links in the ring topology (*ring links*), and links added as shortcuts in the ring topology (*shortcut links*). Since all layer-(k-1) GFBs are equivalent in layer-k GFB, the number of flows passing through each ring link is independent of the GFBs connected to the link. Similarly, the number of flows passing through each shortcut link is also independent of GFBs connected to the link. Therefore,

$$x_l^k = \begin{cases} \frac{M_k^{\mathrm{ring}}}{2\prod_{i=1}^{k} N_i}, & \text{if } l \text{ is a ring link,} \\ \frac{M_k^{\mathrm{shortcut}}}{(L_k-2)\prod_{i=1}^{k} N_i}, & \text{if } l \text{ is a shortcut link,} \end{cases} \quad (9)$$

where M_k^{ring} is the total number of ring links passed through by traffic between layer-(k-1) GFB source-destination pairs, and M_k^{shortcut} is the total number of shortcut links passed through by traffic between layer-(k-1) GFB source-destination pairs. $2\prod_{i=1}^{k} N_i$ is the number of ring links between layer-(k-1) GFBs, and $(L_k-2)\prod_{i=1}^{k} N_i$ is the number of shortcut links between layer-(k-1) GFBs.

Traffic between layer-(k-1) GFBs passes through at most one shortcut link because the portion of GFBs connected to a certain GFB is constant. The number of flows that do not pass through a shortcut link is $2h_k\prod_{i=1}^{k} N_i$. Thus,

$$M^{\mathrm{shortcut}} = \prod_{i=1}^{k} N_i(\prod_{i=1}^{k} N_i - 1) - 2h_k\prod_{i=1}^{k} N_i.$$

In addition, M^{ring} is obtained by subtracting M^{shortcut} from the total number of links passed through by traffic between layer-(k-1) GFBs;

$$M^{\mathrm{ring}} = \sum_{i=1}^{h_k} i s_k(i) - M^{\mathrm{shortcut}},$$

where $s_k(i)$ is the number of layer-(k-1) GFB source-destination pairs whose traffic passes through i links in the abstracted topology.

$s_k(i)$ is obtained as follows. $s_k(1)$ has the same value as the number of links in layer-k GFB. That is,

$$s_k(1) = \begin{cases} N_k(N_k-1), & \text{if } L_k^{\mathrm{GFB}} \geq (N_k-1), \\ N_k L_k \prod_{i=1}^{k-1} N_i, & \text{otherwise.} \end{cases} \quad (10)$$

$s_k(i)$ for $i > 1$ is obtained by dividing the topology constructed of layer-(k-1) GFBs into groups, similar to the case of calculating h_k. By dividing the topology, $s_k(i)$ is obtained as the sum of the number of layer-(k-1) GFBs located i hops away from the source layer-(k-1) GFB in each group. In other words,

$$s_k(i) = N_k \sum_{m_j \in M} U_{(k,m_j)}(i), \quad (11)$$

where $U_{(k,m_j)}(i)$ is the number of layer-(k-1) GFBs located i hops away from the source layer-(k-1) GFB in group m_j. For GFBs in each group, the source GFB and GFBs directly connected to the source GFB form a ring topology,

$$U_{(k,m_j)}(i) = \begin{cases} 0, & \text{if } i > \left\lceil \frac{m_j+2}{2} \right\rceil, \\ 1, & \text{if } i = \left\lceil \frac{m_j+2}{2} \right\rceil \text{ and } |m_j| \text{ is odd,} \\ 2, & \text{otherwise.} \end{cases} \quad (12)$$

We determine the number of flows between each two layer-(k-1) GFBs (denoted as F_k), which is independent of the IDs of the source and destination GFBs. Thus, we determine the number of flows between layer-(k-1) GFBs s and d (denoted as $F_k^{s\to d}$) as follows:

$$\begin{aligned} F_k^{s\to d} &= f_k^{s\to s\to d\to d} + \sum_{n\in G} f_k^{n\to s\to d\to d} \\ &+ \sum_{n\in G} f_k^{s\to s\to d\to n} + \sum_{n_1,n_2\in G} f_k^{n_1\to s\to d\to n_2}, \end{aligned} \quad (13)$$

where $f^{a\to b\to c\to d}$ is the number of flows whose source and destination switches belong to layer-(k-1) GFBs a and d, respectively, and that traverse layer-(k-1) GFBs b and c. G is the set of switches that do not belong to layer-k GFB including layer-(k-1) GFBs s and d. $f_k^{s\to s\to d\to d}$ is obtained as the product of the respective numbers of switches in layer-(k-1) GFBs s and d. That is,

$$f_k^{s\to s\to d\to d} = \prod_{i=1}^{k-1}(N_i)^2. \quad (14)$$

$\sum_{n\in G} f_k^{s\to s\to d\to n}$ shows the number of flows from layer-(k-1) GFB s to the outside of layer-k GFB via layer-(k-1) GFB d. Because all layer-(k-1) GFBs are equivalent in GFB, $\sum_{n\in G} f_k^{s\to s\to d\to n}$ is obtained by dividing the number of flows whose source and destination switches belong to layer-(k-1) GFB s and a different layer-k GFB, respectively, by the number of layer-(k-1) GFBs in layer-k GFB.

$$\sum_{n\in G} f_k^{s\to s\to d\to n} = \frac{(\prod_{i=1}^{k-1} N_i)(\prod_{i=1}^{K_{\max}} N_i - \prod_{i=1}^{k} N_i)}{N_k}. \quad (15)$$

Similarly, $\sum_{n\in G} f_k^{n\to s\to d\to d}$ is obtained by

$$\sum_{n\in G} f_k^{n\to s\to d\to d} = \frac{(\prod_{i=1}^{k-1} N_i)(\prod_{i=1}^{K_{\max}} N_i - \prod_{i=1}^{k} N_i)}{N_k}. \quad (16)$$

$\sum_{n_1,n_2\in G} f_k^{n_1\to s\to d\to n_2}$ shows the number of flows that travel from the outside of layer-k GFB via layer-(k-1) GFB s to the outside of layer-k GFB via layer-(k-1) GFB d. The number of flows traveling from the outside of layer-k GFB via layer-(k-1) GFB s is the sum of flows through links that connect switches in layer-(k-1) GFB s and switches outside layer-k GFBs, which is obtained by

$$\prod_{j=1}^{k-1} N_j \sum_{i=k+1}^{K} (X_l^i L_i). \quad (17)$$

We finally obtain the number of flows that travel from the outside of layer-k GFB via layer-(k-1) GFB s to layer-(k-1) GFB d by dividing Eq. (17) by the number of layer-(k-1) GFBs in layer-k GFB. The resulting value includes the flows whose destination switches belong to layer-(k-1) GFB d, whose number is $\sum_{n_1\in G} f_k^{n_1\to s\to d\to d}$. Therefore, $\sum_{n_1,n_2\in G} f_k^{n1\to s\to d\to n_2}$ is obtained by

$$\begin{aligned} \sum_{n_1,n_2\in G} f_k^{n_1\to s\to d\to n_2} &= \frac{\prod_{j=1}^{k-1} N_j \sum_{i=k+1}^{K}(X_l^i L_i)}{N_k} \\ &\quad - \sum_{n_1\in G} f_k^{n_1\to s\to d\to d}. \end{aligned} \quad (18)$$

c) Difference from FB: GFB is an extension of FB, and FB is obtained from GFB if we set L_k = N_k - 1 and set N_k the same for all layers. In FB, we cannot set the number of links independently from the number of nodes N_k. Thus, if the number of nodes in the data center is large, N_k should be large, requiring a large number of links even when the traffic amount is small. In GFB, the parameter L_k can be set independently from the other parameters for each layer. Thus, we construct a topology where only links required to accommodate the instantaneous traffic are established.

IV. Virtual network reconfiguration for data center networks

In this section, we propose a method for adjusting the topology of a virtual network to meet the requirements for minimizing the energy consumption of data center networks. In VNR-DCN, the virtual network is constructed by calculating the GFB parameters to satisfy these requirements.

A. Overview

The VNR-DCN considers the both types of the traffic changes; frequent changes in traffic pattern and gradual changes of the total amount of traffic. The VNR-DCN reconfigures the virtual network to handle the changes of the total amount of traffic, while the frequent changes of the traffic pattern are handled by the load balancing over the virtual network.

1) Virtual Network Control: In VNR-DCN, the virtual network and the traffic routing are adjusted by a single network controller (NC) and multiple route controllers (RCs). NC ensures that the requirements on the network set by the data center administrator are met, and collects information about the amount of traffic from or to ToR switches. Then, NC sets the GFB parameters to satisfy the current requirements on the network and to accommodate the instantaneous traffic demand, after which it configures the OCSs and sends the GFB parameters to RCs. Each server rack hosts a RC. When the virtual network is reconfigured, RCs set the routing rules for the ToR switch in the same rack based on the GFB parameters sent by NC.

In VNR-DCN, NC reconfigures the virtual network by the following steps as shown in Figure4.

Step VN-1 Collect traffic information and set the GFB parameters to satisfy the requirements.

Step VN-2 Configure OCSs to add virtual links that are not included in the current virtual network but are included in the virtual network with the new GFB with the parameters set in the previous step.

Step VN-3 Send the GFB parameters to RCs, after which RCs update the routing table for ToR switches.

Step VN-4 Wait for a notification of the completion of routing update from the RCs.

Step VN-5 Configure OCSs to delete unused virtual links.

In the above procedure, the routing tables are updated by the method described in Section III-C2.

2) Load balancing over the Virtual Network: In VNR-DCN, we use a load balancing technique called valiant load balancing (VLB) [25] combined with the GFB routing. In VLB, we select the intermediate switches randomly, regardless of the destination, in order to avoid the concentration of traffic at any particular links, even when the traffic amount flowing between a certain pair of switches is large. Then, traffic is sent from the source switch to the destination switch via an intermediate switch. The VLB is combined with the GFB routing as follows; (1) Each server encapsulates the packets from it with the addresses of the randomly selected ToR switches, and (2) the encapsulated packets are relayed by the GFB routing.

By applying VLB, the amount of traffic between each ToR switch pair T is obtained by the following equation:

$$T \leq \frac{T^{\mathrm{toSW}} + T^{\mathrm{fromSW}}}{N_{\mathrm{all}}}, \tag{19}$$

where T^{toSW} is the maximum traffic amount to a ToR switch, T^{fromSW} is the maximum traffic amount from a ToR switch, and N_{all} is the number of ToR switches in the virtual network. Thus, we can ensure sufficient bandwidth by setting the virtual network so that the number of flows passing a link is smaller than a certain threshold obtained by dividing the capacity of a link by the traffic amount between each switch pair calculated from Eq. (19). The rest of this section explains how NC set the GFB parameters considering the load balancing.

B. GFB Parameter Setup

1) Outline: We propose a method to set GFB parameters so as to minimize the number of used ports by considering two requirements: large bandwidth and short delay between servers. When using VLB, based on Eq. (19), the traffic amounts between ToR switch pairs depends only on the total amount of traffic from or to ToR switches. Delays are also difficult to predict when designing a virtual network. In this study, we avoid long delays by providing sufficient bandwidth and ensuring that the maximum number of hops does not exceed a certain threshold.

2) Steps to Set Suitable GFB Parameters: In this subsection, we describe a method to set up the GFB parameters so as to minimize the number of used ports and to satisfy the requirements on bandwidth and maximum number of hops between servers. In VNR-DCN, the GFB parameters are set based on the number of switches connected in the virtual network (N_{all}), the maximum number of hops ($H_{\max}$), the maximum traffic amount from a ToR switch (T^{fromSW}), and the maximum traffic amount to a ToR switch (T^{toSW}) by the following steps.

First, we obtain the candidates for the number of layers. Because the maximum number of hops in GFB cannot be smaller than 1 (Eq. (5)) for any layer, to take the maximum number of hops to be no more than $H_{\max}$, and the number of layers ($K_{\max}$) must satisfy the following condition.

$$2^{K_{\max}} - 1 \leq H_{\max}. \tag{20}$$

We define K_{can} as the set of numbers of layers satisfying Eq.(20). We consider all $K_{\max}$ ($K_{\max} \in K_{\mathrm{can}}$) as candidates

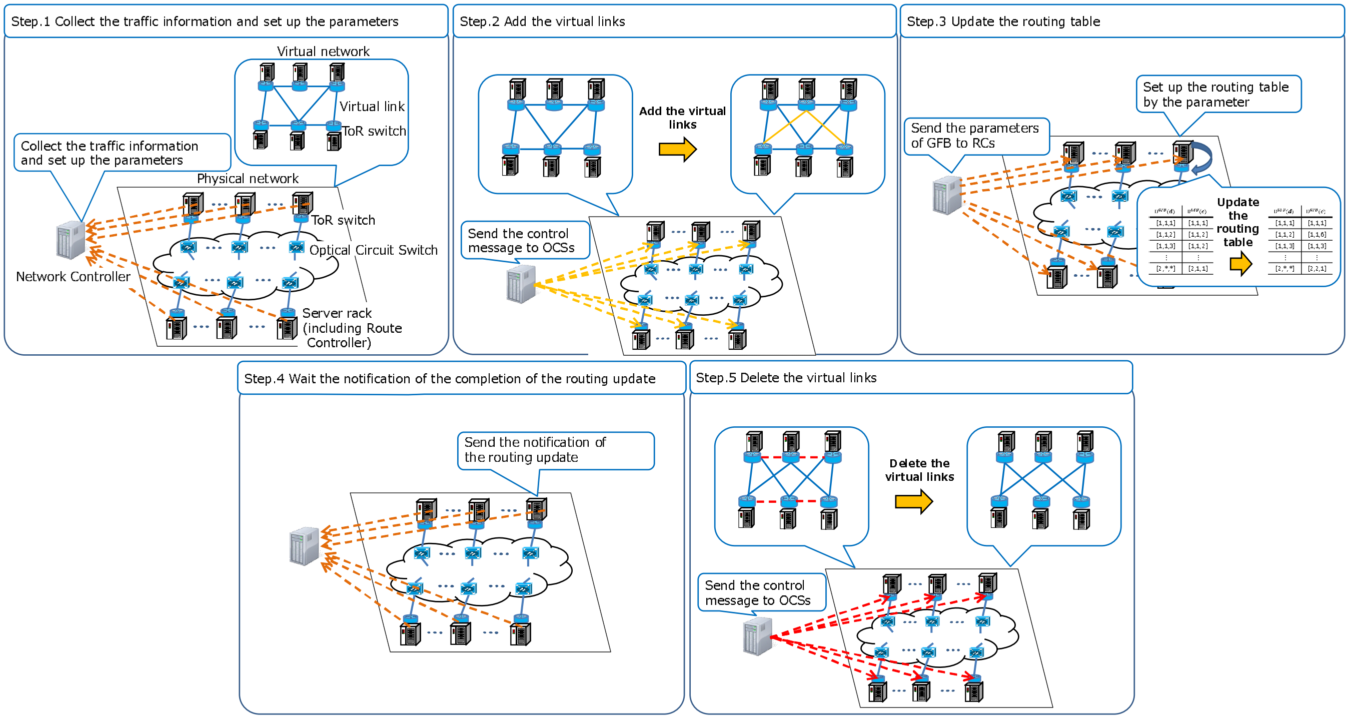

Step.1 Collect the traffic information and set up the parameters
Virtual network
Virtual link
ToR switch
Collect the traffic information and set up the parameters
Physical network
ToR switch
Optical Circuit Switch
Network Controller
Server rack (including Route Controller)
Step.2 Add the virtual links
Add the virtual links
Send the control message to OCSs
Step.3 Update the routing table
Set up the routing table by the parameter
Send the parameters of GFB to RCs
Update the routing table
Step.4 Wait the notification of the completion of the routing update
Send the notification of the routing update
Step.5 Delete the virtual links
Delete the virtual links
Send the control message to OCSs

Fig. 4. Overview of VNR-DCN

of the number of layers. For each candidate, we choose suitable parameters by the following steps.

- Step P-1 Set the parameters considering the acceptable number of hops.
- Step P-2 Modify the parameters to provide sufficient bandwidth.

Then, we construct the topology that uses the smallest number of virtual links from among the candidates. The details of the above steps are described in the following paragraphs.

a) Parameter Setting Considering the Acceptable Number of Hops: We set parameters N_k and L_k so as to ensure that the maximum number of hops is no more than $H_{\max}$. In this step, to reduce the number of variables, we set N_k to $\prod_{i=1}^{k-1} N_i + 1$ for $1 < k < K_{\max}$. In this way, h_k becomes 1 even when $L_k = 1$. To connect $N_{\rm all}$ switches, $N_{K_{\max}}$ must satisfy the following equation.

$$N_{K_{\max}} = \left\lceil \frac{N_{\rm all}}{\prod_{i=1}^{k-1} N_i} \right\rceil . \tag{21}$$

In this step, we also set $L_{K_{\max}}$ so that $h_{K_{\max}}$ becomes 1 to reduce the number of variables. To ensure that $h_{K_{\max}} = 1$, $L_{K_{\max}}$ should satisfy the following equation.

$$L_{K_{\max}} = \left\lceil \frac{N_{K_{\max}}}{\prod_{i=1}^{k-1} N_i} \right\rceil . \tag{22}$$

To ensure that the maximum number of hops is no more than $H_{\max}$, h_1 must satisfy the following condition, according to Eq. (7).

$$h_1 \le \left\lceil \frac{H_{\max}+1}{2^{K_{\max}-1}} - 1 \right\rceil . \tag{23}$$

To satisfy Eq. (23), L_1 should satisfy the following equation.

$$L_1 = \begin{cases} N_1 - 1, & \text{if } h_1 = 1 \\ 2, & \text{if } h_1 \ge \lfloor \frac{N_1}{2} \rfloor \\ \lfloor \frac{N_1}{2h_1} + 1 \rfloor, & \text{otherwise.} \end{cases} \tag{24}$$

In the above condition, all N_k $(k > 1)$ and L_k $(k \ge 1)$ are obtained in order from N_1 to $N_{\max}$. The objective of our parameter setting procedure is to minimize the number of used ports of ToR switches. That is, we minimize $\sum_{1 \le k \le K_{\max}} L_k$. Since $\sum_{1 \le k \le K_{\max}} L_k$ is a convex function of N_1, we find the N_1 that minimizes $\sum_{1 \le k \le K_{\max}} L_k$ by incrementing N_1 as long as $\sum_{1 \le k \le K_{\max}} L_k$ decreases.

b) Parameter Modification to Ensure Sufficient Bandwidth: If GFB with the parameter set obtained at Step 1 cannot provide sufficient bandwidth, we add links to the layers with insufficient bandwidth. To detect insufficient bandwidth, we check whether the following condition is satisfied for each layer-k.

$$TX_l^k \le BU, \tag{25}$$

where B is the bandwidth of one link, U is the target maximum link utilization, and T is obtained by Eq. (19). If Eq. (25) is not satisfied, L_k is incremented until Eq. (25) is satisfied.

3) Calculation Time: The virtual network control method should be applicable to large data center networks, and the calculation time for the suitable virtual network parameters should be short. The computational complexity for determining the GFB parameters is $\mathcal{O}(N|K_{\rm can}|)$, where N is the number of ToR switches and $K_{\rm can}$ is the set of candidate for the number of layers in GFB. From Eq. 20, $|K_{\rm can}| \sim \log 2H_{\max}$

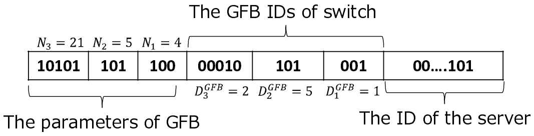


Fig. 5. Encoding GFB IDs

where $H_{\max}$ is the acceptable number of hops which is set by the administrator. Therefore, unless the administrator sets $H_{\max}$ to the significantly large value, the calculation time for the determining the GFB parameters is small. In the evaluation discussed in Section V, the parameters can be obtained within few milliseconds for a network with 420 ToR switches, by using a computer with a 3.06 GHz Intel Xeon X5675 processor.

C. Implementation Issues

In this section, we discuss certain issues related to the implementation of VNR-DCN.

1) Collection of Traffic Information: In VNR-DCN, NC needs to collect traffic information to compute the GFB parameters. NC requires only the amount of traffic from or to ToR switches, which can be obtained by collecting information from the management information base from ToR switches via Simple Network Management Protocol.

2) Configuration of an Optimal Virtual Network: In VNR-DCN, the virtual network is constructed by setting the GFB parameters. Thus, VNR-DCN can be implemented by deploying an NC implementing the method described in Subsection IV-A. OCSs accept remote commands that represent the configuration of the OCSs. By sending such commands, NC adds or deletes virtual links.

3) Routing Table Update: Routing in GFB can be implemented by using generic routing encapsulation (GRE) and OpenFlow. In this approach, we use GRE to establish a tunnel from any ToR switch to ToR switches connected to different GFBs. Then, each ToR switch selects the next packet destination from among the directly connected ToR switches and GRE tunnels according to the rules within the ToR switch set by the RC.

One approach to implementing GFB routing in the rules of OpenFlow is encoding GFB IDs into an IPv6 address and using OpenFlow switches. An example of encoding GFB IDs is shown in Figure 5, where the GFB parameters are stored in the top 11 bits of the IPv6 address. The GFB ID of the ToR switch is stored in the following 11 bits of the IPv6 address, so that the GFB ID of the upper layer becomes the prefix. The remaining bits represent the ID of the server. By encoding the network in this manner, the GFB routing table can be implemented by using the longest matching prefix. In addition, we can also find packets whose GFB parameters are different from the current parameters. Then, we add rules stating that such packets are relayed to the RC, and RC changes the IPv6 address to fit to the current parameters. In this way, we avoid packet loss even if there are packets in the virtual network during its reconfiguration.

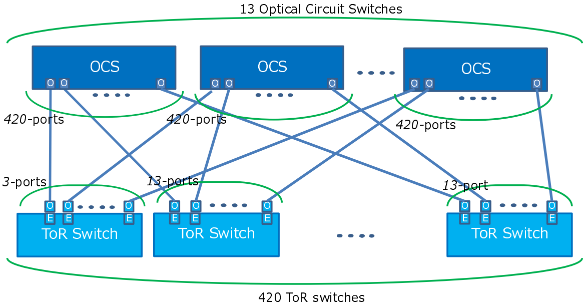


Fig. 6. Overall structure of physical network used in this evaluation

V. Evaluation

In this section, we evaluate the performance of a virtual network constructed by VNR-DCN. First, we investigate whether the virtual network constructed by VNR-DCN satisfies the requirements given as input to VNR-DCN. Next, we investigate the number of virtual links required to meet the requirements of VNR-DCN by comparing the results with existing data center networks constructed by parameter setting. Finally, we investigate the impact of reconfiguring the virtual network in terms of reduction in energy consumption by comparing the results with the method based on powering down the ports of ToR switches of a static electronic network.

In the evaluation, we use a physical network shown in Figure 6. This network includes 420 server racks and multiple OCSs with 420 ports. Each OCS is connected to all server racks. We assume that the number of ports of the ToR switches and the number of OCSs are 13.

In our evaluation, the traffic amount between ToR switch is generated by the following steps, considering the fact that each server communicates with about 1–10% of other severs in a data center [20]. First, we select the communicating ToR switch pairs randomly so that 5% of all ToR switch pairs communicate with each other, and generate traffic amount based on the uniformly distributed random values. Then, we scale the traffic amount so that the maximum amount of traffic from or to each ToR switch becomes the predefined traffic amount from the ToR switch.

In this section, we discuss the results of the evaluation based on the traffic amount normalized by the bandwidth of one virtual link, because the number of required virtual links depends on the ratio of the generated traffic amount to the bandwidth of each virtual link instead of the traffic amount itself.

A. Evaluation of the Virtual Network Satisfying the Bandwidth and Delay Requirements

In this subsection, we show that VNR-DCN can construct a virtual network satisfying the requirements that the virtual network accommodate all traffic demand and that the maximum number of hops be no more than a certain threshold.

1) Routing: Routes over the virtual network are established by the following three policies: shortest path (SP), a combination of VLB and shortest path routing (SP-VLB), and a

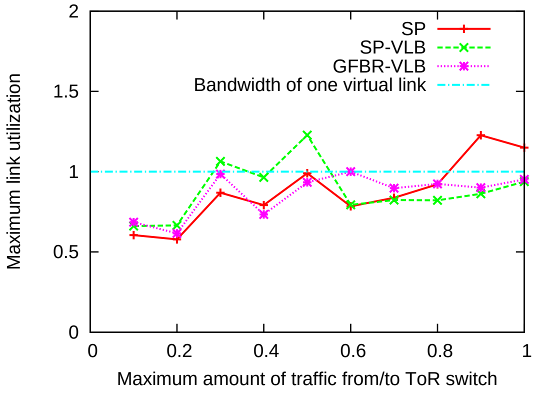


Fig. 7. Maximum link utilization required to accommodate traffic from/to ToR switches

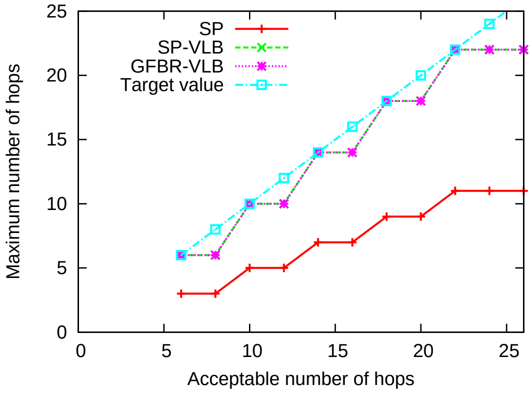


Fig. 8. Maximum necessary hop count vs. hop limit

combination of VLB and GFB routing (GFBR-VLB). In the cases of SP-VLB and GFBR-VLB, the traffic from a source ToR switch to a destination ToR switch is distributed by selecting the intermediate ToR switch randomly, as mentioned in Subsection IV-B1.

2) Evaluation Metrics: Although the total amount of traffic in the data center changes gradually, the connections between communicating servers may change significantly even within a few seconds [19, 20]. Therefore, we generate multiple random traffic patterns under the constraint that the total amount of traffic from or to each ToR switch be less than a predefined value, and consider the maximum link utilization among all traffic patterns to check whether the virtual network can accommodate all traffic. In this evaluation, we generate 10 traffic patterns. Each traffic pattern is generated by selecting the communicating ToR switches randomly and generating traffic between the selected ToR switches under the constraint on the total traffic from or to each ToR switch.

We also investigate the number of hops in the virtual network constructed by VNR-DCN to examine whether the constructed virtual network satisfies the requirements on the number of hops. In addition, we investigate the impact of GFB routing compared with SP routing.

3) Result: First, we show the maximum link utilization in Figure 7, where the horizontal axis denotes the maximum amount of traffic from or to ToR switches, and the vertical axis denotes the maximum link utilization. Note that the maximum link utilization is independent from the maximum amount of traffic from/to ToR switch, because the number of constructed virtual links are different; as the amount of traffic increases, the number of constructed virtual links becomes large. This figure indicates whether the constructed virtual network can accommodate traffic without congestion.

In the cases of SP and SP-VLB, the virtual network cannot accommodate traffic without high link utilization, even when the virtual network is constructed by VNR-DCN. This is caused by insufficient load balancing. In contrast, in the case of GFBR-VLB, the maximum link utilization is always lower than the bandwidth of one virtual link, indicating that VNR-DCN can configure a suitable virtual network that can accommodate any traffic demand. In addition, this result shows that GFBR-VLB is required in order to implement load balancing capable of accommodating any traffic demand under the constraints on the amount of traffic from or to each ToR switch.

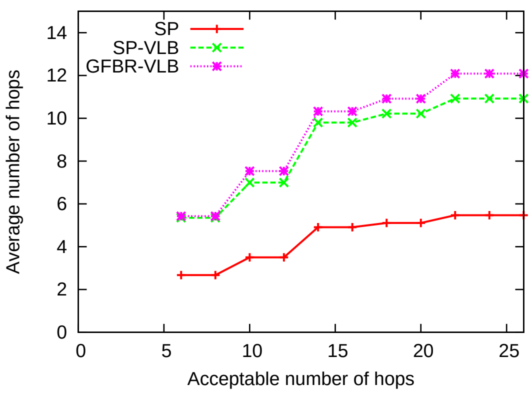


Fig. 9. Average vs. acceptable number of hops

Figures 8 and 9 respectively show the maximum and average number of hops in GFB under the constraint that the maximum number of hops be no more than the acceptable number of hops. The horizontal axes in both figures denote the acceptable number of hops, and the vertical axes denote the maximum and average number of hops, respectively. The virtual network constructed by VNR-DCN meets the requirements on the acceptable number of hops in all cases. GFBR-VLB uses a similar number of hops to SP-VLB. Although the average number of hops in GFBR-VLB is slightly larger than that in SP-VLB, GFB routing does not cause a significant increase in the number of hops.

As discussed above, GFBR-VLB can successfully balance the load and does not use an excessively large number of hops. In addition, the routing table for GFB routing can be constructed immediately after virtual network reconfiguration. Therefore, GFB routing is suitable for application to our virtual network reconfiguration.

B. Comparison with Existing Data Center Network Topologies

In this subsection, we show that GFB is appropriate for use as the base topology in VNR-DCN. In this evaluation, we investigate the number of virtual links required by GFB under the constraints that it can provide sufficient bandwidth and that the maximum number of hops be no more than $H_{\max}$. We compare the number of virtual links in GFB with existing data center network topologies (FatTree, Torus, Switch-based DCell, and FB). These data center network topologies are configured by setting their corresponding parameters so as to satisfy the same constrains as in the case of GFB. The original FatTree topology is not suitable for implementing a virtual network considering the high energy consumption, as discussed in Subsection III-B. Therefore, in this evaluation, unlike the FatTree topology proposed by Al-Fares et al. [5], we assume that all switches are connected to servers. In VNR-DCN, the GFB parameters are set to minimize the number of virtual links required by the topology under the constraints that it can provide sufficient bandwidth and that the maximum number of hops be no more than $H_{\max}$. Thus, the parameters of the other topologies are also set to minimize the number of virtual links required under these constraints.

First, we investigate the number of required virtual links when the amount of traffic from or to ToR switches is changed. In this evaluation, the amount of traffic from or to each ToR switch is the same. Also, we assume that the traffic from ToR switches is balanced by VLB. The results are shown in Figure 10, where the horizontal axis denotes the maximum traffic amount from or to ToR switches that must be accommodated, and the vertical axis denotes the number of used links required to meet the traffic demand.

Clearly, VNR-DCN uses the smallest number of links to accommodate traffic, regardless of the amount of traffic, while other topologies either require a large number of ports (FB) or cannot accommodate the required amount of traffic with any parameter settings (Switch-based DCell, FatTree, and Torus). This is because in setting the GFB parameters, VNR-DCN adds only links that are necessary to accommodate the traffic. Therefore, the topology constructed by VNR-DCN satisfies the requirement on bandwidth with the lowest energy consumption.

We also compare the number of used links required to meet the requirements on the acceptable number of hops. In this comparison, we assumed that the capacity of each virtual link is sufficient. The results are shown in Figure 11, where the horizontal axis denotes the maximum number of hops, and the vertical axis denotes the number of virtual links required to satisfy the requirements. In all cases of the acceptable maximum number of hops, the topology constructed by VNR-DCN uses the smallest number of virtual links to satisfy the requirements. This is again because VNR-DCN adds only links that are necessary, thus maintaining the maximum number of hops at no more than the required value.

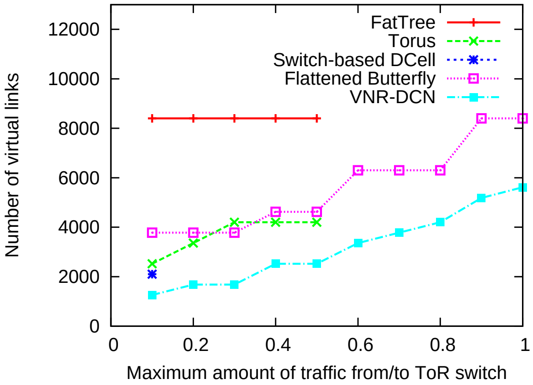


Fig. 10. Number of virtual links required to accommodate the traffic from ToR switches

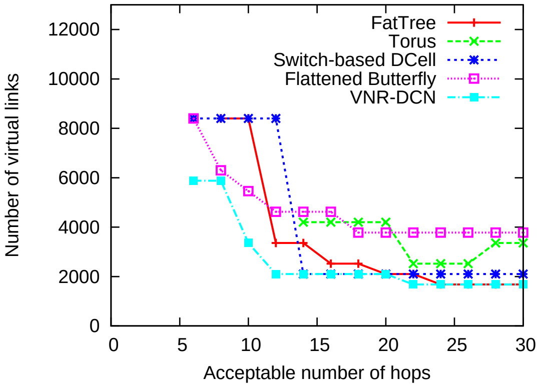


Fig. 11. Number of virtual links required to ensure that the maximum number of hops is no more than the target value

C. Impact of virtual network reconfiguration on energy consumption

In this subsection, we demonstrate that the proposed virtual network reconfiguration reduces the energy consumption, even though the addition of the OCSs is required. In this evaluation, we change the maximum amount of traffic from or to ToR switches from 1 to 0.1 but maintain the same number of physical links.

1) Comparison Method: In this evaluation, we compare VNR-DCN with the following two cases where the network is constructed without OCSs.

a) Static Electronic Network: This network is constructed using only electronic switches, and all ports of the switches are always powered on. In our evaluation, we construct this network to connect the electronic ports of ToR switches. Comparing VNR-DCN with this network construction method, we demonstrate the strong impact of powering down switch ports on the energy consumption. In our evaluation, the topology of the static electronic network is set as in the case of GFB, whose parameters are set so as to accommodate the maximum amount of traffic generated in the evaluation because GFB accommodates the maximum amount of traffic with the smallest number of links.

TABLE II
ENERGY CONSUMPTION OF DATA CENTER NETWORK ELEMENTS IN THIS EVALUATION

Element	Power consumption (W)
ToR (48 ports)	600
OCS (420 ports)	100.8
Transceiver	1

b) Elastic Tree: The network topology can be reconfigured without using OCSs. Heller et al. proposed a method for reconfiguring the topology by powering down the ports of electronic switches [17]. In our evaluation, we also use this method in the abovementioned electronic network to reduce the energy consumption. Comparing VNR-DCN with this method, we demonstrate the impact of the reconfiguration of the virtual network using OCSs. The method used in Heller et al. is based on FatTree, which requires additional switches, resulting in increased energy consumption compared with the network topology using only ToR switches connected to communicating servers. Thus, in our evaluation, we use GFB as the base topology whose parameters are set so as to accommodate the maximum amount of traffic generated in our evaluation, similarly to the static electronic network.

In the evaluation, we power down the ports of the ToR switches when the following two constraints are satisfied: (1) routes exist between all ToR switches, and (2) the capacity of each link is larger than the traffic passing through the link. When determining the electronic ports to be powered down, we set the routes of the traffic by SP or SP-VLB.

2) Energy Consumption Model: In our evaluation, we models the energy consumption based on the catalogs of the following devices. We use the 48-port Arista 7148 switches [22] as the ToR switches. We need to connect some ports of ToR switches to OCSs via optical transceivers. We use Delta 10 GBASE SR transceivers [26]. In our architecture, we need an OCS with 420 ports, and a OCS with such a large number of ports has been proposed and implemented [27]. However, we do not have the data of the energy consumption of the OCS with a large number of ports. Thus, we estimate the energy consumption of the OCS by assuming that the energy consumption of each port of the OCS equals to that of a 192-port Glimmerglass OCS [21]. This assumption may overestimate the energy consumption of OCS, because most of the energy of the OCS is consumed by the management function that handles the remote commands from the controller and its energy consumption is independent from the number of ports. Thus, the actual energy consumption of VNR-DCN may be smaller than the following results.

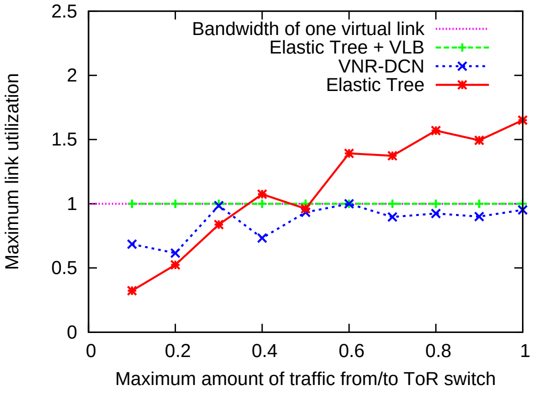


Fig. 12. Maximum link utilization for different amounts of traffic

Table II summarize the energy consumed by the devices used in the network. In the VNR-DCN, only the ports of ToR switches, optical transceivers, and OCSs that are required to construct the current virtual network are powered on. The energy of the ToR switch is consumed by the two kinds of components; one is the component that cannot be shut down even if all ports are shut down, and the other is the component that can be shut down if the corresponding port is not used. However, the ratio of the energy saved by shutting down the ports is not described in the spec sheet of the switches, and depends on the switch architectures [28]. Therefore, we introduce a parameter α indicating the ratio of the energy consumed by the ports to the total energy consumption of the switch. The results by Reviriego et al. [28] indicates that α of the commercial energy efficient Ethernet switches is from 0.5 to 0.7. Thus, in this evaluation, we use two kinds of α, 0.5 and 0.7.

By using this model, the energy consumption of the virtual network P_{logical} constructed over an OCS network is obtained by

$$P_{\text{logical}} = 100.8 S_{\text{active}} + (12.5\alpha + 1)L + 252000(1 - \alpha), \quad (26)$$

where S_{active} is the number of active OCSs, and L is the number of virtual links, respectively. In contrast, when we construct the topology without OCSs, the energy consumption P_{physical} is modeled by

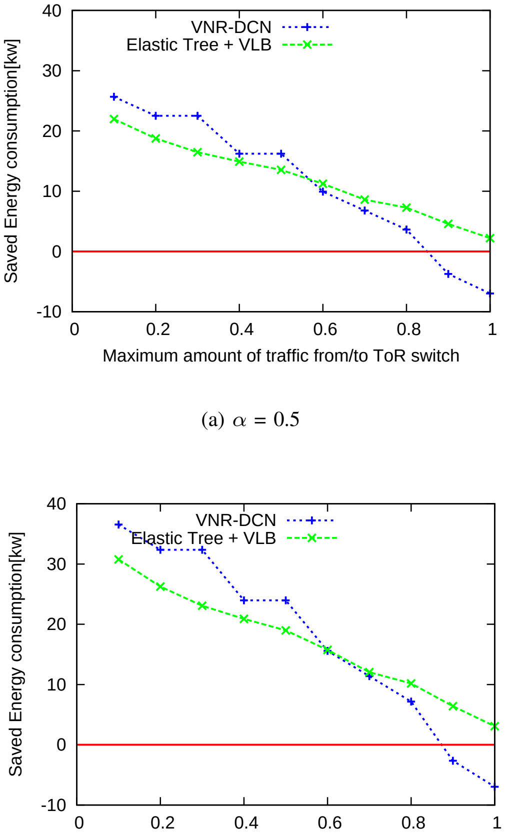
$$P_{\text{physical}} = 12.5\alpha L + 252000(1 - \alpha), \quad (27)$$

where L is the number of active links. The energy consumed by the servers or by the ports connected to the servers is ignored in our evaluation because it is the same in all cases.

3) Results:

a) Maximum link utilization for different amounts of traffic: Before comparing the energy consumption of virtual networks constructed by each method, we check whether the constructed network can accommodate the required amount of traffic. In a data center, traffic patterns change within a few seconds [19, 20], and the virtual network should accommodate such frequently changing traffic as discussed in Subsection V-A. Thus, we investigate the maximum link utilization when the traffic pattern changes from the initial traffic pattern, while the virtual network is constructed by using the initial traffic pattern. The initial and changed traffic patterns are generated in the same manner as in Subsection V-A, namely by generating 10 traffic patterns after the change and showing the maximum link utilization in all patterns.

The results are shown in Figure 12, where the horizontal axis denotes the maximum traffic amount from or to ToR switches that must be accommodated, and the vertical axis



(a) α = 0.5

(b) α = 0.7

Fig. 13. Energy consumption required to accommodate the traffic from/to ToR switches

shows the maximum link utilization when the traffic demand is changed. In the figure, the labels “VNR-DCN”, “Elastic Tree”, and “Elastic Tree + VLB” denote the results for the cases where the topology is constructed by using VNR-DCN, the elastic tree method (see paragraph V-C1b), and the elastic tree method with considering VLB, respectively. “Electronic” denotes the results for the static electronic network.

In the case of using an elastic tree, the maximum link utilization is larger than the bandwidth of one virtual link when the maximum amount of traffic is larger than 0.4. The elastic tree method uses a small number of links if they are sufficient to accommodate the initial traffic. However, the link utilization increases after changing the traffic demand. In contrast, the maximum link utilization of the topology constructed using an elastic tree by considering VLB is smaller than the target value. This is because VLB avoids the concentration of traffic at any particular ToR switch pair by balancing the traffic across all ToR switch pairs. Then, the virtual network constructed using an elastic tree and VLB accommodates the traffic balanced by VLB. As a result, the virtual network accommodates all traffic without excessive link utilization, even when traffic demand changes.

b) Evaluation of the energy consumption: We investigate the energy consumption of the topologies. The results are shown in Figures 13(a) and 13(b), where the horizontal axis denotes the maximum amount of traffic from or to ToR switches that must be accommodated, and the vertical axis denotes the saved energy consumption compared with the static electronic network.

In this evaluation, we exclude the result for elastic tree because in that case the network cannot accommodate the required amount of traffic, as discussed in the previous paragraph. When the maximum amount of traffic from or to ToR switches is large, the energy consumption of the network constructed by VNR-DCN is larger than that in the case of the static network based on electronic switches alone, and the saved energy consumption becomes lower than zero. This is caused by the additional devices required by VNR-DCN, such as the optical transceiver and the OCS.

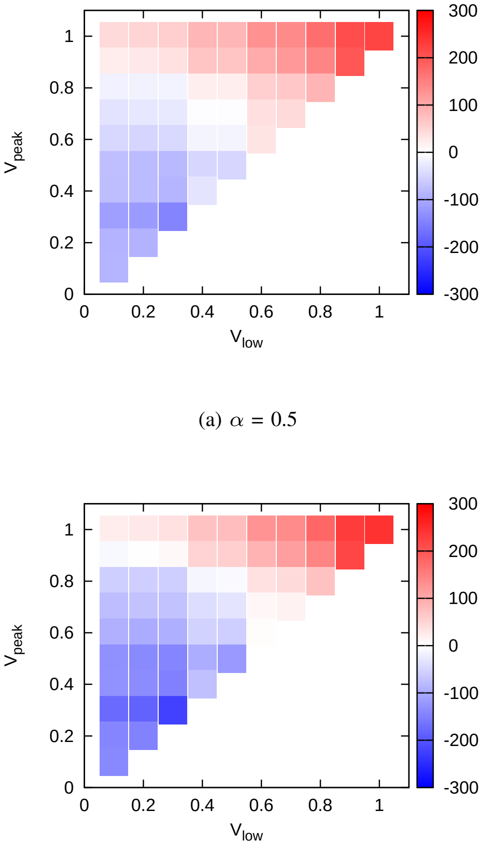
Figure 13 also shows that in the case of using the elastic tree combined with VLB, the energy consumption can be reduced in comparison to the static electronic network, even when the maximum amount of traffic is generated. This is caused by the fact that all switches in GFB have the same number of links in order to facilitate the calculation of the number of flows passing through each link. However, even though the elastic tree allows switches to have different number of links, the reduction in energy consumption brought by the elastic tree is marginal. When the maximum amount of traffic from or to ToR switches is small, the energy consumption of the network constructed by VNR-DCN is the smallest. This is because VNR-DCN reconfigures the virtual network so as to reduce the number of links based on the instantaneous traffic load. Although the elastic tree also powers down ports to save energy, it cannot reconfigure the network to a topology vastly different from the initial topology because it has to maintain the connectivity between ToR switches, and many links cannot be powered down.

Comparing Figure 13(a) with Figure 13(b), the saved energy becomes large in the case of large α, because shutting down each port reduces energy consumption more. However, even in the case of $\alpha = 0.5$, VNR-DCN saves more energy than the elastic tree when the traffic amount becomes small.

c) Evaluation of energy consumption over 24 h: As can be seen in Figure 13, the energy consumption of the network constructed by VNR-DCN is lower than others when the maximum amount of traffic from or to ToR switches is small. To clarify the impact of the reduction in energy consumption, we compare the total energy consumed over 24 h. Heller et al. [17] found a clear pattern in the total traffic rate at switches in a data center, where traffic was found to peak during the day and drop at night. In this evaluation, we use the following simple model of variation in traffic over 24 h:

$$T_{\max}(x) = \frac{V_{\text{peak}} - V_{\text{low}}}{2} \sin x + \frac{V_{\text{peak}} + V_{\text{low}}}{2} (0 \leq x \leq 2\pi), \quad (28)$$

where $T_{\max}(x)$ is the traffic rate from or to each ToR switch at time x, V_{peak} is the peak traffic rate, and V_{low} is the lowest traffic rate. We investigate the energy consumption in various scenarios by changing V_{peak} and V_{low}. In this evaluation, the



(a) α = 0.5

(b) α = 0.7

Fig. 14. Energy consumption required to accommodate the traffic from/to ToR switches

elastic tree method and VNR-DCN are used to configure the virtual network 24 times over 24 h.

Figure 14 shows the results, where vertical axis denotes the peak rate and horizontal axis denotes the lowest rate. In this figure, each area is colored based on the value of $P^{\mathrm{VNR-DCN}}_{\mathrm{logical}} - P^{\mathrm{elastic}}_{\mathrm{physical}}$ where $P^{\mathrm{VNR-DCN}}_{\mathrm{logical}}$ is the energy consumption when we use the VNR-DCN, and $P^{\mathrm{elastic}}_{\mathrm{physical}}$ is the energy consumption when we use the elastic tree combined with the VLB. The area where VNR-DCN saves more energy than the elastic tree is colored in blue, while the area the elastic tree saves more energy is colored in red. Clearly, VNR-DCN is effective when traffic changes drastically even in the case of $\alpha = 0.5$. This is because VNR-DCN changes the virtual network topology so as to reduce the energy consumption, while the elastic tree cannot change the network topology. In this regard, Kandula et al. showed that traffic rate changes drastically in an actual commercial data center [20]. VNR-DCN is expected to be effective for reducing the energy consumption in such data centers.

VI. Discussion on the Scalability

In this section, we discuss the scalability of the VNR-DCN, and explain how the VNR-DCN works in a large data center.

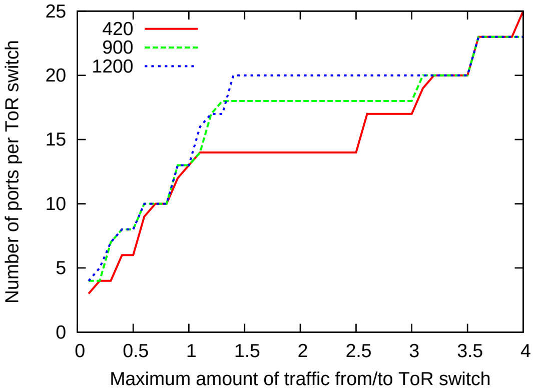


Fig. 15. The number of ports per ToR switch in the case of increasing number of servers and traffic

A. Scalability of the controllers

1) NC: In our method, we introduce the centralized controller NC, which collects the traffic information, and sets the parameters of the GFB. As the size of the network becomes large, the number of ToR switches whose traffic information is required to be collected by the NC increases. However, the amount of the collected traffic information is not large, because the VNR-DCN requires only the total traffic amount from/to each ToR switch and does not require the detailed traffic information.

The calculation time of the GFB parameters may increase as the size of the network becomes large. However, the calculation time of the GFB parameters is only $\mathcal{O}(N) \log 2H_{\max}$, where N is the number of ToR switches and $H_{\max}$ is the acceptable number of hops set by the administrator, as discussed in Section IV-B3. Therefore, the calculation time does not become large, even in a large network.

In addition, because the short-term traffic changes are handled by the load balancing, the interval of calculation of the parameters of the GFB is not necessarily short. Therefore, the NC can work even in a large data center.

2) RC: We also introduce another kind of the controller, RC. The only task of the RC is to calculate the routing table from the GFB parameters. The calculation time for the routing table in GFB routing is $\mathcal{O}(\sum_{1<=i<=K_{\max}} N_i)$, which is significantly shorter than $\mathcal{O}(N)$, as discussed in Section III-C2.

B. Scalability of the physical network structure

In our architecture, we need the network of the OCSs that can connect any ToR switch pairs. Though we use the OCSs with the same number of ports as the number of ToR switches in the evaluation in Section V, we can construct the network by using OCSs with a small number of ports. For example, by constructing the Clos network using the OCSs, we can construct the network that can connect any ToR switch pairs.

C. Scalability of the constructed virtual network

Finally, we discuss the virtual network by the VNR-DCN when the number of ToR switches or the amount of the traffic becomes large. Figure 15 shows the number of used ports per ToR switch. In this figure, the horizontal axis is the maximum amount of traffic from a ToR switch, and the vertical axis is the number of used ports per ToR switch. We plot three lines; the red line indicates the case with 420 ToR switches, the green line indicates the case with 900 ToR switches, and the blue line indicates the case with 1200 ToR switches.

As shown in this figure, the number of required ports per ToR switch is almost similar even when the number of servers is large. That is, our method is applicable to a large data center.

This figure also indicates that the number of used ports per ToR increases linearly as the traffic amount increases. This increase of the number of used ports is not large, because at least a proportional number of virtual links to the amount of traffic is required to accommodate the generated traffic without congestion even if any kind of the network topology is used. Therefore, the VNR-DCN constructs the virtual network with a small number of ports even when the traffic amount becomes large.

VII. Conclusion

In this paper, we introduced the concept of a virtual network configured over a data center network consisting of both OCSs and electronic switches. We proposed a method for constructing a virtual network by setting the parameters of its topology as well as a method for reconfiguring the virtual network within a short period of time by adjusting these parameters. In addition, we discussed implementation issues related to VNR-DCN and demonstrated that it can be applied to networks consisting of existing devices.

Through evaluation, we clarified that VNR-DCN constructs a topology satisfying the requirements on bandwidth and delay in the case of using a routing method based on GFB combined with VLB. We demonstrated that VNR-DCN is effective for reducing the energy consumption of the network by comparing it with the method based on powering down the ports of the ToR switches of a static electronic network.

One of our future research topics is to construct the distributed algorithm to further reduce the time required to respond to traffic changes.

Acknowledgment

This work was supported in part by the National Institute of Information and Communications Technology (NICT).

References

[1] A. Greenberg, J. Hamilton, D. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM Computer Communication Review*, vol. 39, pp. 68–73, Jan. 2008.

[2] L. Barroso and U. Holzle, "The case for energy-proportional computing," *Computer*, vol. 40, pp. 33–37, Dec. 2007.

[3] C. Patel, C. Bash, R. Sharma, M. Beitelmal, and R. Friedrich, "Smart cooling of data centers," in *Proceeding of International Electronic Packaging Technical Conference and Exhibition*, pp. 129–137, July 2003.

[4] D. Abts, M. Marty, P. Wells, P. Klausler, and H. Liu, "Energy proportional datacenter networks," *ACM SIGARCH Computer Architecture News*, vol. 38, pp. 338–347, June 2010.

[5] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *ACM SIGCOMM Computer Communication Review*, vol. 38, pp. 63–74, Aug. 2008.

[6] J. Kim, W. Dally, and D. Abts, "Flattened butterfly: a cost-efficient topology for high-radix networks," *ACM SIGARCH Computer Architecture News*, vol. 35, pp. 126–137, May 2007.

[7] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "DCell: a scalable and fault-tolerant network structure for data centers," *ACM SIGCOMM Computer Communication Review*, vol. 38, pp. 75–86, Aug. 2008.

[8] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "BCube: a high performance, server-centric network architecture for modular data centers," *ACM SIGCOMM Computer Communication Review*, vol. 39, pp. 63–74, Aug. 2009.

[9] D. Guo, T. Chen, D. Li, Y. Liu, X. Liu, and G. Chen, "BCN: expansible network structures for data centers using hierarchical compound graphs," in *Proceedings of IEEE INFOCOM*, pp. 61–65, Apr. 2011.

[10] D. Li, C. Guo, H. Wu, K. Tan, Y. Zhang, S. Lu, and J. Wu, "Scalable and cost-effective interconnection of data-center servers using dual server ports," *IEEE/ACM Transactions on Networking*, vol. 19, pp. 102–114, Feb. 2011.

[11] A. Greenberg, J. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. Maltz, P. Patel, and S. Sengupta, "VL2: a scalable and flexible data center network," *ACM SIGCOMM Computer Communication Review*, vol. 39, pp. 51–62, Aug. 2009.

[12] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, "PortLand: a scalable fault-tolerant layer 2 data center network fabric," *ACM SIGCOMM Computer Communication Review*, vol. 39, pp. 39–50, Aug. 2009.

[13] A. Singla, C.-Y. Hong, L. Popa, and P. B. Godfrey, "Jellyfish: Networking data centers randomly," in *Proceedings of USENIX Symposium on Networked Systems Design and Implementation*, pp. 1–14, Apr. 2012.

[14] A. R. Curtis, T. Carpenter, M. Elsheikh, A. López-Ortiz, and S. Keshav, "REWIRE: an optimization-based framework for unstructured data center network design," in *Proceedings of IEEE INFOCOM*, pp. 1116–1124, Mar. 2012.

[15] X. Wang, M. Chen, C. Lefurgy, and T. W. Keller, "Ship: A scalable hierarchical power control architecture for large-scale data centers," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, pp. 168–176, Jan. 2012.

[16] Y. Zhang and N. Ansari, "On architecture design, congestion notification, TCP incast and power consumption in data centers," pp. 39–64, Feb. 2013.

[17] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "ElasticTree: saving energy in data center networks," in *Proceedings of USENIX Symposium on Networked Systems Design and Implementation*, pp. 1–16, Apr. 2010.

[18] A. Singla, A. Singh, K. Ramachandran, L. Xu, and Y. Zhang, "Proteus: a topology malleable data center network," in *Proceedings of ACM SIGCOMM Workshop on Hot Topics in Networks*, pp. 8–13, Oct. 2010.

[19] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *Proceedings of ACM SIGCOMM conference on Internet measurement*, pp. 267–280, Nov. 2010.

[20] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: measurements & analysis," in *Proceedings of ACM SIGCOMM conference on Internet measurement conference*, pp. 202–208, Nov. 2009.

[21] "Glimmerglass intelligent optical system 600." http://www.glimmerglass.com/products/intelligent-optical-system-600/.

[22] "Arista 7148sx switch." http://www.aristanetworks.com/media/system/pdf/Datasheets/7100_Datasheet.pdf.

[23] S. Ghemawat, H. Gobioff, and S. Leung, "The google file system," in *Proceeding of ACM SIGOPS Operating Systems Review*, vol. 37, pp. 29–43, ACM, 2003.

[24] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[25] M. Kodialam, T. Lakshman, and S. Sengupta, "Efficient and robust routing of highly variable traffic," in *Proceedings of HotNets*, Nov. 2004.

[26] "Delta 10gbase-sr sfp+ optical transceiver." http://www.deltaww.com/filecenter/Products/download/04/0405/DataCenter/LCP-10G3A4EDRx-G_S2.pdf.

[27] Y.-K. Yeo, Z. Xu, C.-Y. Liaw, D. Wang, Y. Wang, and T.-H. Cheng, "A 448× 448 optical cross-connect for high-performance computers and multi-terabit/s routers," in *Proceeding of Optical Fiber Communication, collocated National Fiber Optic Engineers Conference*, pp. 1–3, Mar. 2010.

[28] P. Reviriego, V. Sivaraman, Z. Zhao, J. A. Maestro, A. Vishwanath, A. Sánchez-Macián, and C. Russell, "An energy consumption model for energy efficient ethernet switches," in *Proceeding of International*

Conference on High Performance Computing and Simulation (HPCS), pp. 98–104, July 2012.

Yuya Tarutani received an M.E. degree in Information Science and Technology from Osaka University in 2012.
He is currently a Ph.D. student in Information Science and Technology at Osaka University. His research interest includes traffic matrix estimation and data center networks. He is a Student Member of IEICE and IEEE.

Yuichi Ohsita received M.E. and Ph.D. degrees in Information Science and Technology from Osaka University in 2005 and 2008, respectively.
He is currently an Assistant Professor at the Graduate School of Information Science and Technology, Osaka University. His research interests include traffic matrix estimation and countermeasures against DDoS attacks. He is a Member of IEICE, IEEE, and the Association for Computing Machinery.

Masayuki Murata received M.E. and D.E. degrees in Information and Computer Sciences from Osaka University in 1984 and 1988, respectively.
In April 1984, he joined the Tokyo Research Laboratory of IBM Japan as a researcher. From September 1987 to January 1989, he was an Assistant Professor at the Computation Center, Osaka University. In February 1989, he moved to the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University. From 1992 to 1999, he was an Associate Professor at the Graduate School of Engineering Science, Osaka University, and became a Professor at the same school in April 1999. He moved to the Graduate School of Information Science and Technology, Osaka University in April 2004. He has published more than 300 papers in international and domestic journals, and has given presentations at numerous conferences. His research interests include computer communication networks, as well as performance modeling and evaluation.
He is a Fellow of IEICE and a Member of IEEE, the Association for Computing Machinery (ACM), The Internet Society, and IPSJ.