Distributed Search for Exchangeable Service Chain Based on In-Network Guidance

Yutaro ODA†, Student Member, Yosuke TANIGAWA†, Member, and Hideki TODE†, Senior Member

SUMMARY   Network function virtualization (NFV) flexibly provides services by virtualizing network functions on a general-purpose server, and attracted research interest in recent years. In NFV environment, providing service chaining, which dynamically connects each network function (virtual network function: VNF), is critical issue. However, as it is challenging to select the optimal sequence of VNF services in the service chain in a decentralized manner, the distances between the VNFs tend to increase, leading to longer communication and processing delays. Furthermore, it has never considered that certain VNFs that can be exchange the order of services with one another. To address this problem, in this paper, we propose a distributed search method for ordered VNFs to reduce delays while considering the load on control server, by exploiting an in-network guidance technology, called Breadcrumbs, for query messages.

key words: breadcrumbs, NFV, VNF, service chaining

1. Introduction

Networks are currently constructed using network functions composed of a wide variety of network devices. Conventionally, for such changes to the network as updating or adding a network function, it is necessary to replace the network equipment, which entails considerable cost in terms of equipment and labor.

To address this problem, Network Function Virtualization (NFV) virtualizes network functions on a general-purpose server, and has thus attracted considerable research interest in recent years. For example, it is supposed that network functions such as network address translation, firewall, intrusion detection and prevention system, and so on, will be virtualized as Virtual Network Functions (VNFs). In addition to these fundamental and conventional network functions, many kinds of as-yet-unknown network functions would emerge in near future; for example, network functions for big data processing. As NFV replaces dedicated network devices with general-purpose servers, there is no need to prepare specific devices for each network function. Thus, equipment cost can be reduced. Moreover, as new functions can be implemented in the network by updating functions on the general-purpose server, the new network services and operations can be flexibly implemented.

As network functions are virtualized, it also becomes possible to share these virtualized network functions (VNFs) on the network. Thus, each user on the network can utilize these network functions without the need to prepare specific network devices by other hardware. As there are several VNFs with same function on the network, it is possible to use an alternative VNF, even when a certain VNF becomes unusable due to breakdown, i.e., it is possible to construct a network with high fault tolerance.

There are two types of NFV environments: a centralized type, in which numerous VNFs are arranged in a data center, and a distributed type, where several small-scale VNFs are arranged at edges near to users in a decentralized manner. The former environment is the mainstream at present, but the latter environment is also being studied[1], [2]. In this paper, a distributed environment, that is complex but offers considerable potential for scalability and resilience, is assumed. For service provision in NFV environments, data are transferred and adequately via multiple network functions in sequence, which is called service chaining. However, in a distributed environment, when there are many VNFs, it is impractical to select a globally optimal solution because collecting many types of states in VNFs, deployed over in a wide area, is difficult process in real time. In other words, it is difficult to select nearest server considering statuses of all VNF servers and positions of users and VNF servers on time.

In this paper, to search VNFs by configuring a service chain efficiently in a realistic way, we propose a distributed search method for VNFs by applying in-network guidance technology as basis. We aim to build a decentralized NFV infrastructure based on the design policy whereby appropriate VNFs are selected by logging and using past usage histories of VNFs configuring service chains for simple but effective search. This is because there are several patterns in the types that need to be combined as a service chain. To satisfy this design policy, we focus on Breadcrumbs [3], [4] as a in-network guidance method. Breadcrumbs framework realized in-network guidance like Information Centric Network (ICN) over IP network. In this framework, queries are guided towards nodes with caches of the target content, by using past guidance histories logged at routers in a decentralized manner. This framework is appropriate for our design policy described above.

The main contributions of this paper are as follows: We propose and design distributed infrastructure for NFV, based on in-network guidance. In this infrastructure, we establish a distributed search method for ordered VNFs configuring service chaining. We show that the proposed method re-
duces the load on control server and redundant communication, through extensive computer simulation. As future challenge, finally, we insist on the need for decentralized NFV infrastructure.

The remaining of this paper is organized as follows. We show background technologies and related works in Sect. 2. In Sect. 3, we explain our proposed method. The proposed method is evaluated in Sect. 4. Focusing on VNFs which have exchangeability, we extend our proposed method in Sect. 5, and evaluate it in Sect. 6. We describe the conclusion of this paper in Sect. 7.

2. Background Technologies and Related Works

2.1 Network Function Virtualization

NFV virtualizes various network functions such as router, switch, firewall on general-purpose servers, and provides virtualized network functions (VNFs) on cloud computing platforms [5], [6]. Through communication and process via a series of intermediate VNFs from each user to its destination, a specific services are provided. The performance of VNF has been investigated in various environments [7], [8].

The advantages of NFV environments are as follows:

- They reduce capital investment and maintenance cost by eliminating the need for specific devices [9].
- They improve fault tolerance by providing multiple VNFs of a same type.
- They improve network flexibility, e.g., through easier provision of new services.

On the contrary, the following are disadvantages of NFV environments:

- They increase the end-to-end delay from each user to their destination because of detouring through multiple VNFs.
- They require complicated management because all operators need to manage several VNFs.

2.2 Service Chaining

In NFV environments, services are provided by transferring data through multiple types of VNFs from users to their destinations. This series of VNFs is called a service chain. In environments assumed in this paper, there are various types of VNFs, and many VNF servers support certain types of VNF. Of these VNF servers, good solutions for selecting a series of VNF configuring service chains need to be determined by a control server in charge of network and service administration by considering the resource usage at each VNF server and the distances between neighboring VNF servers in a service chain. In general, it is difficult to provide the number of VNF types which is enough to provide all kinds of service chains at a few VNF servers. Thus, the selection of VNFs configuring a service chain, so that data transfer delay through the VNFs in the order of the processing becomes as small as possible, is a challenging problem.

There are several patterns in VNF types that need to be combined as a service chain. Such patterns have been discussed in Internet Engineering Task Force (IETF) [10]. An example of a service chain is shown in Fig. 1; a user uses VNF 1, VNF 2, and VNF 3 as the service chain by transferring data via one of the VNF servers corresponding to each VNF type.

In configuring service chain, we need to solve problems in the arrangement of VNFs [11], which determine where the VNFs are placed and which VNFs are selected. M. T. Beck et al. [12] proposed a heuristic algorithm for optimizing VNF arrangement to reduce communication resource consumption among data centers and the conversion between optical and electrical signals. M. Xia et al. [13] proposed a VNF allocation method for minimizing bandwidth usage in the network within a realistic execution time. S. Ahvar et al. [14] proposed to search VNFs for service chains by using Tabu search to reduce some costs like network bandwidth, operation cost, and so on. However, the hop counts of query and data are not considered. On the other hand, one objective of our proposed method is to reduce the hop counts as explained in Sect. 5.

The European telecommunication standards institute (ETSI) has defined the NFV framework [15]. According to it, the NFV framework can be roughly divided into three parts: the first is VNF, the second is NFV infrastructure representing physical resources and virtualization functions for VNFs, and the third is NFV management and orchestration (MANO), which manages and orchestrates the physical and virtual resources and VNFs.

R. Mijumbi et al. [16] claimed that a MANO framework is needed to manage NFV; The model-oriented repository for scalable access (MORSA) [17] has been proposed as a MANO frameworks. It performs resource scheduling for NFV infrastructure and satisfies various objective functions using a multi-objective genetic algorithm. A distributed and privacy-preserving algorithm using non-cooperative game theory [18] has also been proposed.

Our proposed method is a type of MANO. It is a distributed search method for VNFs that provides resource management. VNF search using simple and passive Information Centric Network (ICN) feature, such as in-network guidance technology, according to the requesting resource attribute, has not been studied to date, to the best of the au-

![Fig. 1 Example of a service chain](image-url)
Fig. 2 Processes to search for and provide resources in the RBC method

thors knowledge.

2.3 Resource Breadcrumbs

Resource Breadcrumbs (RBC) [19] is an overlay method for searching computer resources in a distributed manner in edge computing environments over an IP network. The RBC method extends the method of content search and acquisition based on breadcrumbs (BC) method [3], [4], whereby a query to the content cache is guided by in-network guidance information.

As shown in Fig. 2, a system based on RBC consists of suppliers providing resources, routers that autonomously guide queries while storing resource guidance information, and a control server that is a central database.

2.3.1 Distribution of the Guidance Information

Suppliers, which are resource provider nodes, autonomously distribute RBCs with guidance information centering themselves to their surrounding nodes by disseminating messages. At the same time, suppliers register their resource information with the control server. The distributed RBCs are appropriately updated according to the changes in their resource utilization.

2.3.2 Search for Resources

Each user searches for resources by sending a query to the control server. When a query passes through a router on the route from a user to the control server, if an RBC with guidance information to a supplier that can provide the corresponding resource is found, the query is forwarded to the supplier by using the guidance information. However, if the corresponding RBC cannot be found on the route, the query reaches the control server, which transfers the query to the corresponding supplier, based on such registered information as the registered resource utilization. Upon receiving the query, the supplier retains the requested resource for the user and sends an ACK to the user. The user determines the supplier to be used, based on the received ACKs.

In Fig. 2, the user in the top of the figure sends queries searching for blue and red resources. The query for the blue resource finds the corresponding RBC and is guided to the supplier by the RBC. However, the query for the red resource does not find the RBC for red on the way to the control server and is guided to the supplier by the control server.

2.4 En-Route RBC

In the RBC method, guidance information should be distributed within a limited range because the load on the network to disseminate messages increases as the range of distribution expands. En-route RBC (ERBC) has been proposed as an extension to compensate for this drawback [20].

In the ERBC method, in addition to RBC distribution, when an ACK message is returned from the supplier to the user, information concerning the resource is added to the ACK message. Further, each router on the forwarding route reads this information and creates guidance information called ERBC. Queries are guided by the ERBC to the supplier, but if the corresponding RBC is found during guidance by the ERBC, the query is forwarded by RBC guidance because of a higher priority. The ERBC operation is shown in Fig. 3. The supplier sends an ACK message to the user who uses the resource of the supplier. Routers that forward the ACK message constitute the ERBC for the supplier.

The RBC and ERBC frameworks described in Sects. 2.3 and 2.4 perform a resource search with the in-network guidance capability. However, they cannot be applied for VNF environments with service chaining which orders a series of VNFs according to the user’s request.

3. Proposed Method

Conventional in-network guidance techniques including BC method are designed to search not resources but contents, and such techniques are not applicable to resource and VNF searching. Thus, RBC method is designed to search resources by extending the content search technology in BC method. In RBC method, users request combination of some computer resources like CPU and memory, and the resources are reserved and used at the same time. In contrast, in NFV environment, VNFs configuring a service chain are used not simultaneously but step by step. Thus, if such VNFs are reserved simultaneously based on RBC method, the VNFs’ resources are reserved for longer time than necessary. To solve this problem, in the proposed method, each VNF configuring a service chain is searched not by the user
Table 1  FBC entry

| VNF_ID | Server_ID | Max_resource | Used_resource | Upstream_node (only EFBC) | Downstream_node (only EFBC) | Hop_count | TTL |
|--------|-----------|--------------|---------------|--------------------------|---------------------------|-----------|-----|

itself but by the VNF server which has just processed the data of an intermediate VNF in the service chain, except for the firstly used VNF that is searched by the user. In RBC method, each user sends some queries for all requesting resources. On the other hand, in the proposed method, each user sends a query for the service chain. The query has information about the requested VNFs of the service chain. VNF servers are informed of a service chain by the query. In addition, information of VNF servers in a service chain, including the information of adjacent downstream and upstream VNF servers, is preserved on data route in the proposed method. While, in RBC method, information of only a resource supplier is preserved on data route by ERBC.

Thus, we design a decentralized NFV infrastructure where appropriate VNFs configuring service chains are searched simply and effectively by exploiting past usage histories. This is inspired by the in-network content guidance technology in information centric network (ICN). To implement this search scheme, we propose the Function Bread-crumbs (FBC) method that guides queries to appropriate VNF servers by considering the processing orders of VNFs in service chains. The FBC method can be divided into the following two types according to the method of distribution: spread FBC (SFBC) and en-route FBC (EFBC) methods. SFBCs are distributed by limited flooding and EFBCs by the en-route dissemination, which is explained in Sect. 3.2.

In the FBC method, FBCs that have guidance information concerning the VNF servers are distributed to routers. The information in each FBC entry is shown in Table 1. VNF_ID represents the type of VNF, Server_ID is the VNF server that distributed the FBCs. Max_resource shows the maximum size of the resource that the server can supply for the VNF, and Used_resource indicates the current resource utilization for the VNF on the VNF server. Upstream_node and downstream_node represent EFBC message trails, where upstream is the server side that sends the EFBC message. Data are divided into packets, and control messages are represented by a packet.

The routers refer to the time-to-live (TTL) of the FBC and deletes it if it is zero. The router that receives the query checks whether it has an FBC corresponding to the target VNF server. If the router has the FBC, the query is guided to the VNF server specified by the FBC. However, if the router does not have the FBC, the query is forwarded to the control server, according to normal IP routing. A control server has FBCs for all types of VNF servers, and guides the query to the VNF server specified by FBC.

3.1 Service Provision Flow in the Proposed Method

In the proposed method, a series of VNF servers in service chaining are searched for, according to the following procedures (1–5).

1. Data to be processed by service chaining with \( n \) types of VNFs \((v_1, v_2, \ldots, v_n)\) are generated by a user. The destination of the data is also specified by the user.
2. The user sends a query to the control server to find \( v_k = v_1 \).
3. If the SFBC of \( v_k \) found along the route, the query is directed to the VNF server specified in the SFBC; otherwise, if the EFBC of \( v_k \) found along the route, the query is directed to the VNF server specified in the EFBC. Otherwise, the query reaches the control server, and is transferred to the VNF server specified by the control server.
4. Upon completion of the process at the VNF server for \( v_k \), the VNF server (not the user) sends a query to the control server to find \( v_{k+1} \).
5. Procedures 3) to 4) are repeated, until the data have been processed at the VNF server of \( v_n \). The data are finally transferred to the destination.

The details of the procedures 1, 3–5 are explained as follows.

3.1.1 Detail of Procedure 1

Assume that data to be processed by a service chain with \( n \) types of VNFs \((v_1, v_2, \ldots, v_n)\), where the order of specification of \( v_i \) \((i = 1, 2, \ldots, n)\) represents the processing order of these VNFs, are generated by a user. The destination user of the data is also specified by the user.

3.1.2 Detail of Procedure 3

The flowchart of procedure 3 is shown in Fig. 4. The query to search for \( v_k \) is transferred to the control server according to normal IP routing.

When the query encounters a router with an SFBC whose VNF_ID is \( v_k \), the query is transferred by IP routing to the VNF server specified as Server_ID in the SFBC. If there are multiple SFBCs whose VNF_IDS are the same but Server_IDS are different on the router, the SFBC with the minimum Hop_count (as the first priority) or minimum Used_resource (as the second priority) is selected, and the query is transferred to the corresponding VNF server. In this transfer process, the query is no longer guided by another FBC by setting the FBC guidance prohibition flag to the query. This prevents a loop of query forwarding.

On the contrary, if the query encounter a router with EFBC, and not SFBC, whose VNF_ID is \( v_k \), the query is
transferred by IP routing to the VNF server specified as Server_ID in the EFBC. If there are multiple EFBCs on the router whose VNF_IDS are the same but Server_IDS are different, an EFBC is selected in the same manner as in the case of SFBC, and the query is transferred to the corresponding VNF server. In this forwarding, prohibition flag with EFBC as guide, and not FBC, is set, and the query is no longer guided by another EFBC. However, if the query finds an SFBC whose VNF_ID is also \(v_k\), the query is redirected to the server specified as Server_ID in the SFBC. This is because SFBC is distributed in a limited area from the corresponding VNF server by flooding, which is explained in Sect. 3.2. Thus, the SFBC entry implies the existence of the VNF server in a smaller area than that of the EFBC.

Otherwise, no FBC including SFBC and EFBC for \(v_k\) is found on any routers to the control server, and the control server forwards the query to the VNF server, supporting \(v_k\), which is nearest in terms of IP hop count from the control server. If there are multiple candidate VNF servers with a same IP hop count, the VNF server with lowest resource utilization is selected. Each VNF server notifies the control server of its resource utilization for its VNF type whenever its use status changes, and the control server manages the VNF type, resource utilization for the type, and the IP hop count of each VNF server.

Following this, when the selected VNF server receives the query, it sends an ACK back to the user (in the case of \(v_1\) is searched) or the VNF server that has processed \(v_{k-1}\) (in the case of \(v_i\) \((i = 2, 3, \ldots, n - 1)\) is searched), if it can provide the requested resource. On the contrary, if the requested resources cannot be provided, the query is forwarded to the control server, and it searches for another VNF server that can provide \(v_k\). If the control server cannot find any VNF servers, it finally sends a NACK.

Then, when the user/VNF server that has processed \(v_{k-1}\) receives the ACK from the VNF server for \(v_k\), it sends its data to the VNF server, and the VNF server processes \(v_k\) for the data.

### 3.1.3 Detail of Procedures 4 and 5

Once the process has finished, the VNF server sends the query to the control server to search for a server providing \(v_{k+1}\), according to normal IP routing. Such procedures are repeated until the data have been processed at the VNF server of \(v_n\). Thus, the VNF server that processed \(v_k\) searches for a VNF server providing \(v_{k+1}\), where \(k = 1, \ldots, n - 1\).

Finally once the processing of \(v_n\) is completed, the VNF server of \(v_n\) sends the data to the destination user.

### 3.2 Distribution and Update of FBC

FBCs are distributed to routers in the following two ways: limited flooding and en-route dissemination. SFBCs are distributed by the limited flooding, and EFBCs are distributed by the en-route dissemination. When the load of each VNF server is changed, the information of distributed SFBCs is updated by distributing control messages from the VNF server according to the SFBC distribution process described in Sect. 3.2.1. This updating process needs the duration for control messages to reach routers. On the other hand, the information of distributed EFBCs is not updated except for the case that new EFBCs with the same information are deployed based on the process in Sect. 3.2.2. Thus, the proposed method cannot use latest information for the load optimization. However, the proposed method is designed to achieve better load optimization with information that can be obtained in a realistic way, and it is also effective if the performance indices evaluated in Sects. 4 and 6 are improved even with the information.

#### 3.2.1 SFBC Distribution Process

SFBCs are distributed by limited flooding from each VNF server. Specifically, they are distributed using control messages that are distributed by flooding within the range of the specified number of hops from each VNF server. In the distributed SFBCs, Server_ID, VNF_ID, Max_resource, and Used_resource are set according to the server that distributes them, where VNF_ID, Max_resource, and Used_resource represent the VNF type that the server provides, the maximum amount of resources that it can provide for the VNF type, and the amount of used resources for the VNF type, respectively. These items of required informations are included in the control message. Hop_count in each of the SFBCs is set depending on the number of hops from the server. When the amount of used resources for the VNF type changes at the VNF server, another control message to update the SFBCs is distributed by the server in the same manner, and the Used_resource in each SFBCs is updated.

The range of distribution of the SFBC should not be wide because the loads used to create them and forward messages on the routers exponentially increase, and this
must be avoided.

3.2.2 EFBC Distribution Process

To compensate for the limitation in the range of distribution of the SFBC, EFBCs are deployed by en-route dissemination. The order of processing of some VNF types is similar among some service chains due to their characteristics. Thus, the proposed method deploys EFBCs on nodes through which data are passed between various VNFs, instead of nodes between users and the VNF servers, to distribute FBCs more effectively. This is the most distinct procedure of the FBC method to provide service chaining. This enables the system to find VNFs for various service chains without sending queries to the control server. Therefore, the workload on the control server can be reduced, and the response delay in the transfer of query from source user to destination one can be suppressed.

For the specific EFBC distribution scheme, when data processing is completed at a VNF server, the VNF server distributes EFBCs, to redirect queries to itself, by sending a message for EFBC dissemination to the VNF server that has processed the previous type of VNF and the server that will process the next type of VNF in the service chain. The EFBC is registered on each relay node that forwarded the message. In addition to containing the same information as the SFBC, the EFBC contains the information pertaining to Upstream_node and Downstream_node. This is included in the control message. Note that the VNF server for the first VNF type of a service chain also disseminates the EFBCs to itself and the user, and the VNF server for the last VNF type of the service chain disseminates EFBCs to itself and the destination.

3.2.3 FBC Delete Process

The router refers to the TTL of the FBC and deletes it if TTL is zero. Furthermore, when the amount of used resources for the VNF type is zero at the server, the control message to delete the SFBCs is distributed by the server by flooding. Each router that received the message deletes the SFBC. If the router has an EFBC for the VNF type of the server, the router sends a control message to downstream node of the EFBC to delete the EFBCs.

3.3 Data Processing Example with a Series of VNF Servers

Figure 5 shows an example of the dissemination of the EFBC. In this case, the user has requested three types of VNFs: VNF 1, VNF 2, and VNF 3 in order. The arrows indicate the data flow. There are two servers for VNF 1 and among them the one at the bottom of Fig. 5 is used. There are two servers for VNF 3 as well, and the one at the top of Fig. 5 is used. Data sent by the user pass the VNF 1, VNF 2, and VNF 3, and the destination user, in order. Thus, between the user and the VNF 1 server, EFBCs for VNF 1 are disseminated along with the data trail. Moreover, between the VNF1 and VNF 2 servers, EFBCs for VNF 1 and VNF 2 are disseminated separately. Furthermore, between the VNF 2 and VNF 3 servers, EFBCs for VNF 2 and VNF 3 are disseminated. Finally, between the VNF 3 server and the destination user, FBCs for VNF 3 are disseminated. Thus, when the same type of service chaining request occurs subsequently, it is possible to discover these EFBCs.

4. Performance Evaluation in Static VNF Processing Order

The performance of the proposed method in Sect. 3 under the assumption of the static VNF processing order in each service chain was evaluated with computer simulation by varying the range of SFBC through limited flooding.

The proposed method is operated in distributed VNF environment and VNFs are searched in the decentralized manner. At present, few researchers study VNFs search in distributed VNF environment. The method [18] uses game theory. However, users need to know all information about other users and VNF servers for using game theory. On the other hand, we suppose that only the control server knows information about all users and VNF servers in the network though it might not be the latest. Thus, we cannot use the method [18] as comparable conventional method. Thus, for performance comparison, we used a method where the control server selected all VNFs for all the queries without using FBC (No-FBC), and another one where only SFBCs were distributed and EFBC was not used (Only-SFBC).

One objective of our proposed method is to reduce the delays to search VNFs configuring service chains and to transfer data from each user to its destination via VNFs configuring the service chain. The other objective is to reduce the load on the control server. The searching delay can be reduced by decreasing the number of query hops, and data transfer delay can be reduced by decreasing data hops. Thus, we used the number of hops of query and data, and the workload on the control server. Reducing these two delays is important for users to complete their tasks as quickly as possible, and reducing the workload is also important to avoid the processing bottleneck at the control server. In addition, the proposed method uses some control messages, and we should evaluate how the messages increase the network load. Thus, the other performance index is the number
Table 2  Parameter settings

| Parameters                  | Value |
|-----------------------------|-------|
| Number of routers           | 1000  |
| Number of users             | 5000  |
| Number of VNF servers       | 100   |
| Number of overall VNF types | 6     |
| Number of VNF types held by one VNF server | 1 |
| VNF server processing time  | 1000  |
| Packet size                 | 1500 Byte |
| Link capacity               | 5 Mbps |
| Data size                   | 50 MByte |

Table 3  Service chain patterns of users’ request

| Pattern | VNFs |
|---------|------|
| Pattern 1 | A     |
| Pattern 2 | A-B   |
| Pattern 3 | A-B-C |
| Pattern 4 | D-A-B-C |
| Pattern 5 | D-A-E-B-C |
| Pattern 6 | F-B-E-C |

Fig. 6  Number of query hops averaged by each adjacent VNF server pair (including source to $v_1, v_n$ to destination)

Fig. 7  Number of data hops averaged by each adjacent VNF server pair (including source to $v_1, v_n$ to destination)

Fig. 8  Number of messages in the network per unit time

of messages in the network.

The parameters used in this evaluation are shown in Table 2. We used the Waxman model [21] to create the network topology composed of routers and links. Five users were then connected to each router by following the evaluation setting in RBC[19]; each VNF server was connected randomly to a router. Thus, when the range of SFBC distribution was set to one, only the router connected directly to a VNF server held the FBC of the VNF server. Thus, in this case, limited flooding was not performed. The amount of resources on each VNF server was uniformly distributed from 50–100 unit, where processing a VNF according to a user’s request consumed one unit. The interval of request occurrence from each user followed an exponential distribution with an average of 3,600 unit. The service chain of each request was randomly selected from the six patterns in Table 3 with equal probability. There were six VNF types (VNF A, B, C, D, E, and F), and there are 28, 25, 20, 12, 10, and 5 A–F VNFs on the network, respectively. We chose these numbers of the VNF types by taking into account the occurrence frequency of each VNF type in the service chain patterns of Table 3. In all the service chain patterns randomly generated with same probability, each VNF type A-F appears 5, 5, 4, 2, 2, 1 times, respectively (see Table 3). According to the occurrence frequency, 95 VNF servers were assigned to each VNF type A-F as 25, 25, 20, 10, 10, and 5 VNFs. In addition, considering the position in the service chain pattern, VNF A was assigned 3 more servers and VNF D was assigned 2 more servers.

Figures 6 and 7 show the average number of hops averaged by each adjacent VNF server pair (including source to $v_1, v_n$ to destination). In these figures, the case where the range of SFBC distribution was zero corresponds to No-FBC method. From Fig. 6, the Only-SFBC method reduced the number of query hops as the range of SFBC distribution was extended because more queries were guided by the SFBC and did not need to reach the control server. The proposed method further reduced the query hop count because EFBCs in addition to SFBCs guided queries to their VNF servers. From Fig. 7, the number of data hops was reduced by both the Only-SFBC method and the proposed method. This reduction means that the hop count of VNF servers whose VNF types neighbored in the order of processing of the service chain decreased. When a VNF server, which had finished processing data by its VNF, searched for a VNF server providing the next type of VNF in the service chain, the SFBC and EFBC guided the query to nearby VNF server providing the next VNF. Thus, the proposed method can find a series of VNFs whose distances are close. These results of Figs. 6 and 7 show that the average delay to finish services was reduced by the proposed method.

Figure 8 compares the number of control messages required to disseminate and update FBCs, in the network per unit time by different methods. The results show that the increase in the number of messages in the proposed method was not significant compared with that in the Only-SFBC because it was negligible compared with the number of messages needed to distribute SFBCs by flooding. Note that
range of SFBC distribution needed to be limited even in the proposed method to reduce the number of control messages.

Figure 9 depicts the workload on control server. Note that the workload is defined as \( 1 - \frac{\text{number of queries guided by SFBC}}{\text{number of generated queries}} \). As shown in Fig. 9, the proposed method reduced the load on the control server compared with the Only-SFBC method, irrespective of the range of FBC distribution, by using limited flooding. In the “range of SFBC distribution 1”, flooding was not used. However, the workload was reduced drastically because of the EFBC, as many queries guided by SFBCs did not need to use the control server. The result shows that FBCs reduce the load on the control server.

From the above results, it is clear that the proposed method can select VNF servers that are close to one another. These indices evaluated in Figs. 6–9 change non-linearly. Performance improvements of the number of the hops and workload become small as SFBC distribution range is extended. In particular, the performance differences are very small when SFBC distribution range is increased from 4 to 5. On the other hand, the number of control messages increases exponentially as SFBC distribution range is extended. Thus, we conclude that SFBC distribution range 4 is reasonable. An extended method is proposed for the adaptation to exchangeable VNFs.

5. Extended Proposed Method for the Adaptation to Exchangeable VNFs

We consider functions of data processing like big data processing will be used in the future, as one of forthcoming network functions. These functions have exchangeability in some cases. For specific exchangeability, we considered the following three cases.

1. Two VNF types, VNF-A and VNF-B, can be exchanged with each other.
2. A VNF type VNF-A could be processed in an arbitrary order within a certain section of the service chain.
3. VNFs were classified into groups, and VNFs in each group could be processed in an arbitrary order, while the order of each group was preserved.
4. VNFs were classified into groups, and each group could be processed in an arbitrary order while the order of each member of a given group was preserved.

For order exchange, we extended our proposed method. We added two functions to the initially proposed method. One was for VNF search, where the VNF that could be exchanged was also searched for at the same time. The other function was one using which the EFBC was distributed to connect VNF servers of all patterns that could be exchanged.

5.1 VNF Search Method Considering Order Exchange

In this method, a query requests multiple types of VNF. When a query for VNF-A and VNF-B, which can be exchanged, encounters a router with an SFBC whose VNF_ID is VNF-A or an EFBC whose VNF_ID is VNF-B, the query is transferred by IP routing to the VNF server corresponding to the SFBC with the minimum Hop_count (first priority) or the minimum Used_resource (second priority). Moreover, if the query encounters a router with the EFBC, and not SFBC, corresponding to either of the VNF types, the query is transferred by IP routing to the VNF server selected in the same manner as the SFBC. In this forwarding, not the FBC but the EFBC’s guide prohibition flag is set, and the query is no longer guided by another EFBC. However, if the query finds an SFBC whose VNF_ID is also either of the VNF types, the query is redirected to the server specified as Server_ID in the SFBC.

If the query reaches the control server but finds neither an SFBC nor an EFBC, the control server forwards the query according to the same rules.

In the proposed method, there is no problem of the complexity of searching processes when the chain is long or many VNFs are exchangeable. Even in such case, although the number of candidates for search is increased, the increase is linear.

5.2 VNF Distribution Considering Order Exchange

To deal with order exchange, it is effective to distribute the EFBC along each service chaining in which the order is changed. Therefore, in the service chain consisting of \( v_1 \) to \( v_n \), if \( v_k \) and \( v_l \) are exchangeable in terms of order, in addition to normal EFBC distribution, EFBCs are distributed to connect \( v_k \) to \( [v_{k-1}, v_{k+1}] \) and \( v_l \) to \( [v_{l-1}, v_{l+1}] \), respectively, where \( [a, b] \) represents each element; \( a, b \), and \( v_0 \) represents the source user, and \( v_{n+1} \) represents the destination user. Figure 10 shows the additional distribution of EFBC. In this
case, VNF 2 and VNF 4 were exchangeable. The servers VNF 2 and VNF 4 were already connected to VNF 3 server with normal EFBC distribution. Therefore, VNF 2 sent an additional EFBC message to the destination user and VNF 4 sent additional one to VNF 1.

6. Performance Evaluation in Exchangeable VNF Processing Order

As in the evaluation in Sect. 4, the performance was evaluated with computer simulation by varying the SFBC distribution range by limited flooding. We used the same simulation settings as Sect. 4 except for the service chain patterns of user request. The patterns in this section considered exchangeable VNF types. We changed the patterns of user requests as shown in Fig. 11. Patterns 3 to 6 had exchangeable VNF types. In pattern 3, VNF-C could be processed before VNF-A or VNF-B. In patterns 4 and 6, VNFs specified by the dotted bidirectional arrows could be exchanged. In pattern 5, the VNF-D and VNF-A could be exchanged with the other group VNF-B and VNF-C while maintaining order inside the groups. We compared the proposed method between the one that can deal with order exchange (Exchangeable) and the one that cannot (Not-Exchangeable) for quantitative evaluation. This comparison is important to quantitatively evaluate how the performance indices used in Sect. 4 are changed by applying exchangeable VNF processing order.

Figures 12 and 13 show the average number of hops through which the queries and data passed when moving from the previous VNF server (including source) to the next one (including destination). From these figures, it is clear that regardless of the range of flooding, Exchangeable reduced a greater number of hops of queries and data than Not-Exchangeable. Moreover, the reduction in the number of query hops was larger than that in the number of data hops. This is because the data were transmitted through the shortest route to the server found by the query. However, because the query first went to the control server, it did not take the shortest route. Therefore, by reducing the number of queries arriving at the control server, the number of queries transferred from the control server to the VNF server is decreased. This also reduced the number of hops through which queries passed. Reducing the number of data hops means that a closer VNF server could be used by Exchangeable.

Figure 14 shows the number of control messages for FBC. In Exchangeable, the number of EFBC messages in-
increased for order exchange. However, the influence in terms of other aspects of performance improvement was greater than the increase in the number of messages.

We show the workload of a control server in Fig. 15. Due to order exchange, the likelihood of FBCs as guide queries increased in Exchangeable. For this reason, the number of queries that reached a control server decreased. Therefore, the workload of a control server decreased.

7. Conclusion

In this paper, we proposed a method that suppresses delays by considering the constraint pertaining to load on the control server by using in-network guidance technology that pursues the local optimum in a passive and distributed manner. Its effectiveness was shown through extensive computer simulations. In the proposed method, the user can use the neighboring VNF server to form the desired service chain, and at the same time, the network administrator can reduce the workload on the control server and distribute it among the corresponding VNF servers. We intend to conduct a more detailed performance evaluation by proposing a more efficient FBC distribution method to reduce the load on the network in future work.

Acknowledgements

This research was supported by JSPS KAKENHI Grant Number JP17H00734.

References

[1] H. Feng, J. Llorca, A.M. Tulino, D. Raz, and A.F. Molisch, “Approximation Algorithms for the NFV Service Distribution Problem,” IEEE INFOCOM 2017, pp.1–9, May 2017.
[2] X. Fei, F. Liu, H. Xu, and H. Jin, “Towards Load-Balanced VNF Assignment in Geo-distributed NFV Infrastructure,” Proc. IEEE/ACM 25th International Symposium on Quality of Service (IWQoS 2017), pp.1–10, June 2017.
[3] E.J. Rosensweig and J. Kurose, “Breadcrumbs: efficient, best-effort content location in cache networks,” Proc. IEEE INFOCOM 2009, pp.2631–2635, April 2009.
[4] E.J. Rosensweig and J. Kurose, “Breadcrumbs: efficient, best-effort content location in cache networks,” UMass Amherst, MA, Tech. Rep. UM-CS-2009-005, pp.2631–2635, Aug. 2008.
[5] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouren, F. De Turck, and R. Boutaba, “Network Function Virtualization: State-of-the-Art and Research Challenges,” IEEE Communications Surveys & Tutorials, vol.18, no.1, pp.236–362, Sept. 2015.
[6] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, “Network function virtualization: Challenges and opportunities for innovations,” IEEE Communications Magazine, vol.53, no.2, pp.90–97, Feb. 2015.
[7] R. Kawashima, H. Nakayama, T. Hayashi, and H. Matsu, “Evaluation of Forwarding Efficiency in NFV-Nodes Toward Predictable Service Chain Performance,” IEEE Transactions on Network and Service Management, vol.14, no.4, pp.920–933, Dec. 2017.
[8] N. Pitaev, M. Farkner, A. Leivadeasy, and I. Lombadaris, “Multi-VNF Performance Characterization for Virtualized Network Functions,” Proc. IEEE Conference on Network Softwarization (NetSoft 2017), pp.1–5, July 2017.
[9] E.H.-Valencia, S. Izzo, and B. Polonsky, “How will NFV/SDN transform service provider opex?,” IEEE Network, vol.29, no.3, pp.60–67, June 2015.
[10] S. Kumar et al., “Service Function Chaining Use Cases In Data Centers draft-ietf-sfc-dc-use-cases-05,” https://tools.ietf.org/html/draft-ietf-sfc-dc-use-cases-05, Feb. 2017.
[11] J.G. Herrera and J.F. Botero, “Resource Allocation in NFV: A Comprehensive Survey,” IEEE Transactions on Network and Service Management, vol.13, no.3, pp.518–532, Aug. 2016.
[12] M.T. Beck and J.F. Botero, “Coordinated Allocation of Service Function Chains,” Proc. IEEE GLOBECOM 2015, pp.1–6, Dec. 2015.
[13] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, “Network Function Placement for NFV Chaining in PacketOptical Datacenters,” Journal of Lightwave Technology, vol.33, no.8, pp.1565–1570, Jan. 2015.
[14] S. Ahvaz, M.M. Mirzaei, J. Leguay, E. Ahvar, A.M. Medhat, N. Crespi, and R. Glitho, “SET: a Simple and Effective Technique to improve cost efficiency of VNF placement and chaining algorithms for network service provisioning,” Proc. IEEE NetSoft 2018, pp.293–297, June 2018.
[15] ETSI GS NFV-MAN 001, “Network Functions Virtualisation (NFV); Management and Orchestration,” http://www.etsi.org/, Dec. 2014, ETSI ISG.
[16] R. Mijumbi, J. Serrat, J.-L. Gorricho, S. Latre, M. Charalamides, and D. Lopez, “Management and orchestration challenges in network functions virtualization,” IEEE Communications Magazine, vol.54, no.1, pp.98–105, Jan. 2016.
[17] M. Yoshida, W. Shen, T. Kawabata, K. Minato, and W. Imajuku, “MORSA: A multi-objective resource scheduling algorithm for NFV infrastructure,” Network Operations and Management Symposium, pp.1–6, Sept. 2014.
[18] S. D’Oro, L. Galluccio, S. Palazzo, and G. Schembra, “Exploiting Congestion Games to achieve Distributed Service Chaining in NFV Networks,” IEEE Journal on selected areas in communications, vol.35, no.2, pp.407–420, Feb. 2017.
[19] K. Nishii, Y. Tanigawa, and H. Tode, “Decentralized Resource Sharing Platform Exploiting In-Network Guidance Information,” Proc. IEEE CCNC 2014, pp.287–292, Jan. 2014.
[20] K. Nishii, Y. Tanigawa, and H. Tode, “A Participating Fine-Granular Cloud Computing Platform with In-Network Guidance,” IEICE Trans. Commun., vol.E98-B, no.6, pp.1008–1017, June 2015.
[21] B.M. Waxman, “Routing of multipoint connections,” IEEE Journal on selected areas in communications, vol.6, no.9, pp.1617–1622, Dec. 1998.

Yutaro Oda received the B.E. degree in information networking from Osaka Prefecture University in 2017. At present, he is a student in the Department of Computer Science and Intelligent Systems, Graduate School of Engineering, Osaka Prefecture University. His current research interests includes next generation network for cloud computing. Mr. Oda is a member of IEICE Japan.
Yosuke Tanigawa received the B.E., M.E., and Ph.D. degrees in information networking from Osaka University, in 2005, 2007, and 2009, respectively. He was appointed as an assistant professor in the Department of Computer Science and Intelligent Systems, Graduate School of Engineering, Osaka Prefecture University in April 2009. Since April 2016, he has been an associate professor in the same department. His current research interests include access controls, QoS-aware network controls, and power saving in wireless network. Dr. Tanigawa is a member of IEEE and IEICE Japan.

Hideki Tode received the B.E., M.E., and Ph.D. degrees in communications engineering from Osaka University in 1988, 1990, and 1997, respectively. He was adopted as assistant professor at the Department of Communications Engineering, Osaka University from a doctor’s course student on Dec. 1, 1991. In 1998 and 1999, he shifted and promoted to a lecturer and associate professor in the Department of Information Systems Engineering, and, in 2002, shifted in the Department of Information Networking, Graduate School of Information Science and Technology, Osaka University. From 2008, he is a professor in the Department of Computer Science and Intelligent Systems, Graduate School of Engineering, Osaka Prefecture University. His current research interests include high speed router architecture, QoS-aware network controls, optical network architecture and application-level contents distribution technologies. Dr. Tode is a member of IEEE, IEICE Japan.