Biochemical-inspired autonomous control of virtualized network functions

Ryota Kurokawa
Graduate School of
Information Science and Technology
Osaka University
1-5 Yamadaoka, Suita,
Osaka 565-0871, Japan
Email: r-kurokw@ist.osaka-u.ac.jp

Go Hasegawa
Cybermedia Center
Osaka University
1-32, Machikaneyama-cho, Toyonaka,
Osaka 560-0043, Japan
Email: hasegawa@cmc.osaka-u.ac.jp

Masayuki Murata
Graduate School of
Information Science and Technology
Osaka University
1-5 Yamadaoka, Suita,
Osaka 565-0871, Japan
Email: murata@ist.osaka-u.ac.jp

***Abstract*—In Network Function Virtualization (NFV), various Virtual Network Functions (VNFs) are placed on general-purpose servers. To efficiently operate the NFV system, the placement of VNFs to servers, resource allocation to each VNF, and flow routes are determined carefully and adaptively to handle environmental fluctuations. Our research group has proposed a construction method of service space in virtualized network system, based on biochemical-inspired tuple space model. In this paper, we apply the method to an NFV system to adaptively configure and control VNFs in a distributed manner. We also describe the implementation design of the proposed method, based on the standardization activities to achieve Service Function Chaining. We finally present an application scenario of the proposed method to confirm the effectiveness of the implementation design of the proposed method.**

***Index Terms*—Network Function Virtualization, Software Defined Network, Service Function Chaining, Network Service Header, biochemical reactions**

I. Introduction

Due to the wide and rapid spread of smartphones and tablets, and the development of Internet of Things (IoT), the number of devices connected to the network is increasing. As a result, network services have become more diverse, and network traffic has also increased rapidly. In general, to launch a new network service, new dedicated hardware devices are required. It requires space and power, causing a decrease in revenue and an increase in energy consumption. It also results in low flexibility to deal with system failures, and maintenance and operation of hardware.

Network Function Virtualization (NFV) is considered as one possible technique for resolving such problems [1]. Figure 1 shows a network system based on NFV. Various Virtual Network Functions (VNFs) are placed on general-purpose servers. Since network functions that have been achieved with dedicated hardware are executed as software, it is possible to suppress operational and capital expenditures, and flexibly respond to environmental fluctuations. Multiple VNFs may share the resource of one server or one VNF may be distributed to multiple servers to provide services throughout the network [2] [3].

A flow receiving NFV service may have a Service Function Chaining (SFC) request that describes the order of VNFs to be applied to the flow. As depicted in Figure 1, a flow arriving at the NFV system receives NFV services in accordance with the SFC request and exits the system. Therefore, to efficiently operate the NFV system, placement of VNFs to servers, resource allocation to each VNF, and flow routes are determined carefully in accordance with SFC requests, traffic amount of the flows, and server resources.

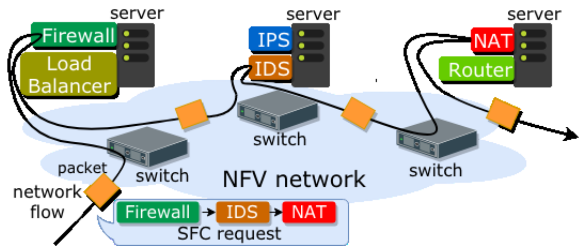

Fig. 1. NFV system and SFC

Our research group has proposed a construction method of service space in virtualized network system, based on biochemical-inspired tuple space model [4] [5]. In this method, we consider a server as a tuple space, and express service requests, service demands and service resources as chemical substances in the tuple space. We then describe the behavior of servers as biochemical reaction equations in the tuple space. Furthermore, by connecting multiple tuple spaces and configuring a network, we represent the movement and spread of services and requests in a large scale network system composed of multiple servers. Since biochemical reaction equations are defined and executed independently in each tuple space, it is suitable for achieving autonomous and decentralized behavior. We consider that one of possible application of the above method is an NFV system. However, to operate an NFV system using the proposed method, it is extended to handle flow routes in accordance with SFC requests, and server resource limitation. In our previous work with the extended method, the effectiveness of the method has been evaluated only with computer simulation and a simple experimental environment.

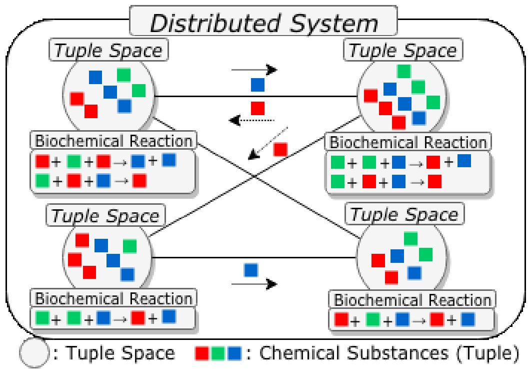

Fig. 2. Tuple space model using biochemical reactions [4]

In this paper, we apply the above-mentioned service space construction method to configure and control an NFV system, and explain how to incorporate the proposed method to NFV framework. We show the detailed implementation environment using Open Platform for NFV (OPNFV) [6]. In addition, we describe the implementation design of SFC using Network Service Header (NSH) [7] with OpenFlow networks. We finally present an application scenario of the proposed method to confirm the effectiveness of the described design.

II. VNF Control by Tuple Space Model using Biochemical Reactions

A tuple space model in [4] is a model that describes a distributed system. Figure 2 depicts the tuple space model used in this paper. Each component of a distributed system is modeled as a tuple space. A tuple space is defined as a cell where biochemical reactions take place. Tuples in the tuple space correspond to chemical substances, and the amount of tuples corresponds to the concentration of chemical substances. By defining biochemical reactions in tuple spaces, various behaviors such as increase and decrease of tuples can be determined. In addition, it is possible to achieve the interaction among multiple tuple spaces by defining biochemical reactions that describe the diffusion and movement of tuples among tuple spaces. Since biochemical reactions in each tuple space occur independently, autonomous decentralized behaviors can be described.

To apply the above model to NFV, we associate a tuple space with a server that executes VNFs. VNFs, flow packets, and server resources are associated with tuples in tuple spaces. In what follows, we present biochemical reaction equations that achieve various behaviors in the NFV system.

A. *Service Function Chaining (SFC)*

An SFC request of a flow, represented by a series of VNFs, $f_1, f_2, \cdots, f_{end}$, is described as Equation (1).

$$c = \{f_1, f_2, f_3, \cdots, f_{end}\} \tag{1}$$

When VNF f_1 is executed to the flow with an SFC request c, c changes as Equation (2).

$$c \leftarrow c \backslash \{f_1\} = \{f_2, f_3, \cdots, f_{end}\} \tag{2}$$

A VNF that is executed at first in c is represented by $f^1(c)$. In Equation (2), $f^1(c) = f_2$. In what follows, the subscript f of chemical substances represents a VNF, subscript c represents an SFC request, and subscript t represents a server.

B. *Resource allocation and execution of VNFs*

When a packet of a flow with an SFC request c arrives at a server, VNF $f^1(c)$ is applied to the packet. Then, when c is composed of one VNF, the packet disappears. On the other hand, when c is composed of multiple VNFs, the applied VNF is deleted from c as in Equation (2). It is also required that VNFs in low demand have low priority in the server and those in high demand have high priority to be executed. The above behaviors are described by Reaction Equations (3) and (4).

$$VNF_{f^1(c)}|PKT_c \xrightarrow{r_{us}} \begin{cases} VNF_{f^1(c)}|VNF_{f^1(c)}|PKT_{c\backslash\{f^1(c)\}} \\ |toserve(VNF_{f^1(c)}, PKT_c) \quad (c\backslash\{f^1(c)\} \neq \emptyset) \\ VNF_{f^1(c)}|VNF_{f^1(c)} \\ |toserve(VNF_{f^1(c)}, PKT_c) \quad (c\backslash\{f^1(c)\} = \emptyset) \end{cases} \tag{3}$$

$$VNF_f \xrightarrow{r_{ds}} 0 \tag{4}$$

In the above Equations, substance $VNF_{f^1(c)}$ indicates the VNF to be applied for a flow. A VNF with a large concentration value means that its execution is highly demanded. Substance PKT_c represents packets constituting a flow with c. Substance $toserve(VNF_{f^1(c)}, PKT_c)$ indicates results of applying the VNF to packets of a flow with c. r_{us} and r_{ds} are the rate coefficients of Reaction Equations (3) and (4), respectively, to determine the rate of reactions. Reaction Equation (3) indicates that a VNF is executed to packets of a flow on a server, and the concentration of VNF increases to represent the demand increase for the corresponding VNF. Reaction Equation (4) represents the decay of VNFs.

The execution rate of a biochemical reaction is determined in proportion to the product of its reactant's concentrations. Therefore, in Reaction Equation (3), as the concentrations of VNF and PKT increase, the reaction rate increases without limitation. However, in general, servers have their performance constraints determined by server resources such as CPU capacity and memory size. To describe such constraints, Reaction Equation (3) is extended into Reaction Equations (5) - (7) by applying enzyme-catalyzed reactions mechanism [8].

$$VNF_{f^1(c)}|PKT_c|RSRC_t \xrightarrow{r_{us1}} MEDIATE_{f,t} \tag{5}$$

$$MEDIATE_{f,t} \xrightarrow{r_{us2}} VNF_{f^1(c)}|PKT_c|RSRC_t \tag{6}$$

$$MEDIATE_{f,t} \xrightarrow{r_{us3}} \begin{cases} VNF_{f^1(c)}|VNF_{f^1(c)}|PKT_{c\backslash\{f^1(c)\}}|RSRC_t \\ |toserve(VNF_{f^1(c)}, PKT_c) \quad (c\backslash\{f^1(c)\} \neq \emptyset) \\ VNF_{f^1(c)}|VNF_{f^1(c)}|RSRC_t \\ |toserve(VNF_{f^1(c)}, PKT_c) \quad (c\backslash\{f^1(c)\} = \emptyset) \end{cases} \tag{7}$$

The concentration of substance $RSRC_t$ represents the amount of available resources of a server t. The concentration of substance $MEDIATE_{f,t}$ represents the amount of resources of a server t allocated to VNF f. r_{us1}, r_{us2} and r_{us3} are the rate coefficients for Reaction Equations (5), (6) and (7), respectively. We exploit the enzyme-catalyzed reactions mechanism to describe the server resource limitation, where an enzyme corresponds to $RSRC$ in the Reaction Equations.

Reaction Equations (5) and (6) indicate that server resources are allocated in accordance with the demand of each VNF, and that the allocation is controlled by the concentration of $RSRC$. Reaction Equation (7) indicates that VNF f is executed on the basis of the amount of allocated resources.

C. Diffusion of VNFs

To describe the diffusion of highly-demanded VNFs to other servers, Reaction Equation (8) is introduced.

$$VNF_f \xrightarrow{r_{ms}} VNF_f^{\rightsquigarrow} \tag{8}$$

r_{ms} is the rate coefficient for Reaction Equation (8). This Reaction Equation indicates that a highly-demanded VNF in a server diffuses to the surrounding connected servers at a rate proportional to its concentration. The diffusion destination of VNFs are stochastically determined in accordance with the concentration of VNF at connected tuple spaces. As a result, highly-demanded VNFs are distributed to multiple servers.

D. Packet forwarding

When packets remain unprocessed in the server due to a lack of server resource for corresponding VNF, the packets should move to another server. Furthermore, the forwarding direction of packets from each tuple space should be determined in a distributed manner so that the packets would approach a server executing corresponding VNFs with enough resources. To achieve these behaviors, we exploit a gradient field to determine the moving directions of packets. A gradient field for each VNF is constructed on the basis of the VNFs' demand and the available resources on each server. We then determine the moving direction of packets in accordance with the gradient field. For that purpose, Reaction Equations (9) - (12) are introduced.

$$VNF_f|RSRC_t \xrightarrow{r_{rg}} VNF_f|RSRC_t|GRAD_f \tag{9}$$
$$GRAD_f \xrightarrow{r_{dg}} 0 \tag{10}$$
$$GRAD_f \xrightarrow{r_{mg}} GRAD_f^{\rightsquigarrow}(GRAD_f^{-}) \tag{11}$$
$$PKT_c \xrightarrow{r_{mf}} PKT_c^{\rightsquigarrow}(GRAD_{f^1(c)}^{+}) \tag{12}$$

Substance $GRAD_f$ establishes a gradient field for VNF f. r_{rg}, r_{dg}, r_{mg} and r_{mf} are the rate coefficients for Reaction Equations (9), (10), (11), and (12), respectively. Reaction Equation (9) indicates that $GRAD$ is generated at a rate proportional to the concentrations of VNF and $RSRC$. Reaction Equation (10) indicates that $GRAD$ decays at a rate proportional to its concentration. Reaction Equation (11) indicates that $GRAD$ spreads to the surrounding servers with smaller concentration of $GRAD$. Therefore, the gradient field is constructed so that the server providing VNFs with enough resources becomes a summit with the largest concentration of $GRAD$, and the surrounding servers have smaller concentration of $GRAD$ in accordance with the distance from the summit. Reaction Equation (12) describes the movement of PKT to the surrounding servers with large concentration of $GRAD$. The forwarding direction of packets are stochastically determined in accordance with the concentration of $GRAD$ at connected tuple spaces. Figure 3 depicts the movement of packets with the SFC request $c = \{f_0, f_1\}$. Gradient fields are respectively generated for each VNF. First, packets move in the direction of the summit of the gradient field for f_0. Then, after applying VNF f_0 to the packets, they move in the direction of the summit of the gradient field for f_1.

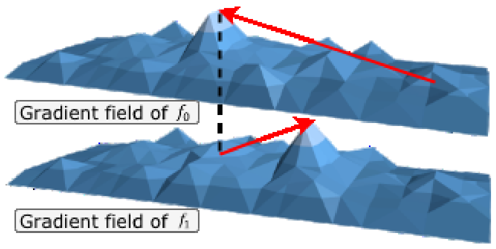

Fig. 3. Movement of a packet in accordance with the gradient fields

E. Coexistence of multiple VNFs

When multiple VNFs coexist on one server, it is required to share server resources by allocating them in accordance with the demand of each VNF. Therefore, we define the above-mentioned biochemical reaction equations for each VNF.

F. Update chemical substance concentration

In the NFV system, the concentrations of the above-mentioned substances are determined as follows. The initial concentration value of VNF is configured in accordance with the initial location of the VNF in the system. The concentration of PKT is determined in accordance with the number of packets arriving at a server and that leaving the server. The initial concentration value of $RSRC$ is determined on the basis of CPU resources in a server. The concentrations of $MEDIATE$ and $GRAD$ change in accordance with the reactions in each server. The concentration of $MEDIATE$ corresponds to the resource utilization ratio allocated to each VNF. The concentration of $GRAD$ determines the direction of packet movement, as explained in Subsection II-D. As a result, CPU resources are allocated to each VNF in accordance with the concentration of $MEDIATE$. The packets also move to servers where highly-demanded VNFs exist in accordance with the concentration of $GRAD$. In addition, VNFs are placed in the server with highly-demanded VNFs in accordance with the concentration of VNF.

III. Implementation Design of the Proposed Method with the NFV Framework

We describe the implementation design of the method proposed in Section II, based on the NFV framework [9] and its integration with SDN [10] proposed by ETSI ISG. In the NFV framework, there are three main components: VNF, NFV Infrastructure (NFVI), and NFV Management and Orchestration (NFV MANO). In [10], SDN Controller is utilized to manage the physical resources in NFVI. SDN Switches are included in network resources in NFVI.

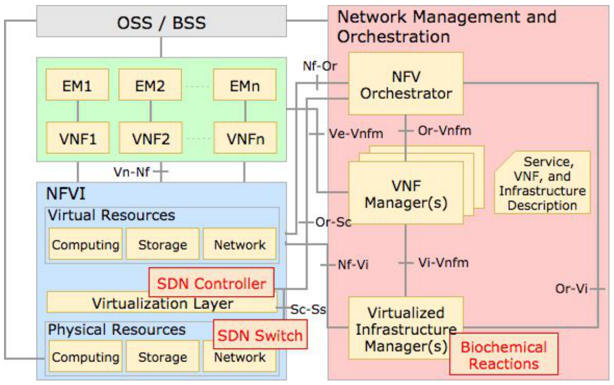

Fig. 4. Proposed method in NFV/SDN framework

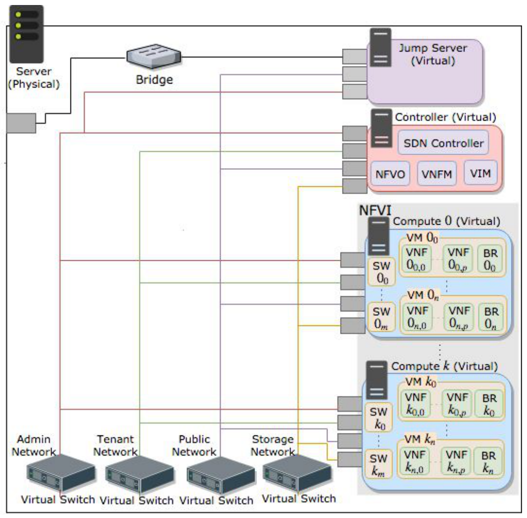

Fig. 5. The system configuration of OPNFV

In what follows, we describe the placement of the functions to apply the proposed method to the ETSI NFV framework and show the detailed implementation environment.

A. Function Placement of the proposed method

Figure 4 depicts the placement of the functions to apply the proposed method to the NFV framework. For the proposed method, VIM is extended to include the management function of biochemical reaction equations ("Biochemical Reactions" in the figure).

By placing the functions of the proposed method in the framework, the following communication interfaces between functions are added. Nf-Vi is used to monitor the state of NFVI and biochemical reaction equations by VIM. Nf-Or is used to receive the state information of biochemical reaction equations by NFVO. Or-Sc is used to receive flow routes by SDN Controller.

We adopt OPNFV [6] to implement the NFV framework. OPNFV aims at implementing the NFV framework by integrating OSS such as OpenStack [11], OpenDaylight (ODL) [12], Open vSwitch (OVS) [13] and Kernel-Based Virtual Machine (KVM) [14]. In OPNFV, since OpenFlow is used as a southbound protocol for SDN, we introduce the implementation design of the proposed method with OpenFlow.

B. Implementation environment of the proposed method

Figure 5 depicts the system configuration of OPNFV and the placement of functions shown in Figure 4. In Figure 5, the NFV framework is implemented on one physical machine, and there are multiple VMs and virtual switches. It includes three types of VMs: Jump Server, Controller, and Compute. Jump Server is utilized for installing and maintaining Controller, Compute, and networking environment in the system. Controller is utilized for implementing NFV MANO and SDN Controller on NFVI. Compute is utilized for implementing VNFs and NFVI. VMs for executing VNFs can be deployed on Compute. There are four types of networks in the system: Admin, Tenant, Public and Storage. Admin network is used for installing and maintaining the NFV system. Tenant network is used for network traffic generated by tenants on the NFV system. Public network is used to connect to external networks. Storage network is used for I/O processing of storage.

In our implementation design, NFVO, VNFM and VIM are implemented on Controller using OpenStack. SDN Controller and Switch are implemented on Controller with ODL and Compute with OVS, respectively. VNFs are implemented on VMs on Compute. NFVI is constructed with Compute, and virtual resources are provided with KVM.

In Figure 5, BR is a program that implements the tuple space of the proposed method, and runs as one process on each VM hosting VNFs on Compute. BR creates a tuple space and executes biochemical reaction equations. The concentration of PKT can be updated in accordance with the flow rate monitored at VNF. However, considering the software implementation of existing network functions and implementation difficulties, one possible alternative is to monitor the flow rate at the corresponding port of the SDN Switch. Resource allocation to VNFs, and activation and deactivation of VNFs can be performed by each VM on Compute executing VNFs in a distributed manner, in accordance with the concentrations of $MEDIATE$ and VNF of the corresponding tuple space. For ease of implementation, such VNF control can be conducted at VIM on the Controller in a centralized manner.

In the proposed method, the moving direction of packets is stochastically determined as explained in SubSection II-D. However, such stochastic behavior may cause a routing loop in the actual network environment. Therefore, in our implementation, flow routes are determined by SDN Controller in a centralized manner. In detail, NFVO collects the information of the concentration of $GRAD$ at each tuple space, and determines the active flow routes. The flow routes are then installed in SDN Switches via the SDN Controller. In the next section, we explain the implementation of SFC in our system, which is an important issue in determining flow routes.

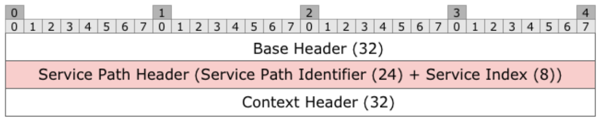

Fig. 6. Network Service Header (NSH) [7]

IV. Service Function Chaining

In this section, the mechanism of Network Service Header (NSH) [7] proposed by IETF is briefly explained. The implementation design of SFC with NSH is then described.

A. Network Service Header (NSH)

NSH is a header added to flow packets to control the flow with an SFC request in the NFV system. Figure 6 depicts the format of NSH, which is composed of three fields: Base Header, Service Path Header, and Context Header. Base Header contains the basic information of NSH such as version, header length, and payload information. Service Path Header contains the identifier of a flow route and the state of SFC of the flow. Context Header contains the metadata. Service Path Header is composed of Service Path Identifier (SPI) and Service Index (SI). SPI has an ID of Service Function Path (SFP), which is described below. SI has the remaining number of functions to be executed to the flow. Therefore, with a pair of SPI and SI, the next function to be executed to the flow can be identified.

Figure 7 depicts the implementation design of SFC with NSH, as described in RFC 8300 [15]. In the figure, Service Function (SF) means the function to be executed to flow packets. Service Function Forwarder (SFF) forwards flow packets to the specified SF or another SFF. SFP represents a flow route with detailed location of servers in which required SFs exist, while an SFC request only includes the chain of functions. SFP is used for forwarding packets to the designated server in accordance with the SFC request. Service Classifier (SC) is located at the entrance of the NFV system, which determines an SFP for a flow, and inserts an NSH into the packets.

In Figure 7, we consider the situation where flow packets having an SFC request of {Firewall $\rightarrow$ NAT} arrive at the NFV system. Since there are two SFs of firewall (“Firewall0” and “Firewall1”) and one SF of NAT (“NAT0”) in the system, the SFP for the flow could be {Firewall0 $\rightarrow$ NAT0} or {Firewall1 $\rightarrow$ NAT0}. These candidates are managed by the SC. The SC determines an SFP from the candidates for the flow and inserts NSH into the flow packets arriving at the SC. Here, NSH includes SPI of 0x000064 and SI of 0x10. When an SF is executed to a flow packet, the SF decrements the value of SI by one and forwards the packet to corresponding SFF. When SI becomes zero, the SFF deletes the NSH from the packet. Otherwise, the SFF forwards the packet to the next SF.

B. Implementation design of NSH

In this work, for ease of implementation, we only utilize Service Path Header for NSH and other two fields (Base

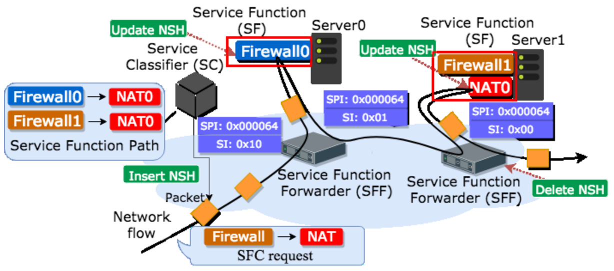

Fig. 7. Implementation design of SFC with NSH in RFC 8300 [15]

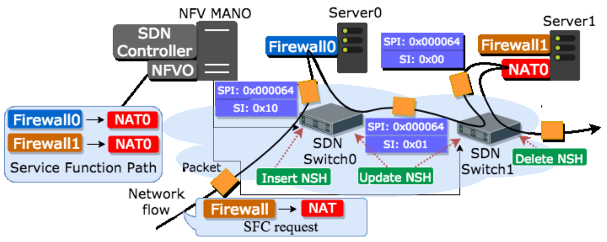

Fig. 8. Implementation design of SFC with NSH in this paper

Header and Context Header) are not implemented. To handle NSH in the NFV system, we exploit encapsulation and decapsulation functions of OpenFlow. Note that the latest version of OVS can encapsulate and decapsulate packets with NSH.

Figure 8 depicts the implementation of SFC with NSH in our implementation. SFs and SFFs are respectively provided as VNFs on the servers and SDN Switches. SFP is managed by NFVO on NFV MANO. SC is implemented in NFVO and SDN Controller on NFV MANO.

C. Stochastic determination of flow routes

In NFV, a route of packets is generally determined in a flow-by-flow manner. On the other hand, in the proposed method, as described in Section II-D, it is stochastically determined in a packet-by-packet manner. Therefore, the stochastic determination of flow routes is proposed to fill the gap.

Figure 9 depicts the stochastic determination of a flow route with the proposed method. In the figure, we consider the situation where flow packets having an SFC request of {Firewall $\rightarrow$ NAT} arrive at the NFV system. BRs are running on servers (“Server0” and “Server1”) to create tuple spaces. Since there are two VNFs (SFs) for firewall (“Firewall0” and “Firewall1”) and is one SF of NAT (“NAT0”), the SFP for a flow could be {Firewall0 $\rightarrow$ NAT0} or {Firewall1 $\rightarrow$ NAT0}. When flow packets arrive at the SDN Switch, the SFP for the flow is stochastically determined on the basis of the concentrations of $GRAD$ at both VNFs. In the figure, since the concentraions of $GRAD$ at Firewall0 and Firewall1 are respectively 2,000 and 1,000, {Firewall0 $\rightarrow$ NAT0} or {Firewall1 $\rightarrow$ NAT0} is assigned to the flow with the probability of 0.66 and 0.33, respectively.

V. Application Scenario

To exhibit the effectiveness of the proposed method, we describe an application scenario to a video streaming. In this

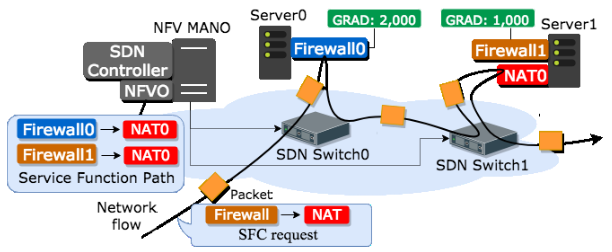

Fig. 9. Stochastic determination of flow route with the proposed method

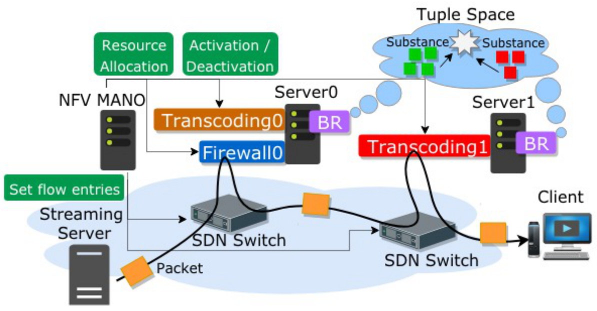

Fig. 10. Implementation design of the proposed method in video streaming application

scenario, as depicted in Figure 10, a client sends an HTTP request to the server to watch a video. Flow packets as an HTTP response from the server are sent to the client. To achieve secure and flexible video streaming service, firewall and video transcoding functions are required. We then apply the NFV system with the proposed method to the application. By this scenario, we expect that resource allocation to each VNF, adaptive load distribution of VNFs, and diffusion and aggregation of VNFs can be confirmed to assess the capability of the proposed method.

Figure 10 depicts the video streaming system with the proposed method. In the figure, Streaming Server is a server providing video streaming service. Firewall0 is a VNF providing the firewall function. Transcoding0 and Transcoding1 are VNFs providing the transcoding function of video. BR is a program that implements the tuple space of the proposed method. Server0 and Server1 are servers for hosting the VNFs and executing BR. Client is a client watching the video. NFV MANO is a server achieving NFV MANO shown in Figure 4. Flow packets sent from Streaming Server arrive at the Client after receiving the NFV service. Since firewall and video transcoding functions should be applied to flow packets, the SFC request becomes $\{$Firewall $\rightarrow$ Transcoding$\}$. The SFP for the flow could be $\{$Firewall0 $\rightarrow$ Transcoding0$\}$ or $\{$Firewall0 $\rightarrow$ Transcoding1$\}$ because there are functions of one firewall and two video transcoding in the NFV system.

In Server0 and Server1, BR respectively creates a tuple space and executes biochemical reaction equations. The concentration of PKT can be updated in accordance with the flow rate arriving at each server. Resource allocation to each VNF is executed in accordance with the concentration of $MEDIATE$ of the corresponding tuple space. $\{$Firewall0 $\rightarrow$ Transcoding0$\}$ or $\{$Firewall0 $\rightarrow$ Transcoding1$\}$ is assigned to the flow in accordance with the concentration of $GRAD$. Furthermore, activation and deactivation of VNFs can be executed in accordance with the concentration of VNF.

VI. Conclusion and Future Work

In this paper, we applied the construction method of service space in virtualized network system, based on biochemical-inspired tuple space model, to the NFV system, and examined the implementation design of the proposed method with the NFV framework. Specifically, we defined chemical substances and biochemical reaction equations to describe various behaviors in the NFV system. We then described the placement of the proposed method in the NFV framework and showed the detailed implementation environment. Furthermore, we described the implementation design of SFC with NSH and presented an application scenario.

For future work, we plan to implement and evaluate the proposed method on the basis of the described design in this paper. In addition, it is also important to extend the proposed method to describe the effect of the propagation delay and link bandwidth between tuple spaces. Furthermore, it is necessary to achieve discrete resource allocation to VNFs with the proposed method to accommodate the CPU core-based resource control in the current cloud computing environment.

References

[1] ETSI, "Network Function Virtualisation - White Paper 1." available at https://portal.etsi.org/nfv/nfv_white_paper.pdf.
[2] X. Li and C. Qian, "A Survey of Network Function Placement," in *2016 13th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pp. 948–953, Jan. 2016.
[3] 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, Sep. 2016.
[4] M. Viroli, M. Casadei, S. Montagna, and F. Zambonelli", "Spatial Coordination of Pervasive Services Through Chemical-Inspired Tuple Spaces," *ACM Trans. Auton. Adapt. Syst.*, vol. 6, no. 2, pp. 1–24, June. 2011.
[5] G. Hasegawa and M. Murata", "Biochemically-inspired Method for Constructing Service Space in Virtualized Network System," *in Proceedings of ICIN 2016*, Mar. 2016.
[6] "Home - OPNFV." available at https://www.opnfv.org/.
[7] "Network Service Header (NSH)." available at https://www.rfc-editor.org/rfc/pdfrfc/rfc8300.txt.pdf.
[8] R. Goldberg, Y. B Tewari, and T. Bhat, "Thermodynamics of enzyme-catalyzed reactions," *Science Direct*, vol. 20, no. 16, pp. 2874–2877, Dec. 2004.
[9] ETSI GS NFV 002, "Network Functions Virtualisation (NFV); Architectural Framework." available at http://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.02.01_60/gs_NFV002v010201p.pdf.
[10] ETSI GS NFV 005, "Network functions virtualisation (nfv); ecosystem; report on sdn usage in nfv architectural framework." available at https://www.etsi.org/deliver/etsi_gs/NFV-EVE/001_099/005/01.01.01_60/gs_NFV-EVE005v010101p.pdf.
[11] "Open source software for creating private and public clouds.." available at https://www.openstack.org/.
[12] "Home - OpenDaylight." available at https://www.opendaylight.org/.
[13] "Open vSwitch." available at https://www.openvswitch.org/.
[14] "KVM." available at https://www.linux-kvm.org/page/Main_Page.
[15] "Network Service Header (NSH)." available at https://www.rfc-editor.org/rfc/pdfrfc/rfc8300.txt.pdf.