International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 12 (2017) pp. 3439-3443

A Study on Reinforcement Learning based SFC Path Selection in SDN/NFV

Hye-Jin Ku[1], Joo-hee Jung[2] and Gu-In Kwon[3*]
Department of Computer Engineering, Inha University, 100 Inharo, Nam-Gu, Incheon 22212, Korea.

*(*Corresponding author 0000-0003-0065-4330)*

[3]ORCID: 0000-0003-0065-4330

Abstract

In the era of Internet of Things, SDN (Software Defined Networking) and NFV (Network Function Virtualization) technologies are introduced in order to secure network flexibility and process massive data efficiently. In particular, SFC (Service Function Chaining) technology makes it possible to deliver traffic flow to service functions in sequence. It is being researched with much interest. This study proposes a reinforcement learning based algorithm which selects a path from multiple paths for one service function chain in the SDN/NFV environment by taking into account the usage of each node and the link bandwidth between the nodes. The proposed algorithm selects a proper path depending on network conditions and thereby guarantees efficient service chaining environment.

Keywords: SDN, NFV, SFC, Reinforcement Learning

INTRODUCTION

Internet of Things (IoT) is the technology of transmitting data in real time between things with sensors. In the era of IoT, the scalability and flexibility of network infrastructure are of significance. If SDN (Software Defined Networking) and NFV (Network Function Virtualization) technologies are applied to the IoT network for the IoT service establishment and application, it is possible to process rapidly increased data of terminal devices and sensors efficiently, and save the cost and time for network establishment and operation in new service introduction.

In SDN, network control function is separated from physical network. Through software programming in SDN, the next-generation networking technology makes it possible to set up and control a network path configuration and process complicated operation & management conveniently. NFV is a virtualization technology that virtualizes the main functions for network (L4, firewall, etc.) in a commodity server with high-performance, instead of using expensive dedicated equipment. In other words, SDN supports easier management and setting of network, and NFV supports easy establishment and scalability of network. Therefore, a network is established using NFV, and after that SDN can be used for maintenance and management of the network [1][2]. In the point that NFV helps to add or update a network function easily and reduces the burden of network operation, SFC (Service Function Chaining) technology draws a lot of attention.

SFC technology processes network traffic and delivers virtualized network functions and services along with the one sequential chain. Depending on many different requirements, one logical chain is created in accordance with purpose and policy. The SFC technology can be provided more flexibly and effectively through SDN and NFV technologies.

This study tries to propose a reinforcement learning based path selection technique that can efficiently select one path from multiple paths, or a set of multiple chains for one service function chain.

This thesis is comprised of as follows: 'INTRODUCTION' describes overall contents of the background technologies of this research, including SDN, NFV, and SFC; 'BACKGROUND INFORMATION' explains the background knowledge related to the research; 'PROPOSAL' presents the proposed reinforcement learning based SFC path selection technique; 'EVALUATION' evaluates the performance of the proposed technique in experiment; 'CONCLUSION AND FUTURE RESEARCH' describes the conclusion of this research and the future study.

BACKGROUND INFORMATION

SFC:

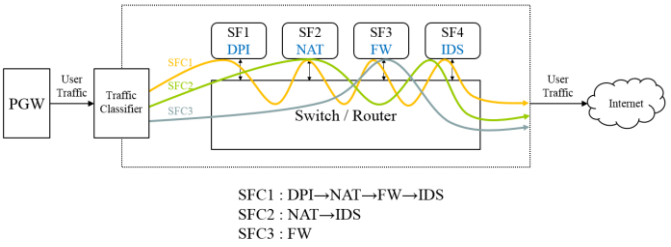

Figure 1: Overview of SFC Technology

SFC is the technology which is used to define a path of component services, create a network service timely, and control it actively. [Fig. 1] illustrates the overview of SFC technology whose basic operation is presented as follows: When user traffic goes into the network, traffic classification is performed in accordance with already-defined policy. As a result, a particular service function chain for the traffic is selected (in Fig. 1, there are three service function chains: SFC1, SFC2, and SFC3). After that, the user traffic is sequentially delivered to the service functions (DPI, NAT, FW, and IDS as shown in Fig. 1) defined in the service function chain. After the functions are executed, the traffic is transmitted to its destination. The basic components of the SFC technology defined for the above operation are described below [3][4].

Service Function (SF):

The SF means a function to perform specific processing on the packet. As a logical component, the SF can be implemented in virtual or physical network equipment. More than one service function can be operated in the same network equipment, and more than one instance can exist for a single service function.

Service Node (SN):

The SN is an object that is connected to the network and can have one or more service function instances.

Service Function Chain (SFC):

The SFC defines service functions and a combination order of the functions in order for the packets, frames, and flows selected in classification process. And it is defined in accordance with network service policy.

Service Function Path (SFP):

The SFP refers to an instance of a logically defined service chain. As the result of mapping a logical service chain to a service function instance and a physical service node in a real network, the path is used for transmitting actual network packets and traffic.

SFC Forwarding Node:

This node sequentially transmits network packet and traffic to a path through the connection between the service functions and service nodes defined in a service path. Through the data transmission between service functions, it plays a role of connecting service functions.

Traffic Classifier:

In order for the network service functions required for a corresponding application, a traffic flow is matched with a policy. At this time, the policy is different depending on user, network, and service. The equipment for classification is called classifier.

SFC Encapsulation

It plays a role to add information or metadata in order to distinguish the service function path selected in inflow traffic.

Reinforcement Learning :

Reinforcement learning is explained by the interaction between an agent and its neighboring environment. An agent means a learning entity. The environment is the learning place and presents the object to learn. The environment plays a role to use a reward function to provide information about how good other strategies are.

[Fig. 2] illustrates the step of reinforcement learning. A learning agent obtains a reward from the environment by doing a particular action in the current state. At this time, its goal is to maximize the reward. After the reward is set, it is required to select an action suitable for the current state, which is called a policy. Policy is a combination of exploration and exploitation. It is able to use exploitation to re-select the action that gave the highest reward in the current state, or use exploration to select a different action in expectation of better reward [5][6].

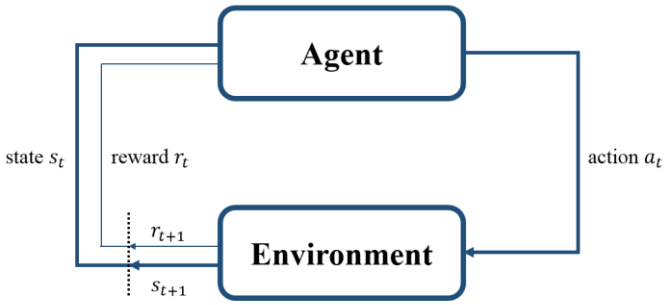

Figure 2: Step of Reinforcement Learning

Q-value (action-value function):

This is the value as to how good a certain action is and the expectation value for return of the action 'a' is taken in the state 's'. It is called Q-value or action-value function.

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

Formula 1. Q-value (action-value function)

Temporal Difference (TD) Algorithm:

This is a reinforcement learning method of using the difference between the currently and previously estimated values. On the assumption that more previous information is known, it is possible to update information about more states at the time of achieving a goal. TD(0) uses only the current state, and TD(1) takes into account all states visited so far. This study uses TD(0) algorithm.

PROPOSAL

SFC Path Selection Method based on TD(0) Algorithm for Q-value :

This study tries to propose a technique of selecting one chain efficiently from multiple paths, or a set of multiple chain candidates for one service function chain. The proposed method takes into consideration the usage of each node and the link bandwidth between the nodes which are prioritized in mentioned order. In other words, the method consider the node first and then consider link later.

As shown in [algorithm 1], at the beginning, QTD(0)-Matrix(the proposed method is TD(0) algorithm based path selection technique for Q-value so that it is called QTD(0)) value and R-Matrix value are set to '0', respectively. After the initialization, R-Matrix is set to the value updated repeatedly each time, since all instances and link conditions are changed depending on network conditions and thereby a reward value is also changed. The [algorithm 1] is applied to a list of service function chains already used or in use over network. Although SFC does not exist in a list, it is possible to perform the SFC and execute the same algorithm. In [algorithm 1], the service function chain of the first type 'A' is comprised of the service functions 1, 3, …, N, and the service function chain of the second type 'B' consists of the service functions 2, 4, …, M. In this case, as a candidate set of the service function chains, the type 'A' has a total of P chains (SFC1, SFC2, …, SFCP), and the type 'B' has a total of Q chains (SFC1, SFC2, …, SFCQ). Each chain in each set is given a number from '1'. Therefore, SFC1 in a different type is not the chain of the same service functions. For this reason and convenience, the chains are described as SFCA_1, SFCA_2, …, SFCA_P, SFCB_1, SFCB_2, …, SFCB_Q.

If traffic flows in a network, one path is selected from a set of candidates for the selected service function chain type. Packets are sent to the path, and are applied to the service functions through the chain so that an information on CPU and link bandwidth of each node is recorded. Based on the information, a reward value is adjusted.

[Algorithm 2] is the pseudocode of the reward function for setting a R-Matrix value. If the total CPU usage of the nodes used by the service functions in the currently selected particular type of service function chain and the total usage of link bandwidth are better (smaller) than previous values, a positive reward value (REWARD) is given; if the values are worse than before, a negative reward value (PENALTY) is given; if there are no changes, a reward value is not changed.

In the above procedure, the changed reward value is used for adjusting a QTD(0)-Matrix value for Q-value. As a result, if there are multiple paths for a service function chain of a particular type, it is possible to estimate and select a service function chain in the current network topology efficiently. When a particular service function chain should be used actually, a QTD(0)-Matrix value is used for routing traffic flow to a path that is considered efficient.

Algorithm 1. TD(0) Algorithm about Q value for SFC selection

```
Initialize: QTD(0)-Matrix ← {0,},
            R-Matrix ← {0,}
SFCs :      SFCA = {SF1, SF3, ···, SFN},
            SFCB = {SF2, SF4, ···, SFM},
            ···
SFC_candidates for each SFC :
            SFCA_candidates = {SFC1, SFC2, ···, SFCP},
            SFCB_candidates = {SFC1, SFC2, ···, SFCQ},
            ···
            ( = SFCA_1, SFCA_2, ···, SFCA_P,
               SFCB_1, SFCB_2, ···, SFCB_Q)
while network traffics exist
    while
        Select a state SFC type s_t
        Select an action(SFC type s_t's candidate) a_t and Excuete it
        R-Matrix ← reward(Node CPU and Link Bandwidth information of
                          selected a_t based on current network topology)
        Receive immediate reward
        Compute Q value
        Update QTD(0)-Matrix
        if a service uses a SFC then
            break
        end if
    end while
    Route using QTD(0)-Matrix
end while
```

Algorithm 1: TD(0) Algorithm about Q value

Algorithm 2. $reward(SFC\ candidate\ information)$ **function**

```
If      current SFC candidate's (CPU usage, Bandwidth usage)
      < previous SFC candidate's (CPU usage, Bandwidth usage)   then
        R-Matrix[SFC][candidate] ← R-Matrix[SFC][candidate] + REWARD (positive number)
else if current SFC candidate's (CPU usage, Bandwidth usage)
      > previous SFC candidate's (CPU usage, Bandwidth usage)   then
        R-Matrix[SFC][candidate] ← R-Matrix[SFC][candidate] + PENALTY (negative number)
else
        R-Matrix[SFC][candidate] ← R-Matrix[SFC][candidate]
end if
```

Algorithm 2: reward function

EVALUATION

SFC Selection Scenario:

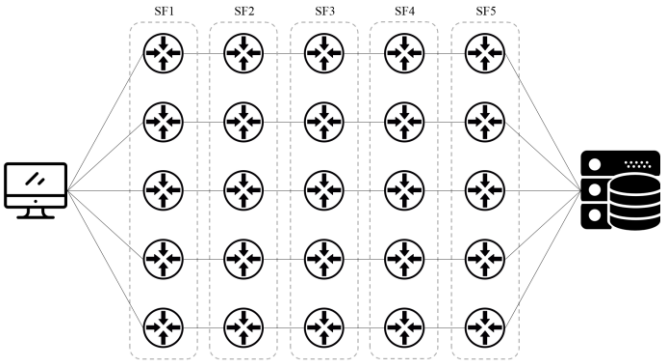

Figure 3: Scenario

[Fig. 3] shows the network topology for testing the service function chain path selection technique proposed in this study. In a list of service function chains, the algorithm selects one SFC representing a state. This SFC is considered to be Type A. This SFC is a chain of service functions {SF1, SF2, SF3, SF4, SF5}. A set of the possible candidates for the chain is {SFCA_1, SFCA_2, SFCA_3, SFCA_4, SFCA_5}.

In the above scenario figure, although the same service function (SF) is applied, it is represented each service function (SF) for convenience. In other words, although different chain candidates use the same SF1 node, it is represented each node as shown in [Fig. 3]. Accordingly, for five service functions, there are five nodes, and therefore the scenario has 25 service function nodes.

The network that has actually used SFCA applies the above algorithm before service function chain SFCA is used. It sends a packet to the candidates which is an action and gives a reward value to the set of candidates. At this time, if a SFC needs to be used, the value is applied for selecting a path and routing traffic flow.

As such, in case of SFC selection, network conditions are taken into account for path selection and routing, and thereby efficient service chaining environment is guaranteed.

Experimental Environment:

There are many limitations to apply service function chaining to actual network environment. Although there are researches on the simulation in Amazon EC2, they fail to take into consideration the virtualization environment [7][8]. Therefore, to test the proposed algorithm, this study implemented a simulator with Java.

Experimental Result:

To evaluate the performance of the proposed algorithm, this study compared the proposed method with Greedy method at the time of service function chain path selection. The comparison result is shown with graphs in [Fig. 4].

Before a particular type of service function chain is used, proposed algorithm is performed for a given network topology. After that, when the chain should be actually used, Greedy method and the proposed method are applied to path selection, respectively. When the flow routing is complete, the two methods are compared in terms of a path using smaller node and link usage in ratio. At the end of routing for a flow, the path with the smallest node and link resources is compared with the path selected by each method in order to find their consistency. As a result, although the path selected by Greedy method was the best select at that time, it didn't show a good outcome with the lapse of a certain time in changed network conditions (for instance, after one flow is all routed to a particular service function chain). In other words, the proposed method more efficiently uses the CPU and link bandwidth of each node in the service function chain than Greedy method, since it takes into account network conditions in order for path selection and routing.

As shown in [Fig. 4], in the QTD(0) Method of vertical axis, ver2 gives more enough time to figure out the network topology information than ver1 through the proposed technique so that it has a better outcome (in short, closest to the Best SFC path).

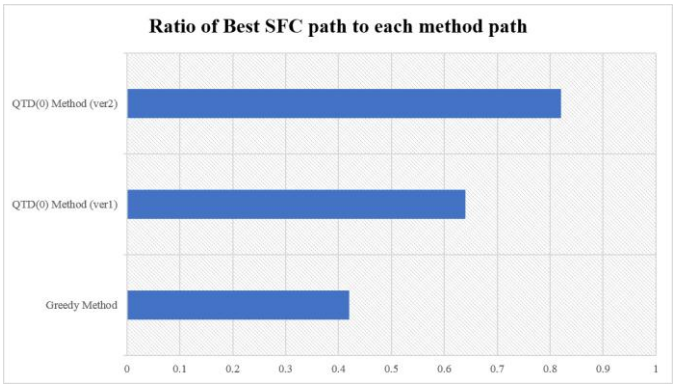

Figure 4: Experimental Result

CONCLUSION AND FUTURE RESEARCH

This study proposed a flow routing technique that takes into account constantly changing network conditions by continuing to send packets to the service function chains that have used before in the SDN/NFV environment, gives a reward value depending on the nodes and links in a path in order to evaluate a better path in a set of path candidates, and selects the best path in the current network conditions at the time of using the path actually.

In this paper, Q-value based TD(0) algorithm was proposed and this algorithm continues to give a reward value depending on network conditions in order to select an efficient path when a particular service function chain needs to be used.

The future research is to change the reward function proposed in [algorithm 2] in order to apply other path selection method as like selection of load balanced path.

ACKNOWLEDGEMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by Korea government (MSIP) (no. NRF-2015R1A2A2A01003501).

REFERENCES

[1] Stallings, W. (2015). *Foundations of Modern Networking: SDN, NFV, QoE, IoT, and Cloud*. Addison-Wesley Professional.

[2] Jain, R., & Paul, S. (2013). Network virtualization and software defined networking for cloud computing: a survey. *IEEE Communications Magazine*, *51*(11), 24-31.

[3] Medhat, A. M., Taleb, T., Elmangoush, A., Carella, G. A., Covaci, S., & Magedanz, T. (2017). Service Function Chaining in Next Generation Networks: State of the Art and Research Challenges. *IEEE Communications Magazine*, *55*(2), 216-223.

[4] Halpern, J., & Pignataro, C. (2015). *Service function chaining (sfc) architecture* (No. RFC 7665).

[5] Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction* (Vol. 1, No. 1). Cambridge: MIT press.

[6] Marsland, S. (2015). *Machine learning: an algorithmic perspective*. CRC press.

[7] Lee, G., Kim, M., Choo, S., Pack, S., & Kim, Y. (2015, June). Optimal flow distribution in service function chaining. In *The 10th International Conference on Future Internet* (pp. 17-20). ACM.

[8] Kim, T., Kim, S., Lee, K., & Park, S. (2015, May). A QoS assured network service chaining algorithm in network function virtualization architecture. In *Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on* (pp. 1221-1224). IEEE.