LETTER

Flow Clustering Based Efficient Consolidated Middlebox Positioning Approach for SDN/NFV-Enabled Network

Duc Tiep VU[†a)], ***Nonmember and*** **Kyungbaek KIM**[†b)], ***Member***

SUMMARY Recently in an SDN/NFV-enabled network, a consolidated middlebox is proposed in which middlebox functions required by a network flow are provided at a single machine in a virtualized manner. With the promising advantages such as simplifying network traffic routing and saving resources of switches and machines, consolidated middleboxes are going to replace traditional middleboxes in the near future. However, the location of consolidated middleboxes may affect the performance of an SDN/NFV network significantly. Accordingly, the consolidated middlebox positioning problem in an SDN/NFV-enabled network must be addressed adequately with service chain constraints (a flow must visit a specific type of consolidated middlebox), resource constraints (switch memory and processing power of the machine), and performance requirements (end-to-end delay and bandwidth consumption). In this paper, we propose a novel solution of the consolidated middlebox positioning problem in an SDN/NFV-enabled network based on flow clustering to improve the performance of service chain flows and utilization of a consolidated middlebox. Via extensive simulations, we show that our solution significantly reduces the number of routing rules per switch, the end-to-end delay and bandwidth consumption of service flows while meeting service chain and resource constraints.

key words: *consolidated middlebox, flow clustering, placement, SDN, NFV, service chain*

1. Introduction

A traditional middlebox is a computer appliance including both hardware and software designed to provide specific network service functions such as firewalls, proxies, load balancers and intrusion detection systems. A network flow usually requires a service chain, which is a set of service functions in a particular order. The rapid development of the Network Function Virtualization (NFV) allows implementing a middlebox function as an application that can be installed at a Virtual Machine (VM) running inside a physical machine. By using this virtualized middlebox, we can deploy a service chain flexibly in an SDN/NFV environment. Moreover, the NFV technology enables us to implement a consolidated middlebox in which each network flow receives all its required service function at a single physical machine [1], [2]. By using the consolidated middlebox, we can improve the performance of a service chain flow in an SDN/NFV enabled network through simplifying network traffic routing and saving resources of switches and machines.

Recently, many efforts on how to implement efficient service chains with both traditional middlebox and consolidated middlebox have been published [3]–[5], which focus on creating an efficient routing scheme for the controller to steer traffic through the required middleboxes. However, these works basically assume that the middleboxes have fixed location. Actually, the NFV technology allows to change the location of consolidated middlebox easily by migrating VMs to other physical machine when necessary. In this situation, the locations of consolidated middleboxes may have a significant impact on the performance of a service chain. A recent work tried to solve the placement problem of traditional middlebox but they did not consider the characteristics and properties of consolidated middlebox [6]. The consolidated middlebox positioning problem in an SDN/NFV-enabled network must be addressed adequately with service chain constraints (a flow must visit a specific type of consolidated middlebox), resource constraints (switch memory and processing power of machine), and performance requirements (end-to-end delay and bandwidth consumption).

In this paper, we propose a novel placement method for consolidated middleboxes in an SDN/NFV-enabled network based on flow clustering to improve the performance of service chain flows and utilization of a consolidated middlebox.

2. Related Works

One similar problem is the well known facility location problem which suggests the location to deploy a fixed number of service facilities to minimize the distance between clients and the closet facility capable of delivering the service [7]. The classic solution of the problem is the centralized approach, which needs to calculate distance between clients and all possible positions, hence it requires knowledges of global topology and service demand information. These requirements are not practical for large networks. A more effective approach is the distributed method as proposed in [10]. Generally, the network will be divided into small r-ball regions or r hops away from the facility. Then the topology and demand information within the r-ball are observed to re-optimize the current location of facility. However, these works did not consider an SDN/NFV environment where consolidated middleboxes are used with

Manuscript received March 18, 2016.
Manuscript revised May 6, 2016.
Manuscript publicized May 19, 2016.

[†]The authors are with the Department of Electronics and Computer Engineering, Chonnam National University, Gwangju, Korea.
a) E-mail: ductiep91@gmail.com
b) E-mail: kyungbaekkim@jnu.ac.kr (Corresponding author)

DOI: 10.1587/transinf.2016EDL8064

service chain requirements and resources constraints.

Another similar problem is the VM placement problem as in [11]. The proposed VM placement scheme is based on mutual bandwidth usage between VMs. Those VMs with large mutual bandwidth usage are assigned to host machine closed to each other. However, the end-to-end delay time is not considered in this scheme.

3. Flow Clustering Based Consolidated Middlebox Placement

In this work, we consider a directed graph $G_0 = (V_0, E_0)$, where V_0 is the set of nodes and E_0 is the set of edges. Each node corresponds to a switch, and each edge is a link connecting two switches. We assume that the costs to connect a consolidated middlebox to a single switch is insignificant. The number of rules that is currently stored in a switch s is denoted as $r(s)$, and the maximum number of rules a switch s can store is $R(s)$. The set of all service chains is denoted as P. Let us consider a consolidated middlebox type m that supplies a set of service chains $P_m \subset P$. The number of available consolidated middleboxs for type m is denoted as q_m. Each consolidated middlebox type m has a processing power O_m (Mbps), which is how much bandwidth it can deal with per unit of time.

A flow f_k can be described as $f_k = \{src_k, p_k, dst_k, dmd_k\}$, where src_k is the source node, dst_k is the destination node, and p_k is the service chain that traffics of flow f_k must visit. dmd_k is the processing demand of flow f_k, which represents how much bandwidth a flow occupies per unit of time (Mbps). Given that all flow information is known in advance, we can generate a set of flows F_m that require services from consolidated middlebox type m. The end-to-end delay time of a flow f (d_f) is determined by the sum of the delay from ingress switch of the flow f (i_f) to the corresponding consolidated middlebox of the flow f (m_f) and from the corresponding consolidated middlebox (m_f) to the egress switch of the flow f (e_f) as $d_f = d_{i_f,m_f} + d_{m_f,e_k}$. Our problem is find the location of all consolidated middleboxes type m so that the end-to-end delay time of each flow f in F_m is minimized:

$$\min_{\forall f \in F_m} d_f,$$

$$\sum_{f_i \in F_m} dmd_{f_i} \leq \sum_{j=1}^{q_m} O_{m_j} \tag{1}$$

$$r(s) \leq R(s), \forall s \in V_0 \tag{2}$$

Constraint 1 is a processing demand constraint to assure that a consolidated middlebox has enough processing power to process the corresponding flows, that is the total demand of all required flows must not exceed the processing capacity of the consolidated middlebox. Constraint 2 is a switch memory constraint to make sure a switch has available memory to store new flow table entries.

To solve the problem, we considered two intuitive properties. (1) It is better that a consolidated middlebox is located near ingress switch of its corresponding flows. (2) Accordingly, it is better that these flows with the set of ingress switches closed to each other share the same consolidated middlebox. Based on these intuitions, our solution consists of two steps as expressed in the Algorithm 1. First, we split flows into the given number of clusters based on the end-to-end delays between the ingress switches of flows. The number of cluster is equal to the number of available consolidated middlebox, q_m, so that each consolidated middlebox serves each cluster of flows. In the second step, we further analyze each cluster to find the best switch which satisfies the defined constraints and which is closest to the cluster of flows.

Algorithm 1 *Flow Clustering based Consolidated Middlebox Placement*

Input: $G = (V_0, E_0), F_m, V_m$
Output: Set of switches to connect consolidated middlebox type m

Step 1: Flow Clustering
Initial cluster $C_0 = \{v_0\}$
while there is a flow f_i with ingress switch v_i **do**
    Calc delay time from v_i to all existing clusters
    Find the closest cluster C_k with delay time d_{v_i,C_k}
    if $d_{v_i,C_k} > \theta$ and there is an available consolidated middlebox **then**
        Create a new cluster to contain v_i
    else
        $C_k = C_k \cup v_i$
Step 2: Find the closest switch to each obtained cluster
for each ingress switch $v_i \in C$ **do**
    for each adjacent switch $a_i \in A_{v_i}$ **do**
        Calculate $d^{tot}_{a_i} = \sum_{v_j \in C} d_{a_i,v_j}$
        Find a_i with $\min_{d^{tot}}$ and add a_i to the possible location list L
Sort list L in ascending order of $d^{tot}_{a_i}$
for each $a_i \in L$ **do**
    Calculate $r(s)$ for each switch $s \in V_0$
    if $\exists s \in V_0$ **such that** $r(s) > R(s)$ **then**
        a_i is unqualified
        continue to a_{i+1}
    else
        a_i is selected
        break

Step 1: Flow Clustering

The goal of this step is to gather the flows with ingress switches which are closed to each other into a same cluster. A cluster C is composed of the ingress switches of each corresponding flows. That is, $C = (v_1, v_2, \cdots, v_i)$ where v_f is the ingress switch of a flow f. Let V_m is set of ingress switchs of flows in F_m. First, we determine the end-to-end delay time between each pair of switches $v_i, v_j \in V_m$ as the shortest path (SP) delay time between them: $d_{v_i,v_j} = d_{\mathrm{SP}(v_i,v_j)}$, for all $v_i, v_j \in V_m$. We also defined a delay thresholds θ as:

$$\theta_m = \min d_{v_i,v_j} + \frac{\max d_{v_i,v_j} - \min d_{v_i,v_j}}{q_m} \text{ for } \forall v_i, v_j \in V_m$$

Initially, there is only one cluster. The delay time from switch v_i to cluster C_k is defined as $d_{v_i,C_k} = \min d_{v_i,v_j}$ with all $v_j \in C_k$. After the delay times from v_i to all existing cluster are calculated and the closest cluster C_k can be de-

termined. By comparing d_{v_i,C_k} with the threshold θ, we can decide either to assign the flow to the closest cluster C_k or create a new cluster to contain the flow if there still is a room for a new cluster.

Step 2: Find the closest switch to each cluster

In this step, we locate the best node (or switch) for each cluster of flows, where the total end-to-end delay to reach all ingress switches of the flows in a cluster is minimized. Let us consider a cluster of flows as C, and let A_{v_i} is the set of adjacent node of switch v_i with $v_i \in C$. First, we calculate the total end-to-end delay time, $d^{tot}_{a_i}$, from each adjacent node a_i to each node v_j of the cluster. Then, we can determine the adjacent node a_i with smallest $d^{tot}_{a_i}$, and add it to a list L of possible locations. The list L will be sorted by ascending order of total delay time (d^{tot}). Finally, we choose the first node which satisfies the given contraints among the nodes in the list L.

4. Evaluation

To evaluate the performance of our proposed method, we implemented a SDN testbed by using Opendaylight SDN controller [9] and Mininet [8]. Opendaylight and Mininet are installed in two separate machines which have a quad-core CPU of 3.4 GHz and 8 GB memory. On this testbed, we calculate the location of given number of consolidated middleboxes on a given network topology by using various placement methods, and simulate the SDN flows and measure the parameters such as end-to-end delay per flow, bandwidth consumption of flows and number of rules per switch. Iperf is used to generate traffic of each flow, and measure the end-to-end delay and bandwidth consumption. The number of rules per a switch is collected by using Opendaylight flow table statistics.

Through the evaluation, we compare our method with the other two placement methods: the random placement and the most used switch placement. For random placement, we simply place each consolidated middlebox at a random switch in a network. In the most used switch method which is the baseline method in [6], the consolidated middleboxes are placed at the most common ingress switch of multiple flows.

We use well known network topologies such as FatTree and Abilene, which were used in previous works [6]. The FatTree has a layered structure and usually can be seen in data center, and the FatTree-4 is the FatTree topology where each switch has four ports for connecting to other switches. The Abilene represents an irregular structure network which can be seen in most ISP networks. The characteristic of these topologies are summarized in Table 1.

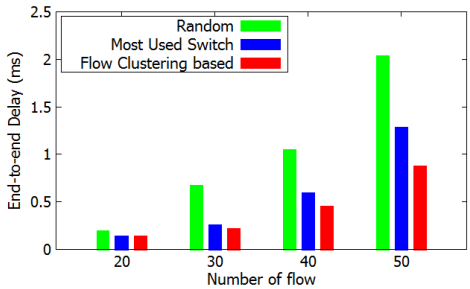
In this evaluation, we assume that there are various kinds of service chains and various numbers of flows which use a specific service chain randomly. We assume that the link delays of edges and the processing demand of each flow is randomly assigned according to normal distribution. The parameters setting of our evaluations are summarized in Table 2.

Table 1 The characteristic of topologies used in evaluation.

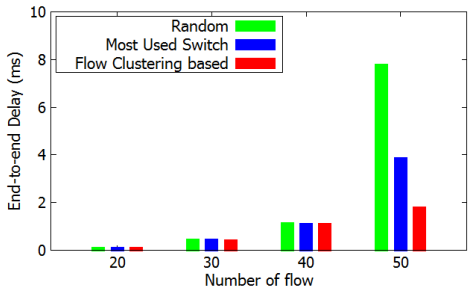
Topology	Abilene	FatTree-4
Number of nodes	11	20
Number of links	14	32

Table 2 Parameters setting in the performance evaluation.

Parameter	Value/Range
FatTree-4 Link delay (ms)	From 0.01 to 0.3
Abilene Link delay (ms)	From 0.23 to 2.1
The number of service chain	5
The number of consolidated middleboxes	10
The number of flow	20, 30, 40, 50
The processing demand of each flow (Mbps)	From 0.2 to 0.4
Maximum number of rules in each switch	30
Maximum bandwidth of each port in a switch (Mbps)	10



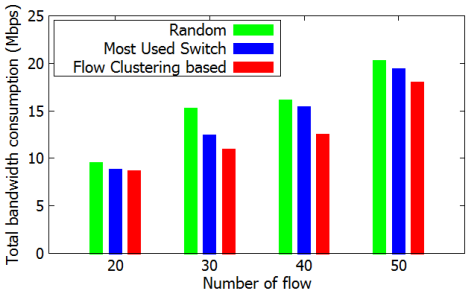
(a) FatTree-4



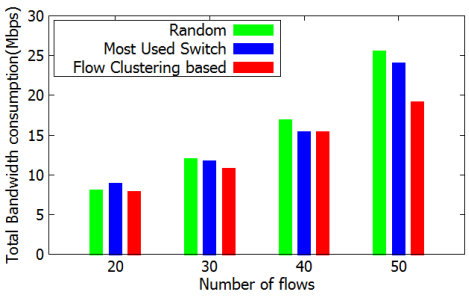
(b) Abilene

Fig. 1 The average end-to-end delay time per flow.

Figure 1 illustrates the average end-to-end delay time per flow in FatTree-4 and Abilene networks. In both networks, as the number of flow increases, the random placement has the highest end-to-end delay while the flow clustering based method has the lowest. In FatTree-4 network, the flow clustering based method shows an average of 53% and 27% smaller end-to-end delay compared to the random and the Most Used Switch method respectively. In Abilene network, the improvements are around 23% and 17% respectively. In terms of the total bandwidth consumption of service flows, the flow clustering method also shows significant improvements over the other two methods as depicted in Fig. 2. In FatTree-4 network, the flow clustering method

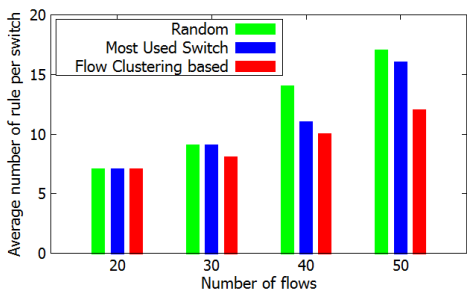


(a) FatTree-4

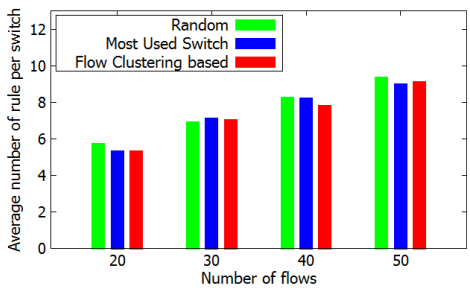


(b) Abilene

Fig. 2 The total bandwidth consumption.



(a) FatTree-4



(b) Abilene

Fig. 3 The average number of rules per switch.

reduces around 17% and 10% of bandwidth consumption compared to the random and the Most Used Switch methods respectively. The improvements in Abilene network are around 12% and 10% respectively.

The average number of rules per switch are presented Fig. 3. In FatTree-4 network, when the number of flow increases, the random placement has the highest number of rules per switch while the flow clustering based method has the lowest. On an average, the number of rule per switch that the flow clustering based method requires are around 17% and 11% less than the random and Most Used Switch methods respectively. In Abilene network, the average number of rule per switch increases slowly as the number of flow increases, and also the three methods have comparable results. The reason is that the Abilene topology has a smaller number switches and links compared the FatTree-4. Therefore, the number of routing hops between two nodes in Abilene is small.

In brief, our flow clustering based placement method is efficient to reduce the number of rules per switch, the end-to-end delay per flow and total bandwidth consumption of service flows.

5. Conclusion

In this paper, a new placement method for consolidated middlebox in an SDN/NFV network is proposed based on flow clustering to improve the performance of service chain flows and utilization of a consolidated middlebox. Through extensive evaluations, we demonstrated that our proposed method outperforms the random placement method and the Most Used Switch placement method. The proposed method can be applied to the network planning and designing phase or to the run-time location optimization of consolidated middleboxes. As a future work, the impact of neighboring clusters on each other and their joint impact on the overall network performance can be considered.

Acknowledgments

This work was supported by the National Research Foundation of Korea Grant funded by the Korean Government (NRF-2014R1A1A1007734).

References

[1] V. Sekar, N. Egi, S. Ratnasamy, M. Reiter, and G. Shi, “Design and implementation of a consolidated middlebox architecture,” Proc. NSDI, 2012.
[2] J.W. Anderson, R. Braud, R. Kapoor, G. Porter, and A. Vahdat, “xOMB: Extensible open middleboxes with commodity servers,” Proc. ANCS, pp.49–60, 2012.
[3] Z.A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu, “Simple-fying middlebox policy enforcement using SDN,” Proc. ACM SIGCOMM, pp.27–38, 2013.
[4] Z. Cao, M. Kodialam, and T.V. Lakshman, “Traffic steering in software defined networks: Planning and online routing,” Proc. ACM DCC, pp.65–70, 2014.
[5] A. Gushchin, A. Walid, and A. Tang, “Scalable routing in SDN-enabled networks with consolidated middleboxes,” Proc. ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization, HotMiddlebox 15, pp.55–60, 2015.
[6] J. Liu and Y. Li, “Improve service chaining performance with optimized middlebox placement,” IEEE Trans. Services Computing, no.1, p.1, PrePrints, 2011.
[7] P. Mirchandani and R. Francis, Discrete location theory, John Wiley & Sons, 1990.
[8] Mininet, http://mininet.org/
[9] Opendaylight, https://www.opendaylight.org/

[10] G. Smaragdakis, N. Laoutaris, K. Oikonomou, I. Stavrakakis, and A. Bestavros, "Distributed server migration for scalable Internet service deployment," IEEE/ACM Trans. Netw., vol.22, no.3, pp.917–930, June 2014.

[11] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," Proc. IEEE INFOCOM, pp.1–9, 2010.