Network-aware Service Function Chaining Placement in a Data Center

Cheng-Husan Hsieh[2], Je-Wei Chang[2], Chien Chen[12], and Ssu-Hsuan Lu[2]
[1]Information Technology Service Center
[2]Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan
{hsiehch, changzw, chienchen}@cs.nctu.edu.tw, shlu@nctu.edu.tw

***Abstract*—Network function virtualization (NFV) has drawn much attention in recent years, where some network functions that used to be deployed on specific hardware have become virtualized instances on general servers to achieve more scalability and flexibility. In a data center, service function chaining (SFC) makes a workflow traverse different network functions in a specific order to provide different levels of service for its customer. Because the distance between any adjacent network functions in a service chain will decide the total bandwidth consumption for that chain, the placement of the virtualized network functions in a data center becomes an important problem. In this study, this placement problem is treated as a multi-layer bin packing problem. Two greedy algorithms are proposed for the tree-like network topology: Multi-layer Worst-Fit (MWF) and Multi-layer Best Fit (MBF). Furthermore, the placement problem is formulated as an integer linear programming. The experimental results show that MWF can reduce bandwidth consumption by 15% while only increasing the number of used servers by 1% compared to the traditional Best-fit algorithm.**

***Index Terms*—Network Function Virtualization (NFV), Service Function Chaining (SFC), middlebox, Bin packing.**

I. Introduction

Network Function Virtualization (NFV) has drawn much attention recently. NFV transfers network functions from dedicated physical hardware (also called middlebox) to a Virtual Machine (VM) on general purpose servers to achieve flexibility and scalability. For instances, virtual network functions can rapidly scale on demand when burst requests come in a very short time.

Service function chains (SFC) make different traffic workflows traverse different network functions in some specific order to provide different levels of service in data centers. In this study, we focus on the SFC placement problem in the NFV environment (SFC-NFV environment). It is easy to treat the SFC-NFV placement problem as a special VM placement problem where each VM has inter-communication with at most two other VMs. We name this special VM placement problem the chained VM placement problem.

Previous works [4], [10], [12], [13] discuss the virtual network function placement problem in the SFC-NFV environment. The works [10], [13] have a single aspect of focusing on the servers' cost by reducing the number of usage servers or usage virtual instances. Work [12] focuses on the backbone topology with multiple objective functions in multiple data centers. The objective functions are minimizing the latency of the chaining request, maximizing the remaining bandwidth, and minimizing the number of used network nodes (e.g., network points of present). Alternately, works [10], [12] and [13] use integer linear programming to get the optimal result. Stratos [4] uses minimum K-cut to deploy network functions on top of a rack which will decrease the cost of the network in the cloud.

However, there are few solutions combining the cost of the servers and the network in the SFC-NFV environment. The cost of a cloud [6] reveals that the average cost of servers and network exceeds half of the cost. Misplacing the virtual network functions on the different servers may increase the number of usage servers and total bandwidth consumption. This study was inspired by the work of [9], which gives a new point of view on the VM placement problem in a data center by reducing the cost of both the server and the network. The work of [9] solves the problem using bin packing where the cost function in the bin packing algorithm puts reducing cost of network as higher priority than the cost of servers. In reality, the server cost is higher than the network cost. Besides, they consider the network cost a fixed cost. Since the topologies of the underlay network in the data center are tree-like topologies such as Fat-Tree [1] and VL2 [7], the network cost between two servers is easy to evaluate based on the positions of the servers in the underlay network. Finally, they do not consider a bin packing algorithm suitable for the chained VM placement problem.

Therefore, this study presents the chained VM placement problem under a tree-like topology as a multi-layer bin packing problem. To the best knowledge, we are the first one to solve this problem as a multi-layer bin packing problem. We assume that (1) a network function can use more than one virtual machine; (2) a virtual machine runs only one network function; (3) a network function instance must be deployed on a server; and (4) the amount of traffic in the same chaining request will not change when the packets are carried out by network functions. There are different layers of bins, and a server may belong to many bins and a bin may cover many servers. For consistency, a server is treated as a bin at layer 1. We group k bins at layer i to form a bin at layer i+1. It is easy to evaluate the network cost between two servers using multi-layer bins.

Specifically, the number of hops between any two servers is decided by the common bin at the lowest layer of these two servers (see Fig. 1). Thus, the network functions deployed on the bin at a lower layer will have a smaller network cost. For instance, the number of layers in Fat-Tree [1] is four due to the fact that the distance among network functions on the servers has four different hops: 0 hop, 2 hops, 4 hops, and 6 hops, respectively.

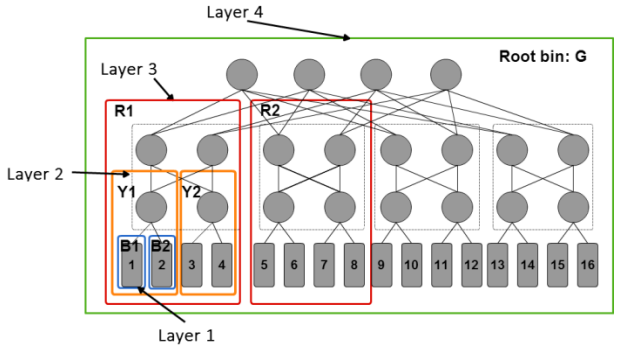


Fig. 1. Multi-layer Bin Packing in Fat-tree Topology

Multi-layer bin packing first tries to minimize the number of usage servers, and then tries to deploy network functions on the bin at a lower layer to reduce network cost based on SFC requests. We propose two greedy algorithms to solve the multi-layer bin packing problem: Multi-layer Worst-Fit (MWF) and Multi-layer Best Fit (MBF). Both MWF and MBF try to deploy network functions on a SFC request one-by-one by using local search on bins at different layers with the worst-fit scheme and the best-fit scheme, respectively.

The main contributions of this study are: (1) the network function placement and chaining problem is formulated as a multi-layer bin packing problem taking inter-communication traffic into account, (2) the proposed algorithms can cover many tree-like topologies in the data center, and (3) our algorithms can reduce the bandwidth consumption by at least 15% while only increasing the usage servers by 1% compared to the best-fit algorithm.

The remainder of this study is organized as follows. Section II introduces the related works. Section III formulates the multi-layer bin packing placement problem and proposes two greedy algorithms for tree-like topologies. Section IV discusses the experimental results of the proposed algorithms. Finally, Section V summarizes this study with a brief conclusion.

II. RELATED WORKS

Traditionally, network functions such as firewall and IDS are installed in dedicated physical hardware, which is expensive and difficult to scale or migrate. Some works [2], [3], [5], [8], [15], [16], [17] show that the network functions come with high infrastructure and management costs because of the behavior of functions (e.g., rewrite packet header/payload or map the sessions). Therefore, network functions that used to be deployed on specific hardware have become virtualized instances in general servers to achieve more scalability and flexibility.

The term service function chaining is used to describe the ordered list of network functions which packets need to traverse by sequence. Software-Define Networking (SDN) [11] gives more flexibility to steering the traffic between network devices. To keep the ordered sequence, packets can be tagged in an unused field [3], [15] (e.g., ToS, MPLS, and VLAN field) or use metadata and multiple tables [18] in the switch. Another way is using Network Service Header (NSH) [14], which will maintain the identity of the chaining request and the index of the function.

It is easy to consider the SFC problem a VM placement problem. The work of [9]: traffic communication between VMs is end-to-end, network cost can be reduced if VMs are deployed nearly in a data center. Besides, the number of usage servers dominates the overall cost in the cloud. This work will reduce not only servers cost but also network cost The work of [4] uses Minimum k-cut to solve SFC. In this approach, servers can be treated as vertices and traffic among servers can be treated as edges, then partitioning servers into a group will decide the total amount of traffic in the topology. Hence, this approach aims at minimizing bandwidth consumption when the original topology is partitioned into k connective components. Many works [10], [12], [13] use linear integer programming to get the optimal results, and they have different objective functions, constraints, and environment. [12] has three different objective functions; and they are (1) minimizing the number of usage data centers, (2) maximizing the remaining bandwidth consumption, and (3) minimizing the total delay of the chaining requests. The environment of [12] is an internet service provider environment which has multiple data centers. [13] aims at minimizing the number of usage servers, and its environment doesn't take the underlying network into account. The cost of licenses will be large if the provider uses more virtual instances which need the license. Hence, [10] aims at minimizing the number of virtual instances of each type of function.

Unlike previous works, this study focuses on formulating the SFC problem as a multi-layer bin packing problem to minimize the network cost and server cost.

III. NETWORK FUNCTION PLACEMENT AND CHAINING PROBLEM

In this section, we model our multi-layer bin packing problem. Subsection A formulates this problem as an integer linear programming problem in the tree-like network topology. Subsection B defines the multi-layer bin structure. Subsection C proposes two greedy algorithms.

A. Network Model

The traffic pattern of SFC is chain-like, and the traffic of the SFC request must follow the specific *ordering* to traverse functions. We consider the number of hops among the adjacent network functions and use Fat-Tree as the topology. Fat-Tree is a commonly used topology in the data center, which has three-tier architecture and k ports at each switch. As shown in Fig. 1, we have a Fat-Tree with k=4. There are four distances

between network functions, which are (1) hosted by the same server, (2) hosted by different servers and under the same edge switch, (3) hosted by different servers, under different edge switches and under the same pod, and (4) hosted by different servers and pods. The numbers of the hops between network functions in each category are 0 hops, 2 hops, 4 hops, 6 hops respectively.

We describe the inputs, variables, and constraints of this model. In a real environment, the physical machine has limited capability, such as CPU utilization of each server. In our model, the set N represents the server set and the set UN represents the unused servers. The set c represents the chaining request. The traffic amount of c is W_c which is a constant. The Set $F_c = \{f_1, f_2, \ldots, f_n\}$ the network functions are used in the request c, where f_i indicates the i-th function in chain c.

The variables of this model are listed as follows:

- $X_{i,n} \in \{0,1\}$: 1 if f_i is deployed on the server n, 0 otherwise.
- $T_{i,n} \in \{0,1\}$: 1 if f_i and f_{i+1} are deployed on the server n, 0 otherwise.
- $E_{i,e} \in \{0,1\}$: 1 if f_i and f_{i+1} are deployed under the edge switch e, 0 otherwise.
- $P_{i,p} \in \{0,1\}$: 1 if f_i and f_{i+1} are deployed under the pod p, 0 otherwise.
- $U_v \in \{0,1\}$: 1 if unused server v is used now, 0 otherwise.

Based on these inputs and variables, we present the objective function and constraints of this model. The objective function aims at minimizing the network and server cost simultaneously for deploying a chaining request c.

Objective function:

$$Min\left(\begin{array}{l} W_c \sum_{i}^{|F_c|-1}(6-2\sum_{n\in N}T_{i,n}-2\sum_{e\in E}E_{i,e}-2\sum_{p\in P}P_{i,p}) \\ +Cost_{Server}\sum_{v\in UN}U_v \end{array}\right) \quad (1)$$

Subjected to:

$$\sum_{n\in N} X_{i,n} = 1, \forall \mathrm{i} \in 1,2,\ldots,|\mathrm{F}_c| \quad (2)$$

$$CurrentLoading_n + W_c \sum_{i}^{|F_c|} P_{f_i} X_{i,n} \le 1, \forall n \in N \quad (3)$$

$$U_v \le \sum_{i}^{|F_c|} X_{i,v}, \forall v \in UN \quad (4)$$

$$X_{i,v} \le U_v, \forall v \in UN \quad (5)$$

$$X_{i,n} + X_{i+1,n} - 1 \le T_{i,n}, \forall n \in N, i \in 1,2,\ldots,|\mathrm{F}_c|-1 \quad (6)$$

$$T_{i,n} \le \frac{X_{i,n}+X_{i+1}}{2}, \forall n \in N, i \in 1,2,\ldots,|\mathrm{F}_c|-1 \quad (7)$$

$$\sum_{n=\frac{k}{2}(\mathrm{e}-1)+1}^{\frac{k}{2}e} (\mathrm{X}_{i,n}+\mathrm{X}_{i+1,n}) - 1 \le \mathrm{E}_{i,e}, \forall e \in E, i \in 1,2,\ldots,|\mathrm{F}_c|-1 \quad (8)$$

$$E_{i,e} \le \frac{1}{2}\sum_{n=\frac{k}{2}(\mathrm{e}-1)+1}^{\frac{k}{2}e} (\mathrm{X}_{i,n}+\mathrm{X}_{i+1,n}), \forall \mathrm{e} \in \mathrm{E}, \mathrm{i} \in 1,2,\ldots,|\mathrm{F}_c|-1 \quad (9)$$

$$\sum_{n=\frac{k^2}{4}(\mathrm{p}-1)+1}^{\frac{k^2}{4}p} (\mathrm{X}_{i,n}+\mathrm{X}_{i+1,n}) - 1 \le \mathrm{P}_{i,p}, \forall p \in P, i \in 1,2,\ldots,|\mathrm{F}_c|-1 \quad (10)$$

$$P_{i,p} \le \frac{1}{2}\sum_{n=\frac{k^2}{4}(\mathrm{p}-1)+1}^{\frac{k^2}{4}p} (\mathrm{X}_{i,n}+\mathrm{X}_{i+1,n}), \forall \mathrm{p} \in \mathrm{P}, \mathrm{i} \in 1,2,\ldots,|\mathrm{F}_c|-1 \quad (11)$$

Equation (2) ensures that a network function is exactly deployed on a server. Equation (3) ensures that the loading of a server will not exceed its capability. Equations (4) and (5) calculate if any unused server v must be used at this time. Equations (6) and (7) consider the network costs for adjacent network functions deployed on the same server. Equations (8) and (9) consider the network costs for the adjacent network functions deployed under the same edge switch. Equations (10) and (11) consider the network costs for adjacent network functions deployed under the same pod.

We solve this problem using multi-layer bin packing. The next section will introduce the multi-layer bin structure.

B. Multi-layer Bin Structure

This study uses the multi-layer bin structure to identify the distance relation between any two servers. Since there are four kinds of distances between servers in the Fat-Tree topology, we have four different bin structures at their individual layers. For consistency, a server is treated as the bin with the smallest size at layer 1. We group k bins at layer i to form a bin at layer i+1, and the number of hops between the servers in the different neighboring bins at layer i, who have a common parent bin at layer i+1, is the same. The bin at the highest layer is the root bin r. The functions deployed on the bin at lower layers will have a smaller network cost. In Fat-Tree topology, servers are covered by multiple bins with different sizes.

Take Fig. 1 as an example. All servers are under the root bin G. Servers 1 and 2 are individual bins at layer 1 (i.e., B_1, B_2), and servers 1 and 2 can form a bin at layer 2 (i.e., Y_1). Server 3 and server 4 can also form another bin at layer 2 (i.e., Y_2). Hops between servers in these two bins are the same. For instance, the number of hops between servers 1 and 3 is the same as the number of hops between servers 1 and server 4. Moreover, Y_1, Y_2 can form a bin at layer 3 (i.e., R_1). In this example, the number of hops among servers in the different bins at layer 3 is six (i.e., R_1, R_2), the number of hops among servers in the different bins at layer 2 is four (i.e., Y_1, Y_2), and the number of hops between servers in different bins at layer 1 is two (i.e., B_1, B_2).

C. Multi-layer Worst-fit / Multi-layer Best-fit

The idea of Multi-layer worst-fit (MWF) and Multi-layer best-fit (MBF) is that we have two basic operations to search

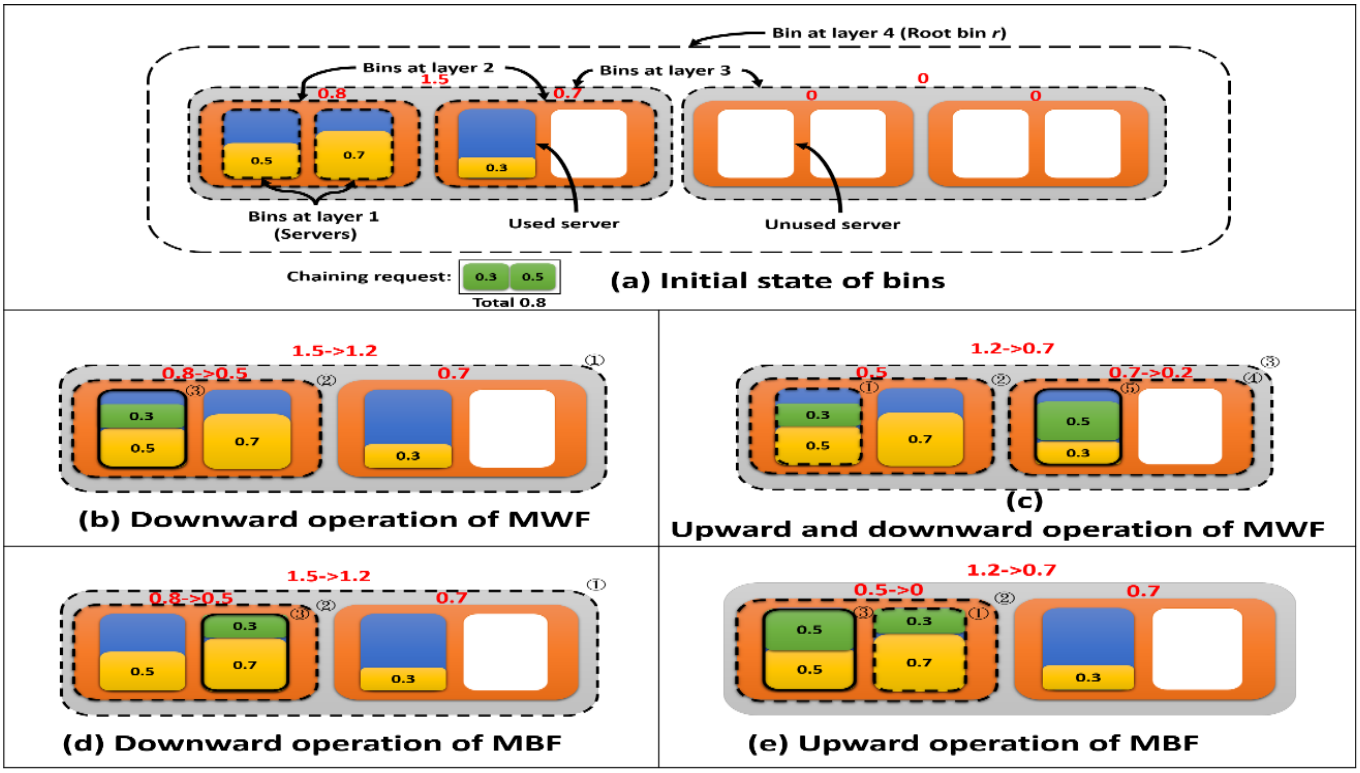


Fig. 2. A use case of MWF and MBF

the bins at different layers to deploy network functions. They are downward and upward operations.

- **Downward Operation:** the downward operation is used to select a bin to deploy the network function. Suppose that we start from bin b at layer j. The downward operation selects one of the children bins from layer j to layer 1, recursively. In each layer i, suppose that bin c is selected. We will select a child bin of bin c at layer i-1 based on worst-fit or best-fit (depends on using MWF or MBF), where the current capacity of this child bin is aggregated from the remaining capacity of used servers in that bin and has a satisfied server to deploy. If we select a bin at layer 1, we will deploy the network function to that bin.
- **Upward Operation:** when a network function is deployed, we use the upward operation to find a satisfied server to deploy the next network function as close as possible to the deployed function. The method is to find the first common bin with the previous network function from layer 1 to the highest layer, which contains a used server with enough capacity to deploy the next function.

Using these two operations, the multi-layer worst-fit scheme and multi-layer best-fit scheme are designed to deploy a chaining request $c = \{f_1, f_2, \ldots, f_n\}$, where f_i represents the i-th function in chain c.

- **Multi-layer Worst-Fit (MWF):** the multi-layer worst-fit scheme starts from the root bin r and uses the downward operation to find a satisfied server for the first network function f_1. Then, the multi-layer worst-fit scheme iteratively tries to deploy the remaining network functions f_k for (k=2, 3,. . . , n) as follows. First, the multi-layer worst-fit scheme uses the upward operation to find a common bin b for f_{k-1} and f_k. Next, the multi-layer worst-fit scheme uses the downward operation starting at bin b to select the best server in bin b to deploy f_k.
- **Multi-layer Best-Fit (MBF):** the multi-layer best-fit scheme starts from the root bin r and uses the downward operation to find a satisfied in bin r with the least remaining capacity to deploy the first network function f_1. (Using the total of network functions' loading as basis when there is a bin that can satisfy; otherwise, using the first network function as basis.). Then, the multi-layer best-fit scheme iteratively tries to deploy the remaining network functions f_k for (k=2, 3,. . . , n) as follows. First, the multi-layer best-fit scheme uses the upward operation to find a common bin b for f_{k-1} and f_k. Again, the multi-layer best-fit scheme selects the best server in bin b to deploy f_k (i.e., the server with the least remaining capacity).

When both MWF and MBF schemes cannot find a feasible solution for the network function f_j, we will add an unused server approaching as close to the previous network function f_{j-1} as possible.

Figure 2 is an example of how the MWF and MBF schemes work, where the red number on a bin represents the aggregated remaining capacity of the used servers in that bin. The initial state and the input of the chaining request are shown in Fig. 2(a). Figure 2(b) and (c) demonstrate how the MWF scheme works. Figure 2(b) shows how to deploy the first network function in the MWF scheme. Starting from the root

bin r, MWF uses the downward operation to find the satisfied server (i.e., the steps (1)-(3) in Fig. 2(b)). Figure 2(c) shows how to deploy the second network function in the MWF scheme. First, MWF uses the upward operation (the steps (1)-(3) in Fig. 2(c)) to find a common bin with the previous network function (i.e., the gray bin in Fig. 2(c)). Then, MWF again uses the downward operation to find a satisfied server to deploy the second network function (i.e., the steps (4)-(5) in Fig. 2(c)).

Figure 2(d) and (e) demonstrate how the MBF scheme works. Figure 2(d) shows how to deploy the first network function in the MBF scheme. Starting from root bin r, the MBF scheme uses the downward operation to find a satisfied server (i.e., the step (1)-(3) in Fig. 2(d)). Figure 2(e) shows how to deploy the second network function in the MBF scheme. First, MBF uses the upward operation (i.e., the steps (1)-(2) in Fig. 2(e)) to find a common bin with the previous network function (i.e., the left-orange bin in Fig. 2(e)). Then, MBF again finds the satisfied server under the common bin using the best-fit method (i.e., the step (3) in Fig. 2(e)).

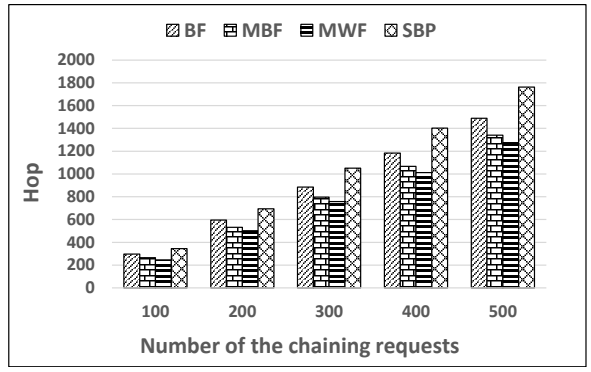
IV. Simulation

In our simulations, we consider possible lengths of the chaining requests varying from 2 to 6, and we formulize each server's capacity as 1. The loading of the network functions are 0.2, 0.23, 0.26, 0.29, 0.32, and 0.35, and each network function's loading is unique. The traffic amount of each chaining request is the same and it is fixed as 1. The number of the ports in a switch in Fat-Tree is 20. The number of chaining requests is from 100 to 500. The performance metrics are network cost and server cost, where network cost is the total number of hops for deploying all of the chaining requests and server cost is the total number of the used servers. Each experimental result is the average result of 100 times.

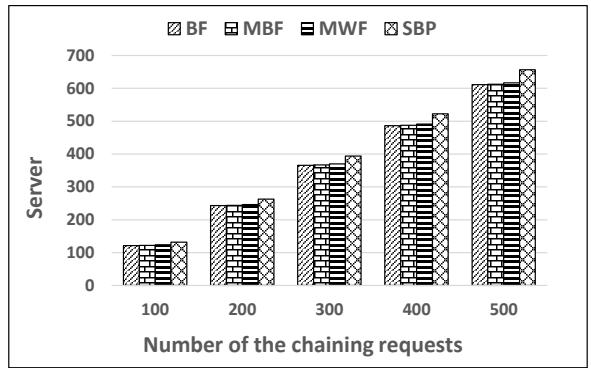
The comparison algorithms are as follows:

- Best-Fit (BF): All network functions are using best-fit to deploy on the servers.
- Sorted-Based Placement (SBP) [9]: SBP sorts the VM requests by the number of requested VMs in decreasing order. Try to deploy the VMs in the request in the sorted list from the first to the last, and this approach will use first-fit if all VMs in the requests can be deployed at least on a server. Otherwise, this approach uses worst-fit to find the server and deploys the most VMs in that server, and the remaining VMs will be put into the sorted list and retain their ordering.
- Multi-layer Worst-Fit (MWF): the detailed description of MWF is mentioned in section III.C.
- Multi-layer Best-Fit (MBF): the detailed description of MBF is mentioned in section III.C.

Figure 3 compares all of the algorithms under different numbers of chaining requests. Figure 3(a) shows the network cost of all of the algorithms. The network costs of MWF and MBF are smaller than those of BF and SBP due to these approaches trying to deploy adjacent network functions as closely as possible. MWF has the smallest network cost due to this approach focusing on selecting the bin which has the largest remaining capacity first, and MWF can reduce network cost by 14.2% to 15.6% compared to BF, and by 27.5% compared to SBP. MBF can reduce network cost by 10% compared to BF, and by 23.4% compared to SBP. SBP has the largest network cost due to this approach just focusing on deploying the VMs on the same server but not considering distance among the remaining VMs in requests which cannot be deployed on that server. Figure 3(b) shows the server cost of all of the algorithms. BF has the smallest usage of the servers, and MBF has the second smallest usage of the servers. MBF increases the number of the usage servers by 0.2% compared to BF because this approach is based on best-fit, hence, the usage of the servers is similar to BF. MWF's increases the number of usage servers by 1% compared to BF due to this approach being based on worst-fit, and the drawback of worst-fit is contributing fragments. Both MBF and MWF's number of the usage servers are lower than SBP; MBF can reduce the server cost by 6.6% and MWF can reduce it by 6%. SBP has the largest server cost due to this approach not considering the used or unused servers.



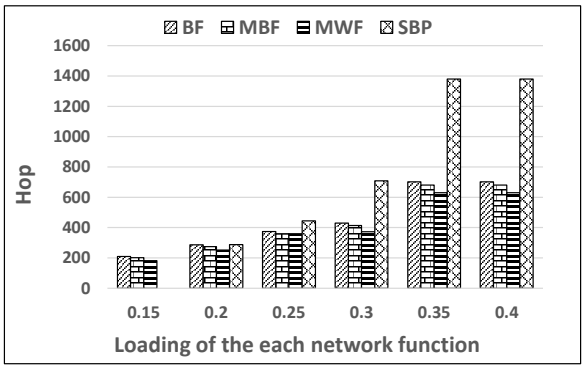
(a) Network Cost



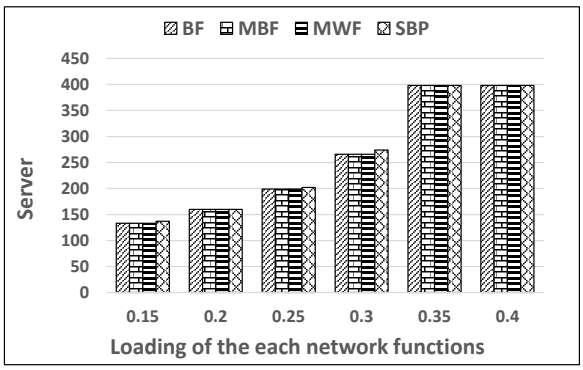
(b) Server Cost

Fig. 3. Under Different Number of the Chaining Requests

Figure 4 compares all of the algorithms under different loadings of the network functions. We fix the number of chaining requests at 200 and each network function has the same loading. As shown in Fig. 4(a), when the loading of network functions is fixed at 0.15, all network functions in a chaining request can be deployed on a server due to there being at most 6 network functions in a server, hence, the network cost of SBP is zero. When the loading of a network function



(a) Network Cost



(b) Server Cost

Fig. 4. Under Different Loading of Network Function

increases, SBP has the worst network cost. MWF can reduce network cost by 10% to 13% compared to BF and can reduce network cost by 47.5% to 54.3% compared to SBP. MBF can reduce network cost by 3.1% to 3.7% compared to BF and can reduce network cost by 41.5% to 50.7% compared to SBP. As shown in Fig. 4(b), the server cost among BF, MBF, MWF, and SBP are similar and each network function has the same loading. SBP uses more servers than others due to its approach not considering deploying the network functions on the used server. Since the loading of the network function is fixed in each case, the difference of the server cost among these algorithms is insignificant.

V. Conclusions

Deploying network functions in the chaining requests in a data center with low cost becomes an important issue. We treat service function chaining and placement problem as a multi-layer bin packing problem and discuss this problem in a tree-like network topology. Our approach uses multi-layer greedy algorithms for a tree-like network topology which deploys network functions as close as possible to the previous network functions in the same chaining request. The experimental results show that MWF can reduce bandwidth consumption by 15% while only increasing the number of used servers by 1% compared to the traditional Best-fit algorithm.

Acknowledgment

This research is supported in part by the Ministry of Science and Technology of Taiwan under Grant: MOST 104-2622-8-009-001, and is also supported by D-Link.

References

[1] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. A scalable, commodity data center network architecture. *ACM SIGCOMM Computer Communication Review*, 38(4):63–74, 2008.

[2] Jeremias Blendin, Julius Ruckert, Nicolai Leymann, Georg Schyguda, and David Hausheer. Position paper: software-defined network service chaining. In *Software Defined Networks (EWSDN), 2014 Third European Workshop on*, pages 109–114. IEEE, 2014.

[3] Seyed Kaveh Fayazbakhsh, Vyas Sekar, Minlan Yu, and Jeffrey C Mogul. Flowtags: Enforcing network-wide policies in the presence of dynamic middlebox actions. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 19–24. ACM, 2013.

[4] Aaron Gember, Anand Krishnamurthy, Saul St John, Robert Grandl, Xiaoyang Gao, Ashok Anand, Theophilus Benson, Vyas Sekar, and Aditya Akella. Stratos: A network-aware orchestration layer for virtual middleboxes in clouds. *arXiv preprint arXiv:1305.0209*, 2013.

[5] Aaron Gember, Prathmesh Prabhu, Zainab Ghadiyali, and Aditya Akella. Toward software-defined middlebox networking. In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, pages 7–12. ACM, 2012.

[6] Albert Greenberg, James Hamilton, David A Maltz, and Parveen Patel. The cost of a cloud: research problems in data center networks. *ACM SIGCOMM computer communication review*, 39(1):68–73, 2008.

[7] Albert Greenberg, James R Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David A Maltz, Parveen Patel, and Sudipta Sengupta. Vl2: a scalable and flexible data center network. In *ACM SIGCOMM computer communication review*, volume 39, pages 51–62. ACM, 2009.

[8] Dilip A Joseph, Arsalan Tavakoli, and Ion Stoica. A policy-aware switching layer for data centers. *ACM SIGCOMM Computer Communication Review*, 38(4):51–62, 2008.

[9] Xin Li, Jie Wu, Shaojie Tang, and Sanglu Lu. Let's stay together: Towards traffic aware virtual machine placement in data centers. In *INFOCOM, 2014 Proceedings IEEE*, pages 1842–1850. IEEE, 2014.

[10] Marcelo Caggiani Luizelli, Leonardo Richter Bays, Luciana Salete Buriol, Marinho Pilla Barcellos, and Luciano Paschoal Gaspary. Piecing together the nfv provisioning puzzle: Efficient placement and chaining of virtual network functions. In *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, pages 98–106. IEEE, 2015.

[11] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008.

[12] Sevil Mehraghdam, Matthias Keller, and Holger Karl. Specifying and placing chains of virtual network functions. In *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*, pages 7–13. IEEE, 2014.

[13] Hendrik Moens and Filip De Turck. Vnf-p: A model for efficient placement of virtualized network functions. In *Network and Service Management (CNSM), 2014 10th International Conference on*, pages 418–423. IEEE, 2014.

[14] Network Service Header. https://datatracker.ietf.org/doc/draft-quinn-sfc-nsh/.

[15] Zafar Ayyub Qazi, Cheng-Chun Tu, Luis Chiang, Rui Miao, Vyas Sekar, and Minlan Yu. Simple-fying middlebox policy enforcement using sdn. In *ACM SIGCOMM computer communication review*, volume 43, pages 27–38. ACM, 2013.

[16] Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, and Vyas Sekar. Making middleboxes someone else's problem: network processing as a cloud service. *ACM SIGCOMM Computer Communication Review*, 42(4):13–24, 2012.

[17] Justine Sherry, Sylvia Ratnasamy, and Justine Sherry At. A survey of enterprise middlebox deployments. 2012.

[18] Ying Zhang, Neda Beheshti, Ludovic Beliveau, Gregoire Lefebvre, Ravi Manghirmalani, Ravishankar Mishra, Ritun Patneyt, Meral Shirazipour, Ramesh Subrahmaniam, Catherine Truchan, et al. Steering: A software-defined networking for inline service chaining. In *Network Protocols (ICNP), 2013 21st IEEE International Conference on*, pages 1–10. IEEE, 2013.