Minimizing Transmission Cost for Multiple Service Function Chains in SDN/NFV Networks

Faqiang Liu[1,2], Xin Chen[1], Wei An[1*], Yong Peng[3], Jiuyue Cao[1,2], Yan Zhang[1]
[1] State Key Laboratory of Information Security, IIE, Chinese Academy of Sciences, Beijing 100093, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China
[3] Beijing University of Posts and Telecommunications, Beijing 100876, China

***Abstract*—Service Function Chaining (SFC) has received considerable attentions due to its potential in remarkably improving the flexibility and efficiency of networks. Network Functions Virtualization (NFV) and Software-Defined Networking (SDN) are becoming promising ways for realizing SFC. A key feature in SDN/NFV networks is the capability of dynamically composing network functions into complex services. In this paper, we first introduce SFC instantiation and form it as the model of the Shortest Path Tour Problem, then find the minimum transmission cost path with network function order constraints for a single SFC by exploiting a constructed multistage graph. Next, we derive minimum transmission cost paths for multiple SFC classes using the Dijkstra's Shortest Path Algorithm with resource constraints in a flexible way. Finally, some experiments are carried out and the results show the effectiveness and efficiency of our proposed method.**

I. Introduction

To improve the performance of current networks, lots of middleboxes have been introduced into the networks for providing all kinds of services, such as middleboxes for improving security (e.g, firewalls and intrusion detection systems) and for reducing bandwidth costs (e.g, WAN optimizers). However, those middleboxes usually come with high infrastructure and management costs which results in the complex and specialized processing in management tools across devices and vendors and the necessary for taking policy interactions between different network infrastructures into account. To address this issue, a service deployment concept, namely Service Function Chaining (SFC) [1], has received considerable attentions in research communities. It can provide the ability to define an ordered list of Network Functions (NFs) chained together and make paths for packets to be processed for improving the flexibility and efficiency of networks.

In the current networks, NFs are deployed with a static or semi-static manner, which is very ossified and inefficient for realizing SFC. Fortunately, SFC can be realized with high efficiency and flexibility with the help of virtualization. As two most attractive representatives of virtualization technology in network area, Network Functions Virtualization (NFV) [2] and Software-Defined Networking (SDN) [3] are becoming the most promising ways to realize SFC. NFV replaces dedicated and hardware-based middleboxes with software-based alternatives running on inexpensive commodity hardware (e.g., x86 server) for deploying NFs with high efficiency, while SDN steers flows through appropriate NFs to manage the network and enforce policies for combining NFs with high flexibility. The joint applications of NFV and SDN will provide a huge impetus for the rapid development of the current networks.

Numerous works such as optimization strategies for function placement [4], policy enforcement [9] and path selection [10] have been done extensively by research communities. These works have reached important achievements in developing SFC, but some challenges still need to be addressed. Firstly, the strict order of different NFs in an SFC must be guaranteed, which is a fundamental requirement for realizing SFC. Secondly, the response time of finding minimum transmission cost path for an SFC should incur low delay. Since every NF may have more than one instance in the network and many different paths exist for a specific SFC, choosing optimal paths for a specified SFC is challenging. Thirdly, resource constraints should be considered in routing process, such as link capacity and NF processing ability. In our work, the process of selecting NFs to compose an SFC is defined as SFC instantiation.

In this paper, we propose a multistage graph method for SFC instantiation in SDN/NFV networks and derive minimum transmission cost paths for multiple SFCs using the Dijkstra's Shortest Path Algorithm (DSPA) with resource constraints. Our main contributions are as follows:

- We propose a method of constructing a multistage graph to guarantee the strict order of NFs in SFC instantiation and get the minimum transmission cost path for a single SFC with the DSPA based on the multistage graph.
- Minimum transmission cost paths are derived for multiple SFCs with link capacity and NF processing

ability constraints, which can reduce the risk of link congestion and node overload.

- Some experiments are designed and conducted to evaluate the performance of our proposed algorithms. The results show that our method performs much better than some traditional methods in effectiveness and efficiency.

The rest of the paper is organized as follows. We briefly review the related works in Section II and present the system model and formulate the problem in Section III. Section IV proposes a multistage graph method for SFC instantiation and derive the minimum transmission cost paths for multiple SFCs with network resource constraints. Section V shows the experiment results and Section VI concludes this paper.

II. Related Works

As an emerging research topic in SDN and NFV, research questions about SFC include optimization strategies for function composition, enforcement of policies, path selection, etc. The work in [4] took traffic engineering into account to make a better decision for placement and selection of network services. In work [5], a software-defined middlebox networking framework was proposed to simplify the management of complex and diverse functionalities. Further OpenNF [6] provided an efficient and cooperative control for the reallocation of flows across network functions, which presents the combined benefits of NFV and SDN.

Recently, middleboxes are implemented in SDN data plane in [7]–[9]. The functions of their middleboxes can be realized based on the original SDN switches by embedding additional hardware or adding new features with the same basic protocol APIs as the normal switches. In [10], several algorithms were proposed for the path selection in SFC in order to ensure the quality of service (QoS) such as the service time and the resource cost. However, these works did not consider the processing order specified in an SFC, which is a fundamental requirement for SFC instantiation. The work in [11] introduced several greedy algorithms and a tabu-search-based heuristic algorithm to handle online mapping and scheduling problems of virtual network functions. However, link capacity and node processing ability constraints are not considered in this work.

Those promising works did not deal with how to combine disheveled NFs to satisfy special demands (e.g., the sequence of NFs in an SFC is not guaranteed) for SFC instantiation. How to realize SFC optimally and dynamically is still an challenging problem at present.

III. System Model and Problem Formulation

A. System Model

We consider SDN/NFV networks that NFs run on commodity servers connected by SDN routers. Different NFs such as Load Balancer (LB), Intrusion Detection System (IDS), Network Address Translation (NAT) and Firewall (FW) have been placed on servers. When the SDN controller receives an SFC request from the SDN router, the controller analyzes the request and computes optimal paths, then sets flow tables to the corresponding routers. After these operations, the flow can be routed along the desired paths. A simple case of two flow paths with the combination of NFs is shown as follows:

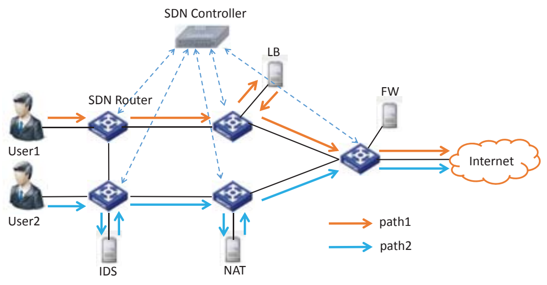


Fig. 1. Illustration of two flow paths with network functions combination.

Each flow enters the network at its ingress node and has to visit a set of NFs complying with the SFC policy before leaving the network at its egress node. All flows that share the same SFC are defined as an SFC class. The path that a flow goes through in an SFC order is defined as Service Function Path(SFP) in our work.

B. Problem Formulation

The minimizing transmission cost issue can be solved by two subproblems. The first subproblem is to satisfy the order constraints of NFs for a single SFC, and the second subproblem is to find the minimum transmission cost paths for multiple SFC classes without violating link capacity and NF processing ability constraints.

For the first subproblem, SFC can be realized in segmentations. For an SFC with N NFs, copy the network graph N times and construct a multistage graph. Every copy is a stage and one NF will be selected in a stage in the routing process. The sequence of the passed NFs can be guaranteed automatically due to the properties of the multistage graph. We call this method as the Multistage Graph (MG) method and the details are shown in Section IV.

For the second subproblem, a network graph $G = (V, A, W)$ is given, where V is the set of nodes, A is the set of arcs between different nodes, and W is the set of weights of different arcs. In our work, we assume that W represents the transmission cost of links. In fact, the transmission cost of a link is decided by many different elements with different optimization targets, such as link delays, hops or link utilization, etc. The details of getting the link transmission cost is not the focus of our work, so we just mark some different numbers to represent the cost of different links in our work. Let

$P_d = \{p_1, p_2, \cdots, p_n\}$ denote the set of paths that are feasible for SFC class d. For path p_i, its cost is defined as $C(p_i) = \sum_{e \in p_i} w_e$. The processing ability of NF f is denoted as c_f and the capacity of link e is denoted as c_e. Our objective is to find the minimum cost path set $\mathcal{P}$ for multiple SFC classes without violating the link capacity and the NF processing ability constraints. Assuming that there are k SFC classes to request services and the number of SFPs that route the flow of SFC class d is n_d. Let f_{dp} denote the flow of class d routed by path p. Then the formulations of the minimum transmission cost paths can be modeled as follows:

$$\textbf{min}: \quad \sum_{d=1}^{k} \sum_{i=1}^{n_d} C(p_i)$$

$$\textbf{s.t.}: \sum_{d} \sum_{p: e \in p} f_{dp} \le c_e, \forall e \tag{1}$$

$$\sum_{d} \sum_{p: f \in p} f_{dp} \le c_f, \forall f \tag{2}$$

$$f_{dp} \ge 0, \forall e, \forall f \tag{3}$$

Eqs. (1), (2) and (3) ensure the link capacity constraints, NF processing ability constraints, and non-negative flow on each candidate path, respectively.

IV. PROBLEM SOLUTION

A. SFC Instantiation

The correct sequence of NFs can be guaranteed by multistage graph method which is similar to Shortest Path Tour Problem (SPTP) proposed in work [12]. The SPTP consists of finding a shortest path from the given origin node to the given destination node in the graph G with the constraints that the optimal path p should successively pass through at least one node in the given node subsets $T_1, T_2, \cdots, T_N$, where different $T_k (k = 1, 2, \cdots, N)$ has no common parts.

According to the theory in [13], SPTP belongs to the complexity class P, since it can reduce to a single-source single-destination Shortest Path Problem (SPP) [12]. We can use multistage graph to transform the SPTP into SPP which can be solved easily. In the multistage graph, every stage is a copy of the original graph. One NF will be selected in a stage in the routing process. Once the multistage graph is constructed, the sequence of the NFs that an SFC goes through can be guaranteed automatically.

Before going further, some explanations are made first. Let $G = (V, A, W)$ be a directed graph where:

- V is a set of nodes numbered with $1, 2, \cdots, n$. $A = \{(i, j) | i, j \in V\}$ is a set of arcs. $W : E \to R^+$ is a function that assigns a non-negative cost to each arc $(i, j) \in A$;
- For each node $i \in V$, let $FS(i) = \{j \in V | (i, j) \in A\}$ and $BS(i) = \{j \in V | (j, i) \in A\}$ be the forward star and backward star of node i, respectively;
- A simple path $p_i = \{i_1, i_2, \cdots, i_k\}$ is without any repetition of nodes;
- The cost $C(p_i)$ of any path p_i is defined as the sum of costs of the arcs connecting consecutive nodes in the path.

The transformation process of the original graph to the multistage graph can be shown in *Algorithm 1*, where $N, T_k, G = (V, A, W)$ and $G' = (V', A', W')$ are the number of stages, the node sets, the original graph and the multistage graph, respectively.

In lines 2-12, at each iteration an arc (a, b) is added into A'. In particular, for each stage $k \in \{1, 2, \cdots, N - 1\}$, each node $v \in \{1, \cdots, n\}$, and each adjacent node w, where w is the forward star of node v. If v and w belong to stage T_k and T_{k+1} respectively, the start point of the new arc is node v in stage T_k and the end point of the new arc is node w in stage T_{k+1}. For node v, its sequence number is $v + (k - 1) \cdot n$, and for node w, its sequence number is $w + k \cdot n$. In fact, the sequence number of node i in stage T_k is $i + (k - 1) \cdot n$, which is always correct for all nodes in the multistage graph. So the arc $(a, b) = (v + (k - 1) \cdot n, w + k \cdot n)$ with cost c_{vw} will be added into the multistage graph. Otherwise, if both nodes v and w belong to the same stage T_k, the arc $(a, b) = (v + (k - 1) \cdot n, w + (k - 1) \cdot n)$ with cost c_{vw} will be added into the multistage graph. In lines 13-19, add arcs in the last stage. It is easy to see that $|A'| = N \cdot m$, where m is the number of the arcs in the original graph. Therefore the computational complexity of *Algorithm 1* is $O(N \cdot m)$. Note that $N \le n$. The worst case computational complexity is $O(n \cdot m)$.

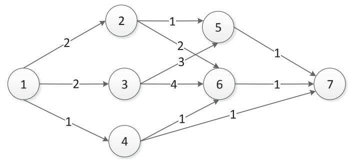


Fig. 2. Original graph of a toy example

The details of the transformation process is illustrated with an toy example as shown in Fig. 2. In this graph, we divide the nodes into four sets: $T_1 = \{1\}, T_2 = \{2, 3\}, T_3 = \{5, 6\}$ and $T_4 = \{7\}$. Node 4 is a common node without running any NF. Assuming that nodes 1 and 7 are the source and destination node, respectively, nodes 2 and 3 run the same NF X, nodes 5 and 6 run NF Y. The flow must enter the network from node 1, go through NF X, then NF Y, and finally reach the destination node 7.

If we use the DSPA in the original graph directly, $\{1, 4, 7\}$ will be the shortest path. In fact, it is not the correct path since NF X and Y are not in this path.

Algorithm 1 Transformation from original graph to the auxiliary graph

```
Require:
    Original Graph $G = (V, A, W)$; The number of stages $N$; Node subsets $T_k (k = 1, 2, \cdots, N)$
Ensure:
    Multistage graph $G' = (V', A', W')$
 1: Initialize:
    $V' := 1, 2, \cdots, i + (N-1) \cdot n$; $A' := \emptyset$; $W' := \emptyset$.
 2: for $k = 1$ to $N - 1$ do
 3:   for $v = 1$ to $n$ do
 4:     for $w \in FS(v)$ do
 5:       if $w \in T_{k+1}$ then
 6:         add $(v + (k-1) \cdot n, w + k \cdot n)$ with cost $c_{vw}$ into $A'$;
 7:       else
 8:         add $(v + (k-1) \cdot n, w + (k-1) \cdot n)$ with cost $c_{vw}$ into $A'$;
 9:       end if
10:     end for
11:   end for
12: end for
13: for $k = N$ do
14:   for $v = 1$ to $n$ do
15:     for $w \in FS(v)$ do
16:       add $(v + (N-1) \cdot n, w + (N-1) \cdot n)$ with cost $c_{vw}$ into $A'$;
17:     end for
18:   end for
19: end for
```

Now we begin to construct the multistage graph. Firstly, copy the original graph for 4 times, because there are 4 sets in the original graph. Then we mark the sequence number of nodes as $1, 2, 3, \cdots, 27, 28$. The sequence number of node i has become $i + (k-1) * n$ in the multistage graph, where k is the sequence number of the stage and n is the number of nodes in the origin graph. The method of adding arcs on the nodes is described as *Algorithm 1*. If all nodes and arcs have been added on the graph, the multistage graph of Fig. 2 is shown as Fig. 3.

When using DSPA in the multistage graph, we can drive the feasible paths directly based on the property of the multistage graph.

B. Minimum Transmission Cost Paths

A single minimum transmission cost path for an SFC class is easy to derive. In a real network, routing process must consider the situation of the network resources. For example, link congestion may occur if no effective methods to ensure that link capacity is enough to process the request flows. If some links or NFs reach their capacity, the remainder traffic flow need to be routed to other paths. So for multiple SFC classes, the

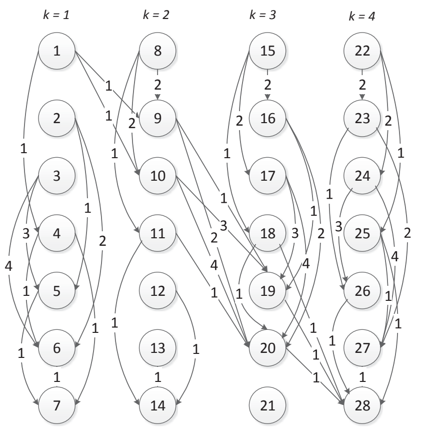


Fig. 3. Multistage graph with 4 stages

routing process must consider the resource constraints. We propose an effective algorithm to find the minimum transmission cost paths for multiple SFC classes with resource constraints. The details of the process is shown in *Algorithm 2*.

Algorithm 2 Minimum Cost Paths for Multiple SFCs

```
Require:
    SFC class set $S = \{SFC_1, SFC_2, \cdots, SFC_k\}$ with traffic demand set $H = \{h_1, h_2, \cdots, h_k\}$;
    The network topology $G$.
Ensure:
    Minimum cost path set $\mathcal{P}$.
 1: Initialize:
    For link $e$ in $G$, the flow $f_e$ = 0;
    Minimum cost path set $\mathcal{P} = \emptyset$;
    Sort $S$ based on $H$ in descending order;
 2: for $i : 1 \rightarrow k$ do
 3:   Construct multistage graph $G'$ with Algorithm 1 according to $SFC_i$;
 4:   while $h_i > 0$ do
 5:     Compute the minimum cost path $p$ for $SFC_i$;
 6:     Add path $p$ into $\mathcal{P}$;
 7:     Path update:
 8:     For all links in path $p$:
 9:     $u$ = min$\{c_e^{'} | e \in p\}$
        ($c_e^{'}$ is the residual capacity of link $e$);
10:     if $h_i < u$ then
11:       $f_e = f_e + h_i$; $c'_e = c'_e - h_i$; $h_i = 0$;
12:     else
13:       $f_e = f_e + u$; $c'_e = c'_e - u$; $h_i = h_i - u$;
14:       Find the bottleneck link $e''$ of the path $p$;
15:       Update the weight of the bottleneck link $e''$: $w(e'')$ = INFINITE;
16:     end if
17:     Until $h_i = 0$;
18:   end while
19: end for
```

In the initialization stage, sort the SFC class set S based on the traffic demand H with the descending order. Since the SFC with larger amount of flows affects the transmission cost heavily, we process such SFCs firstly.

On line 3, the algorithm constructs a multistage graph for the given SFC class with the method introduced in *Algorithm 2*. From line 4 to 18, the detailed procedure of the routing is shown. If some traffics need to be routed, the algorithm finds the minimum cost path with the DSPA and then computes the amount of traffics that can be routed. The traffic demand h_i and the residual capacity u of the bottleneck link in the minimum transmission cost path decide the amount of the flow that can be routed. If the bottleneck link of the path can not route all the request flows, then route flows with maximized value and then set the cost of the bottleneck link as ∞. Continue finding the next minimum cost path to route the residual flow until all the requests are processed.

It is worthy to be mentioned that the link capacity in the algorithm is decided by two variables: the link capacity and the processing ability of the node that connect to the link. In our design, the processing ability of NFs is also considered. Mark the processing ability of the NF connected to link e as c_f, so the real capacity of link e can be calculated as $c_e^* = min(c_e, c_f)$.

V. Experiments and Results

In this section, the performance of our method in SFC instantiation response time and minimum transmission cost are verified by experiments. In particular, Java program is implemented in Eclipse 4.4.2 to emulate the SDN/NFV networks according to the system model described in Section III. The experiments run on a server equipped with Intel i7-6700, quad core CPU at 3.40 GHz, 4GB DDR3 RAM. To evaluate the performance of the proposed algorithms, some test cases are generated to simulate the implementation of the algorithms. A random network graph is generated and the average degree of the random network is 4. Then, we randomly select nodes as candidate NF nodes. To illustrate the performance better, different test scenarios which represent the different combinations of NF types and NF instances are generated.

A. SFC Instantiation Response Time

For this simulation, response time is adopted as the indicator. The response time refers to the time that SDN controller spends on finding the minimum transmission cost path. The response time consists of constructing auxiliary graph and find the minimum transmission cost path for a request. We test three methods to show the performance. The first method is Multistage Graph (MG) method which is introduced in Section IV. The second method is All Pairs Shortest Path (APSP) method, which complexity is $O(n^3)$ [14]. APSP calculates the distance of all node pairs and then choose nodes which are related to the NFs in an SFC to construct a new graph, and run the DSPA on the new graph. Another method is Sort Paths(SP) which realizes SFCs like this, calculate the paths between different NFs that in an SFC, then sort these paths based on their cost value, and route flows on these paths from the minimum cost one. The algorithm of computing paths is the DSPA which complexity is $O(n^2)$. Quick sort which complexity is $O(m{\cdot}\log_2(m))$ is used as the sort algorithm in SP, where m is the number of paths that need to be sorted.

The results of the experiments are shown as follows.

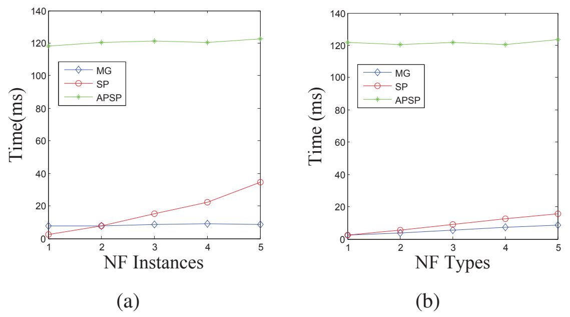


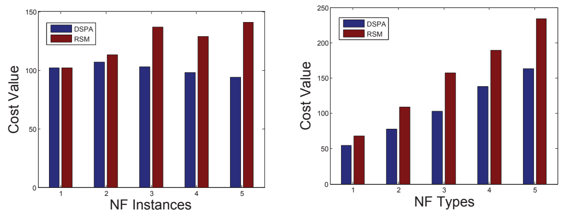
Fig. 4. The response time varies based on the number of NF Instances and Types.

Fig. 4(a) indicates that the response time varies based on the number of NF instances. In this case, the number of NF types is fixed to 3 and the number of NF nodes is 30. The number of NF instances varies from 1 to 5. The results show that MG method uses less time than the other two methods to find the minimum cost path. For APSP, the response time is much larger than the other two methods, since APSP needs to compute all minimum cost paths of different NFs. The response time of SP varies obviously as the number of the NF instances grows. For example, when the number of NF instances is less than 4, the response time of SP is similar to MG, but when the number grows larger than 4, the response time of SP increases obviously. As the number of the NF instances grows, SP need to run the DSPA more times to find all the paths that fit for the SFC, and the number of the paths that need to be sorted also increases, so the response time of SP grows obviously as the number of the NF instances grows.

Similar results in Fig. 4(b) show the response time varies based on the number of NF types. In this simulation, the number of NF instances is set to 3 and the number of NF nodes in the topology is 30. The number of NF types varies from 1 to 5. In this test, MG method is more stable and use less time than SP to find the minimum cost path.

B. Minimum Transmission Cost Paths

In order to evaluate the performance of the minimum transmission cost path algorithm for multiple SFC classes, we compare the DSPA and the Random Selection Method (RSM) in our experiment. The RSM realizes SFCs like this, in each stage of the multistage graph, the RSM select a NF randomly, then calculate the minimum transmission cost paths of the adjacent NFs and add all the minimum transmission cost paths belongs to an SFP to get the total path cost. For each case of the test, we run the experiments for 10 times and calculate the average value to show. The results of the experiments are shown as follows.



(a) (b)

Fig. 5. The cost value varies based on the Number of NF Instances and Types for 10 SFC classes

In Fig. 5(a), the results show that the cost value varies based on the number of NF instances. In this test, the cost of each link varies in [1,5] randomly. The number of NF types is fixed to 3 and the number of NF instances varies from 1 to 5. The experiment results show that when there is only one instance for each type of NF, the cost value of the DSPA and the RSM is the same, since in this case, the two methods can get the same path. As the number of NF instances grows, the cost of the DSPA becomes smaller than the RSM. For example, when the number of the NF instances is 5, the cost value of the DSPA is 94, while the cost value of the RSM is 137. Compared with the RSM, the DSPA reduced by about 31.4% for the same SFC classes to route the traffic in transmission cost.

The variation of the cost value based on the number of NF types is shown in Fig. 5(b). In this case, the number of the NF instances is fixed to 3. Since more NF types denotes more stages in the multistage graph, the cost value becomes larger as more and more segmentations added in the routing process. The advantage of the DSPA is obvious in this experiments, since the cost value of the DSPA is always smaller than the RSM.

From the results above, we find that the performance of MG in response time is much better than APSP and SP based on different NF instances and types, and the DSPA has obvious advantages in deriving minimum transmission cost paths for multiple SFC classes. The experiment results show that our solution has a big advantage in efficiency and effectiveness in solving the SFC instantiation problem.

VI. Conclusion

In this paper, we presented a method for deriving the minimum transmission cost path for an SFC by exploiting a constructed auxiliary multistage graph for the sake of the order of network functions. For multiple SFCs, a polynomial-time algorithm has been provided for finding multiple minimum transmission cost paths under the network resource constraints. The experiment results have shown the effectiveness and efficiency of our proposed method. In our future work, we plan to consider much more constraints for implementing SFCs and focus on the optimization of the MG algorithm to reduce the response time especially for large scale networks.

Acknowledgment

This work is supported in part by the Scientific Research Foundation of the Institute of Information Engineering, Chinese Academy of Sciences (Grant No. Y6Z0011105) and the National Natural Science Foundation of China (Grant No.61572497). The corresponding author is Wei An (anwei@iie.ac.cn).

References

[1] Internet Engineering Task Force. Service Function Chaining (SFC) Architecture: IETF RFC 7665. 2015 .

[2] Telecommunications Standards Institute. Network functions virtualization (NFV): Architectural framework. White Paper, 2014.

[3] Open Networking Foundation. Software-defined networking: the new norm for networks. 2012.

[4] R. Cohen, G. Nakibly, "A traffic engineering approach for placement and selection of network services," IEEE/ACM Trans. Netw. 17 (2)(2009) 487-500.

[5] A. Gember, P. Prabhu, Z. Ghadiyali, "Toward software-defined middlebox networking," in Proceedings of the ACM Hotnets 12, October 29-30, Seattle, WA, USA, 2012.

[6] Gember-Jacobson A, Viswanathan R, Prakash C, et al. "OpenNF: Enabling innovation in network function control," ACM SIGCOMM Computer Communication Review, 2015, 44(4): 163-174.

[7] V. Sekar, N. Egi, S. Ratnasamy, M. K. Reiter, and G. Shi, "Design and implementation of a consolidated middlebox architecture," in Proceedings of the 9th USENIX conference on NSDI, 2012, pp. 24-24.

[8] Zhang Y, Beheshti N, Beliveau L, et al. "Steering: A software-defined networking for inline service chaining," Network Protocols (ICNP), 2013 21st IEEE International Conference on. IEEE, 2013: 1-10.

[9] S. K. Fayazbakhsh, V. Sekar, M. Yu, and J. C. Mogul, "Flow-tags:enforcing network-wide policies in the presence of dynamic middlebox actions," in Proceedings of the second ACM SIGCOMM workshop on HotSDN. ACM, 2013, pp. 19-24.

[10] Kim T, Kim S, Lee K, S Park, "A QoS Assured Network Service Chaining Algorithm in Network Function Virtualization Architecture," in Proc. IEEE/ACM International Symposium on CCGrid, May 2015.

[11] Mijumbi R, Serrat J, Gorricho J L, et al., "Design and Evaluation of Algorithms for Mapping and Scheduling of Virtual Network Functions," in Proc. IEEE NetSoft, April 2015.

[12] Festa, Paola. "The shortest path tour problem: problem definition, modeling, and optimization",Proceedings of INOC. 2009.

[13] R.M. Karp. "Reducibility among combinatorial problems", In R.E. Miller and J.W. Thatcher, editors,Complexity of Computer Computations. Plenum Press, 1972.

[14] Cao Z, Kodialam M, Lakshman T V, "Traffic steering in software defined networks: planning and online routing," ACM SIGCOMM Computer Communication Review. ACM, 2014, 44(4): 65-70.