Cluster Computing (2019) 22:S2331–S2339
https://doi.org/10.1007/s10586-018-1840-9

Performance evaluation of revised virtual resources allocation scheme in network function virtualization (NFV) networks

Hyuncheol Kim[1]

Received: 22 December 2017 / Revised: 8 January 2018 / Accepted: 11 January 2018 / Published online: 27 January 2018

Abstract

The design, management, and operation of network infrastructure have evolved during the last few years, leveraging on innovative technologies and architectures. With such a huge trend, due to the flexibility and significant economic potential of these technologies, software-defined networking (SDN) and network functions virtualization (NFV) are emerging as the most indispensable key catalysers. SDN/NFV enhancing the infrastructure agility, thus network operators and service providers are able to program their own network functions (e.g., gateways, routers, load balancers) on vendor-independent hardware substrate. One of the most important considerations in NFV deployment is how to allocate the virtual resources that are needed to provide flexible virtual network services in an NFV-based network infrastructure. Thus, the most important prerequisite for NFV deployment is achieved fast, scalable and dynamic composition and allocation of networks functions (NFs) to implement network services (NSs). We have proposed a revised RA algorithm that integrates embedding and scheduling of virtual network functions (VNFs) simultaneously. In this paper, performance evaluation of the revised RA algorithm is performed and the proposed algorithm is applied more effectively in NFV environment where the demand for resources is flexible and concentrated.

Keywords Virtual resource allocation · Resource embedding · Resource scheduling · Network function virtualization

1 Introduction

The design, management, and operation of network infrastructure have evolved during the last few years, leveraging on innovative technologies and architectures. With such a huge trend, due to the flexibility and significant economic potential of these technologies, software-defined networking (SDN) and network functions virtualization (NFV) are emerging as the most indispensable key catalysers. SDN/NFV enhancing the infrastructure agility, thus network operators and service providers are able to program their own network functions (e.g., gateways, routers, load balancers) on vendor-independent hardware substrate. They facilitating the design, delivery, and operation of network services in a dynamic and scalable manner [1–3].

NFV is a technology that allows network services to be controlled and managed in software by separating various network functions (NFs) from hardware devices in dedicated network equipment and implementing them in a high-performance general-purpose server. Through the network function virtualization (NFV), companies such as network service providers and carriers have sought to dramatically reduce CAPEX/OPEX by improving the speed of new service provisioning and flexibility of network construction through the S/W-based devices provided by NFV. One of the most important considerations in NFV deployment is how to allocate the virtual resources that are needed to provide flexible virtual network services in an NFV-based network infrastructure. Thus, the most important prerequisite for NFV deployment is achieved fast, scalable and dynamic composition and allocation of networks functions (NFs) to implement network services (NSs) [3–6].

However, until the introduction of NFV to provide commercial services, there are many technical issues to be solved such as guaranteeing performance, stability, and support for the multi-vendor environment, ensuring perfect interoperability and linking existing virtual and non-virtual resources

Funding for this paper was provided by Namseoul University.

✉ Hyuncheol Kim
hckim@nsu.ac.kr

[1] Department of Computer Science, Namseoul University, Cheonan, Korea

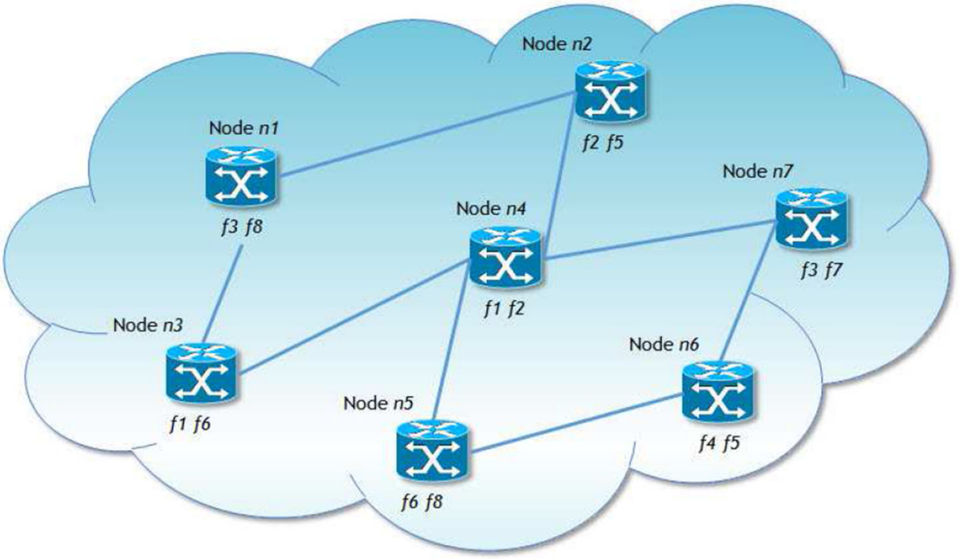

Fig. 1 An example network and simulation network ($|\beta_{i,j}| == 2$)

[7]. We have proposed a revised RA algorithm that integrates embedding and scheduling of virtual network functions (VNFs) simultaneously [8]. In this paper, performance evaluation of the revised RA algorithm is performed and it is proved that the proposed algorithm is effectively applied in NFV environment.

The composition of this paper is as follows. In Sect. 2, we review the revised online combined VNF embedding and scheduling algorithm. In Sect. 3, we describe the network and simulation environment for running the revised RA algorithm. We also analyze the performance of the revised RA algorithm through simulation. Finally, the paper concludes in Sect. 4.

2 Related work and revised virtual resources allocation scheme

Resource allocation (RA) in NFV requires efficient algorithms to determine on which high volume servers (HVSs) VNFs are placed, and be able to migrate functions from one server to another for such objectives as load balancing, reduction of CAPEX and OPEX, energy saving, recovery from failures, etc. [2,9–11].

Riera et al. [12] provided the first formalization of the scheduling problem in NFV as a Resource Constrained Project Scheduling Problem. Mijumbi et al. [2,13,14] proposed an approach to tackle the online VNF Forwarding Graph Embedding (VNF-FGE) and VNFs Scheduling (VNFs-SCH) by proposing greedy and metaheuristic (tabu search) approaches aiming at reducing the flow execution time. The algorithms perform both mapping and scheduling at the same time (one-shot) resulting in high acceptance ratio, low average flow time and low embedding cost. Hyuncheol et al. [8] considered a resource sharing approach that allows a given VM to process multiple VNFs, one after another (possibly) from a queue.

A network example to illustrate the operation process of the revised RA is shown in Fig. 1. As shown in Fig. 1, the network consists of 7 HVS nodes from $n1$ to $n7$, and supports 8 VNFs from VNF $f1$ to $f8$. Node $n1$ supports VNF $f3$, $f8$, and node $n2$ supports VNF $f4$ and each node supports specified VNFs. We first assume that NS $S1 = f8 - f2 - f3 - f6 - f5$ requests arrive at $T1$ time to examine the operations of the revised-RA algorithm. It is also assumed that 3, 2, 4, 2, and 3 TUs are required for VNF $f8$, $f2$, $f3$, $f6$, and $f5$, respectively.

(1) If the scheduling TU space for VNF allocation in one or more nodes is empty
Fig. 2 shows that when the NS $S1 = f8 - f2 - f3-$ $f6 - f5$ request arrives at $T1$ time, at least one of the time slots of nodes $n1$, $n5$ and $n7$ capable of processing VNF $f8$ is empty.

- If only one node has space for VNF $f8$, the VNF is assigned to that node. As shown in Fig. 2(a), at the time of $T1$, only $n1$ that can support VNF $f8$ exists.
- As shown in Fig. 2(b), if the space is empty in one or more nodes, the node with the smallest value of ω_j representing the network distance from the previous VNF service node to the empty node (eg, $n1$, $n5$, $n7$) is seleted to allocate VNF $f8$.
- If there is more than one node that is the first VNF of NS or the same value of ω_j as $f8$ of $S1$, then one node is chosen randomly.

(2) If the space of nodes for VNF allocation is not as empty as desired TU, but $(TU > 1)$

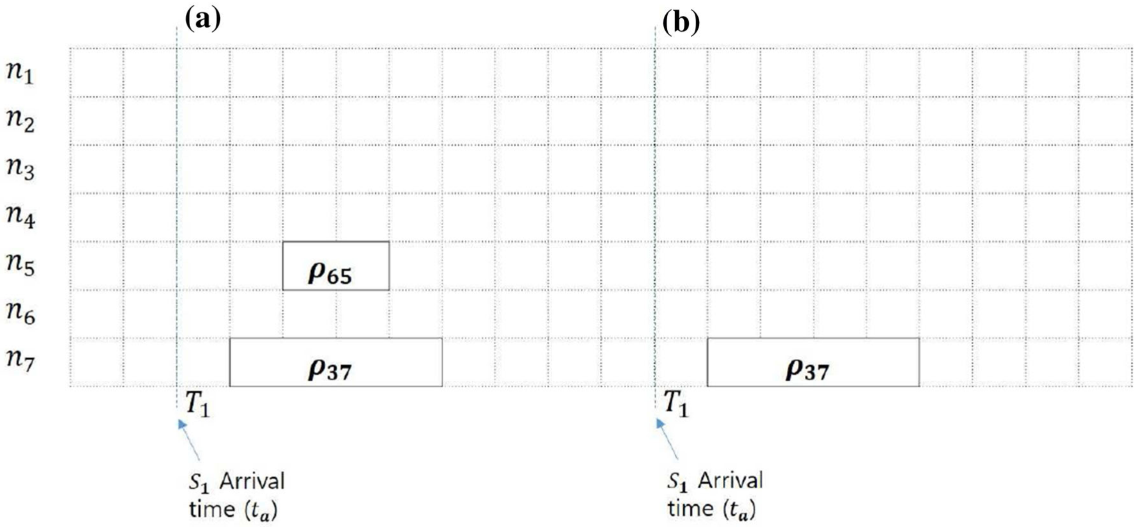

Fig. 2 At least one space of nodes for VNF allocation is empty

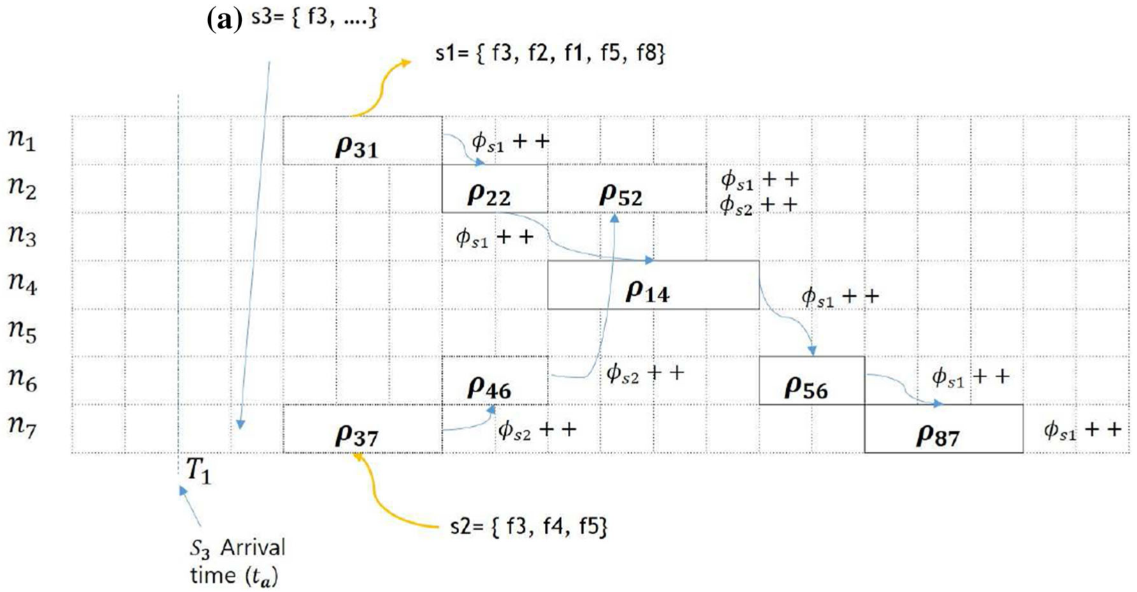

Fig. 3 Key components of rehabilitation

Fig. 3 shows that NS $S1 = f3 - f2 - f1 - f5 - f8$ and $S2 = f3 - f4 - f5$ are already scheduled when the NS $S3$ setup request arrives at time $T1$. In this case, as shown in Fig. 3(a), NS $S3 = f3$.... arrives, there are already scheduled $n1$ ρ_{31} and $n7$ ρ_{37}, so VNF $f3$, which requires 3 TUs, is difficult to allocate directly and the relocation process must be performed. Unlike the existing Mijumbi algorithm, in the revised-RA, allocation starts directly from time $T1$ as shown in Fig. 3(a). Also, since VNF $f3$ is executed for 3TU, $S1$ or $S2$ must be rearranged for 1TU backwards.

To do this, the concept of interference, $\phi_{i,j}$, is introduced in the revised-RA algorithm. $\phi_{i,j}$ denotes the number of VNFs to be relocated due to lack of TU when allocating VNF i to virtual node j. In Fig. 3, in order to schedule $f3$ of NS $S3 = f3$..., two empty TUs exist in $n1$ and $n7$, and $s1$ or $s2$ already scheduled NS must be scheduled back by 1TU.

In order to reschedule S1, ρ_{31}, ρ_{22}, ρ_{52}, ρ_{14}, ρ_{56}, and ρ_{87} must be rescheduled for 1 TU and for rescheduling S2, ρ_{37}, ρ_{46}, and ρ_{52} must be rescheduled for 1 TU. Since $\phi_{3,2}$ is 6, and $\phi_{3,7}$ is 3, S3 is scheduled to $n7$ as shown in In Fig. 4. ρ_{37}, ρ_{46}, and ρ_{52} corresponding to S2 are rescheduled for 1 TU. Fig. 4(b) shows the time in which the VNF supporting the $f3$ of S2 and the $f3$ of S3 are executed in parallel.

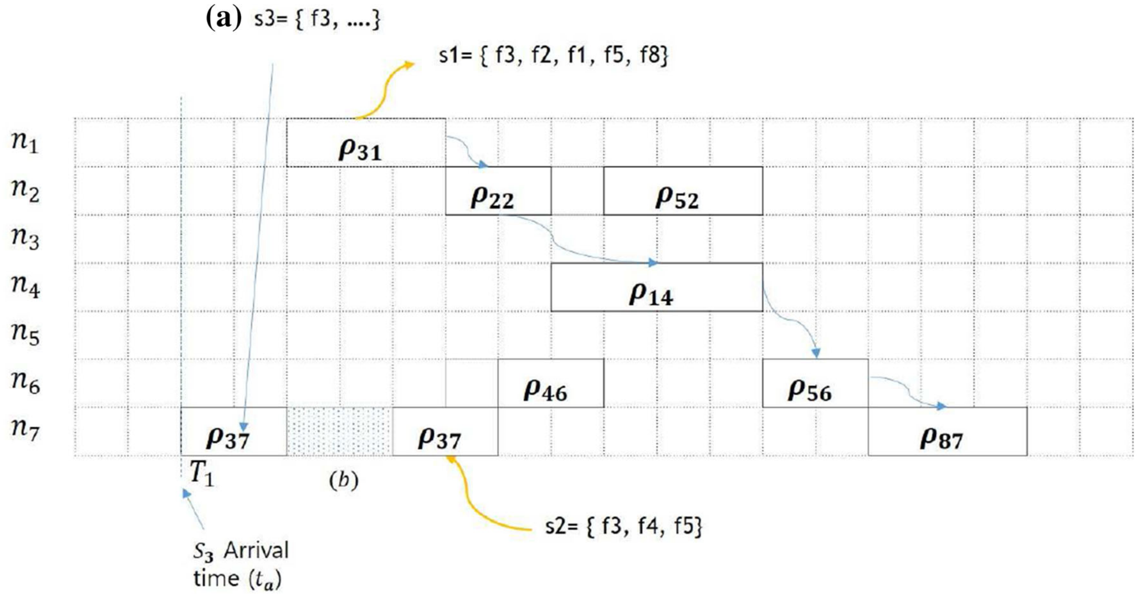

Fig. 4 The scheduling table after the $f3$ of NS $S3$ is rescheduled

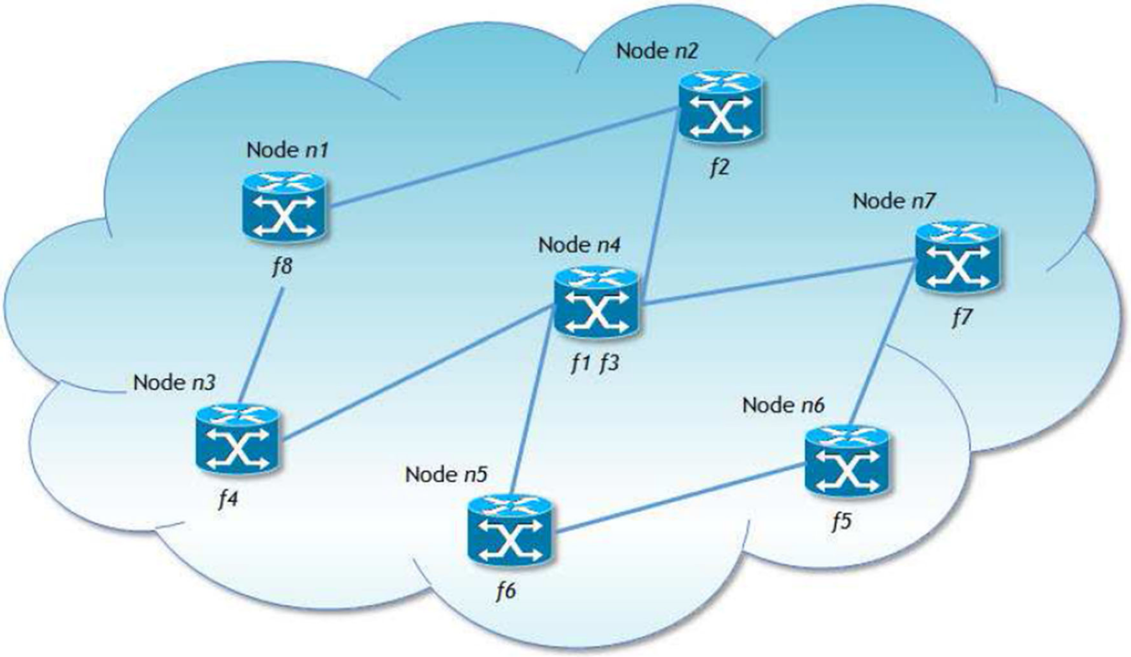

Fig. 5 Simulation network ($|\beta_{i,j}| == 1$)

(3) When the space of nodes for VNF allocation is ($TU == 0$)

In this case, all nodes capable of processing VNF i are currently being serviced. Therefore, in this case ($|\beta_{i,j}| == 1$), the node with the smallest expected completion time, π_j, is selected among the nodes capable of serving VNF i. If there are several nodes with the same π_j value, the node is selected randomly.

3 Performance evaluation

3.1 Simulation environments

To evaluate the performance of the revised-RA algorithm, we implemented discrete simulator using C language. In the performance evaluation, NS requests arrived one by one according to the Poisson distribution and were scheduled with node mapping. For the simulation, the network as shown

Table 1 Simulation parameters

Parameters	Value
Number of nodes	7
Function processed by each node	1 ~ 2
Function processing times	2 ~ 4 TU
Number of functions per service	5
Service processing deadline	60 TU
Number of Network Services	8
	f5-f1-f2- f8-f7
	f6-f1-f7- f5-f8
	f4- f7-f8-f6-f3
	f5-f3-f7-f4- f1
	f7-f1-f5-f3- f8
	f4-f1-f6-f2-f5
	f2-f6-f4-f5-f8
	f8-f4-f2-f3-f6

in Fig. 1 and Fig. 5 was configured, eight different NF (Network functions) were randomly defined, and NS was composed of five of these eight NFs.

To maximize the simulation effect, every node in Fig. 1 can process only two functions per node ($\left|\beta_{i,j}\right| == 2$)). Also, in Fig. 5, all nodes can only process one function per node ($\left|\beta_{i,j}\right| == 1$)). For the simulation, it has ($\left|\beta_{i,j}\right| == 2$)) value only at node n4.

The parameters and contents required for the simulation are shown in Table 1. Each simulation was run with a total of 100 service arrivals in the 400 TU period including service arriving, mapping, scheduling, and departing after processing.

3.2 Simulation results

Figures 6 and 7 compare the revised-RA and the Mijumbi algorithm, respectively, in the Fig. 1 network. In addition, Figs. 8 and 9 compare the revised-RA algorithm and the Mijumbi algorithm in the Fig. 5 network, respectively. Figure 10 shows the completion time results from four simulation results in one graph.

As shown in Figs. 6 and 7, when the NS arrival rate is low, the revised-RA algorithm and the Mijumbi algorithm show similar VNF completion time results. However, revised-RA

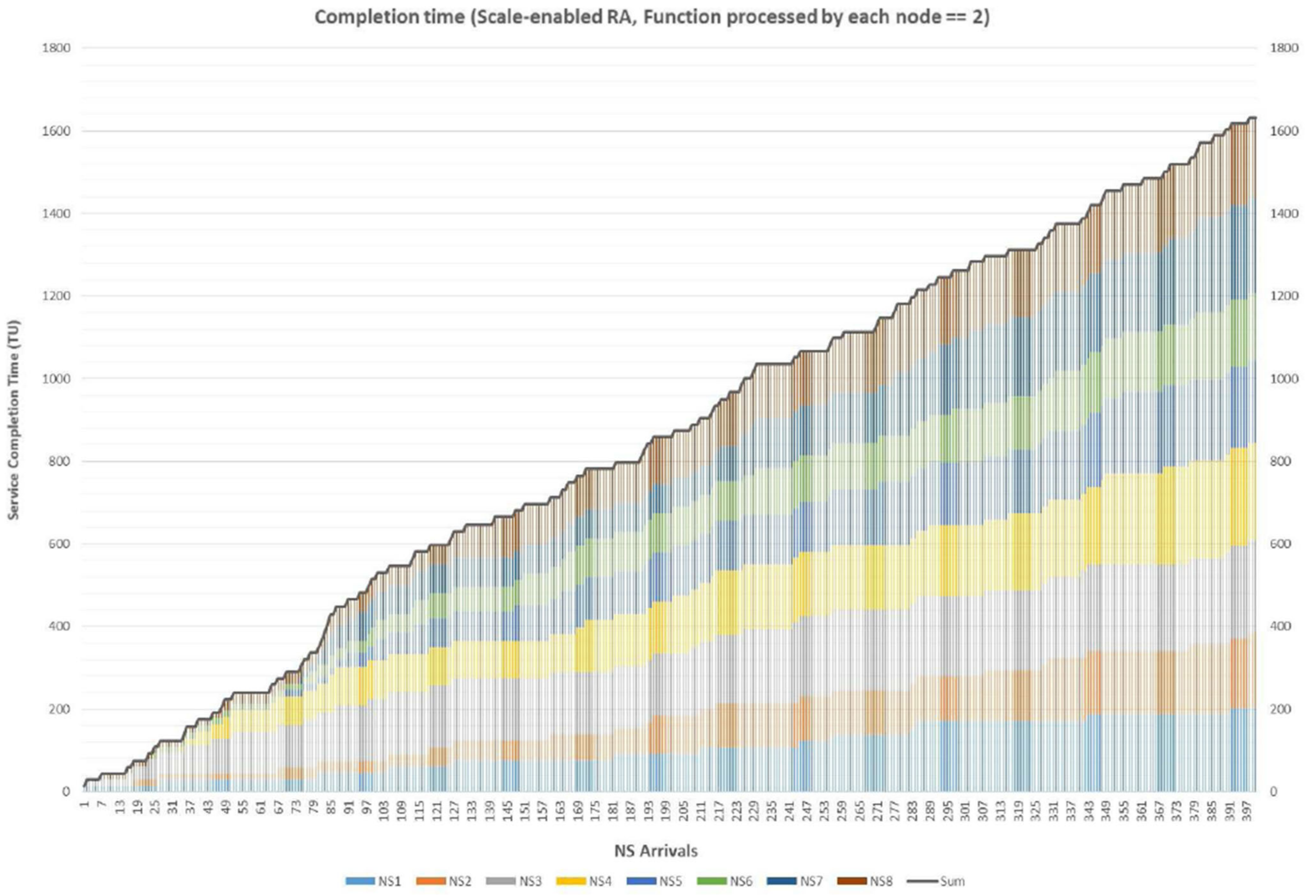

Fig. 6 Simulation results -1 ($\left|\beta_{i,j}\right| == 2$)

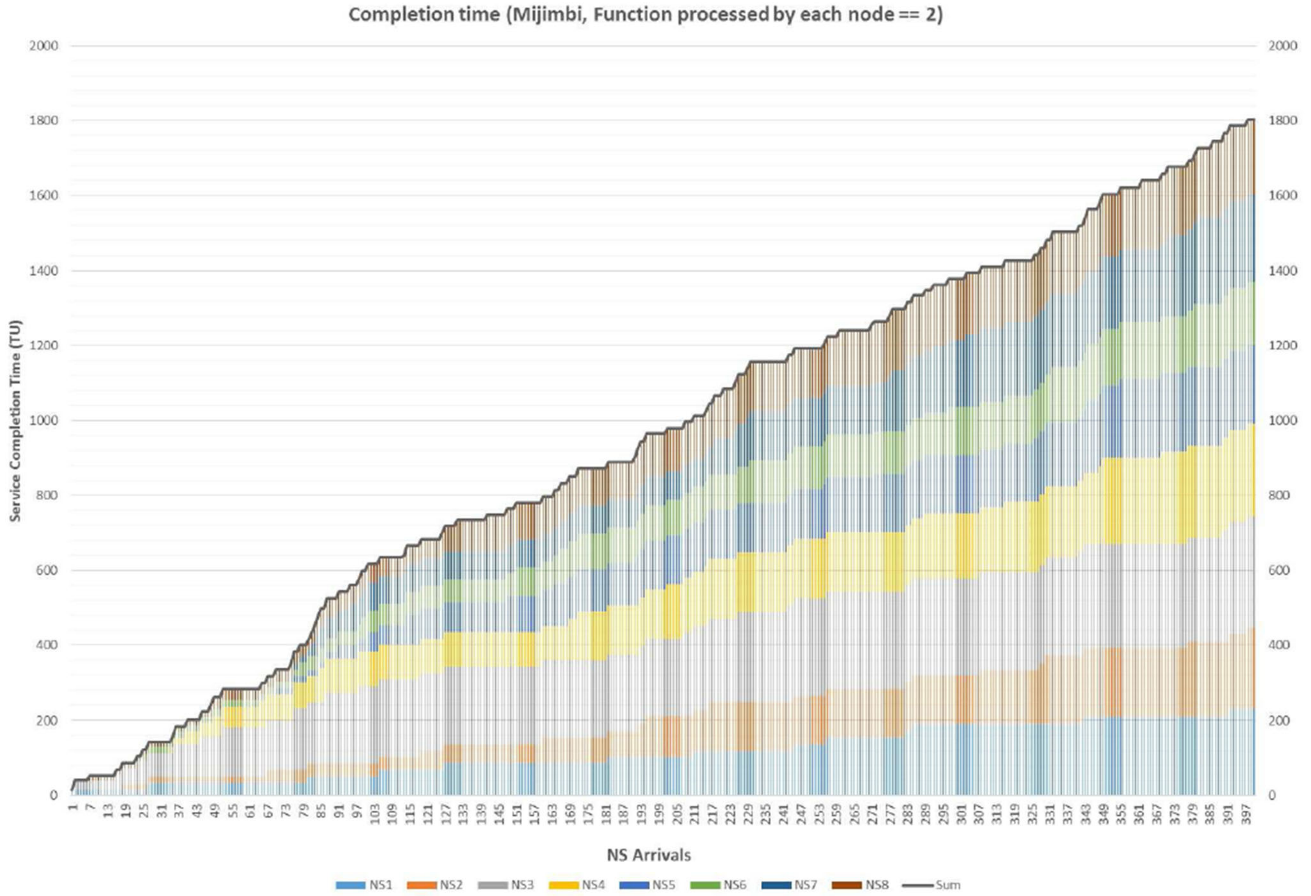

Fig. 7 Simulation results -2 ($|\beta_{i,j}| == 2$)

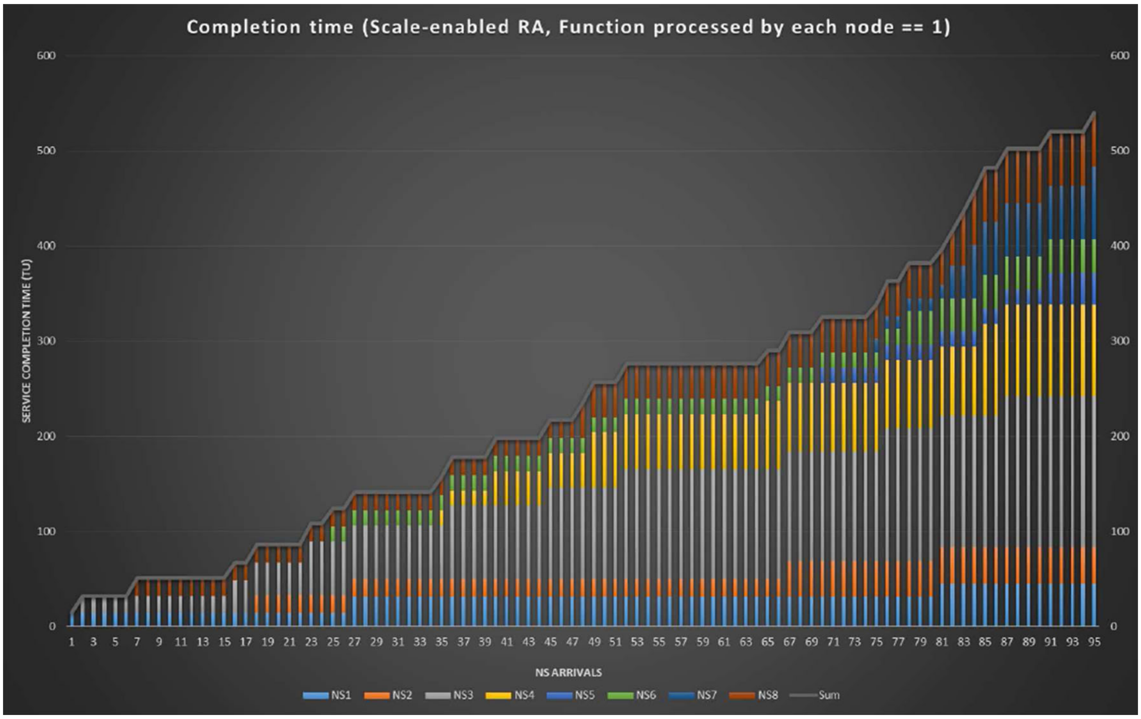

Fig. 8 Simulation results -3 ($|\beta_{i,j}| == 1$)

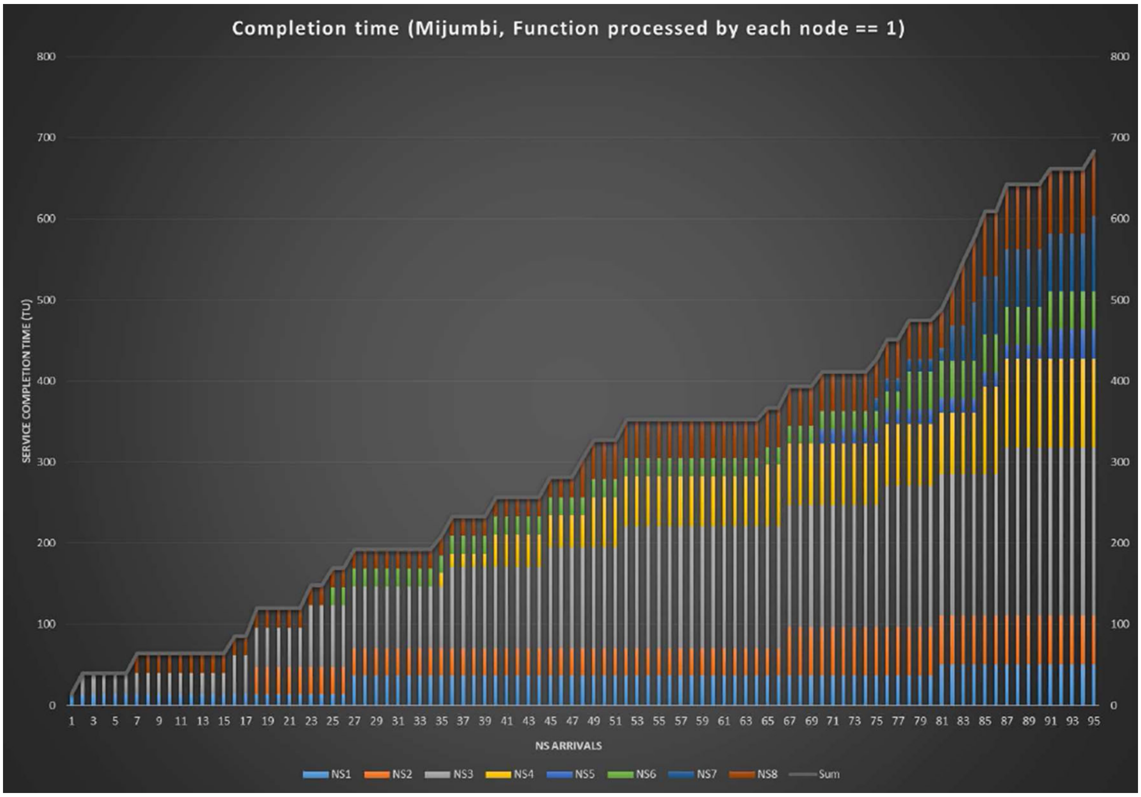

Fig. 9 Simulation results -4 ($\left|\beta_{i,j}\right|$ == 1)

shows a reduction in VNF completion time of 14–16% or more in a region where NS arrival rate is concentrated. If one server can support multiple VNFs, the revised-RA will perform much better if the resource requests are concentrated in a short period of time.

Especially, as shown in Figs. 8 and 9, in the case of ($\left|\beta_{i,j}\right|$ == 1), the revised RA algorithm shows a VNF completion time shortening of 18-22% than the Mijumbi algorithm. In other words, the revised RA algorithm shows better performance for networks with smaller ($\left|\beta_{i,j}\right)$ values.

4 Conclusion

In the NFV network, existing specific network functions are implemented as a single S/W module, which is called VNF. In other words, VNF is a virtualized instance of the existing network NF, and VNF enables modularization and individualization of network functions as well as individual management. In addition, VNFs can be deployed anywhere on the network because VNFs in NFVs can be installed and deployed on a general-purpose server and dynamically migrate between servers. Therefore, one of the most important considerations for establishing an NFV network and providing dynamic services is to determine how to dynamically allocate resources (VNFs) as a basic component of network services. Ultimately, the effective NFV RA algorithm has become a key factor in building a substantial NFV network.

However, until the introduction of NFV to provide commercial services, there are many technical issues to be solved such as guaranteeing performance, stability, and support for the multi-vendor environment, ensuring perfect interoperability and linking existing virtual and non-virtual resources. We have proposed a revised RA algorithm that integrates embedding and scheduling of VNFs simultaneously. In this paper, performance evaluation of the revised RA algorithm is performed and the proposed algorithm is applied more effectively in NFV environment where the demand for resources is flexible and concentrated.

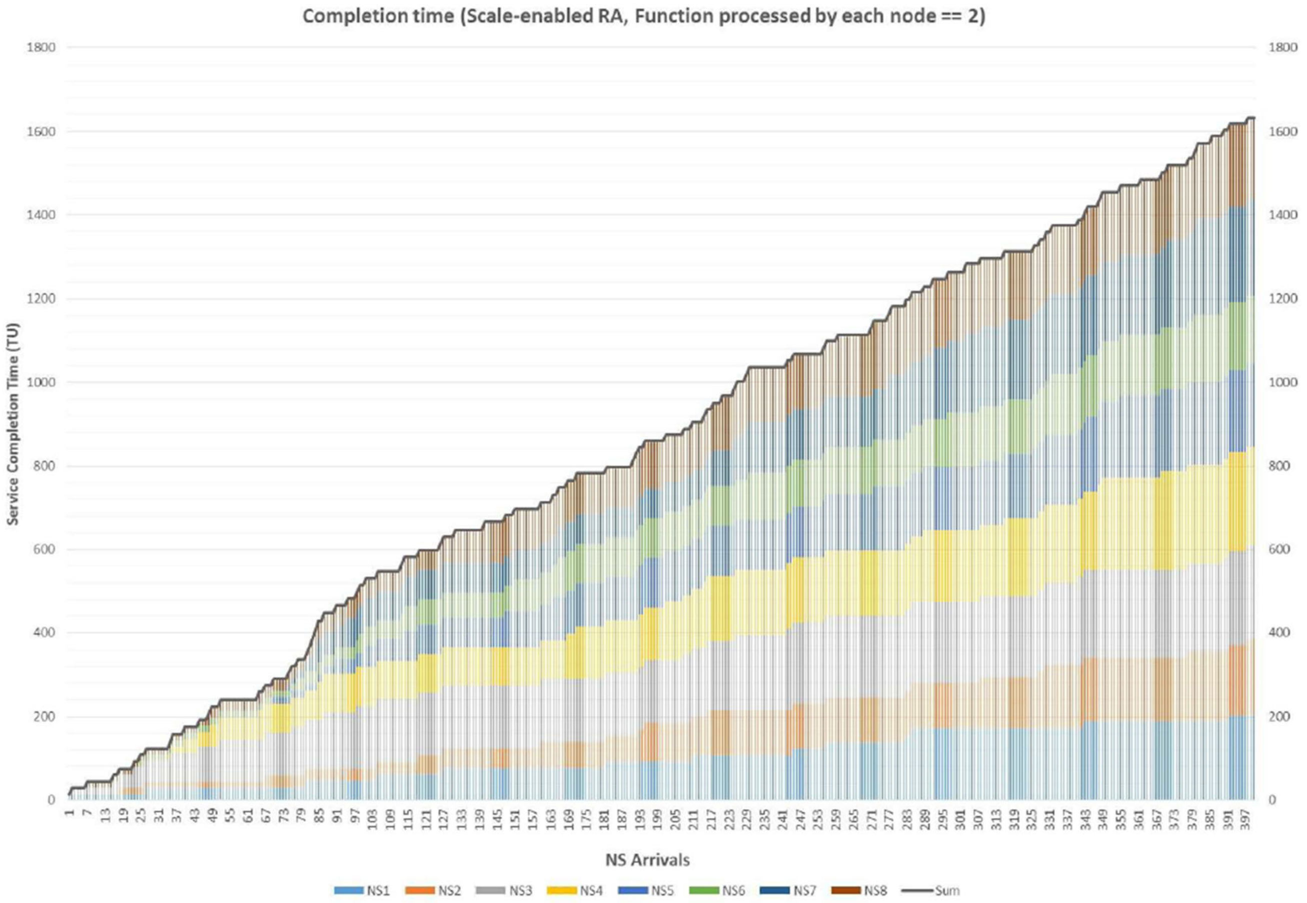

Fig. 10 Simulation results -5

References

1. Omnes, N., Bouillon, M., Fromentoux, G., Le Grand, O.: A programmable and virtualized network & IT infrastructure for the internet of things: How can NFV & SDN help for facing the upcoming challenges. In: International Conference on Intelligence in Next Generation Networks (ICIN), pp. 64–69 (2015)
2. Mijumbi, R., Serrat, J., Gorricho, J., Bouten, N., De Turck, F., Boutaba, R.: Network function virtualization: state-of-the-art and research challenges. IEEE Commun. Surv. Tutor. **18**(1), 236–262 (2015)
3. Akyildiz, H.A., Saygun, E.: SDN-NFV-cloud introduction in the context of service chaining. Signal Processing and Communications Applications Conference (SIU), pp. 2605–2608 (2015)
4. ETSI, Network Functions Virtualization, (2017). http://www.etsi.org/technologiesclusters/technologies/nfv
5. Matias, J., Garay, J., Toledo, N., Unzilla, J., Jacob, E.: Toward an SDN-enabled NFV architecture. IEEE Commun. Mag. **53**(4), 187–193 (2015)
6. Lopez, L.I.B., Caraguay, A.L.V., Villalba, L.J.G., Lopez, D.: Trends on virtualisation with software defined networking and network function virtualisation. IET Netw. **4**(5), 255–263 (2015)
7. Woesner, H., Verbeiren, D.: SDN and NFV in telecommunication network migration. In: European Workshop on Software Defined Networks (EWSDN), pp. 125–126. (2015)
8. Kim, H.: Revised virtual resources allocation scheme in network function virtualization (NFV) enabled network. Lecture Notes in Electrical Engineering, vol. 425, pp. 417–423 (2017)
9. Herrera, J.G., Botero, J.F.: Resource allocation in NFV: a comprehensive survey. IEEE Trans. Netw. Serv. Manag. **13**(3), 518–532 (2016)
10. Beck, M.T., Botero, J.F.: Coordinated allocation of service function chains. In: IEEE Global Communications Conference, pp. 1–6 (2015)
11. Kao, H.Y., Yang, Y.M., Huang, C.H.: Dynamic virtual machines placement in a cloud environment by multi-objective programming approaches. In: International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS), pp. 364–365 (2015)
12. Riera, J.F., Hesselbach, X., Escalona, E., Garca-Espin, J.A., Grasa, E.: On the complex scheduling formulation of virtual network functions over optical networks. In: International Conference on Transparent Optical Networks (ICTON), pp. 1–5 (2014)
13. Mijumbi, R., Serrat, J., Gorricho, J.L., Latr, S., Charalambides, M., Lopez, D.: Management and orchestration challenges in network functions virtualiza-tion. IEEE Commun. Mag. **54**(1), 98–105 (2016)
14. Mijumbi, R., Serrat, J., Gorricho, J.L., Bouten, N., De Turck, F., Davy, S.: Design and evalua-tion of algorithms for mapping and scheduling of virtual network functions. In: IEEE Conference on Network Softwarization (NetSoft), pp. 1–9 (2015)

Hyuncheol Kim received his Ph. D. degree in the department of information engineering from Sungkyunkwan University, Korea, in 2005. From 1992 to 2002, he was senior engineer in Electronics and Telecommunications Research Institute (ETRI), Daejon, Korea. He has been a professor in the department of computer science in Namseoul University, Korea. His research interests include cloud computing, next generation network and embedded computing systems.