Multi-domain VNF mapping

J. Martín-Pérez
IMDEA Networks Institute
Universidad Carlos III de Madrid
Madrid, Spain
Email: jorge.martin@imdea.org

Carlos J. Bernardos
Universidad Carlos III de Madrid
Madrid, Spain
Email: cjbc@it.uc3m.es

I. Introduction

Network Services (NS) like video streaming are usually made up of several Network Functions (NF) that compose a Service Function Chain (SFC). Examples of these NFs can be firewalls, video optimizers, parental control, etc.; and they can be virtualized (VNFs) using Network Function Virtualization (NFV) technologies.

When a NS deployment is requested, an algorithm must decide where to map the SFC in the infrastructure satisfying the imposed link and computational requirements. This is known as the VNF mapping problem, and it is a NP-hard problem [1] that can be solved using techniques as Integer Linear Programming (ILP), Mixed ILP (MILP) [2], [3]; and heuristic algorithms [4], [5].

None of the cited solutions to the problem deal with a multi-domain scenario, i.e. an environment where multiple Service Providers (SPs) share resources to perform NS deployments. This work presents how we solve the VNF mapping problem when multiple SPs are involved, and the performance of several greedy algorithms we have tried.

II. Multi-domain VNF mapping

We solve the VNF mapping problem in a federation of SPs that share fat-tree data centers [6] under their domains.

The simulator developed in our research creates a graph for every SP to represent its resources and the ones other SPs share with it in the federation (Figure 1).

Every time a NS request arrives to a SP, the simulator relies on an algorithm to map the NS's SFC to the SP's graph. In this work the simulator is fed with greedy algorithms that travel through the SP's graph using the link's delay as cost, and they try to find the closest server to host the next VNF present in the SFC.

Our objective is to minimize the delay of going from the first VNF (V_1) to the last one (V_l) of the mapped NS (equation (1)). To do so any mapping algorithm must satisfy that the delay between the mapped VNFs $delay_{map}(V_i, V_k)$ is below the requested delay between them $delay_{req}(V_i, V_k)$ (equation (2)). As well, every link l must have more bandwidth $bw(l)$ than the sum of the bandwidth usage of the VNFs (V_A, V_B) that it connects (term $u_{l,(V_A,V_B)} \cdot bw(V_A, V_B)$ in equation (3)), and every server s must have more computational resources $comput_s$ than the sum of the computational resources used by the VNFs V it hosts (term $u_{l,V} \cdot comput_V$ in equation (4)).

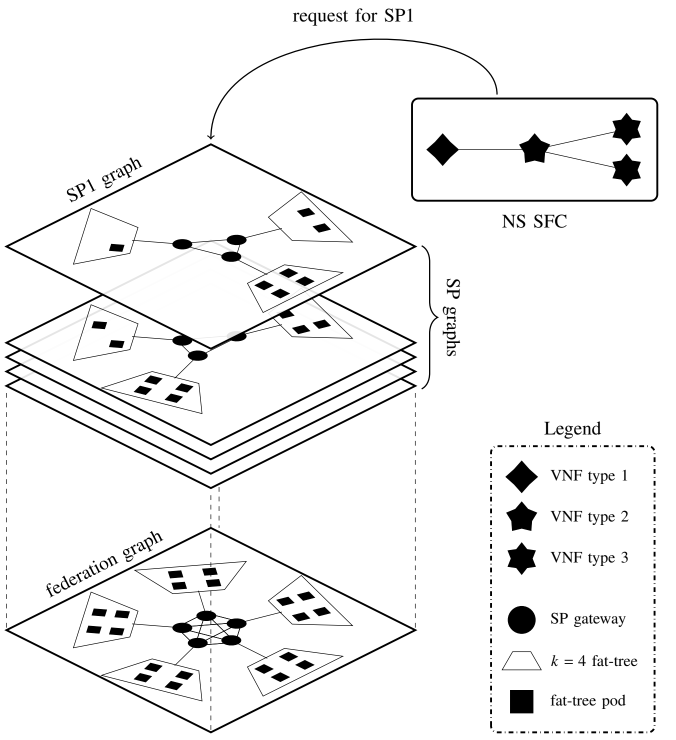

Fig. 1. Illustration of a federated multi-domain infrastructure of 5 SPs and their respective graphs. A NS mapping request arrives to SP1.

$$minimize \sum_{NS \in requests} delay_{map}(V_1, V_l) \tag{1}$$

$$s.t. \quad delay_{map}(V_i, V_k) \leq delay_{req}(V_i, V_k) \tag{2}$$

$$\sum_{NS} \sum_{(V_A,V_B) \in NS} u_{l,(V_A,V_B)} \cdot bw(V_A, V_B) \leq bw(l), \forall l \tag{3}$$

$$\sum_{NS} \sum_{V \in NS} u_{s,V} \cdot comput_V \leq comput_s, \quad \forall s \tag{4}$$

III. Experiment

We test several greedy VNF mapping algorithms in top of our simulator. They rely on Dijkstra, a meta-heuristic called tabu search, and modified versions of Breadth First Search (BFS) and Depth First Search (DFS) [7] that avoid redundant

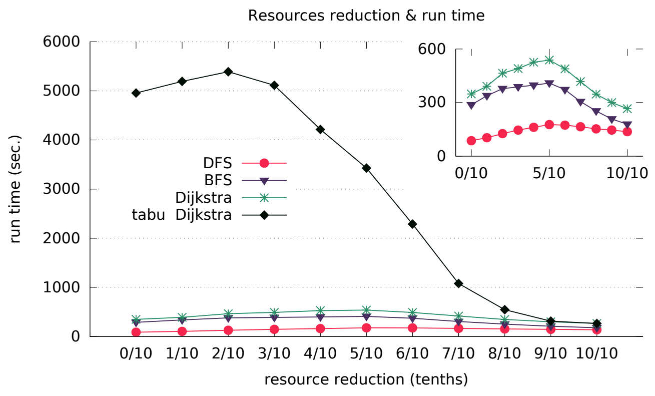

Fig. 2. Running time of 400 NS mapping requests using different algorithms as server resources are reduced.

TABLE I
ACCEPTANCE RATIOS AS RESOURCES ARE REDUCED

	0/10	**2/10**	**4/10**	**6/10**	**8/10**	**10/10**
DFS	100%	90.75%	61.25%	27.25%	3%	0%
BFS	100%	89%	59.75%	27%	3%	0%
Dijkstra	100%	89.75%	60.5%	27.75%	2.75%	0%
tabu	100%	89.75%	61.75%	28.25%	3.25%	0%

operations thanks to our “cutoffs” using infrastructure knowledge.

In the experiment we have 20 SPs, each one with a $k = 4$ fat-tree in its own domain. All the SP’s gateways are connected in a full mesh, and the federation states that every SP shares up to 4 fat-tree pods with other 9 SPs (each fat-tree pod has 4 servers).

On top of that scenario 400 NS are requested among the 20 SPs. We check how many of them can be allocated (acceptance ratio in TABLE I), and how long does it takes to perform the 400 NS mappings when we reduce the fat-trees resources in tenths (Figure 2).

IV. CONCLUSIONS

With this work our contribution to the state of the art is the extension of the VNF mapping problem to the multi-domain scenario, and our DFS with “cutoffs” greedy algorithm, which can map 400 NS requests in 20 SPs in less than 2 min. and 30 sec. (no matter how much we reduce the available resources). The downside of the greedy algorithms we rely on is that they do not retrieve optimal solutions although they are fast.

ACKNOWLEDGMENT

This work has been partially supported by the EU H2020 5G Exchange (5GEx) innovation project (grant no. 671636) and by EU H2020 5G-Transformer Project (grant no. 761536).

REFERENCES

[1] X. Li and C. Qian, “The virtual network function placement problem,” in *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, April 2015, pp. 69–70.
[2] Z. Allybokus, N. Perrot, J. Leguay, L. Maggi, and E. Gourdin, “Virtual Function Placement for Service Chaining with Partial Orders and Anti-Affinity Rules,” *arXiv preprint arXiv:1705.10554*, 2017.
[3] V. S. Reddy, A. Baumgartner, and T. Bauschert, “Robust embedding of VNF/service chains with delay bounds,” in *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Nov 2016, pp. 93–99.
[4] M. A. Lopez, D. M. F. Mattos, and O. C. M. B. Duarte, “Evaluating allocation heuristics for an efficient virtual Network Function chaining,” in *2016 7th International Conference on the Network of the Future (NOF)*, Nov 2016, pp. 1–5.
[5] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and S. Davy, “Design and evaluation of algorithms for mapping and scheduling of virtual network functions,” in *Network Softwarization (NetSoft), 2015 1st IEEE Conference on*. IEEE, 2015, pp. 1–9.
[6] M. Al-Fares, A. Loukissas, and A. Vahdat, “A Scalable, Commodity Data Center Network Architecture,” *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 63–74, Aug. 2008. [Online]. Available: http://doi.acm.org/10.1145/1402946.1402967
[7] S. S. Skiena, *The Algorithm Design Manual*. Springer, 2012.