Towards Reduced-State Service Chaining with Source Routing

Chrysa Papagianni
Institute for Systems Research
University of Maryland
College Park, Maryland, USA
chrisap@isr.umd.edu

Panagiotis Papadimitriou
Department of Applied Informatics
University of Macedonia, Greece
papadimitriou@uom.edu.gr

John S. Baras
Department of Electrical
and Computer Engineering
University of Maryland
College Park, Maryland, USA
baras@isr.umd.edu

***Abstract*—The widespread adoption of Network Function Virtualization (NFV) poses significant dataplane scalability limitations in datacenters (DC), as switches will be required to maintain a large amount of forwarding state for service chaining. This problem is exacerbated by the small Forwarding Information Base (FIB) of commodity switches, which are typically deployed in NFV infrastructures.**

To mitigate this problem, we present a routing fabric that combines source routing with pathlet switching in order to (i) achieve state reduction and (ii) provide support for longer paths, hence, meeting the increased hop-count requirement of service chains. Coupling the proposed source routing fabric with a service chain embedding method, we achieve significant gains in terms of FIB consumption, request acceptance, and revenue generation.

I. Introduction

Middleboxes have become an indispensable part of the network infrastructure. Despite their widespread adoption, middleboxes entail significant flexibility limitations, mainly due to the fact that they are built of specialized hardware, and hence, they cannot be re-purposed for other processing applications. Furthermore, middleboxes are typically provisioned for peak loads, often leading to resource inefficiency. Network Function Virtualization (NFV) has been introduced to cope with some of these limitations and facilitate the deployment of new functionality into network, providing better support for applications and services [1], [2], [3].

Despite its early adoption, NFV introduces significant scalability limitations in datacenters (DC), which may hinder the large-scale deployment of virtualized network functions (vNFs) and associated cloud services (*e.g.,* NF-as-a-Service). More specifically, switches in DCs will be required to maintain a large amount of forwarding state for service chaining, *i.e.,* steering traffic through the sequence of vNFs that comprise the service chain. This problem is further exacerbated by the increasing virtual machine (VM) consolidation level in DCs, in conjunction with the limited Forwarding Information Base (FIB) size in switches (*i.e.,* a commodity switch can typically store a few thousands flow entries in its TCAM) [4], [5].

A potential solution to this problem is to employ source routing in DCs, *i.e.,* encode the path into the packet header [6], [7], [8], [9]. This enables the insertion of a small set of flow-independent forwarding entries into the switches, which leads to a significant reduction in the forwarding state. However, service chains may span a large number of hops, as not all of their vNFs may fit in the same rack, due to capacity or other (*e.g.,* anti-collocation) constraints. In this case, packet header space will be insufficient for the encapsulation of the entire path. The straightforward solution of using additional headers (*e.g.,* VLAN/MPLS) would incur significant transmission overhead, especially for smaller packets, and eventually would not eliminate the scalability limitation of source routing for long paths. For instance, with 64-port switches (hence, 6 bits required for each hop), VLAN and MPLS headers could merely permit the encoding of labels for five additional hops, which may not suffice for vNFs scattered across several racks.

To retain the benefits of source routing while eliminating its main inefficiency in DCs (*i.e.,* transmission overhead), we consider breaking down source-routed paths into pathlets. Fixed middlepoints (*e.g.,* top-of-the-rack switches) at the DC network topology undertake the task of pathlet switching. In particular, each middlepoint encodes into the packet header the sequence of labels required to traverse the pathlet, as well as the pathlet ID. In contrast to middlepoints, intermediate switches (*e.g.,* aggregation and root switches) store a minimum set of flow-independent forwarding rules. We are certainly not the first to conceive pathlet switching [10]; however, we effectively couple pathlets with source routing aiming to increase significantly the scalability of service chaining and NFV deployments in DCs.

Our main goal in this paper is to quantify the gains of our combined source-routing and pathlet switching scheme on service chaining. To this end, we introduce a service chain mapping method which is tailored to source-routed pathlets, as discussed above. Our evaluation results in a FIB-constrained scenario show that embedding service chains with our proposed routing scheme yields significant Ternary Content Addressable Memory (TCAM) savings in aggregation and root switches. In contrast, a baseline service chain embedder coupled with standard L2/L3 forwarding gradually leads to the depletion of TCAM space and the rejection of service chain requests.

The remainder of the paper is organized as follows. Section II discusses the proposed source routing DC network fabric for reduced-state service chaining. In Section III, we present a

978-3-903176-14-0

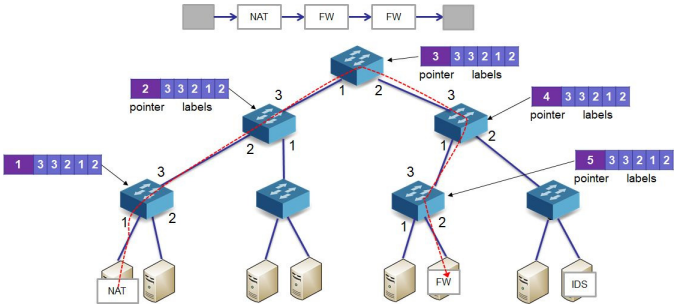

Fig. 1: Source routing example. This example shows the encoded labels and pointer value, as the packet traverses from the NAT to the firewall (FW). The labels match the switch port numbers for simplicity.

service chain embedding method for our source routing fabric. In Section IV, we evaluate the efficiency of the proposed fabric, compared to our baseline. Section V provides an overview of related work. Finally, in Section VI, we highlight our conclusions and discuss directions for future work.

II. Source Routing Fabric

In this section, we elaborate on our proposed source routing DC network fabric for reduced-state service chaining. As already discussed, L2/L3 forwarding (termed as rule-based forwarding) may lead to the depletion of TCAM space in switches, when the DC serves as a NFV infrastructure. This may occur, since with rule-based forwarding the number of switch flow entries is proportional to the end-points (*i.e.,* vNFs). This problem has been observed in the design of DC network fabrics for massive scale (*e.g.,* Portland [11], VL2 [12]). Some of the proposed DC network architectures (*e.g.,* VL2) resort to source routing for better scalability with packet forwarding.

Besides the reduction of FIB size, source routing can reduce the cost of DC networks, since the minimal and nearly-static forwarding tables open up opportunities for simpler and, thus, cheaper switching hardware. Source routing has been further shown to provide higher throughput rates and better exploit the DC path diversity [6]. Network updates can be performed much easier, since any path update can be encoded into the packet header, obviating the need for consistent flow table updates which typically require particular care and the right abstractions [13].

Applying source routing to NFV infrastructures requires attention, as the vNFs of a service chain may be scattered across multiple racks, leading to a much larger hop count, compared to virtual links that simply connect a pair of virtual machines (*e.g.,* SecondNet [7]). Therefore, a straightforward application of source routing would potentially lead to a substantial transmission overhead, due to the use of additional headers (*e.g.,* VLAN/MPLS) and/or stacked labels. To mitigate this, in our source routing fabric, we combine source routing with pathlet switching to effectively retain the benefits of source routing, while avoiding unnecessary transmission overheads.

Source Routing. We first describe the encoding of each pathlet into the packet header. The main idea behind the path encapsulation is to insert a set of labels that correspond to the sequence of switch ports that the packet has to to traverse. In essence, each label is associated with a switch port (without necessarily matching the port number). Hence, the scope of each label is limited to the specific switch. In this case, assuming 64-port switches, each hop requires the use of 6 bits in the packet header. Additional space is needed for a pointer to the next label, which will be updated as the packet is being forwarded along the pathlet. For this purpose, we have allocated an additional 8 bits from the packet header. This port switching scheme, which is inline with other source routing fabrics (*e.g.,* SecondNet [7]), is illustrated in Fig. 1.

Since legacy switches do not support port switching, it is possible to realize such a source routing fabric using OpenFlow switches, which are (incrementally) deployed in modern datacenters. Using the latest OpenFlow specifications that support bit masks, an OpenFlow controller can insert the required flow-independent rules into the switch's flow table, enabling the switch to perform the appropriate matching on incoming packets. In addition, the insertion of a separate flow entry is required to increment the value of the pointer at each switch. In previous work [9], we have implemented and evaluated such a source routing datapath in software, using Click Modular Router [14].

Pathlet Switching. We further discuss our pathlet switching scheme, which alleviates the transmission overhead of source routing over long paths. As already mentioned, paths are decomposed into pathlets. Packet forwarding across each pathlet is performed with port switching, as explained above. Pathlet switching takes place at certain middlepoints; in our source routing fabric, this functionality is delegated into the top-of-the-rack (ToR) switches. In particular, each middlepoint carries out the two following tasks: (i) performs a table lookup to identify the next pathlet and (ii) encodes the sequence of labels for the next pathlet. For pathlet identification and lookup, we use pathlet IDs, which are encoded into the packet header along with information required for source routing (*i.e.,* labels and next-label pointer). Each ToR switch maintains a set of matching rules with pathlet IDs and associated output ports (only the IDs of the pathlets originating from the switch have to be stored). As opposed to the upper-level switches which maintain minimal flow-independant state, the ToR switches are required to store a larger number of forwarding entries for pathlet switching. There are opportunities for the reduction of this forwarding state, *e.g.,* using pathlet IDs with a local scope that require less space compared to globally unique IDs. Regardless of their scope, pathlet IDs can be assigned and distributed among the ToR switches by a centralized controller, which has the knowledge of the whole DC network topology.

In the following, we briefly discuss the packet header space required to implement this source routing fabric in a modern DC that can serve as a NFV infrastructure. We thereby consider a three-layer fat-tree DC topology with $\frac{k}{2}^2$ k-port

switches, k pods each one with two layers of $\frac{k}{2}$ switches and $\frac{k^2}{4}$ servers, based on the DC network model in [15]. In such topology, each ToR switch is connected to the rest of the switches by means of $\frac{k}{2}^3 \times \left(k-1\right) + \frac{k}{2} \times \left(\frac{k}{2}-1\right)$ pathlets. Assuming that k is set to 40, utilizing the MAC address fields is sufficient for the encapsulation of 9 labels, in conjunction with an 8-bit pointer to the next label, and 21 bits for the pathlet ID. Therefore, our routing scheme does not require additional header space for its implementation.

III. Service Chain Embedding

Agile service deployment of a requested service chain at the provider's NFV infrastructure involves the placement of its constituent vNFs and in-sequence routing through them, as prescribed in the service chain. The optimization challenge for the problem at hand is to efficiently allocate computing and bandwidth resources for the vNF-graph. Service chain embedding bears many similarities to the Virtual Network Embedding (VNE) problem, since service chains can be seen as directional graphs that need to be embedded onto a substrate network (*e.g.,* DC). The VNE problem is NP-hard, and service chain embedding, as a generalization of VNE, is considered NP-hard, as well [16].

As explained in Section II, network paths can be decomposed to pathlets, over which traffic is forwarded based on source routing. This routing scheme should be reflected in the service chain embedding method. Therefore, the problem at hand is to embed the vNF-graph by jointly placing the vNFs and mapping the graph edges onto pathlets.

To this end, the directed graph of the substrate DC network can be abstracted into a directed multi-graph, where the additional edges represent the pathlets between the ToR switches. The capacity of each pathlet (edge in the corresponding multi-graph) is constrained by the most utilized link in the path. This topology abstraction is illustrated in Figs. 2 and 3. In particular, Fig. 2 depicts a subset of a 3-layer DC network topology. For the sake of simplicity, the substrate graph is undirected and the corresponding capacities are omitted. For each pathlet between the first ToR switch and every other ToR switch (color-coded in Fig. 2), the corresponding multi-graph edge is added to Fig. 3, illustrated with a dashed line of the same colour. By abstracting the DC graph to a multi-graph, we can easily accommodate the pathlet selection into the service chain mapping formulation (which is omitted, due to space limitations).

Adapting existing VNE methods [17], we further augment the directed multi-graph, by adding the requested vNFs as pseudo-nodes, while each pseudo node is connected to every substrate host with infinite capacity (pseudo-edge). In this augmented directed multi-graph, the bandwidth demand of traffic flows between two consecutive vNFs in the chain (origin-destination pair) is considered a commodity. Hence, the problem is transformed to a Mixed Integer Programming (MIP) multi-commodity flow problem, taking into account substrate node and link capacity constraints (CPU and bandwidth), as well as the FIB capacity for the switches traversed. The

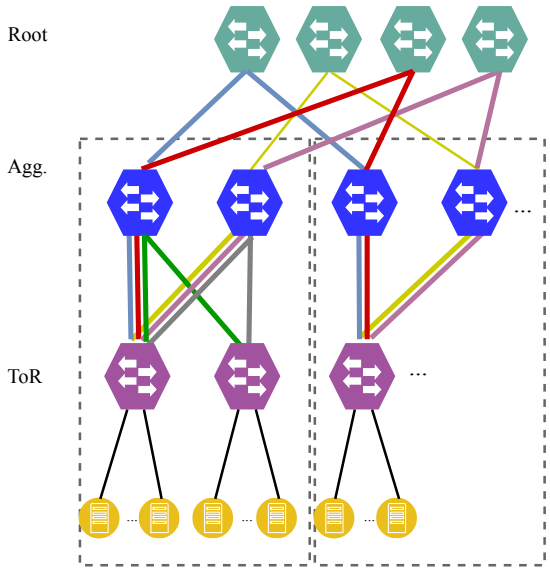

Fig. 2: DC network graph (partial).

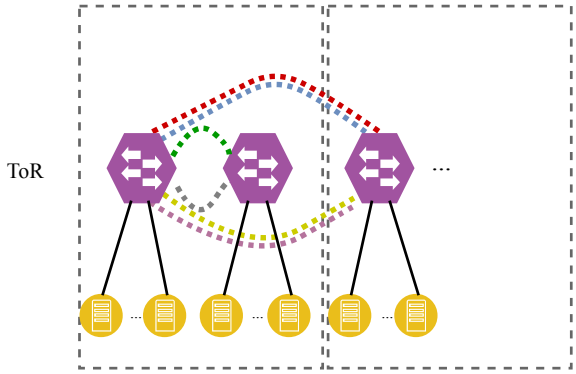

Fig. 3: DC network multi-graph

number of commodities equals the corresponding requested forwarding paths in the vNF-graph. First-hop nodes in the solution (set of paths) denote the servers onto which the requested vNFs will be placed.

IV. Evaluation

In the following, we discuss the evaluation environment (Section IV-A) and the evaluation results (Section IV-B) for service chain embedding with our source routing fabric.

A. Evaluation Environment

We have implemented an evaluation environment for service mapping, including a service chain generator and a DC topology generator, using a discrete event simulator implemented in Java. We use CPLEX [18] for our mixed integer linear programming (MILP) models based on the branch-and-cut method. Our tests are carried out on a server with an Intel i7 CPU at 3.5 GHz and 6 GB of main memory. The proposed embedding method, denoted as *Pathlets*, is compared against a *Baseline* embedder, which assigns service chains onto a DC

with rule-based forwarding (*i.e.,* per destination flow entries), taking into account FIB capacity constraints.

NFV Infrastructure. We have generated a 3-layer fat tree network topology for the DC, consisting of 16 pods. In our evaluations, we use only one portion (or zone) of the DC consisted of 4 (out of 16) pods, utilizing 2 switches per layer and the corresponding set of 2 servers per ToR switch. In order to demonstrate the efficiency of our source routing fabric on state reduction, we use a FIB-constrained simulation setup, at which the root and aggregation switches are equipped with a small FIB (*i.e.,* 100 forwarding entries).

Service Chains. We have implemented an online embedding system, at which each service chain request is embedded as soon as it has arrived. We generate vNF-forwarding graphs based on three service chain templates: (i) the first template corresponds to a chain handling traffic that needs to pass through a particular sequence of vNFs, *i.e.,* a Network Address Translation (NAT) and a Firewall (FW) followed by an Intrusion Detection System (IDS); (ii) the second template reflects the case where traffic in a service chain is split by a particular vNF, according to some predefined policy, *e.g.,* load balancing; (iii) the last template corresponds to chains of virtualized Evolved Packet Core (EPC) elements. For each chain the number of requested VNFs and generated traffic is uniformly distributed, *i.e.,* $U(5,10)$ and $U(50,100)$ Mbps, respectively. The CPU demand of each vNF is derived from the inbound traffic rate and the resource profile of the respective vNF (*i.e.,* CPU cycles per packet). Resource profiles are available for a wide range of vNFs [19], [20], while existing profiling techniques can be employed for processing workloads whose computational requirements are not known in advance [20].

We use the following metrics for the evaluation of the two service chain mapping methods:

- **Request Acceptance Ratio** is the ratio of successfully embedded requests over the total number of requests.
- **Revenue** is the amount of CPU and bandwidth units specified in the SFC request. In particular, we present the cumulative revenue of the successfully embedded SFC requests.

B. Evaluation Results

We executed ten simulation runs, at which 300 non-expiring service function chain (SFC) requests arrive at a DC with FIB-constrained aggregation and root switches, according to a Poisson process at a rate of 4 requests per 100 time units. For our evaluation purposes, we inhibit the co-location of vNFs of the same service chain.

Fig. 4 illustrates the acceptance ratio of the incoming requests. The proposed service mapping method admits all requests, whereas the baseline leads to a significant amount of request rejections. According to Fig. 5, the high request acceptance rate generates more revenue for source routing,

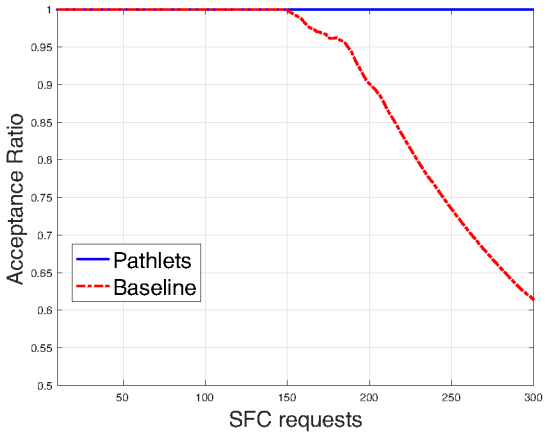

Fig. 4: Request acceptance ratio.

especially after the 150^{th} request, at which the baseline acceptance ratio starts to decay.

Looking closely into the TCAM utilization of the FIB-constrained aggregation and root switches, *i.e.,* Fig. 6 and Fig. 7, respectively, we corroborate the above observations. Following the 150^{th} request, the TCAM utilization of the aggregation switches rises over 80% for the baseline, eventually leading to the depletion of TCAM space. This is the reason for the low acceptance ratio in Fig. 4. The TCAM utilization of the root switches for the baseline is lower than the aggregation switches, albeit still substantial (higher than 60% after the first 200 requests). In contrast, embedding service chains with our approach yields minimal TCAM utilization, which is translated to a higher acceptance ratio, opening up opportunities for larger revenues.

Fig. 8 depicts the increase in the total number of switch forwarding entries across the whole DC. In particular, our approach yields a FIB reduction of approximately 20% on average, prior to the 150^{th} request, after which the baseline experiences requests rejections. After that point, the source-routing based embedding admits requests at much higher rate, which essentially outweighs the FIB savings compared to the baseline. The smaller degree of FIB increase (Fig. 8) compared to the huge margin in terms of TCAM utilization in aggregation (Figs. 6) and root switches (Fig. 7) stems from the much larger number of ToR switches in the DC network, compared to the two upper layers. The TCAM utilization of ToR switches is at the same level for both the proposed method and the baseline, since our source routing fabric requires forwarding state at the ToR switches for pathlet switching and path encapsulation.

Our evaluation results show that source routing in conjunction with pathlet switching can improve the scalability of NFV deployments, alleviating the switch FIB capacity limitations. Based on the TCAM utilizations (Figs. 6, 7), there is also an opportunity for the deployment of cheaper switches at the aggregation and root level of the DC network topology.

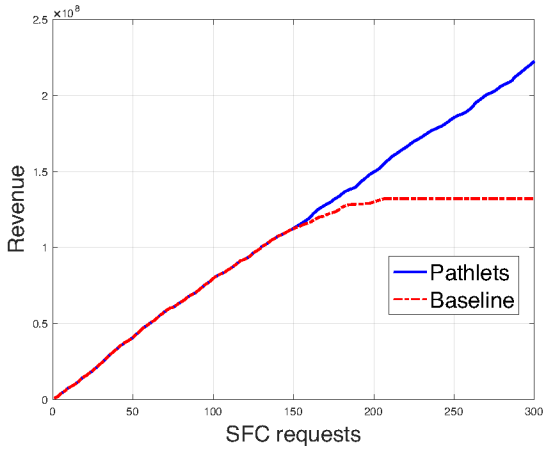

Fig. 5: Cumulative revenue.

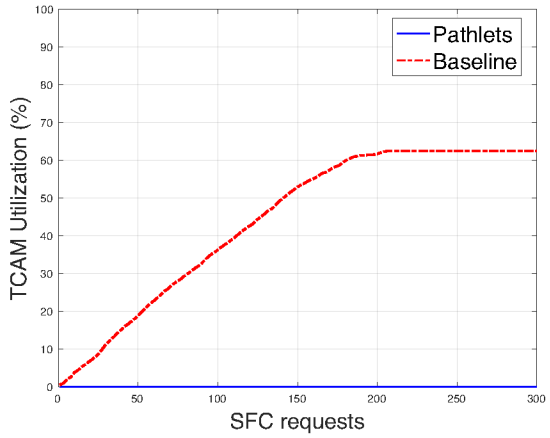

Fig. 7: TCAM utilization in root switches.

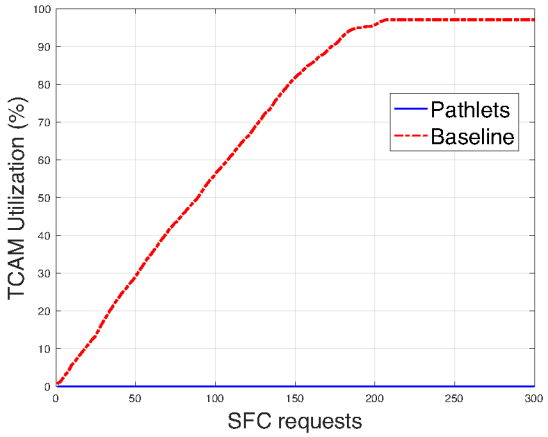

Fig. 6: TCAM utilization in aggregation switches.

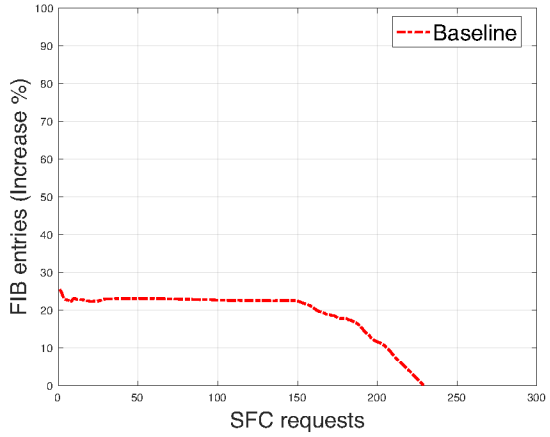

Fig. 8: FIB entries (% increase).

V. Related Work

Hereby, we briefly discuss related work on source routing. Although source routing has not found traction on wide-area networks, mainly due to security issues (*i.e.,* a potential interception of a packet could reveal the flow's path to an adversary), lately source routing has been considered as a viable solution for better dataplane scalability, increased flexibility, and improved resilience in enterprise and cloud datacenters [6], [7], [8], [9].

In particular, *Jyothi et al.* [6] introduce a source-routed fabric for DC networks towards improved flexibility and performance. This work discusses a range of possible implementations for source routing and demonstrates the throughput gains of the proposed source-routed fabric compared to traditional forwarding in leaf-spine DC topologies. SecondNet [7] presents an architecture for DC network virtualization, relying on source routing. Although source routing is implemented using port switching (*i.e.,* similar to our implementation), the path is encoded into the packet at the hypervisor, instead of the ToR switch (as in our case). Furthermore, SecondNet deals with shorter paths, since they connect pairs of virtual machines, instead of a sequence of vNFs in the case of service chaining. Hence, SecondNet does not employ path switching and simply relies on port switching using MPLS.

In previous work of some of the authors [9], a source routing datapath implementation was presented to evaluate the magnitude of switch FIB reduction in DCs. The datapath implementation was based on Click Modular Router [14] and was deployed in an experimental network where performance tests were conducted. *Hari et al.* [8] have also proposed a source routing architecture for datacenters, which is in accordance with our port switching scheme and the source routing mechanisms exemplified in [6]. The work in [8] provides an extensive analysis of the path encapsulation technique, as well as techniques for the compression of the path header.

In contrast to all these aforementioned approaches, we have coupled source routing with pathlet switching to eliminate the switch dataplane scalabilities in DCs and facilitate NFV deployments. We have further outlined a service chain embedding method, applicable to source routing DC network fabrics.

Finally, we briefly discuss new paradigms, *i.e.,* segment routing and the Network Service Header (NSH), that facilitate service chaining. Segment routing [21] is a form of source routing, at which traffic is routed through a set of segments,

which, in the context of NFV, can represent the vNFs comprising the service chain. Segment routing is implemented using an extension header, known as the Segment Routing Header (SRH). NSH comprises an alternative approach for service chaining [22]. In particular, NSH contains a sequence of service nodes (*e.g.,* vNFs) that the packet must be routed before reaching the destination, creating a dedicated service plane for network processing. Our source routing fabric can be reconsidered with segment routing or NSH in mind. We plan to pursue this in future work and perform a comparison among the alternative implementations.

VI. Conclusions and Future Work

In this paper, we tackled the challenging problem of state reduction in DCs, given the increasing interest from most stakeholders in service deployment and chaining. Our approach couples source routing with pathlet switching, since the increased hop-count of service chains tends to outweigh the benefits of source routing, when the latter is used on its own. In this respect, we exemplified our source routing fabric and further outlined a method for the embedding of service chains onto the proposed fabric, with the aim to maximize the acceptance rate of requested service chains and the amount of revenue generated for a NFV provider.

According to our evaluation results, our proposed approach yields significant FIB savings at the aggregation and root switch levels, at which port switching is exercised. In contrast, ToR switch FIB savings are negligible, since in our fabric they serve as path middlepoints, and thereby, have to maintain state for pathlet switching and label encoding. In our FIB-constrained evaluation scenario, these FIB savings have a significant impact on request acceptance and revenue.

Future work will be focused on the experimental evaluation of our source routing fabric, investigating the efficiency of alternative port/pathlet switching implementation techniques, as well as the placement of certain functionality (*e.g.,* pathlet switching elements) at different parts of the network.

References

[1] D. Dietrich, A. Abujoda, A. Rizk, and P. Papadimitriou, "Multi-provider service chain embedding with nestor," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 91–105, March 2017.

[2] M. Kourtis, M. J. McGrath, G. Gardikis, G. Xilouris, V. Riccobene, P. Papadimitriou, E. Trouva, F. Liberati, M. Trubian, J. Batall, H. Koumaras, D. Dietrich, A. Ramos, J. F. Riera, J. Bonnet, A. Pietrabissa, A. Ceselli, and A. Petrini, "T-nova: An open-source mano stack for nfv infrastructures," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 586–602, Sept 2017.

[3] S. Palkar, C. Lan, S. Han, K. Jang, A. Panda, S. Ratnasamy, L. Rizzo, and S. Shenker, "E2: A framework for nfv applications," in *Proceedings of the 25th Symposium on Operating Systems Principles*, ser. SOSP '15. New York, NY, USA: ACM, 2015, pp. 121–136. [Online]. Available: http://doi.acm.org/10.1145/2815400.2815423

[4] Z. Bozakov and P. Papadimitriou, "Towards a scalable software-defined network virtualization platform," in *2014 IEEE Network Operations and Management Symposium (NOMS)*, May 2014, pp. 1–8.

[5] N. Sarrar, S. Uhlig, A. Feldmann, R. Sherwood, and X. Huang, "Leveraging zipf's law for traffic offloading," *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 1, pp. 16–22, Jan. 2012. [Online]. Available: http://doi.acm.org/10.1145/2096149.2096152

[6] S. Jyothi, M. Dong, and P. B. Godfrey, "Towards a flexible data center fabric with source routing," in *Proceedings of the 1st ACM SIGCOMM Symposium on SDN Research*, ser. SOSR '15, 2015, pp. 10:1–10:8.

[7] C. Guo, G. Lu, H. J. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang, "Secondnet: A data center network virtualization architecture with bandwidth guarantees," in *Proceedings of the 6th International COnference*, ser. Co-NEXT '10. New York, NY, USA: ACM, 2010, pp. 15:1–15:12. [Online]. Available: http://doi.acm.org/10.1145/1921168.1921188

[8] A. Hari, T. V. Lakshman, and G. Wilfong, "Path switching: Reduced-state flow handling in sdn using path information," in *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '15, 2015, pp. 36:1–36:7.

[9] A. Abujoda, H. R. Kouchaksaraei, and P. Papadimitriou, "Sdn-based source routing for scalable service chaining in datacenters," in *Wired/Wireless Internet Communications*, L. Mamatas, I. Matta, P. Papadimitriou, and Y. Koucheryavy, Eds. Cham: Springer International Publishing, 2016, pp. 66–77.

[10] P. B. Godfrey, I. Ganichev, S. Shenker, and I. Stoica, "Pathlet routing," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 111–122, Aug. 2009. [Online]. Available: http://doi.acm.org/10.1145/1594977.1592583

[11] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, "Portland: A scalable fault-tolerant layer 2 data center network fabric," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 39–50, Aug. 2009. [Online]. Available: http://doi.acm.org/10.1145/1594977.1592575

[12] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. Maltz, P. Patel, and S. Sengupta, "Vl2: A scalable and flexible data center network." Association for Computing Machinery, Inc., August 2009. [Online]. Available: https://www.microsoft.com/en-us/research/publication/vl2-a-scalable-and-flexible-data-center-network/

[13] M. Reitblatt, N. Foster, J. Rexford, C. Schlesinger, and D. Walker, "Abstractions for network update," *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 323–334, Aug. 2012. [Online]. Available: http://doi.acm.org/10.1145/2377677.2377748

[14] R. Morris, E. Kohler, J. Jannotti, and M. F. Kaashoek, "The click modular router," *SIGOPS Oper. Syst. Rev.*, vol. 33, no. 5, pp. 217–231, Dec. 1999. [Online]. Available: http://doi.acm.org/10.1145/319344.319166

[15] A. Vahdat, M. Al-Fares, N. Farrington, R. N. Mysore, G. Porter, and S. Radhakrishnan, "Scale-out networking in the data center," *IEEE Micro*, vol. 30, no. 4, pp. 29–41, July 2010.

[16] J. G. Herrera and J. F. Botero, "Resource allocation in nfv: A comprehensive survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, Sept 2016.

[17] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Vineyard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Transactions on Networking*, vol. 20, no. 1, pp. 206–219, Feb 2012.

[18] "IBM ILOG CPLEX Optimizer," urlhttp://www-01.ibm.com/software/integration/optimization/cplex-optimizer/, Last 2010.

[19] M. Dobrescu, K. Argyraki, and S. Ratnasamy, "Toward predictable performance in software packet-processing platforms," in *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. San Jose, CA: USENIX, 2012, pp. 141–154. [Online]. Available: https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/dobrescu

[20] A. Abujoda and P. Papadimitriou, "Profiling packet processing workloads on commodity servers," in *Wired/Wireless Internet Communication*, V. Tsaoussidis, A. J. Kassler, Y. Koucheryavy, and A. Mellouk, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 216–228.

[21] A. Cianfrani, M. Listanti, and M. Polverini, "Incremental deployment of segment routing into an isp network: a traffic engineering perspective," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 3146–3160, Oct 2017.

[22] P. Quinn, U. Elzur, and C. Pignataro, "Network Service Header (NSH)," RFC 8300, Jan. 2018. [Online]. Available: https://rfc-editor.org/rfc/rfc8300.txt