Offloading NFV Orchestration to ToR Switches: How to Leverage PDP to Realize Agile Service Function Chaining in HOE-DCNs

Jianquan Peng, Qinhezi Li, Shaofei Tang, Hongqiang Fang, and Zuqing Zhu†
School of Information Science and Technology, University of Science and Technology of China, Hefei, China
†Email: {zqzhu}@ieee.org

***Abstract*—The advances on hybrid optical/electrical DCNs (HOE-DCNs) make it easier to support network function virtualization (NFV) with various quality-of-service (QoS) demands. However, the existing approaches for NFV orchestration rely heavily on the control plane, which might lead to scalability and performance issues. In this work, we propose to offload certain network orchestration tasks from the control plane to top-of-rack (ToR) switches, by leveraging the flexibility of programmable data plane (PDP). Specifically, we architect an HOE-DCN whose ToR switches are PDP switches, design the network orchestration system for it, and program the ToR switches, such that they can make quick decisions locally to assist the control plane to provision the service chains of virtual network functions (vNF-SCs) agilely. We prototype our proposal in a small-scale HOE-DCN testbed, and the experimental results show that without much involvement of the control plane, the ToR switches can make local decisions and readjust the provisioning schemes of vNF-SCs adaptively to satisfy their QoS demands.**

***Index Terms*—Hybrid optical/electrical datacenter networks (HOE-DCNs), Programmable data plane (PDP), Network function virtualization (NFV), Service function chaining, P4, Stateful routing, Network orchestration.**

I. Introduction

Nowadays, the rising of cloud computing and Big Data analytics has promoted rapid development of datacenters (DCs) globally [1, 2], and in the meantime, both the volume of traffic and the variety of network services in DCs have been increasing dramatically. Therefore, DC networks (DCNs) are under great pressure to address the challenges from architecture scalability, energy efficiency, and management agility [3]. The pressure can be potentially relieved by considering hybrid optical/electrical DCNs (HOE-DCNs) [4]. Specifically, in addition to the electrical packet switching (EPS) based inter-rack network, an HOE-DCN introduces optical circuit switching (OCS) to inter-connect top-of-rack (ToR) switches through one or more optical cross-connects (OXCs). Both EPS and OCS have their own pros and cons, and thus they are actually complementary in a few aspects. For instance, EPS can provide flexible traffic forwarding but it has relatively limited throughput and can be power-hungry to achieve high capacities, while OCS can deliver almost unlimited capacity and is much more energy-efficient but it takes longer time to reconfigure [5–7]. Hence, an HOE-DCN can integrate the benefits of EPS and OCS seamlessly, especially for addressing the challenges from architecture scalability and energy efficiency.

Although HOE-DCNs are promising and attracting intensive interests from both academia and industry, to ensure management agility in them is still challenging. This is because an intelligent and effective network orchestration scheme needs to be designed to coordinate the allocations of IT and bandwidth resources timely, for satisfying various quality-of-service (QoS) demands [8–11]. The architecture of HOE-DCNs actually makes the scheme even more difficult to design, because the additional OCS-based inter-rack network complicates the network orchestration. Finally, considering the wide deployment of network function virtualization (NFV) in DCNs, the network orchestration scheme should have special support for it. Note that, NFV realizes network services by instantiating virtual network functions (vNFs) on general-purpose hardware/software platforms [12]. Specifically, with NFV, a service provider (SP) decomposes each network service into a few atomic network functions, instantiates them in a DCN as vNFs, and then steer application traffic through the vNFs in sequence (*i.e.*, realizing the network service with a vNF service chain (vNF-SC) [13, 14]).

Previously, by leveraging software-defined networking (SDN) [15], we proposed and implemented intelligent network orchestration schemes to optimize the resource allocations in HOE-DCNs, and demonstrated that they can effectively reduce the task completion time of network services [16, 17]. However, these schemes proposed still have scalability issues. This is because they only relied on the control plane to manage the placement of vNFs in server racks and the routing of application traffic in EPS/OCS-based inter-rack networks. Considering the huge volume of network services in a typical HOE-DCN, we can hardly imagine that getting the control plane involved in each network readjustment would be a scalable solution. Moreover, for such a control-plane-driven approach, the timeliness of its network readjustments relies heavily on the polling interval that the control plane uses to retrieve data plane status. Nevertheless, using a polling interval that is too short will result in flooding redundant status data to the control plane, while making the interval too long will degrade the timeliness of network readjustments.

To address the aforementioned dilemmas, we, in this work, propose to offload certain network orchestration tasks from the control plane to ToR switches in the data plane, and leverage the flexibility provided by programmable data plane (PDP)

[18] to realize agile provisioning of vNF-SCs in an HOE-DCN. With the P4 language [18], we can program the traffic forwarding logic in a PDP switch, *i.e.*, to define packet fields and to customize packet processing pipelines. Therefore, the packet processing in a PDP switch is much more flexible than that in a conventional SDN switch, and stateful routing [19] can be realized to enable the PDP switch to make local decisions for adapting to dynamic network environment.

Inspired by this mechanism, we architect an HOE-DCN whose ToR switches are PDP switches, design the network orchestration system for it, and program the ToR switches such that they can make quick decisions locally to assist the control plane to realize agile vNF-SC provisioning. We prototype our proposal in a small-scale experimental testbed for HOE-DCN, and conduct experiments in it with the vNF-SCs for Hadoop applications [20]. Our experimental results show that without much involvement of the control plane, the ToR switches can operate in an event-driven way to make local decisions, and readjust the provisioning schemes of vNF-SCs adaptively to satisfy their QoS demands. Moreover, as the readjustment decisions are made locally, the provisioning schemes of vNF-SCs are updated to suitable ones almost instantly when they need to be readjusted, which is much more timely and efficient than the control-plane-driven approaches in [16, 17].

The rest of this paper is organized as follows. Section II reviews the related work. In Section III, we present the design of our system, including its architecture and our detailed implementation. The experimental demonstrations are shown in IV. Finally, Section V summarizes the paper.

II. Related Work

The architecture of HOE-DCN was developed in [4] to address the bandwidth crunch and ever-growing power consumptions in DCNs, and it has the backward compatibility to ensure that the transition from a traditional DCN to it will be smooth. The study in [4] also considered how to configure the EPS/OCS-based inter-rack networks in an HOE-DCN to satisfy the QoS demands of network services. However, it did not address the allocation of IT resources or the network readjustment to adapt to dynamic network environment. Note that, due to the dynamic nature of most network services in DCNs, an effective network orchestration scheme for provisioning vNF-SCs in an HOE-DCN has to: 1) coordinate the IT and bandwidth resource allocations jointly, and 2) readjust vNF-SC provisioning schemes from time to time.

The basic problem of the service provisioning for vNF-SCs can be understood as a variant of the famous virtual network embedding (VNE) problem [21, 22], *i.e.*, mapping chain-type virtual networks to a substrate network while multiple virtual nodes can be embedded onto one substrate node. One can refer to [23] for a comprehensive survey on the algorithms for vNF-SC provisioning. The studies in [24, 25] designed approaches for readjusting the provisioning schemes of vNF-SCs to address time-varying network environment. However, most of the existing studies in this area assumed that the network orchestration is conducted purely by a centralized (or a logically-centralized) control plane, while data plane elements (*e.g.*, switches and servers) cannot make their own decisions to readjust the provisioning schemes of vNF-SCs.

The idea of SDN is to manage data plane elements with a separate control plane [26]. Despite the success of SDN in various networks [26, 27], people also realized that removing intelligence completely from the data plane might not always be good [19]. This is because data plane is the first place to detect network status changes, and thus if it can be programmed to respond to some simple events properly, we can realize network readjustments timely and reduce a large amount of communications between data and control planes. This can be achieved by leveraging the stateful routing [19] supported by PDP. Note that, PDP can be realized with either P4 [18] or protocol-oblivious forwarding (POF) [28], but considering the superior packet processing performance of P4-based hardware PDP switches [29], we develop our system based on them. Previously, Paolucci *et al.* [30] has demonstrated multilayer traffic engineering with P4-based stateful routing. However, they only used stateful routing to address bandwidth crunch, but did not consider an HOE-DCN as the network environment or try to leverage stateful routing to coordinate IT and bandwidth resource allocations jointly for vNF-SC provisioning.

III. System Design

In this section, we first present the architecture of our system to realize agile vNF-SC provisioning with PDP switches, and then explain its functional modules and operation principle.

A. System Architecture

Fig. 1 shows the overall architecture of our proposed system. The data plane is essentially an HOE-DCN, where each rack consists of a few servers on which vNFs can be deployed. The ToR switch of each rack is a PDP switch that can be programmed with the P4 language to realize stateful packet processing pipelines. Specifically, stateful packet processing (or stateful routing) means that in the switch, the procedure for processing packet is not deterministic as in a conventional SDN switch, and it can be switched among a few schemes according to the network status [19]. For instance, the PDP switch can determine the output port of a flow based on its data-rate, *i.e.*, if the data-rate exceeds a preset threshold, the flow will be offloaded to an output port that is different from its commonly-used one. Hence, with stateful routing, the switch can make local decisions to address certain simple events.

The ToR switches are inter-connected by the EPS/OCS-based inter-rack networks. The EPS-based inter-rack network has a hierarchical topology that consists of ToR, aggregation and core switches, while the OCS-based one just interconnects the ToR switches with an OXC. Therefore, a pair of ToR switches can achieve a higher communication throughput if they talk through the OCS-based inter-rack network. The servers in the rack behind each ToR switch provide the IT resources (*e.g.*, CPU, memory and storage) to run virtual machines (VMs). As shown in Fig. 1, the VMs can be used independently by network services or be treated as the vNFs to

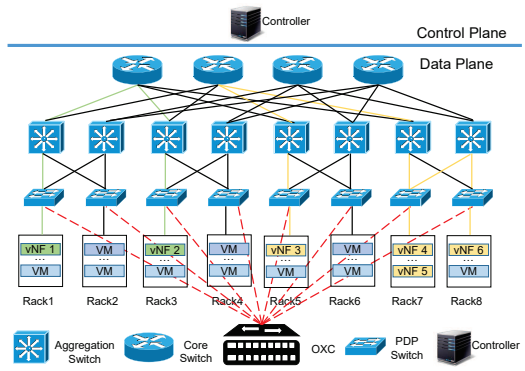

Fig. 1. Overall system architecture.

form vNF-SCs. For each vNF-SC, the traffic between any two adjacent vNFs in it is forwarded by at least one ToR switch.

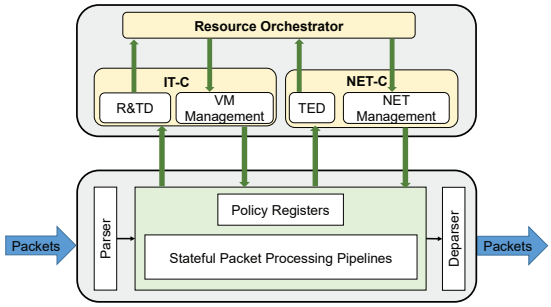

Fig. 2. Functional modules to realize agile vNF-SC provisioning.

B. Functional Modules

The function modules involved in the network orchestration for agile vNF-SC provisioning are illustrated in Fig. 2. The network orchestrator in the control plane mainly includes three modules, which are the resource orchestrator, the IT controller (IT-C), and the network controller (NET-C). Here, the IT-C determines how to deploy/migrate VMs in the servers, while the NET-C manages the switches in the EPS/OCS-based inter-rack networks to steer traffic through the VMs. Meanwhile, as the IT-C and NET-C manage different parts of the HOE-DCN, we place the resource orchestrator on top of them for coordination. For instance, when provisioning a vNF-SC initially, we first use the resource orchestrator to calculate the placement of vNFs and the routing paths among the vNFs, and then let it instruct the IT-C and NET-C to realize the vNF-SC deployment in the HOE-DCN. Note that, in addition to their common tasks on managing the data plane elements, we design the IT-C and NET-C to be able to update the registers in ToR switches, which store the policies for stateful routing.

In our system, the PDP-based ToR switches can sense network status changes and respond locally to certain simple events with stateful routing. This is realized by programming the ToR switches to incorporate stateful packet processing pipelines with the P4 language. Hence, the provisioning scheme of a vNF-SC can be readjusted in realtime to adapt to dynamic network environment, without much involvement of the control plane. Meanwhile, as the ToR switches cannot be shut down frequently for reprogramming, we program them in the way that some key parameters of stateful routing (*e.g.*, preset thresholds) are stored in their registers and can be updated in runtime by the IT-C and NET-C. Once a ToR switch made a local decision to readjust the provisioning scheme of a vNF-SC and has implemented the readjustment, it reports the network status change to the control plane, to ensure that the vNF-SC provisioning schemes recorded in the IT resource and traffic database (R&TD) and traffic engineering database (TED) are always up-to-date. The cooperation procedure of the control plane and ToR switches is shown in Fig. 3.

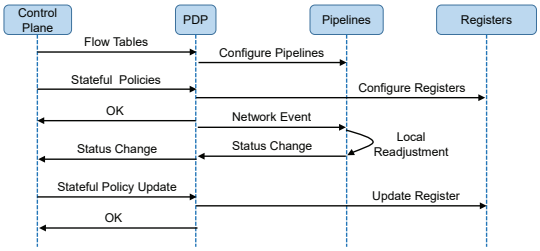

Fig. 3. Cooperation procedure of control plane and ToR switches.

C. Operation Principle

In this work, we specifically consider three scenarios of stateful routing and implement them in our experimental testbed to demonstrate their effects on agile vNF-SC provisioning. The first scenario is *Optical Offloading*, *i.e.*, the ToR switches leverage stateful routing to detect elephant flows from vNFs and instantly reroute them to use the OCS-based inter-rack network. The second scenario is *Optical Bypass*, which means that for each vNF-SC that has stringent QoS demand on end-to-end latency, the ToR switches utilize stateful routing to find the situation where the accumulated latency until the current vNF is close to the latency constraint, and reroute its packets to use the OCS-based inter-rack network, for minimizing future latency. The third scenario is *Server Reselection*, *i.e.*, the ToR switches leverage stateful routing to detect the cases in which one vNF in a vNF-SC runs on a server that is about to be overloaded, and reroute the packets of the vNF-SC to a same vNF on another server to avoid the "hot-spot" server.

The detection of elephant flows in the first scenario can be easily realized by implementing a bandwidth meter in each PDP-based ToR switch. The bandwidth meter measures the data-rate of each flow that is forwarded by the ToR switch. When the measurement shows that the data-rate of a flow, which is currently routed over the EPS-based inter-rack network, is larger than a preset threshold, the stateful routing in the ToR switch will offload the flow to use the OCS-based inter-rack network. Similarly, when the measurement indicates that a flow forwarding by the OCS-based inter-rack network has a data-rate lower than the threshold, the ToR switch will change its output port to use the EPS-based inter-rack network. Here, the threshold on data-rate is stored in a policy register and can be updated in runtime by the control plane.

For the second scenario, the essential part is to measure the accumulated latency until each vNF in a vNF-SC, which includes three types of latencies, *i.e.*, the processing latency in

switches and vNFs, the transmission delay in network interface cards, and the propagation delay on network links. Note that, in an HOE-DCN, the transmission and propagation delays are actually much shorter than the processing latency, and thus we ignore them in our system. The packets of a vNF-SC will pass through a ToR switch twice, if they need to be processed by a vNF that is deployed in the server rack behind the ToR switch. Hence, the total processing latency of the vNF (including that in the ToR switch) can be obtained by time-stamping each packet when it first enters the switch and finally leaves it, and subtracting the first time-stamp from the second one.

Then, we program the ToR switches with *Algorithm* 1 to realize the *Optical Bypass* in the second scenario. Here, we leverage the option fields in an IP header to record two latency parameters, which are the initial ingress time of a packet into a ToR switch (t_1) and the accumulated latency (D), and they are updated each time when the packet passes through a vNF in its vNF-SC. Meanwhile, the threshold on accumulated latency (D_{th}) is given by the control plane to store in a policy register.

Algorithm 1: Procedure for Optical Bypass

```
1  while the ToR switch is operational do
2      if an incoming packet P is for concerned vNF-SC then
3          if this is first time to see P then
4              record current time t1 in the IP header of P;
5              forward P to the vNF in ToR switch's rack;
6          else
7              extract initial ingress time t1 and accumulated
               latency D from the IP header of P;
8              record current time in t2;
9              D = D + t2 − t1;
10             record D in the IP header of P;
11             get latency threshold Dth from policy register;
12             if D > Dth then
13                 forward P to its next vNF through
                   OCS-based inter-rack network;
14             else
15                 forward P to its next vNF through
                   EPS-based inter-rack network;
16             end
17         end
18     end
19 end
```

Finally, the third scenario also utilizes the processing latency in each vNF to estimate whether a vNF runs on a server that is about to be overloaded. Specifically, if the server that carries the vNF is close to be overloaded, there will be IT resource contentions on among the vNFs running on it and thus the processing latency of the vNF will be prolonged [16, 17]. Therefore, we design *Algorithm* 2 to accomplish the *Server Reselection* in the third scenario. Here, the processing latency of a vNF is measured similarly as in *Algorithm* 1. However, as the latency is only for the current vNF, it is removed from a packet, when the packet passes through a ToR switch the second time (*Line* 9). After detecting an excessively long processing latency, the stateful routing in *Algorithm* 2 will change the forwarding rule of the subsequent packets to the vNF to go to the backup vNF on another server in the same rack (*Line* 11-13). Similar to the first two scenarios, we store both the threshold on processing latency (T_{th}) and the output port to the backup vNF in policy registers. The stateful routing in the three scenarios can be realized with P4 programs.

Algorithm 2: Procedure for Server Reselection

```
1  while the ToR switch is operational do
2      if an incoming packet P is for concerned vNF-SC then
3          if this is first time to see P then
4              record current time t1 in the IP header of P;
5              forward P to the vNF in ToR switch's rack;
6          else
7              extract initial ingress time t1 from the IP
               header of P, and record current time in t2;
8              Δt = t2 − t1;
9              remove the option field for t1 from P and
               forward it to next vNF;
10             get latency threshold Tth from policy register;
11             if Δt > Tth then
12                 change forwarding rule of the subsequent
                   packets to the vNF to go to the backup
                   vNF on another server in same rack;
13             end
14         end
15     end
16 end
```

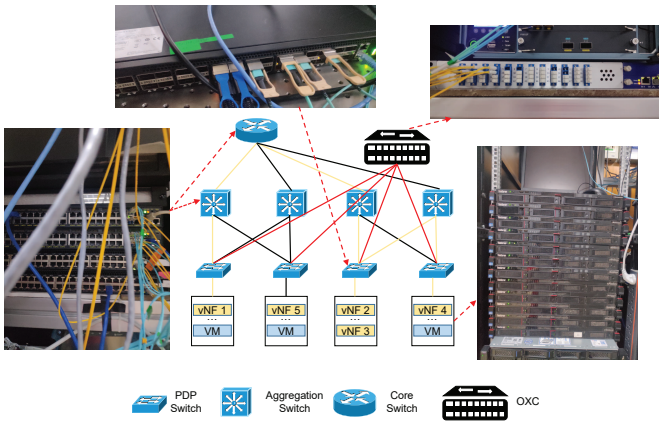

Fig. 4. Experimental setup of our HOE-DCN testbed.

IV. Experimental Demonstrations

A. Experimental Testbed

For the experimental demonstrations, we build a small-scale HOE-DCN testbed that includes four server racks, as shown in Fig. 4. Each server in a rack runs Linux Ubuntu 16.04. The ToR switches are commercial hardware PDP switches [29], each of which has 32 ports and can be programmed with the P4 language. The EPS-based inter-rack network is built with the PDP-based ToR switches and hardware OpenFlow switches that work as aggregation and core switches, and all the network connections in it are based on 1 GbE. The OCS-based inter-rack network consists of a reconfigurable OXC, which interconnects all the ToR switches through 10 GbE optical ports.

We run Hadoop applications in the HOE-DCN testbed to evaluate our proposal. Specifically, each Hadoop application is supported by a cluster that includes one name-node, multiple data-nodes, and a few VMs for intermediate processing. As the

name-node coordinates the tasks on the data-nodes, the data-nodes and the VMs for intermediate processing are the places where the tasks actually get processed. Hence, we can treat the data-nodes and the VMs as vNFs and describe the sequence of the data processing in them as a vNF-SC. For instance, in Fig. 4, *vNF* 5 is the name-node of a Hadoop cluster, *vNFs* 1 and 4 are the data-nodes, and *vNFs* 2 and 3 are the VMs for intermediate processing. Here, *vNFs* 2 and 3 serve as backups of each other for *Server Reselection*. Therefore, the vNF-SC of the Hadoop cluster can be obtained as *vNF* 1→(*vNF* 2/*vNF* 3)→*vNF* 4. As all the vNFs on a vNF-SC need to collaborate to complete tasks together, the performance of each of them can affect the task completion time of the cluster.

B. Functional Verification

We first conduct two simple experiments to verify the functionalities of our proposed system. The first experiment tries to verify *Optical Offloading*. We instantiate a vNF in the HOE-DCN, and let it send time-varying traffic to a vNF in another server rack, using the trace in Fig. 5(a). The control plane instructs the related ToR switch to set the data-rate threshold for elephant flow detection as 300 Mbps. Then, the traffic that gets forwarded in the EPS-/OCS-based inter-rack networks is shown in Fig. 5(b), which indicates that local decisions are made by the ToR switch timely to switch the traffic according to the preset threshold. Hence, without much involvement of the control plane, the ToR switch adjusts it traffic forwarding scheme adaptively to explore the bandwidth in the EPS-/OCS-based inter-rack networks effectively.

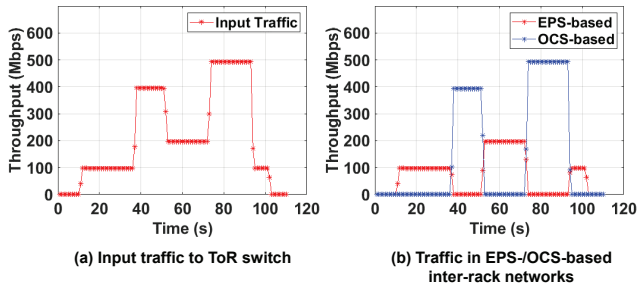

Fig. 5. Experimental results to verify the functionality of *Optical Offloading*.

To further evaluate the performance of *Optical Offloading* achieved by the ToR switch, we use Wireshark to capture the packets forwarded by it, when a local decision is making. The Wireshark capture is illustrated in Fig. 6, which includes 8 packets that are forwarded by the ToR switch during the transition. Here, the interfaces 0 and 1 on the ToR switch connect to the EPS-based and OCS-based inter-rack networks, respectively. In Fig. 6, we can see that after detecting the data-rate of the traffic is above the preset threshold, the ToR switch quickly switches the traffic to use the OCS-based inter-rack network since the fourth packet. Here, the time stamps of the third and fourth packets are 11.577245746 and 11.577313729 seconds, and thus the port switching only takes 68.0 μs. Therefore, the results from the first experiment verify that our proposed system can detect elephant flow correctly and offload them to the OCS-based inter-rack network quickly.

Time	Source	Destination	Interface id
11.577095245	192.168.0.1	192.168.1.1	0
11.577169766	192.168.0.1	192.168.1.1	0
11.577245746	192.168.0.1	192.168.1.1	0
11.577313729	192.168.0.1	192.168.1.1	1
11.577376629	192.168.0.1	192.168.1.1	1
11.577444214	192.168.0.1	192.168.1.1	1
11.577511186	192.168.0.1	192.168.1.1	1
11.577578232	192.168.0.1	192.168.1.1	1

Fig. 6. Wireshark capture of packets during a port switching.

In the second experiment, we try to verify that the functionality of *Optical Bypass* can help a vNF-SC to meet its stringent QoS demand on end-to-end latency. The experimental setup is in Fig. 7(a), where we have *vNF* 1 sending traffic to *vNF* 4 in the HOE-DCN. During the period of $t \in [1, 6]$ seconds, the traffic gets forwarded in the EPS-based inter-rack network and has an end-to-end latency of $\sim$22.6 μs (as shown in Fig. 7(b)). Then, the control plane reduces the QoS demand on end-to-end latency down to 10 μs at $t = 7$ seconds. Hence, the ToR switch makes a local decision to reduce the end-to-end latency with *Optical Bypass*. Fig. 7(b) indicates that after the ToR switch switches the traffic to use the OCS-based inter-rack network, its end-to-end latency reduces to $\sim$2.9 μs. Therefore, the effectiveness of *Algorithm* 1 is confirmed.

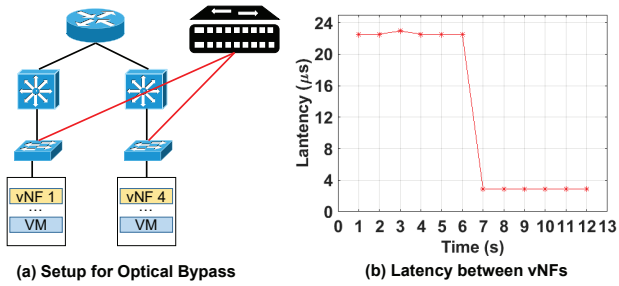

Fig. 7. Experiment for verifying the functionality of *Optical Bypass*.

C. NFV Orchestration for Hadoop Clusters

Next, we conduct two more system-level experiments to demonstrate that our proposed system can leverage local decisions in ToR switches to realize high-performance NFV orchestration for Hadoop clusters. The experimental setup is in Fig. 4, where the vNF-SC is *vNF* 1→(*vNF* 2/*vNF* 3)→*vNF* 4. In the first experiment, we let the vNF-SC use *vNF* 2 first, but when Hadoop tasks are running on the vNF-SC, we launch another VM on the server where *vNF* 2 is deployed to create resource contention there. Hence, as shown in Fig. 8(a), the throughput of *vNF* 2 is limited at $\sim$0.5 Gbps, and in the meantime, the resource contention will increase the processing latency of *vNF* 2. If the ToR switch that connects to *vNFs* 2 and 3 has the functionality of *Server Reselection*, it can use *Algorithm* 2 to switch the vNF-SC to use *vNF* 3. As shown in Fig. 8(b), this can improve the throughput of the vNF-SC to $\sim$1 Gbps, and thus the task completion time gets reduced from 150.18 seconds to 74.51 seconds.

The second experiment still uses the setup in Fig. 4, but tries to demonstrate the effectiveness of *Optical Offloading*. As shown in Fig. 9(a), the throughput of the vNF-SC is limited at $\sim$1 Gbps, if it always uses the EPS-based inter-rack network.

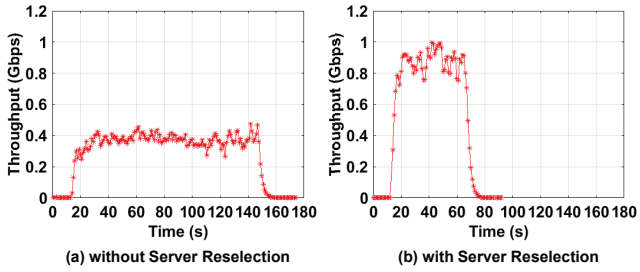

Fig. 8. Experimental results of *Server Reselection* for NFV orchestration.

After enabling *Optical Offloading* on the ToR switches, the throughput of the vNF-SC gets improved to $\sim$2.5 Gbps, which significantly reduces the task completion time of the Hadoop application that it carries (as shown in Fig. 9(b)).

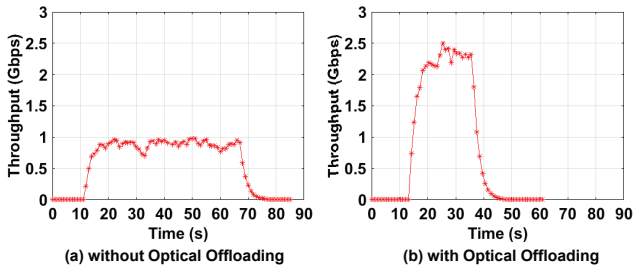

Fig. 9. Experimental results of *Optical Offloading* for NFV orchestration.

V. Conclusion

In this paper, we proposed to offload certain NFV orchestration tasks from the control plane to ToR switches, by leveraging PDP. Specifically, we architected an HOE-DCN whose ToR switches are PDP switches, designed the network orchestration system for it, and programmed the ToR switches such that they can make quick decisions locally to assist the control plane to realize agile vNF-SC provisioning. Our proposal was prototyped in a small-scale HOE-DCN testbed, and experimental demonstrations confirmed its effectiveness.

Acknowledgments

This work was supported in part by the NSFC projects 61871357 and 61771445, CAS Key Project (QYZDY-SSW-JSC003), and SPR Program of CAS (XDC02070300).

References

[1] Z. Zhu, S. Li, and X. Chen, "Design QoS-aware multi-path provisioning strategies for efficient cloud-assisted SVC video streaming to heterogeneous clients," *IEEE Trans. Multimedia*, vol. 15, pp. 758–768, Jun. 2013.

[2] P. Lu *et al.*, "Highly-efficient data migration and backup for Big Data applications in elastic optical inter-datacenter networks," *IEEE Netw.*, vol. 29, pp. 36–42, Sept./Oct. 2015.

[3] W. Lu *et al.*, "AI-assisted knowledge-defined network orchestration for energy-efficient data center networks," *IEEE Commun. Mag.*, vol. 58, pp. 86–92, Jan. 2020.

[4] N. Farrington *et al.*, "Helios: a hybrid electrical/optical switch architecture for modular data centers," *ACM SIGCOMM Comput. Commun.*, vol. 40, pp. 339–350, Aug. 2010.

[5] Z. Zhu, W. Lu, L. Zhang, and N. Ansari, "Dynamic service provisioning in elastic optical networks with hybrid single-/multi-path routing," *J. Lightw. Technol.*, vol. 31, pp. 15–22, Jan. 2013.

[6] L. Gong *et al.*, "Efficient resource allocation for all-optical multicasting over spectrum-sliced elastic optical networks," *J. Opt. Commun. Netw.*, vol. 5, pp. 836–847, Aug. 2013.

[7] Y. Yin *et al.*, "Spectral and spatial 2D fragmentation-aware routing and spectrum assignment algorithms in elastic optical networks," *J. Opt. Commun. Netw.*, vol. 5, pp. A100–A106, Oct. 2013.

[8] H. Bazzaz *et al.*, "Switching the optical divide: Fundamental challenges for hybrid electrical/optical datacenter networks," in *Proc. of SOCC 2011*, pp. 1–8, Aug. 2011.

[9] W. Fang *et al.*, "Joint defragmentation of optical spectrum and IT resources in elastic optical datacenter interconnections," *J. Opt. Commun. Netw.*, vol. 7, pp. 314–324, Mar. 2015.

[10] P. Lu, Q. Sun, K. Wu, and Z. Zhu, "Distributed online hybrid cloud management for profit-driven multimedia cloud computing," *IEEE Trans. Multimedia*, vol. 17, pp. 1297–1308, Aug. 2015.

[11] N. Xue *et al.*, "Demonstration of OpenFlow-controlled network orchestration for adaptive SVC video manycast," *IEEE Trans. Multimedia*, vol. 17, pp. 1617–1629, Sept. 2015.

[12] M. Zeng, W. Fang, and Z. Zhu, "Orchestrating tree-type VNF forwarding graphs in inter-DC elastic optical networks," *J. Lightw. Technol.*, vol. 34, pp. 3330–3341, Jul. 2016.

[13] W. Fang *et al.*, "Joint spectrum and IT resource allocation for efficient vNF service chaining in inter-datacenter elastic optical networks," *IEEE Commun. Lett.*, vol. 20, pp. 1539–1542, Aug. 2016.

[14] B. Li, W. Lu, S. Liu, and Z. Zhu, "Deep-learning-assisted network orchestration for on-demand and cost-effective vNF service chaining in inter-DC elastic optical networks," *J. Opt. Commun. Netw.*, vol. 10, pp. D29–D41, Oct. 2018.

[15] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: an intellectual history of programmable networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 87–98, Apr. 2014.

[16] H. Fang *et al.*, "Predictive analytics based knowledge-defined orchestration in a hybrid optical/electrical datacenter network testbed," *J. Lightw. Technol.*, vol. 37, pp. 4921–4934, Oct. 2019.

[17] Q. Li *et al.*, "Scalable knowledge-defined orchestration for hybrid optical/electrical datacenter networks," *J. Opt. Commun. Netw.*, vol. 12, pp. A113–A122, Feb. 2020.

[18] P. Bosshart *et al.*, "P4: Programming protocol-independent packet processors," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 87–95, Jul. 2014.

[19] C. Sun *et al.*, "SDPA: Toward a stateful data plane in software-defined networking," *IEEE/ACM Trans. Netw.*, vol. 25, pp. A113–A122, 2017.

[20] D. Borthakur, *The Hadoop Distributed File System: Architecture and Design*. Apache Software Foundation, 2007.

[21] L. Gong and Z. Zhu, "Virtual optical network embedding (VONE) over elastic optical networks," *J. Lightw. Technol.*, vol. 32, pp. 450–460, Feb. 2014.

[22] L. Gong, H. Jiang, Y. Wang, and Z. Zhu, "Novel location-constrained virtual network embedding (LC-VNE) algorithms towards integrated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 24, pp. 3648–3661, Dec. 2016.

[23] A. Laghrissi and T. Taleb, "A survey on the placement of virtual resources and virtual network functions," *IEEE Commun. Surveys Tuts.*, pp. 1409–1434, Second Quarter 2018.

[24] J. Liu *et al.*, "On dynamic service function chain deployment and readjustment," *IEEE Trans. Netw. Serv. Manag.*, vol. 14, pp. 543–553, Sept. 2017.

[25] B. Li, W. Lu, and Z. Zhu, "Deep-NFVOrch: Leveraging deep reinforcement learning to achieve adaptive vNF service chaining in EON-DCIs," *J. Opt. Commun. Netw.*, vol. 12, pp. A18–A27, Jan. 2020.

[26] N. McKeown *et al.*, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, pp. 69–74, Mar. 2008.

[27] Z. Zhu *et al.*, "Demonstration of cooperative resource allocation in an openflow-controlled multidomain and multinational SD-EON testbed," *J. Light. Technol.*, vol. 33, pp. 1508–1514, Apr. 2015.

[28] S. Li *et al.*, "Protocol oblivious forwarding (POF): Software-defined networking with enhanced programmability," *IEEE Netw.*, vol. 31, pp. 12–20, Mar./Apr. 2017.

[29] Tofino switch. [Online]. Available: https://www.barefootnetworks.com/products/brief-tofino/.

[30] F. Paolucci, F. Cugini, and P. Castoldi, "P4-based multi-layer traffic engineering encompassing cyber security," in *Proc. of OFC 2018*, pp. 1–3, Mar. 2018.