Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 110 (2017) 242–249

Procedia Computer Science

www.elsevier.com/locate/procedia

The 12th International Conference on Future Networks and Communications (FNC 2017)

A QoS-aware routing in SDN hybrid networks

Chienhung Lin, Kuochen Wang*, Guocin Deng

Department of Computer Science, National Chiao Tung University, Hsinchu, 300, Taiwan

Abstract

Changing the whole network into a software defined network (SDN) is impractical due to high replacement cost. Therefore, there exist SDN hybrid networks, where SDN switches and legacy switches coexist at the same time. In this paper, we propose an SDN hybrid network architecture which can discover existence of legacy switches by using the Spanning Tree Protocol and thus have a global view of the SDN hybrid network. We also enable OpenFlow switches to cooperate with legacy switches by using the Learning Bridge Protocol without requiring any modification on legacy switches. By utilizing the characteristics of SDN, SDN applications can dynamically find routing paths according to pre-defined QoS requirements and current network status. We also propose a *simulated annealing based QoS-aware routing* (SAQR) algorithm which can adaptively adjust weights of delay, loss rate and bandwidth requirements in a cost function to find the best fit path according to QoS requirements. We evaluate the proposed SAQR in a simulated SDN hybrid network which runs applications with different QoS requirements. Simulation results show that the SAQR performs better than related work MINA in terms of the fitness ratios of delay, loss rate and bandwidth, with 88%, 90.8% and 86.5% of flows meeting their respective QoS requirements, in contrast to MINA, with only 63%, 82.4% and 87.5% of flows meeting their respective QoS requirements.

Keywords: *Data center; hybrid network; QoS-aware; routing; SDN*

1. Introduction

The software defined network (SDN)[1] is a novel network architecture which separates the control plane and the data plane, and the network is managed by a logically centralized controller. This architecture has two benefits:

* Corresponding author. Tel.: +886-3-5131363; fax: +886-3-5721490.
E-mail address: kwang@cs.nctu.edu.tw

1877-0509 © 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the Conference Program Chairs.
10.1016/j.procs.2017.06.091

1) *Centralization*: A logically centralized controller can maintain a global view of the whole network, and this enables the controller to make a better decision than that in the traditional network. 2) *Abstraction*: It enables network managers to write applications deployed on the controller to control OpenFlow[2] switches and to prevent vendor lock-in. Although SDN has the benefits mentioned above, changing the whole network into an SDN network is impractical due to high replacement cost. So evolutionary deployment is a feasible way for data center networks to benefit from SDN. This is so called "hybrid networks" or "transition networks" [3, 4]. In an SDN hybrid network, we will encounter two main problems: 1) how to let the SDN controller have knowledge about the SDN hybrid network, 2) how to let OpenFlow switches and legacy switches cooperate with each other.

Related studies designed some SDN applications for SDN hybrid networks, such as link failure recovery and rerouting applications[5, 6]. However, their controllers need pre-built network topologies that include legacy switches. This is inflexible since the controller cannot grasp an SDN hybrid network topology automatically. In this paper, we use characteristics of STP (Spanning Tree Protocol) and LBP (Learning Bridge Protocol), which run on L2 legacy switches to solve the above two problems. Our proposed architecture can build an SDN hybrid network topology by discovering existence of legacy switches. We indirectly control legacy switches without modifying them for elastic routing, and this enables the proposed architecture to have better network utilization than Panopticon[4].

In addition to big data applications, more and more applications run in data centers have their own QoS requirements. Related studies proposed solutions on how to dynamically route flows by QoS requirements, but they considered only one constraint[7, 8]. We deal with this problem by proposing a simulated annealing (SA) based QoS-aware routing (SAQR) algorithm that runs on SDN hybrid networks and addresses multiple constraints. The proposed SAQR can fit the current network status and will not get stuck in a local optimum.

Our research makes two main contributions: 1) An SDN hybrid network architecture is proposed that can deploy partial SDN switches for data centers transiting from traditional networks to SDN networks and let applications do elastic routing by applying network topology discovering and controlling mechanisms to legacy switches. 2) The proposed SAQR can adaptively adjust weights of delay, loss rate and bandwidth deviations in a cost function to find the best fit routing path that meets *multiple* QoS requirements (or constraints). Simulation results show that the proposed SAQR is better than MINA[9], in terms of meeting delay, loss rate and bandwidth requirements.

2. Related work

2.1. SDN hybrid networks

The SDN hybrid network is a widely discussed problem recently because evolutionary deployment of SDN is practical for enterprises. The first thing we need to consider when deploying partial SDN switches into the network is that which and how many legacy switches should be replaced. In Panopticon[4], it simulated an enterprise network using a traffic matrix and VLAN (Virtual LAN) IDs to analyze an SDN deployment plan. In Chu et al.[5], authors used a heuristic algorithm to deploy as few SDN switches as possible, but can do recovery when detect any single link failure in the network. In Das et al.[10], authors used a greedy algorithm to decide which switches should be replaced to increase as many alternative paths as possible. After we decide where to place SDN switches, we need to let SDN switches and legacy switches cooperate with each other. In Panopticon[4], an entire network is separated into several SCTs (Solitary Confinement Trees) that each of them includes a host, a spanning tree built by legacy switches, and at least one SDN switch as a frontier for inter-domain routing. This will separate the whole network into several domains, and SDN switches will act as routers to do routing decisions. In Chu et al.[5], authors proposed an SDN hybrid network link failure recovery strategy. When a link failure occurs, it will use an IP-tunnel to construct a new path under the hybrid network. In the following, we address the issues of pre-built topology or network discovery in SDN hybrid networks. In Vissicchio et al.[3], authors classified hybrid networks into four classes: topology-based, service-based, class-based and integrated SDN hybrid networks, and analyze features and implementation challenges of each class. In Telekinesis[6], authors used SDN switches to indirectly control legacy switches by characteristic of LBP, in order to reroute packets to a new path. The related works mentioned above need a pre-built network topology for the SDN controller to have a global view of the hybrid network. In contrast, in our research, we intend to design a network topology discovery method that allows the SDN controller to build a hybrid network topology automatically, and to indirectly control legacy switches without modifying them for elastic routing.

2.2. QoS-aware routing

In data center networks, there are many big data applications. These applications need to transfer huge amounts of data, so how to dispatch these applications flows around the network to improve network utilization is an important issue. In Hedera[11], authors proposed an SA based algorithm to route flows to different paths by current network status. This work has better performance than the traditional ECMP (Equal Cost Multi-Path) [12]. In Long[7], authors proposed a dynamic rerouting algorithm. When the SDN controller detects network congestion, a single-hop or multi-hop algorithm is used to reroute flows to other paths. This method can improve network utilization compared to the traditional RR (Round Robin) algorithm.

The related works mentioned above only considered one constraint, link bandwidth. However, applications may have multiple QoS requirements, like delay, packet loss rate, bandwidth, etc. To deal with this problem, we may define an SLA (Service Level Agreement) for each application to record its QoS requirements, and allocate a routing path that meets the QoS requirements. This is so called QoS-aware routing. In LARAC[8], authors proposed a LARAC QoS-aware routing algorithm, which uses a Lagrange relaxation based aggregated cost method to find near optimal paths in the DCLC (Delay Constrained Least Cost) problem. In MINA[9], authors proposed a QoS-aware flows scheduling algorithm for IoT (Internet of Things) networks. This algorithm uses a genetic algorithm (GA) to iteratively calculate the best fit path that meets QoS requirements. The evaluation results showed that the algorithm has good performance when there are different QoS requirements in the network. In related works, only MINA addressed multiple constraints QoS-aware routing. However, static weights in the cost function and stuck in a local optimum are the main problems of MINA. To deal with these problems, we proposed a simulated annealing (SA) based QoS-aware routing (SAQR) algorithm to find the best fit path that meets multiple QoS requirements. In summary, Table 1 shows a qualitative comparison among the proposed SAQR and related work.

Table 1. Comparison among the proposed SAQR and related work.

Mechanism	Network discovery	Legacy switches controllable	Routing
Telekinesis[6]	Pre-built (SDN hybrid networks)	Yes	MAC learning
Chu et al.[5]	Pre-built (SDN hybrid networks)	No	IP-tunnel
Panopticon[4]	SCT (SDN hybrid networks)	No	VLAN IDs
LARAC[8]	LLDP (pure SDN networks)	N/A	Dijkstra-based QoS-aware routing (single constraint, delay)
MINA[9]	LLDP (pure SDN networks)	N/A	GA-based QoS-aware routing with static weights (multiple constraints)
SAQR (proposed)	STP (SDN hybrid networks)	Yes	SA-based QoS-aware routing with adaptive weights (multiple constraints)

3. Design approach

3.1. Topology discovery

After switches in an SDN hybrid network are initiated, SDN switches will send LLDP (Link Layer Discovery Protocol) packets to the SDN controller. However, legacy switches won't send such packets, so the SDN controller cannot have a complete topology of the entire network. So we need to "guess" the topology, and report a virtual topology for the controller to make routing decisions. In Panopticon[4], authors mentioned that switches in many data center networks usually run STP (Spanning Tree Protocol) and LBP (Learning Bridge Protocol) to build the networks and transmit data. Therefore, we can get rough data (such as MAC address and port id) of L2 switches information from STP packets. By STP packets, we can create a "virtual switch" in the SDN controller. A virtual switch is composed by one or more legacy switches. Each virtual switch contains these attributes: 1) *Switch id*: It is a MAC address obtained from root id in the STP packet header. 2) *Port id*: It is a string obtained from bridge id and port id in the STP packet header. For example, in Figure 1, the SDN controller will discover that there is a virtual switch (D) with two ports.

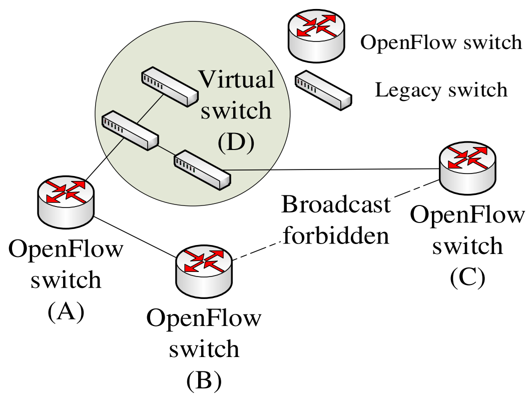

Figure 1. A virtual switch maintained in the SDN controller.

After we construct such a "virtual topology" in the SDN controller, we still have some problems to resolve: 1) *Host location discovery*: We locate a host by using ARP packets. If an OpenFlow switch receives an ARP packet the first time from a port connected to an unknown object, then the host is attached to this OpenFlow switch by that port. If that port is connected to a virtual switch, then the host is attached to the virtual switch. 2) *The loop problem in the network*: If we just simply flood all broadcast packets, it will cause some problems such as broadcast storm when the network has a loop like the topology in Figure 1. Therefore, we need to build a spanning tree and mark some links as broadcast-forbidden to avoid the loop problem in the network.

After the SDN controller gets the virtual topology of a hybrid network, it allows OpenFlow switches and legacy switches to cooperate to do routing decisions. Take Figure 1 as an example. We want to route a packet from OpenFlow switch A to OpenFlow switch C, by going through virtual switch D. On OpenFlow switches A and C, we can install flow entries to tell which output port to forward the packet. In virtual switch D, it will do forwarding itself by the traditional LBP. So the problem occurs on how to assure that virtual switch D will forward the packet to OpenFlow switch C. By the characteristic of LBP, we can send a Packet-out information which contains a "fake packet" from OpenFlow switch C to virtual switch D. This fake packet contains the address of the destination, and it will notify virtual switch D that these packets containing destination address C should be sent to switch C. This characteristic enables multi-path routing on static L2 switches, and makes our routing more elastic.

3.2. QoS-aware routing

In our design, the QoS-aware routing application contains three sub-modules: 1) *Topology discovery module*: As discussed above, this module receives packets from OpenFlow switches, legacy switches and hosts, and builds a virtual topology in the SDN controller. 2) *Network status collecting module*: This module will periodically retrieve network status from OpenFlow switches and estimate network status from legacy switches to help the flow scheduling module to allocate network resources. 3) *Flow scheduling module*: This module will allocate the best fit path by the proposed SAQR according to a pre-defined SLA, and a virtual topology and network status given by the topology discovery module and the network status collecting module, respectively.

For a user with specifying SLA, we give a requirements vector $< R_d, R_l, R_b >$ which means it needs a routing path with $< P_d, P_l, P_b >$, that satisfies delay constraint ($P_d \leq R_d$), packets loss rate constraint ($P_l \leq R_l$), and bandwidth constraint ($P_b \geq R_b$). The multiple constraints path selection is an NP-complete problem that we cannot find an optimal solution in polynomial time. So we need a heuristic algorithm to find a near optimal solution. In our design, the proposed SAQR, as shown in Algorithm 1, is used to find the best fit path. In Table 2, we give definitions of notations used in Algorithm 1 and the following equations.

Table 2. Notations and definitions.

Notation	Definition
P	Selected path
N	Neighbour path generated by selected path
C_P	Cost of selected path
C_N	Cost of neighbour path
t	Iterations count
T	Maximal iterations count
$< P_d, P_l, P_b >$	Delay, loss rate and bandwidth of selected path
$< R_d, R_l, R_b >$	Delay, loss rate and bandwidth of QoS requirements vector
$< W_d, W_l, W_b >$	Weights for delay, loss rate and bandwidth
$< MR_d, MR_l, MR_b >$	Miss rates for delay, loss rate and bandwidth
W_x, MR_x	Weight and miss rate for *x* (delay, loss rate or bandwidth)
c	Coefficient for probability function (equation (4))

Algorithm 1 shows the pseudocode of the proposed SAQR algorithm. When the routing module receives a Packet-in request, the QoS requirements (SLA) will be checked first. Then the routing module generates an initial path by Dijkstra's algorithm and calculates the cost (C_p) of this path by equation (1)[9]. Under different network status, each QoS parameter in a QoS requirements vector may affect the cost function differently, so we use equation (2) to adjust weights. Equation (2) shows how to calculate weight for *x* (*x* can be delay, loss rate or bandwidth). A weight will become larger if the corresponding QoS parameter could not meet in the past too many times, which may be indicated by the high miss rate of that parameter. The miss rate for *x* (MR_x) is defined in equation (3).

$$C_p = W_d \frac{(P_d - R_d)}{R_d} + W_l \frac{(P_l - R_l)}{R_l} + W_b \frac{(R_b - P_b)}{R_b} \tag{1}$$

$$W_x = \frac{MR_x}{MR_d + MR_l + MR_b} \tag{2}$$

$$MR_x = \frac{Number\ of\ flows\ that\ cannot\ meet\ requirement\ x}{Number\ of\ flows\ in\ history} \tag{3}$$

In Algorithm 1, before iterations count *t* decreases to zero, we iteratively do the following two steps: 1) Generate a neighbor path and calculate the path cost. We replace a sub-path in the current path by passing through an unused port on a random selected node on the current path. 2) Move to a neighbor state (move_to_neighbor_state, as shown in Algorithm 1): if a neighbor path has lower cost, then we replace the current path by the neighbor path. Otherwise, we will replace the current path according to a probability calculated by equation (4)[11]. We set $c = -0.75$ since this leads to a better result by simulation. This step allows the algorithm to move to another state randomly, and will not be stuck in a local optimal solution[11], and this is the superiority of SA contrary to other heuristic algorithms.

$$p(C_P, C_N, t) = \begin{cases} 1 & , \quad C_N < C_P \\ e^{\frac{c|C_N - C_P|}{t}} & , \quad C_N \geq C_P \end{cases} \tag{4}$$

Algorithm 1. SA-based QoS-aware routing (SAQR).

```
Input:
        src (source address),
        dst (destination address),
        r_vec (QoS requirements vector)
Output:
        $P$ (routing path)
$P$ = Dijkstra (src, dst)
$C_P$ = calculate_path_cost($P$, r_vec)
for $t$ = $T$ to 0 do
        $N$ = generate_neighbor_path($P$)
        $C_N$ = calculate_path_cost($N$, r_vec)
        if move_to_neighbor_state($C_P$, $C_N$, $t$) then
                $P = N$
                $C_P = C_N$
        end if
end for
return $P$
```

4. Evaluation

4.1. Simulation setup

The simulation environment is shown in Table 3. We used Floodlight[13] as our SDN controller, and used Mininet[14] to simulate a fat tree topology with k = 4 pods. Our simulation included two parts: hybrid network simulation and QoS-aware routing simulation. In the hybrid network simulation, we deployed 4 to 16 OpenFlow switches, and the rest were legacy switches which run STP and LBP. We used hot-spot traffic[7] as the traffic pattern to cause congestion in the hybrid network. In the QoS-aware routing simulation, we replaced four core switches with OpenFlow switches. We set number of iterations T in the proposed SAQR to 16 obtained from simulation. Though the iterations count in the proposed SAQR is larger than that in MINA[9], our algorithm is faster than MINA (3.62 ms contrary to 9.18 ms) in the simulation. We used uniform traffic as the traffic pattern for QoS-aware routing simulation and a pre-defined SLA which records QoS requirements for each flow. Note that MINA is the only related work that addresses QoS-aware routing. Therefore, we compare the proposed SAQR with MINA[9].

Table 3. Simulation environment.

Simulation	Hybrid network	QoS-aware routing
Controller	Floodlight[13]	
Network simulator	Mininet[14]	
Topology	Fat tree (k = 4)	
Number of OpenFlow switches	4 ~ 16	20 (4 core switches)
Test data	Hot-spot traffic[7], more than 80% of traffic sent from one source	Uniform traffic, with QoS requirements for each flow
Routing method	ECMP[12]	SAQR: SA with T = 16 MINA: GA with T = 10

4.2. Simulation results

As mentioned in section 2, our routing is more elastic due to indirect control on legacy switches and thus it enables multi-path routing to improve network utilization. In Figure 2(a), we compare the average throughput per flow under a different number of OpenFlow switches deployed using Panopticon[4] and the proposed SAQR. We found that in Figure 2(a), a network with more OpenFlow switches has better average throughput per flow, because legacy switches will block some links to avoid looping by STP. That is, more OpenFlow switches leads to more available links. In addition, our result is better than Panopticon[4] because being able to indirectly control legacy switches can distribute flows to more links to avoid congestion and to improve average throughput per flow.

We show our simulation results of delay (Figure 2(b)), loss rate (Figure 2(c)), and bandwidth (Figure 2(d)) under a different flow id. The bars in each figure represent the user QoS requirements for each flow, and zero means the flow has no requirement on that attribute. A curve in each figure represents the actual network status of the routing path found by a corresponding algorithms (LARAC[8], MINA[9], and the proposed SAQR). Based on the results of Figure 2(b), 3(c) and 3(d), Table 4 presents the fitness ratio for a respective QoS requirement (delay, loss rate or bandwidth). Note that the fitness ratio means that how many flows meet a respective QoS requirement. As shown in Figure 2, our proposed SAQR and MINA[9] can meet more QoS requirements than LARAC[8] because LARAC only considers a single constraint, delay. The proposed SAQR is better than MINA in terms of delay. This is due to two reasons: 1) *Global optimum*: Our algorithm can move to a worse case randomly and this enables the algorithm not be stuck in a local optimal solution. 2) *Weight adjustment*: We found that bandwidth is an abundant resource in the network. The bandwidth fitness ratios of LARAC and MINA are 86% and 87.5%, respectively. However, the delay fitness ratios of them are only 50% and 63%, respectively. In contrast, the proposed SAQR can dynamically adjust the weight of delay (W_d) to raise the influence of delay to adapt to the current network status for finding better solutions (delay fitness ratio = 88%).

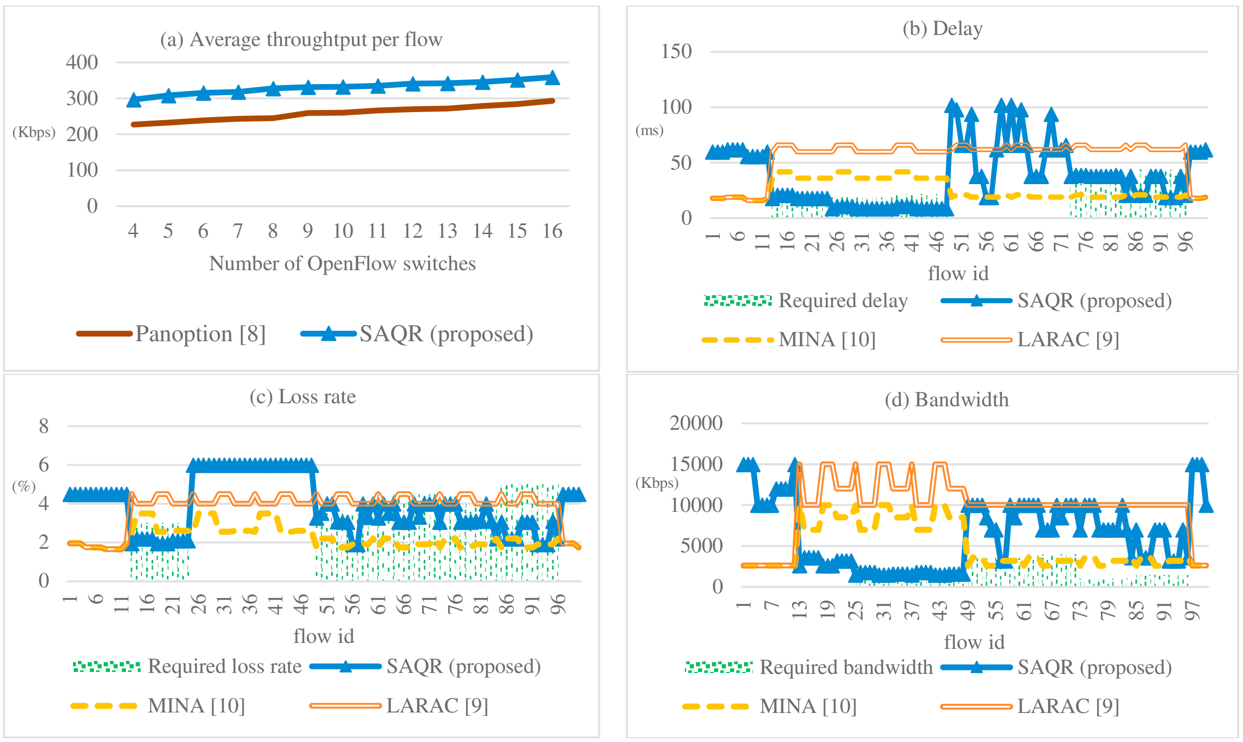

Figure 2. Simulation results of (a) average throughput per flow under a different number of OpenFlow switches deployed (b) delay (c) loss rate (d) bandwidth under a different flow id.

Table 4. Fitness ratios for the simulation results.

Fitness ratio	Delay	Loss rate	Bandwidth
LARAC[8]	0.50	0.729	0.860
MINA[9]	0.63	0.824	0.875
SAQR (proposed)	0.88	0.908	0.865

5. Conclusion

We have presented an SDN hybrid network architecture that SDN switches and legacy switches coexist in the network simultaneously. Our design can discover existence of legacy switches by using STP to build a virtual topology for the SDN hybrid network and it enables OpenFlow switches to cooperate with legacy switches for elastic routing. We have also presented a simulated annealing based QoS-aware routing (SAQR) algorithm which can adaptively adjust the weights of delay, loss rate and bandwidth deviations in the cost function to find the best fit path that meets QoS requirements. The proposed SAQR algorithm is able to adapt to the current network status and not to be stuck in a local optimum. We have evaluated the proposed SAQR in a simulated SDN hybrid network which runs applications with different QoS requirements. Simulation results have shown that the proposed SAQR performs better than MINA, in terms of the fitness ratios of delay, loss rate and bandwidth, with 88%, 90.8% and 86.5% of flows meeting their respective QoS requirements. In contrast, for MINA, only 63%, 82.4% and 87.5% of flows meet their respective QoS requirements. In this work, we only consider L2 legacy switches that coexist with SDN switches. However, there are other types of network devices, such as L3 switches, L3 routers, etc., in the network. In the future, we may take these devices into account as well for QoS-aware routing in SDN hybrid networks.

Acknowledgements

The support by the Ministry of Science and Technology under Grant MOST 105-2221-E-009-091-MY3 and by the Inventec under Contract 105C584 is gratefully acknowledged.

References

1. Rawat Danda B., and Reddy Swetha R. "Software defined networking architecture, security and energy efficiency: A survey." *IEEE Communications Surveys & Tutorials*, Oct. 2016.
2. Lara Adrian, Kolasani Anisha, and Ramamurthy Byrav. "Network innovation using OpenFlow: A survey." *IEEE Communications Surveys & Tutorials,* vol. 16, no. 1, pp. 493–512, Aug. 2014.
3. Vissicchio S. et al. "Opportunities and research challenges of hybrid software defined networks," in *ACM SIGCOMM Computer Communication Review*, pp. 70-75, 2014.
4. Levin D. et al. "Panopticon: Reaping the benefits of incremental SDN deployment in enterprise networks," in *Proc. USENIX Annu. Tech. Conf.*, 2014, pp. 333-345.
5. Chu C. Y. et al. "Congestion-aware single link failure recovery in hybrid SDN networks," in *Proc. IEEE INFOCOM*, Aug. 2015, pp. 1086-1094.
6. Jin C. et al. "Telekinesis: Controlling legacy switch routing with OpenFlow in hybrid networks," in *Proc. 1st ACM SIGCOMM Symp. on Software Defined Networking Research*, June 2015, pp. 1-7.
7. Long H. et al. "LABERIO: Dynamic load-balanced routing in OpenFlow-enabled networks," in *Proc. Advanced Information Networking and Applications*, Mar. 2013, pp. 290-297.
8. Jüttner A. et al. "Lagrange relaxation based method for the QoS routing problem," in *Proc. IEEE INFOCOM*, 2001, pp. 859-868.
9. Qin Z. et al. "A software defined networking architecture for the internet-of-things," in *Proc. IEEE Network Operations and Management Symp.*, May 2014, pp. 1-9.
10. Das T. et al. "Insights on SDN migration trajectory," in *IEEE International Conference Communications*, Sep. 2015, pp. 5348-5353.
11. Al-Fares, et al. "Hedera: Dynamic flow scheduling for data center networks," in *Proc. 7th USENIX Conf. on Networked System Design and Implementation*, Apr. 2010.
12. ECMP. *https://en.wikipedia.org/wiki/Equal-cost_multi-path_routing*.
13. Floodlight. *http://www.projectfloodlight.org/floodlight/*.
14. Mininet. *http://mininet.org/*.