

# Accurate Network Resource Allocation in SDN according to Traffic Demand

Hongyan Cui<sup>1,2,a</sup> Chenhang Ma<sup>1,2,b</sup> Wei Lai<sup>1,2</sup> Lili Zheng<sup>1,2</sup> Yunjie Liu<sup>1,2</sup>

<sup>1</sup>State Key Lab. of Networking and Switching Technology

<sup>2</sup>Beijing Key Lab. of Network System Architecture and Convergence

Beijing University of Posts and Telecommunications, Beijing, China

<sup>a</sup>cuihy@bupt.edu.cn, <sup>b</sup>mch7731@163.com

**Keywords:** SDN, Network Resource Allocation, Quality of Service

**Abstract.** The services carried by the Internet have undergone tremendous change over the past decades, along with the emergence of multimedia services and cloud computing, etc. However, there is little change in the framework of network designed primarily for reliable end-to-end communication, which is not suitable for these new services, resulting in poor performance of services and inefficient utilization of network resource. In this paper, we propose a network resource allocation scheme based on SDN (Software Defined Network) an emerging network paradigm, to provide Quality of Service (QoS) support for various services while balance the link load in the network avoiding congestion and idleness. And we demonstrate its performance via experiments under our SDN test bed.

## I. Introduction

With the development of the Internet, there are more and more types of services carried by the Internet competing for network resource, most of which require performance guarantees. Driven by the demand for multimedia services, a lot of researches about allocating network resource have been carried out over the past decades, archiving several solutions aiming at providing guarantee for specific services, such as IntServ and DiffServ etc. However, limited to the traditional framework, they lack the overview of the network-wide resource and real-time adaption to status of network [1], unable to satisfy the needs of the future network.

As described above, the fundamental problem of traditional solutions lies in the distributed hop-by-hop framework of traditional network, which was designed primarily for reliable end-to-end communication. It is not suitable for the diverse requirements of multimedia services which are more and more popular, resulting in poor performance of services and inadequate utilization of network resource. Noting the limitation of traditional network, we conducted our research based on SDN (Software Defined Network) [2] to address these problems.

SDN is a new proposed architecture of network, with the target to address the limitation of traditional network. It is changing the way networks are designed and managed with two defining characteristics. First, SDN makes the control plane and data plane separate from each other. Second, SDN consolidates the control plane, so that a single software control program can control multiple transmission elements [3]. In this way, the control plane can obtain the network-wide logical view instead of the neighborhood information of traditional routers and the workload of each device. The

logical view is a simplified, abstract representation of the network designed for easier policies specification [4]. And network operators can configure all devices through the control plane, greatly simplifies the network configuration. SDN also introduce the programming capability into network, making network automatic and active. Furthermore, network administrators can develop new services according to the demand of users and current situation, simply through some application interfaces.

Combined with the characteristics of SDN, the management and operation of network become simpler and more flexible, binging new direction to the network resource allocation issue. Many researches about network resource allocation in the SDN environment have been carried out, and even some have been deployed in real network. The most well-known case is B4 [5] designed by Google, a Software Defined WAN for data center to data center connectivity. It archived more than 70% average utilization for all links, while many links are driven utilization close to 100% [6]. In addition to data center and inter-data center network, allocating network resource for services required by end-users is another hot research topic, which is also our focus. In this area, most researches pay attention to resource allocation for a single service, such as streaming service, to guarantee its performance, e.g. Hilmi E. Egilmez et al proposed an optimization framework for dynamic routing of QoS flows to stream scalable coded videos [1]. This is important because of the preference of users and complexity of video streaming service. However lacking global consideration with other services may have negative impact on efficiency of network resource utilization.

Based on this idea, we propose an intelligent network resource allocation scheme based on SDN, for automatically assignment of network resource to various kinds of services on demand in this paper. According to the features of the performance required by each service, the controller organizes them into different groups, and referring the current status of network allocates network resource to satisfy their individual requirements respectively relied on queuing mechanism. The goal of our proposed scheme is to provide QoS support for various kinds of services while balance the link load in the network avoiding congestion and idleness. We implemented the scheme on the SDN test bed and evaluated its performance.

The rest of the paper is organized as follows. In section II the background of our work is presented. Section III provides the related works and our motivation. Section IV introduces our network resource allocation scheme in detail. We discuss the evaluation of our scheme implemented on the SDN test bed in section V and conclude our work in section VI.

## **II. Background**

Due to the initial design of the framework, the number of the devices in the Internet had grown continuously and the management of Internet had become more and more trivial and complicated, with the increase of the kinds of applications in the Internet. Facing these challenges, researchers in the field of network wanted to redesign the framework of the network, and SDN was one of the proposals.

SDN was born from OpenFlow protocol [7], which is proposed by Nick McKeown and his colleagues. OpenFlow protocol defined the specifications of communications between data plane and control plane. Although some enterprises have proposed their own protocols for SDN, OpenFlow is still the most widely applied southbound interface of SDN, and has become the de facto standard [8].

According to OpenFlow protocol, switches handle the traffic in the network based on flow, which is a set of packets with the same characteristics. The controller defines the rules handling each flow and conveys to switches. It allows network administrators to control traffic in a fine-grained way. And network administrators can manage the whole network through the controller, saving a lot of per-device manual configuration. Network administrators also can obtain the overview of the topology and current status of the network thanks to the logically centralized control plane. Furthermore, the control plane provides researchers and administrators with application program interfaces, allowing them to introduce new applications and services quickly and easily. All of the features of SDN make the network active, flexible and easier to manage.

### **III. Motivation and related works**

Subject to numerous restrictions of traditional network framework, many research programs could not get the desired results. With the emergence of OpenFlow and SDN in succession, researchers began to study the solutions to challenges faced by Internet based on the new network paradigm, and there have been some research results.

Whnho Kim et al. extended the OpenFlow version 0.8.9 with priority queues and rate limiters for different services, to satisfy their performance requirements [9]. And OpenFlow protocol has added the priority queues ability from version 1.0.0 [10].

Jonathan Chase et al. proposed an optimal approach integrating virtual machine and network bandwidth provisioning in a cloud computing environment [11]. Rafael L. Gomes et al. presented a framework for dynamic resources allocation in VSDNs based on traffic demand and policy adjustment to support the resource fitting decisions [12]. These works focused on the scenario that Internet Service Providers provide users with network resource not services.

Michael Jarschel et al. presented the improvement of the Quality of Experience (QoE) of Youtube video streaming as an example, to show the benefits of combing application-state information with network management ability of SDN [13]. Panagiotis Georgopoulos et al. introduced an OpenFlow-assisted QoE Fairness Framework (QFF) focusing on the demand of multimedia networks and optimizing the QoE of multiple competing and heterogeneous clients, to achieve user-level fairness in an adaptive video streaming environment [14]. OpenQoS [15] is a novel approach to provide dynamic QoS routing to fulfill end-to-end support for steam video over OpenFlow network. These works only paid attention to video streaming service without consideration of other services. However nowadays there are various kinds of applications requiring performance guarantee in the Internet. Optimization for single service is hard to archive global optimization and efficient use of network resource.

Slavica Tomovic et al. presented new SDN control framework to provide required QoS-level for priority services in automated manner [16]. Kailong Li et al. proposed a novel BRAS architecture using SDN to improve the QoE of uses by adjusting the bandwidth of a specific application according requirements of users [17]. These works focused on the topic allocating network resource for multiple services, but only consider bandwidth as the single factor.

Although bandwidth is the most important, other factors like delay and jitter have significant effect on specific services. Furthermore the requirements of services for these factors are different from each other. That is to say different network parameters (such as bandwidth, delay, jitter, etc)

have different degree of influence on the performance of a specific service. For example, VoIP demands strict guarantee for delay and jitter, but costs little bandwidth. Video on demand and online game also have strict requirements for delay, but they have less strict demand for jitter. While large downloads service has no demand for delay and jitter. Therefore applying network resource with appropriate properties to different services can obtain more efficient utilization of network resource while maintaining the performance of services. Based on this idea, our work focused on a network resource allocation scheme targeted on different requirements of multiple services.

#### IV. Design of network resource allocation

This paper proposes a network resource allocation scheme, aiming to guarantee performance of different services and efficiently use network resource, by providing network resource to services according to their features and current network status. The whole architecture of our scheme is shown in Fig.1. The main modules involved are: traffic classification, queue mechanism, status collection, and route calculation. Below we will describe each of them in detail.

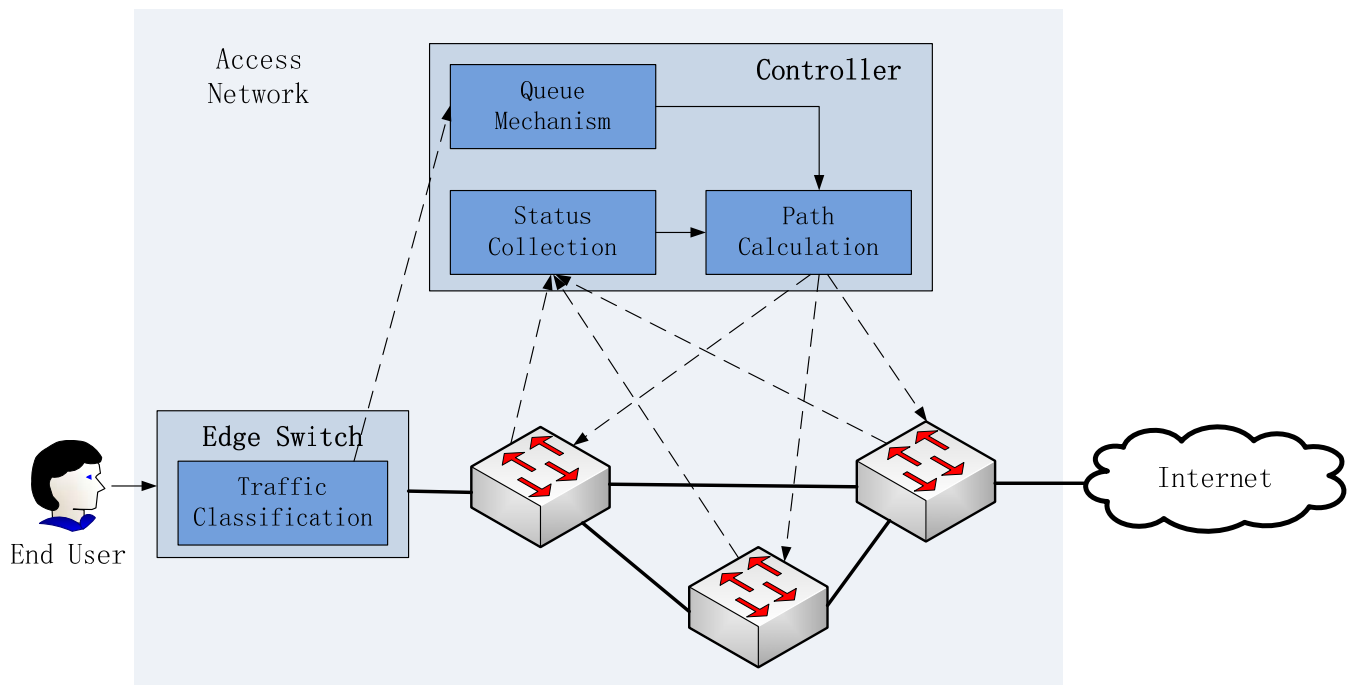


Fig.1. Architecture of our network resource allocation scheme

##### A. Traffic classification

According to OpenFlow protocol, when a new flow enters the network for the first time, the switch will upload it to the controller, if there is no match for it in the flow tables. Then the controller needs to decide the treatment of this new flow and issues corresponding flow entries which contain the properties of this new flow and the actions to handle it, to update flow tables in involved switches. As we described in Section 3, the requirements of different services focus on different network parameters. However, the controller cannot get the service category information from the properties of the flows, which include IP address and transport protocol etc, based on the current OpenFlow protocol.

Therefore, we need to extend the capability of identifying flows of various services. There are many traffic classification technologies and Deep Packet Inspect (DPI) is quite mature one of them. We adopt nDPI [18] software as the traffic classification tool in this scheme. The nDPI software is an open source and cross-platform DPI engine based on the OpenDPI library, allowing application-layer detection of protocols regardless of the port being used [19].

Having in mind the resource consumed by a running DPI engine, we just deploy nDPI software in the edge switches which are directly connected to end-users. We use Open vSwitch [20] running on a Linux operating system to serve as the switch in our test bed. Open vSwitch is a production quality, multilayer virtual switch enabling massive network automation through programmatic extension while still supporting standard management interfaces and protocols. In this way we can install nDPI software on a switch to introduce the traffic classification capability into the network easily.

Equipped with nDPI software, when a new flow arrives, the DPI engine running in the switch will analyze this flow and get the service category information of it. Then the switch uploads the category information with the properties of this flow to the controller, as the reference for priority queue mapping decision.

## B. Queue mechanism

This module is responsible for the queue assignment of different flows based on type of services. After becoming aware of the service category information of flows, we need to provide network resource to them based on the properties of their requirements. In this scheme, we divide the network into small virtual networks, taking advantage of the queue mechanism supported by OpenFlow protocol and Open vSwitch to control the resource allocation at each switch. We can attach one (or more) queues to a port, and flow entries mapped to a specific queue will be treated according to the configuration of that queue [10]. When the controller receives a new flow, it can get the service type that the flow belongs to and the performance the flow demands from the application category information. Then the controller will map the new flow to a virtual network composed by queues of appropriate links which meets its performance requirements.

Always we set multiple queues at a port and assign different priorities to them. However, when a new flow is mapped to a specific queue, the existing flows in the queues with lower priority will be affected, especially for delay and jitter. According to this situation and different dependence of the services on different network parameters, we divide services into different groups based on the performance requirements of them.

TABLE 1. GROUPING RULES OF SERVICE

Group ID	Delay Demand [ms]	Jitter Demand [ms]	Example of Service
No.1	$\leq 40$	$\leq 10$	VoIP
No.2	$\leq 400$	$\leq 50$	VOD
No.3	$\leq 100$	/	Online Game
No.4	$\leq 1000$	/	Web, Email
No.5	/	/	FTP, P2P

We assign the services demanding strict delay and jitter, like VoIP and carrier-class video, to group No.1. For the services demanding less strict delay and jitter, like video on demand thanks to

caching capabilities of the terminal, we assign them to group No.2. The highly interactive real-time services which demand strict delay and are not sensitive to jitter like online game are assigned to group No.3. For the services demanding delay less strict than the services belong to group No.2, like web browser, text chat and email, we assign them to group No.4. The rest services having no demand for delay and jitter like download service of FTP and P2P are assigned to group No.5. These grouping rules are show in Table 1.

Then we will map traffic belonging to different groups to different priority queues based on properties of their requirements. Every group is mapped to one or more queues. Group No.1 is mapped to the highest priority queues, and group No.2 is mapped to secondary priority queues. The rest is mapped in the same manner. In this way we can minimize the influence on existing flows in the network when a new priority flow arrives, while providing the new flow with required network resource.

In this scheme we adopt static queue configuration. For different actual scenes, the configuration of queues should be based on the traffic volume and the proportion of different services under normal conditions to satisfy all the services the network carries.

### **C. Status collection**

In traditional network, the routers which determine the forwarding path only know neighbors information. This incomprehensive information always results in non-optimal solutions and congestion. So we construct the status collection module to collect and maintain the topology and current status of the network aiming to improve the utilization of network resource.

The controller we used in the test bed has built-in topology discovery capability. The controller sends Link Layer Discovery Protocol (LLDP) packets to all switches connecting to it, and orders switches to send out the LLDP packets from all its ports except for the entrance of the LLDP packet. The switch will send the LLDP packet to the controller when it receives one. The controller can confirm a direct link between two switches according to two identical LLDP packets. In this way the controller obtains all link information and calculates the entire network topology.

For the status information of network, we use OFPT\_STATS\_REQUEST message to query the switches about the counter values of flows, ports and queues periodically [10]. According to these data, this status collection module can calculate the available bandwidth and load of different links and queues. The network status information will be sent to route calculation module as reference to reduce the probability of congestion.

### **D. Path calculation**

Most traditional routing protocols only take number of routers involved and link bandwidth as the indicators to calculate the shortest path simply. This leads to the existence of idle and congested links in the network at the same time, greatly reducing link utilization. The path calculation module introduces consideration of network status to avoid this situation. As described in module B, flows are divided into 5 groups based on requirements and are mapped to different priority queues. We will apply different indicators to them.

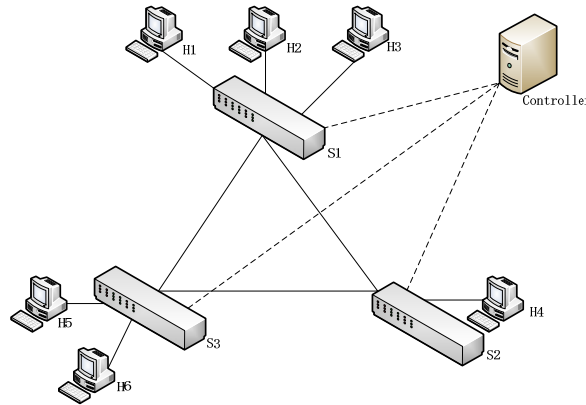
Flows belonging to group No.1-No.4 have requirements for delay to guarantee their QoS. So we set up two indicators, bandwidth and delay, for their route policies. The path calculation module should find out paths satisfies the two indicators at the same time. For flows belonging to group No.5

which have no requirements for delay, namely general best-effort flows, we set up bandwidth as the only indicator. And the calculation basis of the path calculation module is current available resource and relative load instead of the total amount of resource, to balance the utilization of resource.

The problem described above is a Constrained Shortest Path (CSP) problem, and it is known to be NP-complete. In this scheme, we adopt the Lagrangian Relaxation Based Aggregated Cost (LARAC) algorithm to address this CSP problem. LARAC is a polynomial-time algorithm that efficiently finds a route without deviating from the optimal solution in  $O([m + n \log n]^2)$  time, where  $n$  and  $m$  are the number of nodes and links respectively [1, 21].

## V. Implement and evaluation

To evaluate the ability of our network resource allocation scheme to allocate network resource and assure quality of service, we implemented it on our SDN test bed. We constructed the data plane using Open vSwitch running on Linux system, and implemented additional modules on the top of a floodlight controller [22], an Open SDN Controller. The topology of our SDN test bed is shown in Figure 2. Three switches are connected to each other via 1G bandwidth links.



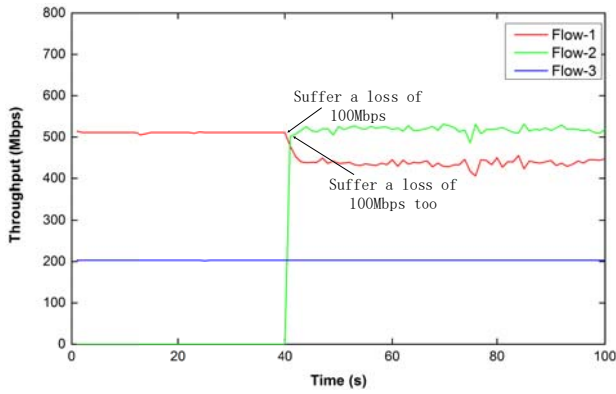
**Fig.2. Topology of our test bed**

To simulate flow of different services, we adopt Iperf [23], a tool for measuring network performance, to generate three test flows between hosts (H1 to H6) as described in Table II. Flow-1 and Flow-2 were started at time zero and Flow-3 was started 40 seconds later. Then we observed the performance of three flows in different situations.

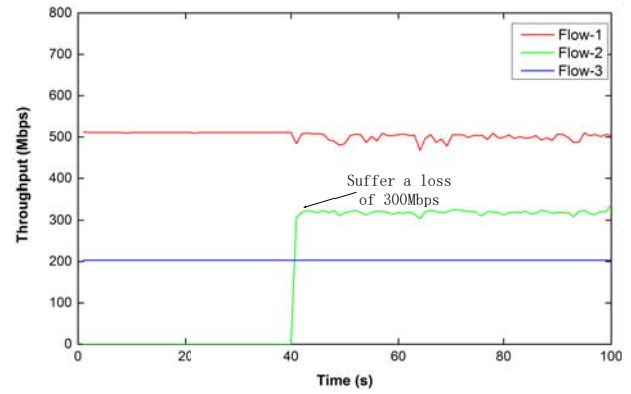
TABLE 2. CHARACTERISTICS OF FLOWS

Flow	Source-Destination	Rate[Mbps]	Group
1	H1-H5	500	No.1
2	H2-H4	200	No.4
3	H3-H6	600	No.2

In the first experiment, we implemented no additional function on the controller, and all flows were forwarded in best-effort way. The time-varying throughputs of each flow are shown in Figure 3. After Flow-3 was started, the rate of Flow-1 decreased about 100Mbps, while Flow-3 also did not get its required 600Mbps.



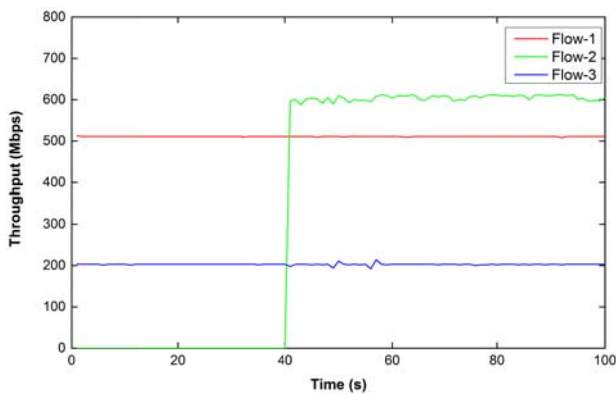
**Fig.3. Throughput in the first experiment**



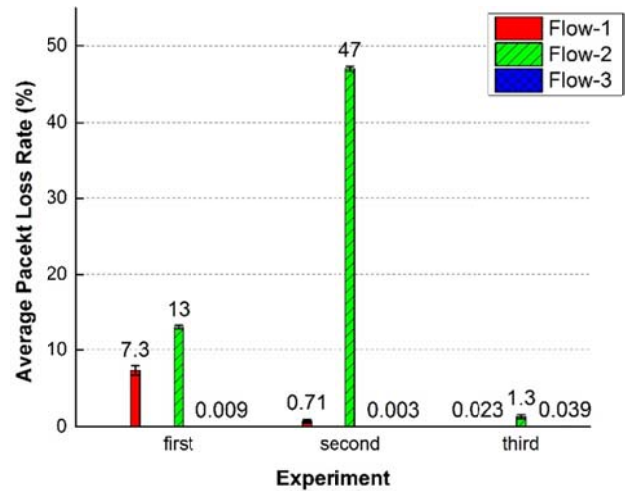
**Fig.4. Throughput in the second experiment**

In the second experiment, we set up priority queues in the links. As shown in Figure 4, the performance of Flow-1 belonging to group No.1 was much better. While for lack of suitable priority queues, Flow-3 suffered a severe degradation, with almost 50% packets failed to transmit. However there was abundant bandwidth left in another path via S2 which had the ability to carry Flow-3. This scenario is similar to the network resource allocation scheme in traditional network. Lacking of global perspective and flexibility, network resource is hard to be fully utilized.

In the third experiment, we implemented our network resource allocation scheme on the controller. Flow-3 was redirected to the light load path which could satisfy its requirements. As shown in Figure 5, the required bandwidth of each flow is guaranteed.



**Fig.5. Throughput in the third experiment**



**Fig.6. Average Packet Loss Rate of three Experiments**

Moreover, as shown in Figure 6, average packet loss rate of three experiments are compared. The results show the improvement our scheme brings to the performance of three flows. And while satisfying the requirements of Flow-3 by redirecting it to a light load path, efficient use of network resource is archived.

The delay and jitter are not shown, because they have little variation for the small scale of our test bed. And static queue mapping we used in the scheme still have problems in terms of efficiency sometimes, although we have do a certain degree of overlap. We will research the two problems in our future work.



## VI. Conclusion

In this paper, we propose an intelligent network resource allocation scheme to assign network resource to different services according to their demands and network status based on SDN. It can guarantee requirements of various services, while archiving efficient utilization of network resource, as show in the evolution of the experiments we implemented in our test bed.

As for our future research, we plan to extend our test bed or conduct experiments in simulation environment to evaluate the impact to delay and jitter in our scheme. And we will research dynamic queue mapping algorithm to further improve the efficiency of network resource utilization. Furthermore, we will extend our scheme to OpenFlow 1.3.0 which is another stable and long term supported version.

## Acknowledgment

This work has been supported by the National Natural Science Foundation of China (61201153), the National 973 Program of China under Grant (2012CB315805), Prospective Research Project on Future Networks in Jiangsu Future Networks Innovation Institute (BY2013095-2-16), the National Basics Research Program 973 of China (2012CB315801), and the Fundamental Research Funds of China for the Central Universities (2013RC0113).

## References

- [1] Egilmez H E, Civanlar S, Tekalp A M. An optimization framework for QoS-enabled adaptive video streaming over OpenFlow networks[J]. Multimedia, IEEE Transactions on, 2013, 15(3): 710-715.
- [2] Fundation O N. Software-defined networking: The new norm for networks[J]. ONF White Paper, 2012.
- [3] Feamster N, Rexford J, Zegura E. The road to SDN[J]. Queue, 2013, 11(12): 20.
- [4] Heller B, Scott C, McKeown N, et al. Leveraging SDN layering to systematically troubleshoot networks[C]//Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. ACM, 2013: 37-42.
- [5] Jain S, Kumar A, Mandal S, et al. B4: Experience with a globally-deployed software defined WAN[C]//ACM SIGCOMM Computer Communication Review. ACM, 2013, 43(4): 3-14.
- [6] Akyildiz I F, Lee A, Wang P, et al. A roadmap for traffic engineering in SDN-OpenFlow networks[J]. Computer Networks, 2014, 71: 1-30.
- [7] McKeown N, Anderson T, Balakrishnan H, et al. OpenFlow: enabling innovation in campus networks[J]. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69-74.
- [8] Song H. Protocol-oblivious forwarding: Unleash the power of SDN through a future-proof forwarding plane[C]//Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. ACM, 2013: 127-132.
- [9] Kim W, Sharma P, Lee J, et al. Automated and scalable QoS control for network convergence[J]. Proc. INM/WREN, 2010, 10: 1-1
- [10] Open Networking Foundation. OpenFlow Switch Specification Version 1.0.0, 2009

- [11] Chase J, Kaewpuang R, Yonggang W, et al. Joint virtual machine and bandwidth allocation in software defined network (SDN) and cloud computing environments[C]//Communications (ICC), 2014 IEEE International Conference on. IEEE, 2014: 2969-2974.
- [12] Gomes R L, Bittencourt L F, Madeira E R M, et al. An architecture for dynamic resource adjustment in VSDNs based on traffic demand[C]//Global Communications Conference (GLOBECOM), 2014 IEEE. IEEE, 2014: 2005-2010.
- [13] Jarschel M, Wamser F, Hohn T, et al. Sdn-based application-aware networking on the example of youtube video streaming[C]//Software Defined Networks (EWSN), 2013 Second European Workshop on. IEEE, 2013: 87-92.
- [14] Georgopoulos P, Elkhatib Y, Broadbent M, et al. Towards network-wide QoE fairness using OpenFlow-assisted adaptive video streaming[C]//Proceedings of the 2013 ACM SIGCOMM workshop on Future human-centric multimedia networking. ACM, 2013: 15-20.
- [15] Egilmez H E, Dane S T, Bagci K T, et al. OpenQoS: An OpenFlow Controller Design for Multimedia Delivery with End-to-End Quality of Service over Software-Defined Networks[C]//Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific. IEEE, 2012: 1-8.
- [16] Tomovic S, Prasad N, Radusinovic I. SDN control framework for QoS provisioning[C]//Telecommunications Forum Telfor (TELFOR), 2014 22nd. IEEE, 2014: 111-114.
- [17] Li K, Guo W, Zhang W, et al. QoE-based bandwidth allocation with SDN in FTTH networks[C]//Network Operations and Management Symposium (NOMS), 2014 IEEE. IEEE, 2014: 1-8.
- [18] nDPI. <http://www.ntop.org/products/ndpi/>
- [19] nDPI QuickStart Guide. <http://www.ntop.org/support/documentation/>
- [20] Open vSwitch. <http://openvswitch.org/>.
- [21] Juttner A, Szviatovski B, Mécs I, et al. Lagrange relaxation based method for the QoS routing problem[C]//INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE. IEEE, 2001, 2: 859-868.
- [22] Floodlight Controller. <http://www.projectfloodlight.org/floodlight>.
- [23] Tirumala A, Qin F, Dugan J, et al. Iperf: The TCP/UDP bandwidth measurement tool[J]. <http://dast.nlanr.net/Projects>, 2005.