A Dynamic Resource Allocation Model for Guaranteeing Quality of Service in Software Defined Networking Based Cloud Computing Environment

Chenhui Xu[1(✉)], Bing Chen[1], Ping Fu[2], and Hongyan Qian[1]

[1] Institute of Computer Science and Technology,
Nanjing University of Aeronautics and Astronautics, Jiangsu 210016, China
{xuchenhui_cd, cb_china, qhy98}@nuaa.edu.cn
[2] Central Washington University, Ellensburg, WA, USA
pingfu@cwu.edu

Abstract. Software Defined Network (SDN) has emerged as a promising network architecture that enables flexible management and global vision for network resource. The centralized control and programmability provide a novel approach to realize resource management for supporting end-to-end quality of service (QoS) guarantee. For real-time computing system and applications, cloud computing provider has to consider QoS metrics when they provide services for cloud user. In this paper, we propose an end-to-end QoS management framework in SDN-based cloud computing architecture. Depending on the characteristic of control plan separates from data plan in SDN, we present a QoS guaranteed approach with allocate resource dynamically for all cloud users by route optimization algorithm. Besides, we combine the queue technology to ensure high priority users request when there no candidate path calculating by routing algorithm. Numerical experiment results show that our propose framework can provide QoS guaranteed for cloud users' requirement.

Keywords: Software defined networking · Cloud computing · Quality of service · Queue

1 Introduction

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [1]. It is parallel and distributed system that provides services dynamically depending on service level agreement (SLA) of cloud computing provider and user. There are three fundamental service models offered by cloud computing provider that are Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). The objective of cloud computing is provide high available, high reliable and elastic service that supported by data center management, virtualization, ability of massive data processing,

DOI: 10.1007/978-3-319-27051-7_18

security etc. And virtualization technology is the core of cloud computing that allows user to access network resources by renting virtual machines (VMs) [2].

There are various network data on cloud as the result of Internet users have increased tremendously over the past few years. Some applications, such as multimedia, video conferencing and voice over IP (VoIP) need high bandwidth resource and computer power, for example throughput for real-time multimedia services, low latency (delay) for VoIP, or online gaming with low jitter. Cloud providers encounter several challenges when they provide services for many cloud users simultaneously [3]. One of the most important is satisfy QoS requirement for each multimedia user or each application user. In the IP network, a variety of techniques have been proposed to achieve preferable quality of service including overprovisioning, buffering, traffic shaping, the bucket algorithm, resource reservation etc. Over the past decade, the Internet Engineering Task Force (IETF) has explored several QoS (quality of service) architectures, but none has been truly successful and globally implemented.

Software Defined Networking (SDN) is a novel network paradigm that decouples the forwarding plane and control plane [4]. The most significant feature of SDN is to provide centralized control and global view of the network. Then it can realize fine-grained control for network flows and allocate the resource reasonably. It enables network as a programmable component of the larger cloud infrastructure and provides virtualization of the underlying network [5]. However, SDN has not been used extensively as the cloud computing provider. Therefore, we present a QoS management framework in SDN-based cloud computing environment, which take full advantage of centrality and programmability of SDN. And the network controller serves as service manager in cloud computing architecture, which offers resource reservation and on-demand bandwidth. OpenFlow is the communication interface between the controller and forwarding layers of SDN architecture. The forwarding layer can be composed of abundant virtual machines (VMs) acting as cloud providers. Our work is focus on SDN controller to implement bandwidth allocation and resource management for the requirement of cloud users.

We classify cloud user service traffic flows into QoS flow and best-effort flow which following shortest path originally. The QoS flow means that it needs more network resource and QoS parameters, such as bandwidth, packet loss, jitter, delay, throughput, must be guaranteed. The controller will run dynamically routing algorithms to calculate available route when the current link load exceed the load threshold. Depending on the requirement of users or application feature, we differentiate traffic flows into different level priority and combine the queue technique to guarantee the transmission of the high priority flow. Some numerical results are provided and analyzed to show that our framework can guarantee the QoS and satisfy the quality of experience (QoE) for cloud users.

The rest of this paper is organized as follows. Section 2 reviews the main work for QoS provisioning in cloud computing environment. Section 3 discusses the QoS implement framework we proposed in this paper and optimization scheme for route calculate. Section 4 presents the experiments verification for bandwidth allocation and resource reservation for cloud users. Finally, Sect. 5 concludes the research work.

2 Related Work

Internet users have shared amount of data such as multimedia data and real-time data on the cloud. Therefore, QoS is one of the essential characteristic to be achieved in the field of cloud computing and SDN. Service Level Agreements (SLAs) are established and managed between cloud service provider and cloud user to negotiate the resource allocation and service requirement [6]. However, SALs are mostly plain written documents from lawyers, which are static and published on the providers' web sites. It is restrict to achieve service satisfaction between network service provider and user dynamically.

Reference [6] presents an automated negotiation and creation of SLAs for network services through combining WS-agreement standard with OpenFlow standard, which deliver QoS guarantees for VMs in the cloud. Their approach allows customers to query and book available network routes between their hosts including a guaranteed bandwidth. Reference [7] proposes a unified optimal provisioning algorithm that places VMs and network bandwidth to minimize users' costs. They have accounted for uncertain demand by formulation and solving a two-stage stochastic optimization problem to optimally reserve VMs and bandwidth. Akella [8] proposes a QoS-guaranteed approach for bandwidth allocation by introducing queue techniques for different priority users in SDN based cloud computing environment. The flow controller selects the new path by using greedy algorithm. However, their approach is implemented in OpenVswitch [9] and they have not employed the SDN controller. Then their scheme lacks of centralized control and programmability. Besides, they will contribute much work in OpenVswitch to meet their design.

In order to implement QoS control, there are many innovation approaches have been researched in SDN. Reference [10] describes an adaptive measurement framework that dynamically adjusts the resources devoted to each measurement task, while ensuring a user specified level of accuracy. To efficiently use TCAM resources, [11] proposes a rule multiplexing scheme, which study the rule placement problem with the objective of minimizing rule space occupation for multiple unicast sessions under QoS constraints. Reference [12] proposed CheetahFlow that a novel scheme to predict frequent communication pairs via support vector machine and detect the elephant flows and rerouted to the non-congestion path to avoid congestion.

Some routing algorithms have been used to realize dynamical management of network resource. Reference [13] describes the two algorithms that are randomized discretization algorithm (RDA) and path discretization algorithm (PDA) to solve the -approximation of delay constrained least-cost routing (DCLC). Their simulations show that RDA and PDA run much faster than other route algorithm on average. Reference [14] described the distributed QoS architectures for multimedia streaming over SDN. They consider an optimization model to meet the multimedia streaming requirements in the inter-domain. However, they only employing dynamic routing to minimize the adverse effects of QoS provisioning on best-effort flows. It is possible that there are no enough available paths to be rerouted for QoS flows which will lead to poor performance for transmission of QoS flow in complex network. In this research, we select available path by routing algorithm when the current link cannot meet the users' requirement.

Particularly, we combine the queue technique to satisfy the requirement of high priority flow when there no candidate paths calculated by routing algorithm.

3 The Propose QoS Controller Architecture

The proposed QoS architecture is designed in SDN controller which provides reservation and on-demand bandwidth service. We send statistics request to forwarding devices by using OpenFlow protocol to acquire state information such as network load, delay, jitter, and available bandwidth of link, which are indispensable element to implement QoS control [15]. In order to mitigate the overload of communication between controller and switches, we create the address resolution protocol (ARP) proxy to learn MAC address. OpenFlow standard, which enables remote programming of the forwarding plan, is first SDN standard and a vital element of an open software defined network architecture. Therefore, our framework can run multiple routing algorithms simultaneously to allocate network resource dynamically in SDN controller. The design principle is that our architecture must be compatible with current OpenFlow specification which need not extra modification for switch hardware, OpenFlow protocol and end hosts. This approach ensures that our solution is practical with little extra effort to upgrade.

3.1 System Modules

The QoS framework our proposed in SDN-based cloud computing environment is shown in Fig. 1. It includes six modules to implement QoS control in SDN controller.

1. State Collector: Controller will send state request instruction periodically to OpenFlow enabled switches to collect the resource state. There are several state messages supported by OpenFlow protocol including port message, flow entry message, queue message etc.
2. State Manager: This function will process the resource statistics received from state collector. It will analysis the state massage and determine whether the link load exceeds the threshold or trigger reroute module to calculate new path.
3. Topology Finder: This module is responsible for discovering the forwarding devices and maintaining the connectivity through data received from switches. It will real-time update the topology structure when the network elements have changed.
4. Flow Classifier: This module will differentiate the traffic flow type depending on type of service (ToS) field in IPv4 header or source IP address when they get to controller. We classify traffic flows into QoS flow and best-effort flow. Furthermore, all traffic flows will be set different priority depending on cloud users' level.
5. Routing Calculation: This function run two routing algorithms in parallel depending on the performance requirements and objective of different flows. It will select the shortest path by Dijkstra algorithm for raw traffic and calculate feasible path for high priority flow according to routing optimization algorithm when their service request cannot be satisfied.

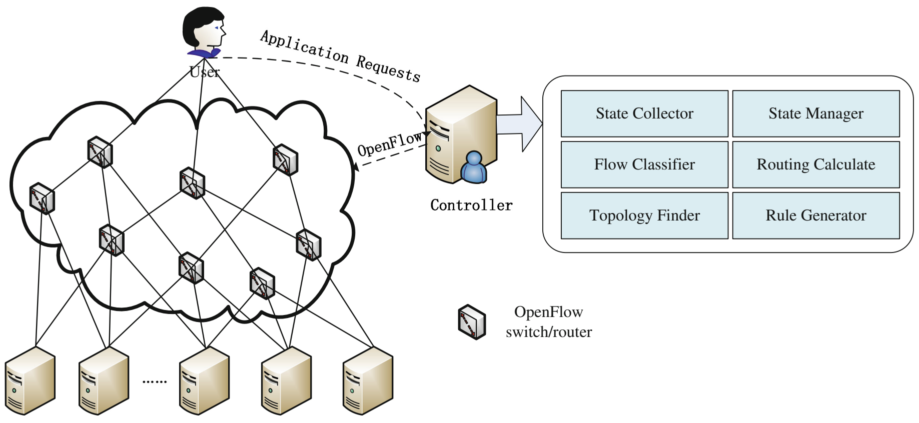

Fig. 1. System Framework

6. Rule Generator: This function will package the route and control information into flow entry and send it to forwarder devices.

In addition to this, we combine queue technique to guarantee high priority flow when there no candidate path calculated by routing optimization algorithm, which will lead to heavy congestion and packet loss. We will configure the multiple queues on forwarders to support resource reservation when the topology starts up.

To implement QoS control in SDN, we ignore the communication overhead produced between controller and forwarder devices and the extra cost from routing optimization algorithm. We assume that there exists multipath in the topology so that the rerouting module can select the candidate path. Our framework resolves three challenge problems. Firstly, we send the stats request message which supported by OpenFlow specification to the OpenFlow-enable switch to acquire the statistics that will be analyzed to determine resource scheduling. Secondly, In order to improve the efficiency of controller, we delete the original flow entry to update flow entry generated by route calculation algorithm instead of modifying the original flow entry directly. Finally, we map different level flow marked by flow classifier module into corresponding queue that had configured by OVSDB protocol. The workflow of our framework is described below:

- We will configure different level queues on each port of OpenFlow switches by using the OVSDB protocol. It is performed when the controller establishes an OpenFlow session with OpenFlow switches.
- When the new flow gets to network, the controller will choose the shortest path depending on Dijkstra algorithm supported by routing calculation module.
- Flow and port statistics are collected from the switches periodically through OpenFlow protocol. They are analyzed in state manager module to estimate the network condition.

- In order to mitigate the impact on best-effort flow, routing calculate function reroutes the feasible path for QoS flow when the state manager module detects the congestion or packet loss. The initial flow entry will be deleted before the new flow entry is installed.
- The QoS flow and best-effort flow are mapped into the different level queues configured in the step 1 when they fight the common path which cannot provide enough bandwidth for all flows.

3.2 Route Optimization

We will reroute the available path for QoS flow by using route optimization algorithm when there exist congestion on shortest path shared by QoS flow and best-effort flow. We consider the route optimization as a Constrained Shortest Path (CSP) Problem, which is the important part in select a cost metric and constrains where they both characterize the network conditions and support the QoS requirements. There are many QoS parameters such as packet loss, delay, delay variation (jitter), bandwidth and throughput to measure the performance of network. Hilmi et al. [14] described that some QoS indicators may differ depending on the type of the application. For example, multimedia applications that have sensitive requirement for total delay and video streaming are based on delay variation to guarantee the quality of experience. Therefore, in this paper we employ the packet loss and delay variations that are universal QoS indicators for a lot of service requirement as the constrained parameters.

We regard a network as a directed simple graph G (N, A), where N indicates the set of node s and A is the set of links. $R_{s,t}$ represents a set of all routes between source node s and destination node t. For any route $r \in R_{s,t}$ we define cost $f_C(r)$ and $f_D(r)$ delay variation as,

$$f_C(r) = \sum_{(i,j)\in r} c_{ij}, f_D(r) = \sum_{(i,j)\in r} d_{ij}. \tag{1}$$

Where $c_{i,j}$ and $d_{i,j}$ are cost and delay variation coefficients for the arc (i,j), respectively. The CSP problem is to minimize path cost function $f_C(r)$ of routing path r subject to a given constraint D_{max} as below,

$$\min\{f_C(r)|r \in R(s,t), f_D(r) < D_{\max}\} \tag{2}$$

We select the cost metric as follows,

$$c_{ij} = (1-\beta)d_{ij} + \beta p_{ij} \text{ for } 0 \le \beta \le 1, \forall(i,j) \in A. \tag{3}$$

The variable $d_{i,j}$ indicates the delay variation for traffic on link (i,j), $p_{i,j}$ is the packet loss measure, and β is the scale factor.

We employ the LARAC (Lagrange Relaxation based Aggregated Cost) algorithm to resolve CSP problem, which follows Hilmis scheme [16, 17]. Firstly, it will find the shortest path r_C by Dijkstra algorithm depending on path cost c. If the shortest path r_C satisfies the constrain condition D_{max}, it will be the feasible path; otherwise, it will calculate the shortest path r_D constrained by delay variation. If this path does not satisfy the delay variation constrain D_{max}, there will no feasible path in the current network and the algorithm will stop; otherwise, it will call circulation iteratively until finding the feasible path that satisfies constraint.

It is shown that LARAC is a polynomial time algorithm that efficiently finds a good route in $O\ ([n + mlogm]^2)$, where n and m are the number of nodes and link, respectively. LARAC also provides a lower bound for the theoretical optimal solution, which leads us to evaluate the quality of the result. Moreover, by further relaxing the optimality of paths, an easy way is provided to control the trade-off between the running time of the algorithm and the quality of the found paths.

3.3 Queue Technique and Policy

There exist multiple types of cloud users on the Internet, which have different needs for network resource. Then we classify the traffic flows into two types implemented by flow classifier module that are QoS flow and best-effort flow. There are many methods to differentiate the type of flows such as the type of service, application type and IP address. We will allocate priority for different level cloud users by above method which will be mapped into corresponding queues. Furthermore, QoS flow was classified as two subtypes in this research, QoS flow-1, QoS flow-2 that represent different level priority. Similarly, it also can be divided into more subtypes depending on the real situation. The QoS flows having higher level than best-effort flow will enter the higher priority queue to acquire sufficient bandwidth resource.

For every router or switch, QoS configurations are assigned to queues, which are related to ports. Each port has a number of queues that have been divided into different levels, assigned to it (The number is about 8). Packets which are going to be forwarded are first assigned to one of these queues, and then getting forwarded from that port. Different QoS parameters, such as maximum and minimum bandwidth cap are configured to these queues. Combining the queue technique in QoS control approach, it can dispose the troublesome situation that there is no candidate path calculated by routing optimization algorithm and have to satisfy the requirement of QoS flow.

4 Experiment Validation

In this section, we introduce the prototype implementation and validation for our approach presented in Sect. 3. We test the emulation experiment in Mininet 2.2.0 [18] environment that will construct the network topology. The hosts created by Mininet can be regard as cloud provider and cloud user. The SDN controller is responsible for resource management and control by connecting the Mininet environment.

Our scheme is implemented in RYU controller 3.15 [19] which is a component based software defined networking framework and it provides software components with well-defined API that make it easy for developers to create new network management and control applications. The experiment topology is shown in Fig. 2.

It consists of eight OpenFlow enabled switches to build prototype network. In this test scenario, we use OpenVswitch 2.3.1 which is a production quality multi-layer virtual switch as forwarding devices and OpenFlow 1.3 as communication protocol between controller and forwarder. We define the H1 as cloud user and H2, H3, H4 is cloud provider that will provide service for H1. We will observe the performance of our QoS control approach by measure the bandwidth, packet loss and delay variation.

In order to simulate the actual cloud user application requirement, we customize iperf network test tool to generate traffic flows [20]. In this test scheme, we employ three test flows shown in the Table 1 as cloud users' application requirement. QoS flow 1 need more network resource then it is given highest priority among the network. The best-effort traffic is the background flow to change the congestion degree in test network. To observe the direct effect of QoS control on the flow performance, we use UDP to set specified bandwidth for two QoS flows and best-effort traffic.

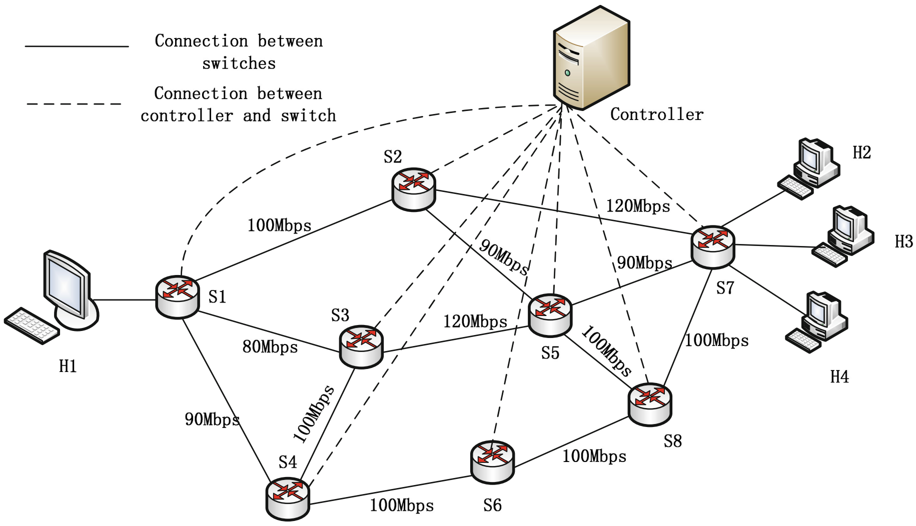

Fig. 2. Network Topology

Tab.1. Network Experiment Flows

Src	Dst	FlowType	Rate
H1	H2	QoS Flow1	50Mbps
H1	H3	QoS Flow2	40Mbps
H1	H4	Best-Effort	80Mbps

For QoS flow1, QoS flow 2 and best-effort traffic, they have different level priority and bandwidth requirement. For the purpose of achieving to congestion, we set specified bandwidth constraint for all links in test network. Firstly, H1 send the QoS flow 1 and QoS flow 2 to the corresponding destination hosts H2 and H3 while background flow started at 20 s. Initially, the controller will calculate shortest path for new flow by Dijkstra algorithm and install flow entries into OpenFlow switches. The total bandwidth of three traffic flows will exceed the link threshold when the best-effort traffic gets to network at 20 s. Then it will be congestion condition on shortest path and cannot guarantee the requirement of services without QoS control. However, our QoS control framework will reallocate the network resource for traffic flows to avoid the congestion problem occurred on shared path.

In the Fig. 3, we present a comparison of our experiment results by using the proposed approach with the ones without a use of the proposed approach. We implement our QoS control scheme at 40 s and observe the change of throughput. The throughput of two QoS flows and best-effort have increased and trended to steady state quickly. The routing algorithm calculates an available route for QoS flow1. Then QoS flow 1 can be guaranteed by selecting another lowest load path. Experiment shows that routing algorithm has not found appropriate route for QoS flow 2 because of the constraint of cost and delay. However, QoS flow 2 has higher priority than best-effort so it can be guaranteed by mapping into high priority queue. In general, our scheme ensures highest priority traffic firstly and tries to mitigate the effect on best-effort traffic.

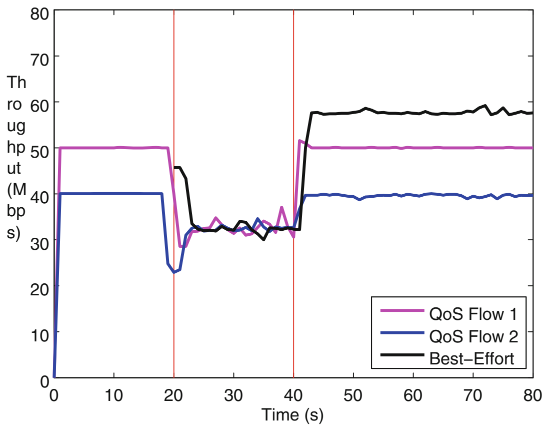

Fig. 3. Throughput for Two QoS Flows and Best-effort

The delay variation of three test flows is shown in Fig. 4. The QoS flow 1 and QoS flow 2 have suffered from great fluctuation when best-effort gets to network at 20 s. Three test flows keep jittering between 20 s and 40 s because of without our QoS control scheme. The delay variation of QoS flow 1 and QoS flow 2 has decreased after we turn on QoS control at 40 s. Although QoS flow 2 still exist fluctuation that influenced by best-effort. It is acceptable for many applications to satisfy the quality of transmission. The background traffic is limited to provide available bandwidth for QoS flow. We also evaluate the packet loss ratio among the implement for our proposed scheme in Fig. 5. Similarly, the phenomenon of packet loss has disappeared when QoS flow 1 reroute another path and best-effort is mapped into low priority queue.

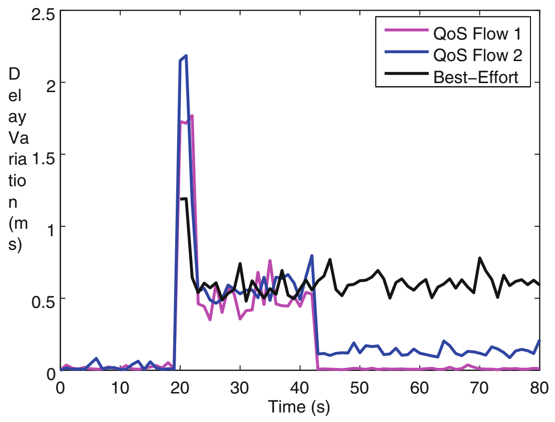

Fig. 4. Delay Variation for Two QoS Flows and Best-effort

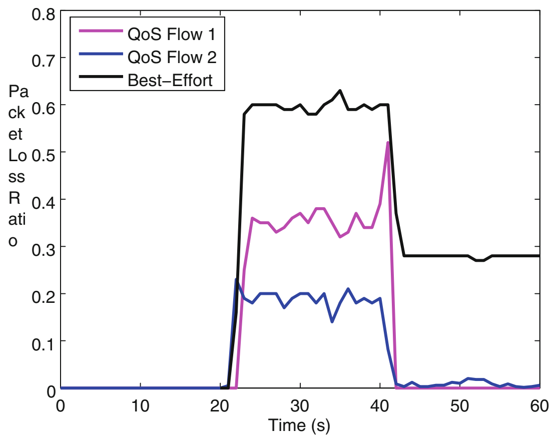

Fig. 5. Packet Loss Ratio for Two QoS Flows and Best-effort

5 Conclusion

We have studied a QoS-guaranteed approach in software defined networking (SDN) based cloud computing environment. SDN provides grate benefit for cloud computing provider to make the network easy to manage and customize. The cloud services usually need to be distinguished according to their services priority and requirement. In this paper, we differentiate the traffic flows into QoS flow and best-effort traffic for different service level. We propose the routing optimization algorithm to reroute feasible path for high priority flow when their service requirement cannot be guaranteed. Furthermore, we combine the queue technique to allocate sufficient bandwidth for high priority flow when the routing algorithm cannot find the available path. We have implemented and tested our approach in Mininet based on

OpenVswitch. Experimental results have shown that our approach can provide an end-to-end QoS guarantees for cloud user. In the future work, we will validate our QoS framework by OpenFlow physical switch in the large-scale network.

Acknowledgements. We thank the reviewer for their valuable feedback. This work was supported by the Industry-University-Research Combination Innovation Foundation of Jiangsu Province (No. BY2013003-03) and the Industry-University-Research Combination Innovation Foundation of Jiangsu Province (No. BY2013095-2-10).

References

1. Mell, P., Grance, T.: The NIST definition of cloud computing (v15). Technical report, National Institute of Standards and Technology (2011)
2. Wood, T., Ramakrishnan, K.K. Shenoy, P., Merwe, J.: Cloudnet: Dynamic pooling of cloud resources by live wan migration of virtual machines. In: 7th ACM SIGPLAN/SIGOPS international conference on virtual Execution Environments, pp. 121–132. ACM Press (2011)
3. Hoefer, C.N. Karagiannis, G.: Taxonomy of Cloud Computing Services. In: IEEEGLOBECOM Workshops, pp. 1345–1350 (2010)
4. McKeown, N., Anderson, T. Balakrishman, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: Openflow: enabling innovation in campus networks. In: ACM SIGCOMM Computer Communication Review, pp. 69–74. ACM Press (2008)
5. Tootoonchian, A., Gorbunov, S. Ganjali, Y. Casado, M. Sherwood, R.: On controller performance in software-defined networks. In: USENIX workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and services (Hot-ICE), pp. 893–898. IEEE Press (2014)
6. Korner, M., Stanik, A. Kao, O.: Applying QoS in software defined networks by using WS-Agreement. In: 6th IEEE International Conference on Cloud Computing Technology and Science, pp. 893–898. IEEE Press (2014)
7. Chase, J., Kaewpuang, R. Wen, Y.G. Niyato, D.: Joint virtual machine and bandwidth allocation in software defined network (SDN) and cloud computing environments. In: IEEE International Conference on Communications (ICC), pp. 2969–2974. IEEE Press (2014)
8. Akella, A.V., Xiong, K.Q.: Quality of Service (QoS) guaranteed network resource allocation via software defined networking (SDN). In: 12th IEEE International Conference on Dependable, Autonomic and Secure Computing, pp. 7–13. IEEE Press (2014)
9. OpenvSwitch: A production quality, multilayer virtual switch. http://openvswitch.github.io/
10. Moshref, M., Yu, M.L., Govindan, R., Vahdat, A.: DREAM: dynamic resource allocation for software-defined measurement. In: ACM SIGCOMM Computer Communication Review, pp. 419–430. ACM Press (2014)
11. Huang, H., Guo, S., Li, P., Ye, B.L., Stojmenovic, I.: Joint optimization of rule placement and traffic engineering for QoS provisioning in software defined network. In: IEEE Transactions on Computers. IEEE Press (2014)
12. Su, Z., Wang, T., Xia, Y., Hamdi, M.: CheetahFlow: towards low latency software defined network. In: IEEE International Conference on Communications, pp. 3076–3081. IEEE Press (2014)
13. Chen, S., Song, M., Sahni, S.: Two techniques for fast computation of constrained shortest paths. In: IEEE Transactions on Networking, pp. 1348–1352. IEEE Press (2008)

14. Egilmez, H.E., Tekalp, M.: Distributed QoS architectures for multimedia streaming over software defined networks. In: IEEE Transactions on Multimedia, pp. 1597–1609 IEEE Press (2014)
15. Kim, H., Feamster, N.: Improving network management with software defined networking. In: IEEE Communications Magazine, pp. 114–119. IEEE Press (2013)
16. Juttner, A., Szviatovski, B., Mecs, I., Rajko, Z.: Lagrange relaxation based method for the QoS routing problem. In: 20th Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 859–868. IEEE Press (2001)
17. Egilmez, H.E. Civanlar, S. Tekalp, A. M.: An Optimization Framework for QoS Enabled Adaptive Video Streaming Over OpenFlow Networks. In IEEE Transactions on Multimedia, pp. 710–715. IEEE Press (2013)
18. Mininet: Network Emulator. http://yuba.stanford.edu/foswiki/bin/view/OpenFlow/Mininet
19. RYU: Component-based Software defined Networking Framework. http://osrggithub.io/ryu/index.htmlMininet
20. Iperf: TCP/UDP Bandwidth Measurement Tool. https://iperf.fr/