Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2018, 5(9):368-375**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

SDN Performance Benchmarking: Techniques and Best Practices

Kodanda Rami Reddy Manukonda

Email: reddy.mkr@gmail.com

Abstract Three benchmarking tools—CBench, PktBlaster, and OFNet—are used to compare nine SDN controllers. NOX, POX, Floodlight, ODL, ONOS, Ryu, OpenMUL, Beacon, and Maestro are tested. A 2.10 GHz i7-3612QM processor with 12 GB of DDR3 RAM and Ubuntu 16.04.03 LTS is used for the virtualized examination. The study examines these controllers' latency and throughput under different network situations. Results demonstrate that controller performance varies greatly between benchmarking techniques. NOX and POX have lesser throughput than other controllers but better latency in CBench and PktBlaster. In contrast, ODL, Beacon, and Maestro have excellent throughput but increased latency in PktBlaster. The study emphasizes the importance of choosing the correct controller based on performance indicators and network needs. It also stresses the necessity for standardized benchmarking methods to compare controllers fairly.

Keywords SDN controllers, performance evaluation, latency, throughput, CBench, PktBlaster, OFNet, virtualized environment.

1. Introduction

a) Project Specification

Three benchmarking techniques are used to evaluate nine SDN controllers in a virtualized context. The study compares latency, throughput, and other performance parameters to assess these controllers' capabilities under different network settings.

b) Aim and Objectives

Aim: This study uses three virtualized benchmarking techniques to evaluate nine SDN controllers.

Objectives:

- To compare latency performance of controllers using CBench and PktBlaster.
- To evaluate controller throughput using CBench and PktBlaster.
- To analyze controller performance, including round-trip time, CPU utilization, missing flows, and OFNet flow messages sending and receiving.

c) Research Question

- Comparison of SDN controller performance in latency and throughput across diverse network situations?

d) Research Rationale

What is the issue?

The network environment and workload on SDN controllers can greatly affect their performance [1].

Why is the issue?

Understanding SDN controller performance is essential for enhancing network performance and running SDN-based networks efficiently [2].

What is the issue now?

Few studies examine numerous SDN controllers' performance in a virtualized context using different benchmarking tools. We desire to fill this vacuum in the writing by offering huge bits of knowledge on these controllers' performance.

2. Literature Review

a) Research Background

This examination analyzes SDN controllers, wired networks, Wi-Fi organizations, and displaying toolboxs for SDN Wi-Fi organizations. Performance evaluation of wired SDN networks has many examination articles:

Bholebawa et al., compared floodlight and POX controllers, [3]. Using Mininet to simulate different topologies, the authors measured round-trip time and bandwidth. Floodlight was outperforming POX. Even if Floodlight performs well, POX is suggested for its usability.

Shamim et al. benchmarked “Ryu”, “POX”, and “Pyretic” SDN controllers [4]. They recreated a wired SDN using Mininet. Performance was largely affected by round- trip time. According to this study, Pyretic outperforms Ryu and POX.

According to Fancy et al. [5], Floodlight and POX should be compared. Their criteria include throughput and latency. In several locations, controller experiments have been done. According to the authors, Floodlight outperforms POX. Floodlight requires preset memory space to execute. Python-dependent POX is the better solution here.

Throughput, dependability, wellbeing, and different factors are utilized to analyze the SDN Controllers "Guide, MuL, Mestro, POX, NOX, Floodlight, and Ryu" in [6]. Under ordinary traffic conditions, Guide, NOX, Floodlight, POX, and Ryu work, the review found. MuL and Maestro show different productivity under various settings.

Pox and RYU, two Python-based SDN controllers, are analyzed by Rastogi et al. [7]. Reenacting networks and making traffic were conceivable with Mininet. Layer 1 exchanging is better with POX, the researchers found. In any case, layer 2 exchanging seems, by all accounts, to be preferable for RYU over POX.

A couple of exploration has inspected "Programming characterized Remote organizations".

In a mimicked remote organization, Islam S. what's more, associates inspected "Ryu, POX, ONOS, and Floodlight" [8]. To pick the best, they looked at jitter and throughput. Jitter low searches in Floodlight. Be that as it may, SDN regulator throughput is fairly reliable.

A SDN-put together Wi-Fi Organization running with respect to Mininet was demonstrated to be adaptable by estimating performance in a few powerful conditions [9]. Expanding the quantity of hosts for the TCP convention diminishes data transmission and speed, while expanding the quantity of hosts for the UDP convention increments jitter however holds transfer speed. Performance investigation of SDN controllers on wired networks seems total. However, SDWN are new, and enormous scope SDN Wi-Fi networks need performance estimation using numerous SDN controllers. Cutting edge investigation proposes this.

A SDN network method for productive and smart parcel bearing change is proposed in [10]. Versatile organization bundle size, exact parcel numbers, the all-out required time stretch, QoS association limit (data transfer capacity), and the quantity of jumps (most brief way) are utilized to appraise way costs, permitting the SDN regulator to limit stream choice time [11]. A strategy is tried with steering postpone information. To find the ideal bundle deferral and way cost, the model purposes parcel size, amount, and time. In a benchmark correlation between the recommended approach and cutting edge other options, distinguishing a suitable recuperation way took a couple of milliseconds less [12] The strategy decreases bottleneck pathways and asset use, further developing goal and abstract video real time QoE. The model limits course determination postpone by 96.3%, further developing end-client fulfillment 13- 15].

b) Critical Assessment

Simulation tools are used to assess SDN controllers in wired and wireless networks. Several research articles present Floodlight, POX, Ryu, and other controller performance evaluations [16]. Due to their novelty, software-defined wireless networks (SDWN) need more research, according to the study. Additionally, an algorithm to optimize packet routing in SDN networks reduces delay time and improves user experience [17]. The study emphasises rigorous evaluation methods and suitable performance indicators for SDN controller performance assessment.

c) Linkage to Aim

SDN controller performance in wired and wireless networks is examined in the study [18]. The study analyzes Floodlight, POX, and Ryu controllers to assess their efficiency and effectiveness [19]. To understand SDN controller performance in different network contexts, this evaluation is essential.

d) Implementation purpose

SDN controllers are tested in wired and wireless networks in the study. The study uses Mininet and benchmarking tools like CBench, PktBlaster, and OFNet to measure latency, throughput, round-trip time, and CPU utilization [20]. This implementation gives empirical evidence and data-driven insights into SDN controller effectiveness to inform network design and optimization [21].

e) Theoretical Framework

The study evaluates SDN controller performance in wired and wireless networks. Previous research has compared SDN controller performance using latency, throughput, and round-trip time. To simulate network scenarios and evaluate controller performance, Mininet, CBench, PktBlaster, and OFNet are used. Intended to reveal SDN controller scalability and efficiency, particularly in network traffic management [22] The study analyses Ryu, POX, ONOS, and Floodlight controllers to provide real- world SDN deployment recommendations.

f) Literature Gap

The review tends to the absence of exhaustive performance evaluations of Software-Defined Networking (SDN) controllers in wired and remote organizations. Some studies have compared SDN controllers in wired networks, but few in wireless networks. Existing studies often use different evaluation methods or focus on a subset of controllers, making it difficult to compare results [23]. This study addresses this gap by systematically evaluating nine SDN controllers using multiple benchmarking tools to better understand their performance across network scenarios.

3. Methodology

a) Research philosophy

A positivist research philosophy was used to objectively evaluate and compare SDN controller performance using quantitative measures. Positivism emphasizes empirical evidence and scientific methods to understand phenomena.

b) Research approach

The performance of SDN controllers was assessed using a quantitative research methodology in this study. To track down examples, relationships, and patterns, quantitative exploration gathers and investigates mathematical information. In this work, the performance metrics of the controllers, including latency, throughput, round-trip time, and CPU utilization, are measured quantitatively. The methodology entails methodical measurement and analysis, yielding statistical information to bolster judgments of the controllers' performance.

c) Research design

This study used an experimental and comparative research approach. It entails utilizing three benchmarking tools in a virtualized environment to compare the performance of nine distinct SDN controllers. Because controlled trials are set up to evaluate and assess the controllers' performance metrics, the study is experimental in character. Because of their architecture, the controllers may be systematically compared under comparable circumstances to provide insight into their respective performances.

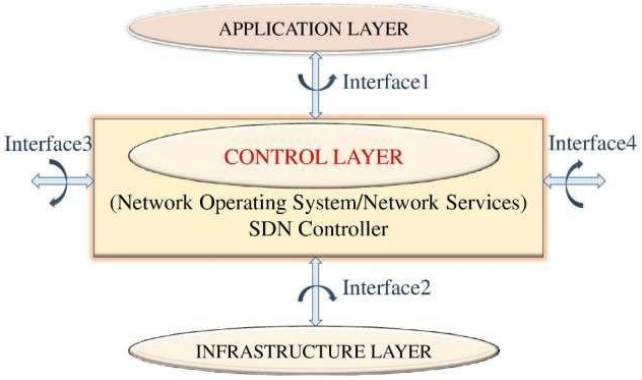

Figure 1: Research Design of the study

d) Environmental Setup

Different evaluation arrangement boundaries. Note that programmable settings in various apparatuses are not same, subsequently we attempted to make them same. Subsequent to setting boundaries, all controllers utilize similar qualities.

CBench sends nonconcurrent messages to test performance. Latency is taken care of by sending a bundle to the imitated switch and sitting tight for a reaction prior to sending another. We run 20 rounds with various quantities of emulated changes to perceive what they mean for the regulator. We test the running regulator's throughput with similar boundaries. In any case, questions are made without hanging tight for a reaction and parcels are not communicated in grouping. One execution of CBench yields a regulator's stream message limit each second. This information are the typical number of responsive switches each second in that activity.

The in-assembled TCP-based traffic copying profile gives an OpenFlow meeting between the recreated switch and regulator in PktBlaster. Since apparatus is free, emphases are restricted to 5. The nine controllers are surveyed by latency (stream establishment rate) and throughput.

e) Data Analysis and Collection Method

This study estimates the performance of nine SDN controllers (NOX, POX, Floodlight, ODL, ONOS, Ryu, OpenMUL, Reference point, Maestro) in a virtualized setting utilizing three benchmarking devices (CBench, PktBlaster, OFNet). This study examined information with CBench, PktBlaster, and OFNet. CBench sends offbeat correspondences to gauge latency and throughput. PktBlaster imitates traffic utilizing a TCP-based profile to construct an OpenFlow meeting between the copied switch and regulator. OFNet examines Full circle Time, normal stream arrangement latency, vSwitch computer chip use, regulator missed streams, and sent and got streams. These apparatuses give quantifiable information expected to think about SDN regulator performance.

4. Results

a) Latency Performance

CBench

We find two latency impacts using CBench. First, we compare latency to topology switches from 2 to 16. Interesting is the Ryu controller's little latency impact. NOX and POX similarly have little latency change as switches grow. However, controller capabilities must also be evaluated; therefore less latency does not guarantee victory. As seen in Table I, ODL constantly performs in the center and offers other features.

Table 1: Latency Performance of SDN Controllers in Various Scenarios

	NOX	Flood light	ONOS	Open MUL	Maestro	POX	ODL	RYU	BEA CON
CBench latency with varying number of switches									
2	15	20	30	45	35	33	50	56	60
4	25	35	89	90	64	65	80	84	75
8	30	45	36	89	81	45	25	30	36
16	35	41	42	43	56	58	56	60	65
CBench latency in different number of iterations (16 switches)									
2	45	36	64	60	58	54	50	37	62
4	84	42	65	75	63	67	80	48	78
8	83	45	80	45	69	78	39	76	80
16	65	56	30	22	48	59	90	45	95
PktBlaster latency in with varying number of switches									
2	85	75	66	78	85	86	34	65	70
4	30	48	65	80	95	53	42	45	65
8	45	35	66	52	63	66	25	50	31
16	36	26	36	45	66	42	26	52	25
OFNet flow setup Latency									
2	65	75	63	67	80	48	78	89	80
4	80	45	69	78	39	76	80	86	65
8	30	22	48	59	90	45	95	58	45
16	41	42	43	56	41	42	43	56	85

In the subsequent analysis, the effect of the device's performance on latency estimation is noticed. For this situation, there are 16 fixed switches; however the quantity of emphases fluctuates. The primary example to be gained from this is that it is generally essential to consider what arrangement environments mean for estimations. It could affect the results with similar boundaries, either emphatically or adversely.

PktBlaster:

PktBlaster is also used to calculate latency versus an increasing number of switches. In this test, NOX and POX have the least deferral, while Floodlight, ODL, and ONOS display the most elevated latency. In the middle are Ryu, OpenMUL, Maestro, and Guide. The crucial thing to remember in this situation is that the latency computation is not significantly affected by the number of switches.

OFNet

OFNet mimics the SDN network like Mininet to assess and report, in contrast to CBench and PktBlaster. Yield values are given against time, not a particular worth. Table 1 shows the 300-second normal of 10 cycles. No regulator has a steady example over the course of time. As the reenactment creates, introducing streams takes less time. The latency fall and spike at 180 sec is because of traffic age relic, where some traffic is made later in the recreation, requiring more streams.

Cross-Tool Analysis

This article makes a contribution by showing how different results can be obtained for the same statistic in possibly comparable network configurations. The Y-hub scale varies altogether between the three instruments, as Table 2shows. The deliberate latency for CBench is in the request for many milliseconds, though similar controllers work in under 10ms for PktBlaster. By examination, OFNet's latency estimations are in the scope of a few hundred milliseconds. The controllers with the best performance in one test system are the most un-powerful in the other. Regardless of OFNet's unmistakable topological arrangement, the noticed results show no affiliation.

b) Throughput Performance

As seen in Table 2 this statistic is only measured with CBench and PktBlaster. Although flow processing cannot be directly measured by OFNet, it can be indirectly measured by sending and receiving flow messages, as will be covered in a later section.

Table 2: Throughput Performance of SDN Controllers in Various Scenarios

	NOX	Flood light	ONOS	Open MUL	Maestro	POX	ODL	RYU	BEA CON
CBench throughput with varying number of switches									
2	75	63	67	80	48	78	85	86	96
4	45	69	78	39	76	80	40	65	88
8	52	63	45	26	38	31	26	67	74
16	53	56	95	75	73	45	69	63	70
PktBlaster throughput with varying number of switches									
2	56	69	63	45	32	18	63	66	12
4	36	33	36	18	56	63	66	67	18
8	48	59	69	64	63	45	67	66	18
16	53	66	64	56	69	67		86	26

CBench:

CBench switches communicate however many bundles as they can without a moment's delay while in throughput mode; they don't hang tight for a reaction. The examination in view of a rising number of switches is shown in Table 2. While controllers like ODL, Guide, and Maestro have up to 100 answers for each millisecond, it is tracked down that NOX, POX, and RYU keep on being the most minimal entertainers. The stream reaction pace of ONOS is recognizably higher at 400 to 500 streams/ms, despite the fact that both OpenMUL and Floodlight reliably performed well at 150 streams/ms.

PktBlaster:

When using PktBlaster for testing, the throughput metrics displayed in Table 2 indicate that the number of switches changed has little influence. Among all the controllers studied, Floodlight, ODL, and ONOS have the best performance, while NOX and POX have the lowest. For NOX, POX, and Ryu, there was a little (insignificant) drop un throughput as the number of switches rose. But even after five iterations apiece, the difference is still negligible.

Cross-Tool Analysis:

The tools differ in the throughput metric as well, in line with previous study, though not significantly. In comparison to CBench, all of the controllers often perform better in PktBlaster evaluations. In particular, ODL and Floodlight demonstrate a notable improvement in performance.

OFNet Specific Measurements

In this series of tests, we pay particular attention to the performance indicators provided by OFNet.

Table 3: OFNet Specific Measurements for SDN Controllers

	NOX	Flood light	ONOS	Open MUL	Maestro	POX	ODL	RYU	BEA CON
Average RTT Measurement									
15	26	36	33	45	96	96	45	38	68
65	87	45	42	50	72	56	36	33	85
125	69	35	35	59	78	66	45	75	48
185	67	26	69	64	55	56	52	88	59
240	89	69	66	62	58	45	15	38	62
CPU utlization of Switch Daemon									
15	36	45	58	69	64	89	96	94	52
65	66	36	59	66	74	35	48	47	36
125	45	66	36	78	89	63	36	36	42
185	50	72	56	36	33	89	66	15	22
240	59	78	66	45	75	72	36	63	69

c) Average Round Trip Time

RTT evaluation is urgent while deciding regulator organization area. It identifies regulator switch correspondence latency. Expanded RTT increments latency on the off chance that the regulator and switches are far separated. Likewise, regulator parcel handling time intricacy influences performance.

d) CPU Utilization of vSwitch Daemon

We utilize the OFNet's traffic reproduction program to send bundles and measure the vSwitch's computer processor utilization while cooperating with a regulator. Switch daemon central processor use around 30% to 40% while working a solitary strung regulator like NOX, POX, and RYU [24]. ONOS, a multi-strung regulator, utilizes 90% of its computer chip. Also, multi-stringing capacities add to's areas of strength for ONOS speed during computer chip use. Notwithstanding, vSwitches might restrict them [25].

We count the regulator's missed streams during the test. Traffic generators give stream solicitations to vSwitches, which sends solicitations to controllers and sits tight for reactions. This testing environment benchmarks SDN controllers with vSwitch responsive streams [26]. Figure 6a shows that ONOS, ODL, and Floodlight miss less streams than NOX, POX, and RYU. Due to their multi-stringing capacities, controllers perform better compared to single-strung ones [27].

5. Findings and Discussion

The nine SDN controllers tested in this study performed differently. NOX, POX, and RYU had lower latency and throughput than ONOS, ODL, and Floodlight, which had higher throughput and higher latency. Open MUL and Beacon performed similarly across measures. The effect of switch count on latency and throughput was intriguing. Other controllers performed differently depending on the number of switches [28-30]. With more switches, ONOS had higher latency but better throughput. The controller's scalability may affect its performance. The results further emphasize the need of benchmarking tool for assessing SDN controllers. Each tool's technique and restrictions affect results. Changes in switch count had less influence on PktBlaster than CBench. Standardized benchmarking methods for SDN controller evaluation are needed.

6. Conclusion

Benchmarking controller performance is difficult. We qualitatively assess 34 controllers and quantitatively benchmark and evaluate 9 controllers. We also classified controller benchmarking metrics during this process. We also analyze benchmarking tools. As per our perceptions, scarcely any controllers consent to OpenFlow 1.3 (or above) and deal vital data for execution. Most prior audits utilized basic measures and advancement objectives. The apparatuses used fluctuate extraordinarily in highlights and abilities.

7. Research Recommendations

The report suggests numerous SDN controller performance evaluation research directions. First, more controllers and benchmarking tools must be assessed to better understand their performance. Secondly, studying how network topologies and traffic patterns affect controller performance may assist find ideal configurations for varied network settings. More practical investigations in real-world network systems could validate virtualized findings. Finally, establishing benchmarking methods and metrics would improve SDN controller comparison and evaluation across studies.

8. Future Work

To improve SDN network comprehension and optimization, SDN controller performance evaluation could focus on numerous aspects. New benchmarking tools and methods to reliably quantify SDN controller performance in dynamic and complex network environments are one research route. Exploring how machine learning and artificial intelligence affect SDN controller performance could lead to more intelligent and adaptive SDN solutions. Investigating SDN controller scalability and robustness in large-scale network installations may reveal limitations and areas for development.

References

[1]. M. Casado, T. Garfinkel, A. Akella et al., "SANE: A Protection Architecture for Enterprise Networks," in USENIX Security Symposium, vol. 49, 2006, pp. 137–151.

[2]. M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "Ethane: Taking Control of the Enterprise," SIGCOMM Comput. Commun. Rev., vol. 37, no. 4, pp. 1–12, 2007.

[3]. I.Z. Bholebawa and U.D. Dalal, "Performance Analysis of SDN/OpenFlow Controllers: POX Versus Floodlight," Wirel. Pers. Commun., vol. 98, pp. 1679–1699, 2017.

[4]. S.M. Shamim, "Performance Analysis of Different Openflow based Controller Over Software Defined Networking," Glob. J. Comput. Sci. Technol. C, vol. 18, pp. 11–15, 2018.

[5]. C. Fancy and M. Pushpalatha, "Performance evaluation of SDN controllers POX and floodlight in mininet emulation environment," in Proceedings of the 2017 International Conference on Intelligent Sustainable Systems (ICISS), Palladam, India, 7–8 December 2017, pp. 695–699.

[6]. L. Mamushiane, A. Lysko, and S. Dlamini, "A comparative evaluation of the performance of popular SDN controllers," IFIP Wirel. Days 2018, 2018, pp. 54–59.

[7]. Rastogi, A.; Bais, A. Comparative analysis of software defined networking (SDN) controllers—In terms of traffic handling capabilities. In Proceedings of the 2016 19th International Multi-Topic Conference (INMIC), Islamabad, Pakistan, 5–6 December 2016; pp. 1–6.

[8]. Islam, S.; Khan, A.I.; Shorno, S.T.; Sarker, S.; Siddik, A. Performance Evaluation of SDN Controllers in Wireless Network. In Proceedings of the 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), Dhaka, Bangladesh, 3–5 May 2019; pp. 1–5

[9]. Kumar, A.; Goswami, B.; Augustine, P. Experimenting with resilience and scalability of wifi mininet on small to large SDN networks Int. J. Recent Technol. Eng. 2019, 7, 201–207.

[10]. Taha, M. An efficient software defined network controller based routing adaptation for enhancing QoE of multimedia streaming service. Multimed. Tools Appl. 2023

[11]. D. Erickson, "The beacon openflow controller," Proceedings of the second ACM SIGCOMM workshop, pp. 13–18, 2013.

[12]. S. H. Yeganeh and Y. Ganjali, "Beehive: Towards a Simple Abstraction for Scalable Software-Defined Networking," Proceedings of the ACM Workshop on Hot Topics in Networks - HotNets-XIII, pp. 1–7, 2014.

[13]. GitHub, "An Open Source SDN Controller for Cloud Computing Data Centers." [Online] Available: https://github.com/China863SDN/DCFa bric

[14]. K. Phemius, M. Bouet, and J. Leguay, "DISCO: Distributed multidomain SDN controllers," IEEE/IFIP Network Operations and Management Symposium: Management in a Software Defined World, 2014.

[15]. J. Bailey and S. Stuart, "Faucet:Deploying SDN in the Enterprise," ACM Queue, no. October, pp. 1–15, 2016.

[16]. R. Sherwood, G. Gibb, K.-k. Yap et al., "FlowVisor: A Network Virtualization Layer," Network, p. 15, 2009.

[17]. A. Tootoonchian and Y. Ganjali, "Hyperflow: a distributed control plane for openflow," Proceedings of the 2010 internet network management conference on Research on enterprise networking, pp. 3–3, 2010.

[18]. S. Hassas Yeganeh and Y. Ganjali, "Kandoo: A Framework for Efficient and Scalable Offloading of Control Applications Soheil,"

[19]. A. Kazarez, "Loom Github Page." [Online].

[20]. Z. Cai, A. Cox, and E. T. S. Ng, "Maestro: A System for Scalable OpenFlow Control," Cs.Rice.Edu, p. 10, 2011.

[21]. A. Voellmy, B. Ford, Y. R. Yang et al., "Scaling Software-Defined Network Controllers on Multicore Servers," in Proceedings of the ACM SIGCOMM Conference on Applications, technologies, architectures, and protocols for computer communication, 2012, pp. 289–290.

[22]. M. Banikazemi, D. Olshefski, A. Shaikh et al., “Meridian: An SDN platform for cloud network services,” IEEE Communications Magazine, vol. 51, no. 2, pp. 120–127, 2013.
[23]. GitHub, “MicroFlow:The light- weighted, lightning fast OpenFlow SDN controller.” [Online] Available: https://github.com/PanZhangg/Microflo w
[24]. N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "NOX: Towards an operating system for networks," ACM SIGCOMM Comput. Commun. Rev., vol. 38, pp. 105–110, 2008.
[25]. H. M. Noman and M. N. Jasim, "POX Controller and Open Flow Performance Evaluation in Software Defined Networks (SDN) Using Mininet Emulator," IOP Conf. Series Mater. Sci. Eng., vol. 881, p. 012102, 2020.
[26]. R. K. Chouhan, M. Atulkar, and N. K. Nagwani, "Performance Comparison of Ryu and Floodlight Controllers in Different SDN Topologies," in Proceedings of the 2019 1st International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE), Bangalore, India, 19–20 March 2019, pp. 188–191.
[27]. B. Xiong, K. Yang, J. Zhao, W. Li, and K. Li, "Performance evaluation of OpenFlow-based software-defined networks based on queueing model," Comput. Netw., vol. 102, pp. 172–185, 2016.