Pakistan Journal of Engineering and Technology
ISSN (Print): 2664-2042; ISSN (Online): 2664-2050
Received: 2023-11-21
Accepted: 2024-08-30
https://doi.org/10.51846/vol7iss3pp111-119

PAKJET
Pakistan Journal of Engineering & Technology

Article

Evaluation of Performance Metrics Required in Proper Selection of SDN Controller for Integrated Satellite-Terrestrial Networks

Oluwatobiloba A. Ayofe [1,2*], Mu'azu J. Musa [2], Zanna M. Abdullahi [2], Abdoulie M. S. Tekanyi [2], Aliyu D. Usman [2], Ezekiel E. Agbon [2], Emmanuel A. Otsapa [2], Agburu O. Adikpe [2], Mathew Iyobhebhe [3], and Paul Thomas Muge [4]

[1] Department of Computer Engineering Technology, Federal Polytechnic, Ede 222001, Nigeria
[2] Department of Electronics and Telecommunications Engineering, Ahmadu Bello University, Zaria 810007, Nigeria
[3] Department of Electrical and Electronics Technology, Federal Polytechnic, Nasarawa 962106, Nigeria
[4] Department of Electrical Engineering, Federal Polytechnic, N'yak Shedam 962106, Nigeria
* Correspondence: Oluwatobiloba A. Ayofe (tobiayofe@federalpolyede.edu.ng)

Abstract: SDN controllers are increasingly becoming an integral part of modern communication networks due to their ability to centrally control and manage networks from a global point of view. One of the application areas in which the SDN paradigm is gaining traction is the Integration of Satellite and Terrestrial Networks (ISTN). Considering that there are different SDN controllers with their intrinsic characteristics, which translate to different performance characteristics, it is pertinent to determine the SDN controllers that will be most suitable for the specific application area. This study embarked on evaluating selected SDN controllers that will help in making informed selection decisions for suitability in a future ISTN architecture. Five SDN controllers were subjected to evaluation and they are Open Network Operating System (ONOS), Opendaylight (ODL), Floodlight (FL), Ryu, and Pox. These controllers are evaluated based on throughput and latency metrics, which are considered some of the factors that could influence the performance of the interworking of both terrestrial and satellite networks. The controllers were subjected under various topologies and traffic densities which are (i) single topologies under no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic conditions (ii) linear topology under no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic conditions (iii) tree topology under no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic conditions (ii) linear topology under no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic conditions (iii) tree topology under no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic conditions. Under all conditions, ONOS shows superiority in terms of throughput while Ryu shows superiority in terms of synchronization time. Focusing specifically on the tree topologies, considering it follows the same pattern with the single and linear topologies, the ONOS exhibited throughput of 815flow/ms under no cross traffic and throughput of 807flow/ms under both half-bandwidth and high-bandwidth band cross traffic. Ryu exhibited a synchronisation time of 19ms, 23ms, and 26ms under the no cross traffic, half-bandwidth cross traffic, and high-bandwidth traffic cross-traffic conditions, respectively. Observing from the values for both the throughput and synchronisation time metrics under all the topologies, it can be inferred that a number of hosts have more impact on the metrics rather than traffic volume, and this could be attributed to caching mechanisms like Address Resolution Protocol (ARP) wherein new traffic from known hosts can quickly be processed compared to new traffic from new hosts. In summary, the ONOS controller can be considered to offer good performance in an ISTN.

Keywords: 5G, Integrated Satellite, Terrestrial Networks, Performance Metrics, SDN Controllers

1. Introduction

1.1. Background and Motivation

Software-Defined Networking (SDN) [1, 2] is a paradigm that revolutionizes communication networks by enabling flexible control and management of network functions. When combined with Network Function Virtualisation (NFV) [3], SDN significantly reduces the cost of network service deployment through efficient provisioning. In traditional communication networks, the management plane (M), control plane (C), and the data (D) or forwarding (F) plane are coupled together. This makes control and management very rigid. The SDN controller in a bid to address this rigidity, decouples these planes. These enables centralisation of control and flexible management of underlying data forwarding equipment. The difference between the traditional networking system and SDN enabled systems is diagrammatically depicted in Figure 1.

As shown in Figure 1, a network device such as a router (R) in a traditional network architecture contains all the M, C, and F planes, thus management and how traffic would be routed will be configured in each of the routers. This results in cumbersomeness in provisioning. In the SDN-enabled

network, the M and C planes is decoupled from the router thereby making the router perform only data forwarding. Thus, the *C* and *M* planes have a global control and management of routers which makes provisioning to be done in a central point. This brings about efficiency and eliminate multiple error instances.

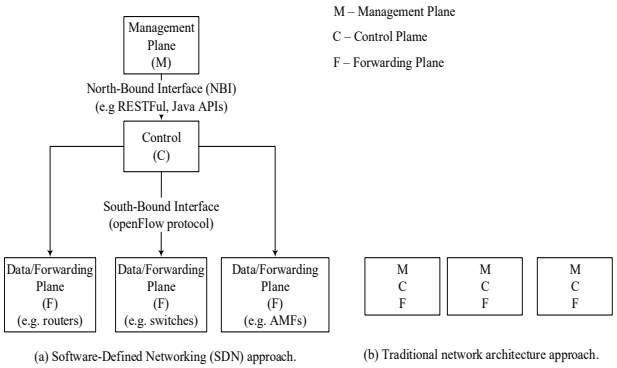

Figure 1. SDN Enable systems versus traditional network architecture.

Following the success of Software-Defined Networking (SDN)-enabled architecture in data centers, the 5G network has also achieved success in its implementation, offering flexible network management and global resource control. Recently, to meet the demand for ubiquitous and resilient connectivity in 5G, there have been proposals to integrate satellite networks with terrestrial 5G networks. However, this integration presents unique challenges that must be addressed for successful implementation. One key challenge stem from the fundamentally different architectures and protocols used in satellite and terrestrial systems [4]. For example, satellites rely on delay-tolerant networking due to intermittent connectivity [5], while terrestrial systems assume continuous connectivity. These divergent assumptions lead to interoperability issues. Additionally, satellite networks introduce longer transmission delays due to space propagation [6], while terrestrial networks are engineered for low-latency communication. Satellite networks typically have lower bandwidth and higher bit error rates than modern terrestrial networks [7], further complicating end-to-end optimisation. The incompatible underlying protocols and divergent evolution [4] of these two network systems also pose interoperability challenges.

To enable seamless integration, it is crucial to reconcile the differences between satellite and terrestrial networks. The integration system will have to address the long propagation delays, intermittent connectivity, and asymmetric link capacities [8, 9]. SDN has emerged as a promising approach to connect these divergent networks through functions such as protocol translation [10]. The introduction of Network Function Virtualization (NFV) can reduce or eliminate the need for specialized hardware, leading to cost savings.

The centralised control plane of SDN can help facilitate resource allocation across the integrated network to achieve reliability and efficiency. However, this requires advanced SDN controllers capable of accommodating the unique demands of satellite systems.

While the SDN/NFV paradigm has been applied to Integrated Satellite-Terrestrial Networks (ISTN) in various capacities, such as softwarerisation of satellite hardware in Network Control Centres (NCCs), the choice of the most appropriate SDN controller for ISTN has not been thoroughly considered. Selecting the right component with the necessary features is crucial for system performance. This article evaluates various SDN controllers and their suitability for different contexts. Therefore, it is essential to assess controller characteristics based on specific metrics to determine the most suitable SDN controller for a particular application area.

This article aims to evaluate the performance of selected SDN controllers to inform decision-making regarding their suitability for ISTN. While the primary focus is ISTN, the insights gained from this study can be leveraged to make informed controller selection decisions in other SDN applications.

1.2. Literature Review

Various studies have focused on assessing the performance of SDN controllers in different application areas. Metrics used to evaluate these performances include synchronisation time or latency, throughput, and similar measures. Latency is the time it takes to send a packet-in message to the controller and receive a response [11, 12]. Throughput is the rate at which a controller processes flow requests, measured by number of packet-in messages sent to controller and corresponding packet-out messages received per unit time [11].

While some research evaluates the performance of a single controller, others consider multiple controllers. Reference [11] assessed the performance of an ODL controller, testing its latency and throughput capabilities in a data center setting using the CBench tool. They compared these results with another SDN controller, FL, finding that ODL did not demonstrate superior performance at the time. Similarly, [13] evaluated the performance of controllers such as MUL, Beacon, and Maestro based on latency and throughput in data center and IoT environments. Reference [14] conducted evaluations to address the need for updated capabilities of controllers with improved features and performance. The authors sampled 4 open-source SDN controllers and evaluated them based on latency and throughput, primarily in the context of data centre networks. [12] provided detailed qualitative comparisons between 34 different SDN controllers and quantitative evaluations of nine controllers, considering latency and throughput as metrics. However, their study focused on a data centre scenario under less stressful conditions, relying solely on fake packets generated by CBench. [15] is one of the recent works that delved into performance evaluation of SDN controllers, measuring latency, bandwidth, and throughput, but only assessed these metrics on one controller, Ryu. Another recent study by [16], performed evaluations in a wireless environment, albeit under stationary conditions, using only a simple linear topology.

In light of the above, this research builds upon existing studies to evaluate SDN controllers, specifically considering

the context of ISTN (Intelligent Software-Defined Transport Network) requirements and expectations.

1.3. Contribution

While numerous studies have conducted performance evaluations of SDN controllers across various application domains, the context of Intelligent Software-Defined Transport Networks (ISTNs) remains relatively unexplored. This research addresses this gap by evaluating SDN controllers within the ISTN framework, specifically accounting for the data volumes and node quantities characteristic of satellite constellation networks.

In contrast to previous studies that relied solely on synthetic data generated by the CBench testing tool, this investigation incorporates real-world network traffic. Specifically, WhatsApp calls and HTTP traffic were captured using Wireshark and injected into the test environment. This additional traffic, termed "cross traffic," was provisioned at varying intensities to occupy minimal, half, and significant portions of the set link bandwidth. This approach provides a more realistic simulation of network conditions in ISTNs.

Furthermore, this study extends beyond the simple linear topologies often used in previous research. Instead, a complex tree topology was implemented, with the complexity further enhanced through the manipulation of depth and fan-out configurations. This sophisticated network structure more accurately represents the intricate interconnections present in satellite constellation networks.

By combining real-world traffic patterns with complex network topologies, this research offers a more comprehensive and applicable evaluation of SDN controller performance in the context of ISTNs. The findings from this study contribute valuable insights into the scalability and efficiency of SDN controllers under conditions that closely mimic those encountered in advanced satellite communication networks.

2. Materials and Methods

There are three topologies that are simulated to measure some key performance metrics. The metrics considered in this article includes synchronisation time and throughput. A single, linear, and tree topologies is setup in a Mininet emulator [17]. Since in a practical situation there would be more nodes and request traffics towards an SDN controller, a more complex node-loaded topology is showcased, and the performance metrics for selected controllers are observed. The configurations for the three topologies are depicted in Figure 2 through Figure 4 for the three topologies. The number of nodes is chosen by arbitrarily increasing the number of nodes considered in the work of Bholebawa and Dalal [18].

Table 1. Various SDN network topologies.

Topology	Number of Nodes	
	Switch	Nodes
Single	1	10
Linear	10	15
Tree (depth=3, fan-out=4)	17	38

In the single topology, there is only one switch connected to a single SDN controller and 10 hosts, h1 through h10, are connected to the switch. This topology is depicted in Figure 2.

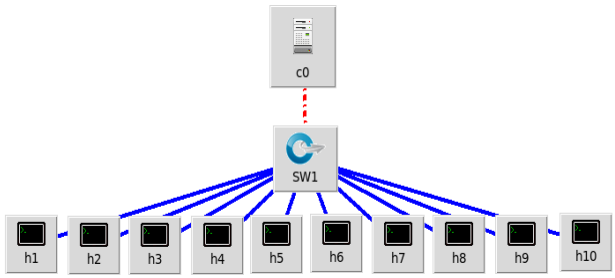

Figure 2. Single switch topology with ten (10) hosts.

In the linear topology, there are 15 switches, SWI through SW15, each of which has a connection to the same SDN controller, and each switch is connected to a single host. Also, the switches SW1 through SW15 are connected to each other serially. The diagram for this topology is shown in Figure 3.

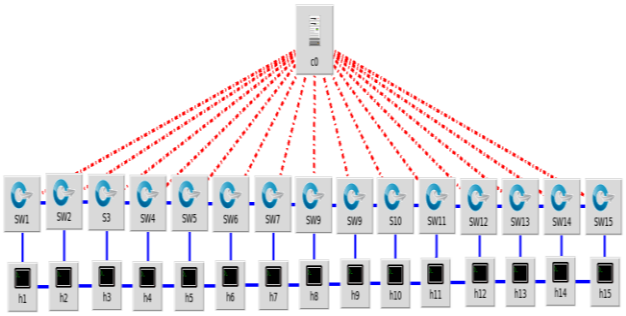

Figure 3. SDN-controlled linear topology with 15 switches and 15 hosts.

Lastly, the tree topology is a more complex topology consisting of 17 switches and 38 hosts. Figure 4 depicts this topology which has three levels of depth. Level 1 consists of only switch S1, level 2 consists of switches S2 through S5, and level 3 consists of switches S6 through S17. Level 3-depth switches are called the access layer switches and are each connected to three hosts. The level 2 switches are called the distribution layer switches, which are fan-outs from the layer 1 switch called the core layer switch.

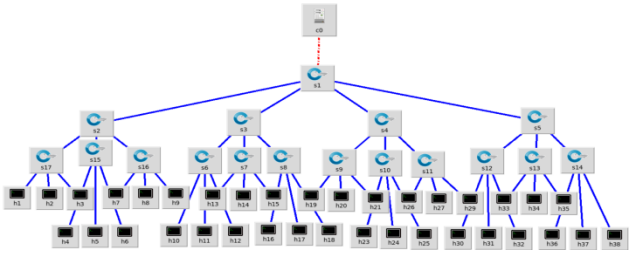

Figure 4. SDN controlled tree topology network.

The performance of the 5 controllers is observed for the three topologies under different conditions. The first condition involves traffic generated with no cross traffic, the second

involves traffic generated with half of the bandwidth crossed with application traffic, and the third involves traffic generated with high-bandwidth cross traffic. In each scenario, simulation run of 30 was made and the synchronisation time and throughput for each controller were recorded.

The 5 SDN controllers selected are considered based on the following criteria:

i. Their popularity among research scholars and industry.
ii. Support for OpenFlow protocol in the South Bound Interface (SBI).
iii. Ability to support clustering
iv. Open source-ness.
v. Openstack support which allows an SDN controller to be operationalised through the cloud.
vi. Support for popular programming language such as Python, C++, and Java.

2.1. Capturing of Cross Traffic

The cross-traffic used to assess the performance of the SDN controllers are Skype and WhatsApp video calls, as well as Skype and WhatsApp voice calls. The least considered cross traffic is Hyper Text Transfer Protocol (HTTP) data which is primarily from web browsing. These cross-traffic are obtained using a tool called Wireshark [19]. Wireshark is a tool used to capture and analyse traffic passing through a Network Interface Card (NIC) of a workstation.

Since WhatsApp calls can only work on mobile devices and it is impossible to use Wireshark to capture traffic on a mobile phone, the mobile device using the WhatsApp application is made to access the internet through a workstation. This is achieved by utilising the hotspot functionality provided by Windows 10 Operating System (OS) [20], where the mobile device is connected to it. This allows any traffic, including WhatsApp traffic emanating from the mobile node, to be captured from the hotspot interface that the mobile device is connected to.

2.2. Performance Measurement

Considering the fact that theoretical comparison based on features and properties of a controller cannot give true performance indices of the controller, it is essential for live deployment and benchmarking to take place for true evaluation. For the 5 controllers under consideration in this work, measurements for latency (sync time) and throughput are conducted.

With the aid of a popular benchmarking tool called CBench [2], the performance metrics of the SDN controllers are observed. CBench test performance by sending asynchronous messages. About 1000 iterations are executed on the different specifications of the Software-Defined Network (SDN) controllers in order to observe quantitatively their behaviour. The sync time and throughput of the considered controllers are observed under the single, linear, and tree topologies using three different scenarios of no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic.

3. Results

Considering the fact that theoretical comparison based on features and properties of a controller cannot give true performance indices of the controller, it is essential for live deployment and benchmarking to take place for true evaluation, for the five different controllers under consideration in this work which are Ryu, Pox, ONOS, FL, and ODL, measurement for latency (synchronisation time) and throughput are conducted.

With the aid of a popular benchmarking tool called CBench [2], the performance metrics of the SDN controllers are observed. CBench test performance by sending asynchronous messages. About 1000 iterations are executed on the different specifications of the Software-Defined Network (SDN) controllers in order to observe quantitatively their behaviour. The synchronisation time and throughput of the considered controllers are observed under the single, linear, and tree topologies using three different scenarios of no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic. This metric is obtained at different instances of time and the mean value is obtained as expressed in (1).

$$p_m = \frac{\sum_{i=i}^{n} p}{n} \tag{1}$$

where: p_m is the mean value obtained for performance metric p over n number of measurements.

To measure the dispersion of the obtained data, the standard deviation, p_{sd}, of the measurements are taking expressed in (2).

$$p_{sd} = \sqrt{\frac{(p - p_m)^2}{n}} \tag{2}$$

An error margin $E_{\leftrightarrow}$ is then obtained using (3).

$$E_{\leftrightarrow} = z \times \frac{p_{sd}}{\sqrt{n}} \tag{3}$$

where: z is known as the z-score that can be obtained with a lookup based on a confidence level table which can be found in [21].

3.1. Single Topology

For the single topology, the metric observed under no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic conditions are as shown in the
Table **2** through Table 4.
Table **2** shows the values obtained for synchronisation time and throughput under the no cross traffic condition in a single topology, for each of the highlighted controllers.

Table 2. Average synchronisation time and throughput for the controllers under no cross-traffic condition in a single topology of Figure 2.

Controllers	Synchronisation Time (ms)	Throughput (flow/ms)
ONOS	75	800
Pox	3	6
Floodlight	85	120
Opendaylight	43	103
Ryu	5	12

The values in Table **2** are compared with the values obtained for the half-bandwidth cross traffic condition, as shown in Table 3. This would help to infer the controller behaviour under different condition. Figure 5 shows the plots of values recorded in Table **2**.

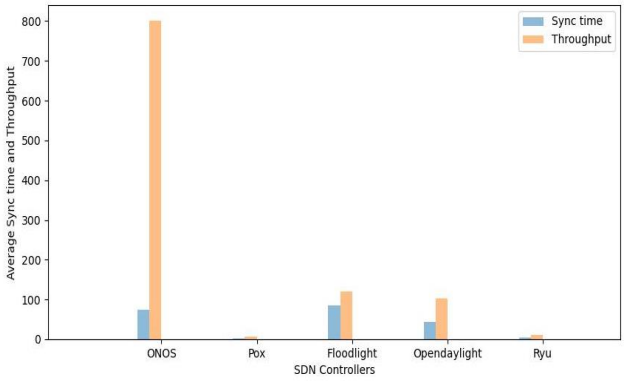

Figure 5. Performance of SDN controllers in a single topology no cross traffic condition.

As shown in Figure 5, the ONOS controller exhibited low synchronisation time and high throughput. The Pox controller exhibited low values for all the metrics while the FL controller presented a slightly higher synchronisation time compared to ONOS with a very low throughput relative to ONOS. ODL presented low values of synchronisation time compared to ONOS and Floodlight but higher than Pox and Ryu. Also, the ODL offers inferior performance in the throughput relative to ONOS but better than other controllers. Ryu offers poor performance in all the metrics and its metrics has close similarities with that of Pox.

For the scenario subjected to half-bandwidth cross traffic, the values obtained are given in Table 3. Comparing these values with the ones in Table **2**, the synchronisation time of ONOS, Pox, and ODL remains unchanged while an insignificant increase is observed for floodlight and Ryu. The throughput of ONOS, Floodlight, and ODL also stayed indifferent, while that of Pox and Ryu experienced a drop.

Table 3. Average synchronisation time and throughput for the controllers under half-bandwidth cross traffic condition in a single topology of Figure 2.

Controllers	Synchronisation Time (ms)	Throughput (flow/ms)
ONOS	75	800
Pox	3	5
Floodlight	86	120
Opendaylight	43	103
Ryu	6	10

The plot for the values shown in Table 3 is depicted in Figure 6.

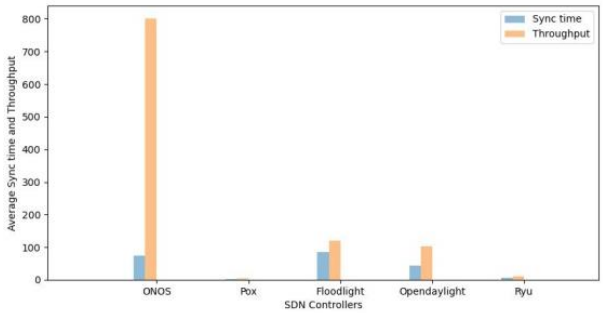

Figure 6. Performance of SDN controllers in a single topology under half-bandwidth cross traffic condition.

The plot in Figure 6 exhibits similar characteristics to the one in Figure 5. With respect to synchronisation time, the Ryu and Pox controllers offer the least and thus considered the best in this regard. ODL and FL are ranked third and fourth, respectively, while ONOS suffers the worst performance. With respect to throughput, the ONOS controller is recorded to have the highest followed by FL and ODL. Very low throughput values are observed for Pox and Ryu.

Table 4 shows the values observed under high-bandwidth cross traffic in the single topology setup and a comparison is made with the values recorded in Table 3.

Table 4. Average synchronisation time and throughput for the controllers under high-bandwidth cross traffic condition in a single topology of Figure 2.

Controllers	Synchronisation Time (ms)	Throughput (flow/ms)
ONOS	81	800
Pox	4	6
Floodlight	88	120
Opendaylight	49	103
Ryu	12	12

As shown in Table 4, the synchronisation time of ONOS increases with a margin of 6 with respect to half-bandwidth cross traffic and no cross-traffic condition. The synchronisation time of Pox, FL, ODL and Ryu under this condition increases with a margin of 1, 2, 6, and 6, respectively with respect to half-bandwidth traffic condition. The plots of the values in the table are shown in Figure 7.

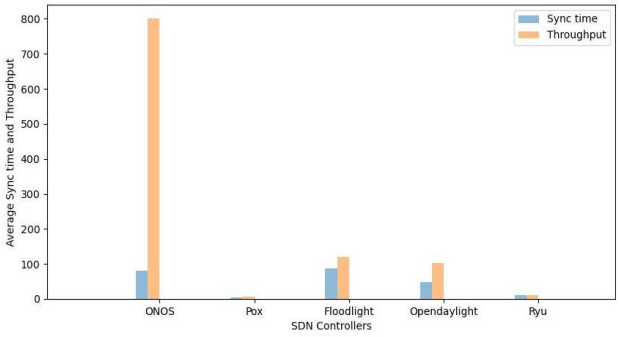

Figure 7. Performance of SDN controllers in a single topology under high-bandwidth cross traffic conditions.

As it can be observed in Figure 7, the plot shows the same pattern as the plots in Figure 5 and Figure 6. Overall, as observed from the

Table **2** through Table 4 and the plots in Figure 5 through Figure 7, the synchronisation time of SDN controller shows no significant difference between when there is no cross traffic and when cross traffic is introduced. When compared to the no cross traffic condition, the ONOS, FL, and ODL experience no drop in throughput when cross traffic is introduced. However, the Python-based controllers Ryu and Pox show a slight decrease in throughput. At high-bandwidth cross traffic, the synchronisation time for all the controllers increase compared to the no-cross traffic and half-bandwidth cross traffic scenarios. Interestingly, the throughput under this condition remains the same for the Java based controllers, that is, ONOS, FL, and ODL.

3.2. Linear Topology

For the linear topology, the metric observed under no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic conditions are observed as shown in Table 5 through Table 7. The metric obtained for the no cross traffic scenario of the linear topology are detailed in Table 5.

Table 5. Average synchronisation time and throughput for the controllers under no cross-traffic condition in a linear topology of Figure 3.

Controllers	Synchronisation Time (ms)	Throughput (flow/ms)
ONOS	75.5	800
Pox	3	5
Floodlight	87	120
Opendaylight	43	103
Ryu	6.5	9.5

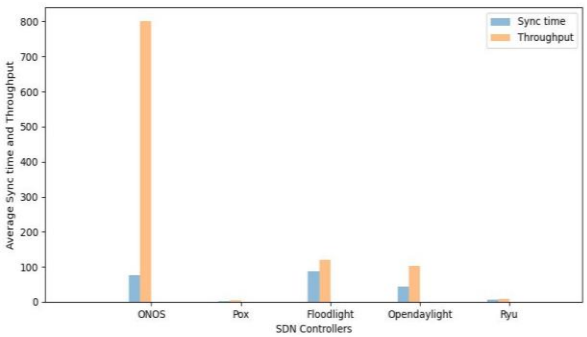

Figure 8. Performance of SDN controllers in a linear topology under no cross-traffic condition.

As shown in Table 5, the synchronisation time of the ONOS, Pox, FL, ODL, and Ryu have relative values with that of the single topology under no cross traffic and half-bandwidth cross traffic but less compared to that of the high-bandwidth cross traffic of the single topology. In the same vein, the throughput observed for all the controllers has similar values compared to the single topology under no cross-traffic condition. Table 6 details the values recorded for synchronisation time and throughput under the half-bandwidth cross traffic in the linear topology. The plots of these values are shown in Figure 8.

The plots in Figure 8 exhibits similarities with the plots shown in the single topology. The ONOS ranks the best in terms of throughput, followed by FL, ODL, Ryu, and Pox, accordingly. The Pox and Ryu exhibit very low synchronisation time as is the situation for single topology scenarios. ODL, ONOS, and FL are ranked well after Pox and Ryu.

For the half-bandwidth cross traffic scenario of the linear topology, the metrics obtained are shown in Table 6.

Table 6. Average synchronisation time and throughput for the controllers under half-bandwidth cross traffic condition in a linear topology of Figure 3.

Controllers	Synchronisation Time (ms)	Throughput (flow/ms)
ONOS	83.5	850
Pox	4	6
Floodlight	91	133
Opendaylight	51	127
Ryu	7	12

As shown in Table 6, there is a tangible increase in the synchronisation time of ONOS when compared to its counterpart in single topology. The Pox, FL, ODL, and Ryu also record higher synchronisation time against both the half-bandwidth cross traffic scenario of single topology and no cross-traffic scenario of linear topology. The throughput recorded in this scenario also increase across all controllers as against no cross-traffic scenario of the linear topology but that of Pox and Ryu are not being significant. The graphical representation of these values is depicted in Figure 9.

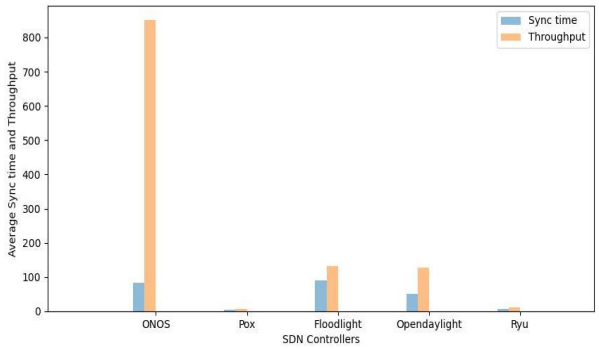

Figure 9. Performance of SDN controllers in a linear topology under half-bandwidth cross traffic condition.

As it can be observed, the graph in Figure 9 is graphically similar to previous plots. In line with previous plots, the ONOS is the best in terms of throughput and the fourth best in terms of synchronisation time. The Pox controller offers the least synchronisation time and thus offers the best performances for the metrics compared to the rest. However, it exhibits very poor throughput. The FL controller is shown to be the least-performed in terms of synchronisation time, while ranking the second best in terms of throughput after ONOS. Ryu ranks second in terms of synchronisation time and exhibits very poor values for throughput.

For the high-bandwidth scenario of the single topology, the metric measured are recorded in Table 7.

Table 7. Average synchronisation time and throughput for the controllers under high-bandwidth cross traffic condition in a linear topology of Figure 3.

Controllers	Synchronisation Time (ms)	Throughput (flow/ms)
ONOS	84	850
Pox	5	8
Floodlight	91	133
Opendaylight	51	127
Ryu	8	13

Comparing the values recorded in Table 7 with the ones in Table 6, the synchronisation time and throughput all increase for all controllers. The plots of the values obtained for this scenario is shown in Figure 10.

Under the high-bandwidth cross-traffic condition of the linear topology as shown in Figure 10, the ONOS controller, once again, leads in terms of throughput while FL, ODL, Ryu, and Pox accordingly rank behind ONOS. With respect to synchronisation time, Pox and Ryu offer the best synchronisation time performance, while ODL, FL, and ONOS follow next.

Generally, in the no-cross traffic scenario, the synchronisation times have close value with half-bandwidth cross traffic condition of the single topology. Similarly, for the throughput, there is insignificant drop when compared to the half-bandwidth cross traffic scenario presented under single topology, especially for Pox and Ryu.

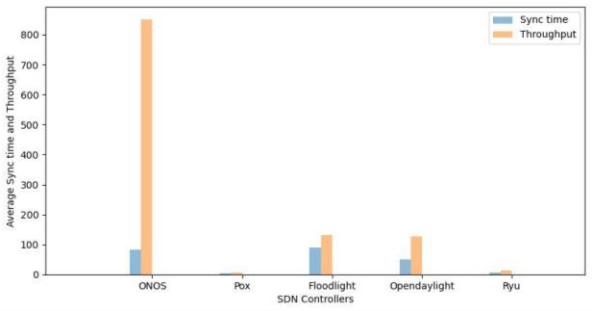

Figure 10. Performance of SDN controllers in a linear topology under high-bandwidth cross traffic condition.

In the half-bandwidth cross traffic condition, there is a slight rise in synchronisation time compared to the single topology high-bandwidth cross traffic for all controllers. There is recorded throughput increase for ONOS, FL, & ODL. Pox and Ryu shared similar throughput values with that of the high-bandwidth cross traffic in the single topology. In the high-bandwidth cross traffic, there is insignificant difference in the synchronisation time compared to that of the half-bandwidth cross traffic of the same topology. However, the python-based controllers, Pox and Ryu, exhibits slight increase in throughput. The throughput for the ONOS, FL, and ODL are relatively the same with those of the half-bandwidth cross traffic scenario of linear topology.

3.3. Tree Topology

For the linear topology, the metric observed under no cross traffic, half-bandwidth cross traffic, and high-bandwidth cross traffic conditions are recorded in Table 8 through Table 10. Table 8 shows the metric recorded for the no cross traffic scenario of the tree topology.

Table 8. Average synchronisation time and throughput for the controllers under no cross-traffic condition in a tree topology of Figure 4.

Controllers	Synchronisation Time (ms)	Throughput (flow/ms)
ONOS	97	815
Pox	23	8
Floodlight	104	126
Opendaylight	65	118
Ryu	19	10

Comparing Table 8 with the values in Table 7, the synchronisation time increases significantly for all controllers. The throughput of ONOS, FL, and ODL significantly decline, while that of Pox and Ryu relatively stay the same. The plots of the values in Table 8 is depicted in Figure 11.

In the previous scenarios under single and linear topologies, there is no obvious graphical view to indicate any significant difference in the plotted values.

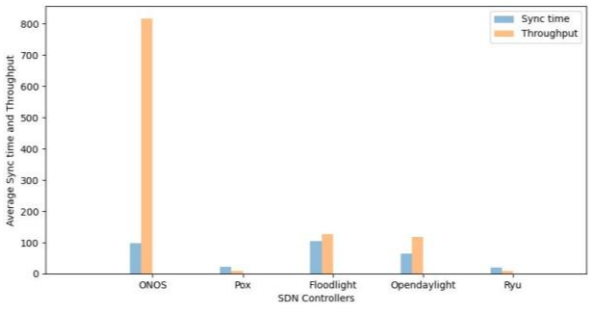

Figure 11. Performance of SDN controllers in a tree topology under no cross-traffic condition.

Here on the tree topology for the no cross traffic scenario as shown in Figure 11, there is an obvious rise in the bars representing the metrics, though, they still follow the same pattern as the previous plots. The ONOS, like in the previous scenarios and topologies, has the highest throughput. It ranks fourth in terms synchronisation time. The Ryu and Pox controllers offer the best of synchronisation time and at the same time, they both offer the poorest throughput performance. The FL controller is ranked the second best after ONOS with the ODL slightly behind, but the ODL has better synchronisation time performance than FL.

Table 9 records the metric values obtained for the half-bandwidth condition of the tree topology.

Table 9. Average synchronisation time and throughput for the controllers under half-bandwidth cross traffic condition in a tree topology of Figure 4.

Controllers	Synchronisation Time (ms)	Throughput (flow/ms)
ONOS	102	807
Pox	26	8
Floodlight	105	122
Opendaylight	69	101
Ryu	23	9

Comparing the values in Table 9 with the ones in Table 8, an increase is experienced in the synchronisation time for all the controllers. Also, there is a drop in throughput performance for ONOS, FL, ODL and an insignificant drop for Ryu, while Pox maintains the same throughput value. Figure 12 depicts the plots of the values in Table 9.

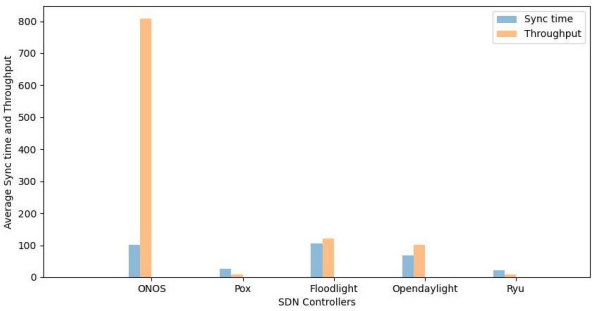

Figure 12. Performance of SDN controllers in a tree topology under half-bandwidth cross traffic condition.

As depicted in Figure 12, there is a drop in the throughput performance of the ONOS controller with increased synchronisation time, nonetheless, it still maintains its first rank position among other controllers. Its synchronisation time ranks the fourth best. Pox and Ryu continue to possess relatively the same synchronisation time and throughput. FL has the second-best throughput value followed by ODL. ODL and FL ranks third and fifth, respectively for synchronisation time.

Finally, Table 10 details the values recorded for the high-bandwidth cross traffic scenario of the tree topology. The values in Table 10 portray similar situations as in Table 9, where the synchronisation time drops for all controllers. Unlike in Table 9, where the Pox value has maintained the same value with respect to the no cross traffic condition of the tree topology, the Pox alongside other controller experienced drop in throughput performance. Figure 13 shows the plots for the values shown in Table 10.

Table 10. Average synchronisation time and throughput for the controllers under high-bandwidth cross traffic condition in a tree topology of Figure 4.

Controllers	Synchronisation Time (ms)	Throughput (flow/ms)
ONOS	108	807
Pox	30	6
Floodlight	106	117
Opendaylight	73	103
Ryu	26	9

The values in Table 10 portray similar situations as in Table 9, where the synchronisation time drops for all controllers. Unlike in Table 9, where the Pox value has maintained the same value with respect to the no cross traffic condition of the tree topology, the Pox alongside other controller experienced drop in throughput performance. Figure 13 shows the plots for the values shown in Table 10.

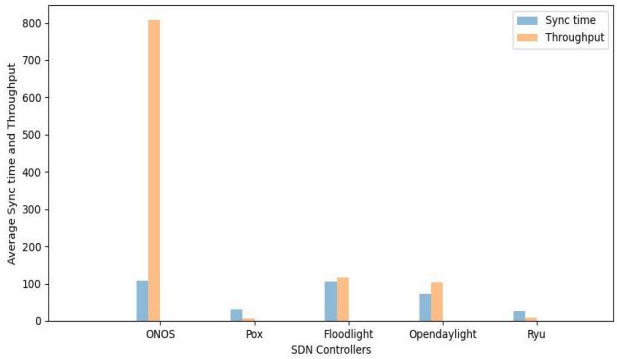

Figure 13. Performance of SDN controllers in a tree topology under high-bandwidth cross traffic condition.

Finally, the plots in Figure 13 show similar patterns with previous plots for various scenarios under single and linear topology. The performance metrics in the tree topology exhibit significant difference compared to the ones observed in single and linear topology, whether under no-cross traffic, half-bandwidth or high-bandwidth cross traffic condition. In the no-cross traffic condition, all controllers experience a significant rise in synchronisation time, while there is significant drop in throughput. In a nutshell, the synchronisation time and throughput for the tree topology experience significant regression with respect to the linear and single topologies for all traffic conditions.

4. Discussion

Based on the overall behaviour of the controllers with respect to the metrics, especially throughput, it is observed that these metrics are affected majorly by an increase in both the volume of traffic and number of hosts rather than on increase in traffic volume. This is contrary to the believe that increased traffic volume would be a major factor that would affects these metrics. Possible reason can be based on the fact that traffic from different applications coming from the same source host would have had their information cached and thus would not require any effort to perform a node location process such as Address Resolution Protocol (ARP) [22].

Judging from all the experiments undertaken under single, linear, and tree topologies, it can be observed that the Java-based SDN controllers, such as ONOS, ODL, and FL, do well in throughput, while the Python-based controllers, such as Ryu and Pox, do well in synchronisation time. While the Java-based controller could be recommended for use in an ISTN system, the cluster of the different controllers is highly recommended. The ISTN systems can greatly benefit from the heterogeneous clustering of SDN controllers, where they can leverage the unique characteristics of, say, ONOS and Ryu SDN controllers. However, this heterogeneous clustering will require developing a unique East-West protocol where they can cooperatively offer control functions, and in such a way, load balancing can be done based on the individual strengths of each member controller cluster.

5. Conclusion

In this article, a brief introduction about the SDN paradigm is introduced and different areas where it has been exploited. In aim to make informed decision for the application area of ISTN, this article present observations for five SDN controllers which are selected based on certain criteria. The controllers are observed for their latency or synchronisation time and throughput efficiency. This will allow to make choice of SDN controller to be used while considering requirement of the ISTN and other application areas.

Because ISTN and other application might require more than one criterion to be considered to determine the choice of SDN controller to be used, a future work considering a Multi-Criteria Decision-Making (MCDM) is suggested for future work. Also, future work will consider heterogeneous clustering of SDN controllers which will be accompanied with suitable communication protocol that will ensure the cooperative control function of the cluster.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: an intellectual history of programmable networks," Commun. Rev., vol. 44, no. 2, pp. 87-98, 2014, doi: 10.1145/2602204.2602219.

[2] L. Zhu, M. M. Karim, K. Sharif, F. Li, X. Du, and M. Guizani, "SDN Controllers: A Comprehensive Analysis and Performance Evaluation Study," ACM Comput. Surv., vol. 53, no. 6, pp. 1-40, 2020, Art no. 133, doi: 10.1145/3421764.

[3] B. Yi, X. Wang, K. Li, S. k. Das, and M. Huang, "A comprehensive survey of Network Function Virtualization," Computer Networks, vol. 133, pp. 212-262, 2018/03/14/ 2018, doi: https://doi.org/10.1016/j.comnet.2018.01.021.

[4] L. Boero, R. Bruschi, F. Davoli, M. Marchese, and F. Patrone, "Satellite Networking Integration in the 5G Ecosystem: Research Trends and Open Challenges," IEEE Network, Conference vol. 32, no. 5, pp. 9 - 15, September/October 2018, doi: 10.1109/MNET.2018.1800052.

[5] C. Caini, H. Cruickshank, S. Farrell, and M. Marchese, "Delay- and Disruption-Tolerant Networking (DTN): An Alternative Solution for Future Satellite Networking Applications," Proceedings of the IEEE, vol. 99, no. 11, pp. 1980-1997, 2011, doi: 10.1109/JPROC.2011.2158378.

[6] Y. Yang, T. Song, W. Yuan, and J. An, "Towards reliable and efficient data retrieving in ICN-based satellite networks," Journal of Network and Computer Applications, vol. 179, p. 102982, 2021/04/01/ 2021, doi: https://doi.org/10.1016/j.jnca.2021.102982.

[7] M. Allman, D. Glover, and L. Sanchez, "Enhancing TCP over satellite channels," Network Working Group, RFC2488, 1999.

[8] H. He, Y. Hou, J. Yang, X. Jiang, S. Chen, and L. Hanzo, "Towards Reliable Space-Ground Integrated Networks: From System-level Design to Implementation," IEEE Network, pp. 1-7, 2022, doi: 10.1109/MNET.124.2200186.

[9] P. Wang, J. Zhang, X. Zhang, Z. Yan, B. G. Evans, and W. Wang, "Convergence of Satellite and Terrestrial Networks: A Comprehensive Survey," IEEE Access, vol. 8, pp. 5550-5588, 2020, doi: 10.1109/ACCESS.2019.2963223.

[10] Y. Bi et al., "Software Defined Space-Terrestrial Integrated Networks: Architecture, Challenges, and Solutions," IEEE Network, Journal vol. 33, no. 1, pp. 22-28, February, 2019 2019, doi: 10.1109/MNET.2018.1800193.

[11] Z. K. Khattak, M. Awais, and A. Iqbal, "Performance evaluation of OpenDaylight SDN controller," in 2014 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS), 16-19 Dec. 2014 2014, pp. 671-676, doi: 10.1109/PADSW.2014.7097868.

[12] L. Zhu et al., "SDN Controllers: A Comprehensive Analysis and Performance Evaluation Study," ACM Comput. Surv., vol. 53, no. 6, p. Article 133, 2020, doi: 10.1145/3421764.

[13] O. Salman, I. H. Elhajj, A. Kayssi, and A. Chehab, "SDN controllers: A comparative study," in 2016 18th Mediterranean Electrotechnical Conference (MELECON), 18-20 April 2016 2016, pp. 1-6, doi: 10.1109/MELCON.2016.7495430.

[14] L. Mamushiane, A. Lysko, and S. Dlamini, "A comparative evaluation of the performance of popular SDN controllers," in 2018 Wireless Days (WD), 3-5 April 2018 2018, pp. 54-59, doi: 10.1109/WD.2018.8361694.

[15] S. Bhardwaj and S. N. Panda, "Performance Evaluation Using RYU SDN Controller in Software-Defined Networking Environment," Wireless Personal Communications, vol. 122, no. 1, pp. 701-723, 2022/01/01 2022, doi: 10.1007/s11277-021-08920-3.

[16] I. Koulouras, I. Bobotsaris, S. V. Margariti, E. Stergiou, and C. Stylios, "Assessment of SDN Controllers in Wireless Environment Using a Multi-Criteria Technique," Information, vol. 14, no. 9, p. 476, 2023. [Online]. Available: https://www.mdpi.com/2078-2489/14/9/476.

[17] R. L. S. d. Oliveira, C. M. Schweitzer, A. A. Shinoda, and P. Ligia Rodrigues, "Using Mininet for emulation and prototyping Software-Defined Networks," in 2014 IEEE Colombian Conference on Communications and Computing (COLCOM), Colombia, 4-6 June 2014 2014, pp. 1-6, doi: 10.1109/ColComCon.2014.6860404.

[18] I. Z. Bholebawa and U. D. Dalal, "Performance analysis of SDN/OpenFlow controllers: POX versus floodlight," Wireless Personal Communications, vol. 98, 2018// 2018, doi: 10.1007/s11277-017-4939-z.

[19] V. Jain, "Getting Familiar with Wireshark," in Wireshark Fundamentals. CA: Apress, Berkeley, CA, 2022, pp. 35-78.

[20] G. Fritsche, "Understanding Windows 10," presented at the Proceedings of the 2015 ACM SIGUCCS Annual Conference, St. Petersburg, Florida, USA, 2015. [Online]. Available: https://doi.org/10.1145/2815546.2815577.

[21] A. Hazra, "Using the confidence interval confidently," (in eng), J Thorac Dis, vol. 9, no. 10, pp. 4125-4130, Oct 2017, doi: 10.21037/jtd.2017.09.14.

[22] J. Arkko and C. Pignataro, "IANA allocation guidelines for the address resolution protocol (ARP)," in "Request for Comments," RFC Editor, RFC 5494, April 2009, issue 2070-1721. [Online]. Available: https://www.rfc-editor.org/info/rfc5494