ISSN 2278-3091
Volume 9, No.4, July – August 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse248942020.pdf
https://doi.org/10.30534/ijatcse/2020/248942020

# Implementation of Queuing Models with SDN for Load Balancing in Multiple Controller Environment

**Deepjyot Kaur Ryait**[1], **Manmohan Sharma**[2]
[1]School of Computer Applications, Lovely Professional University, Phagwara, India,djryait@gmail.com
[2]School of Computer Applications, Lovely Professional University, Phagwara, India,manmohan.sharma71@gmail.com

## ABSTRACT

The Software Defined Networking (SDN) is one of the most promising technology in networking, which decouples the control plane from the data plane. Therefore, all the control logic is transfer to the SDN controller, which provides a centralized logical view of an entire network. But it also increases the chance of a failure in the network due to a single controller. To overcome this situation, multiple controllers are required in the network. When multiple controllers are used in SDN networks which arise some load balancing related issues like controller overloaded when it exceeds the load from its threshold value, then the controller failure or cascaded failure of controllers is happening in the network.

The purpose of this paper is to propose an algorithm for load balancing in multiple controllers in SDN by using the queuing technique. After then compare both queuing models (M/M/1: ∞/∞ and M/M/1: N/∞) on the basis probability of various parameters like Ls, Lq, Ws, and Wq of the system. These parameters help to evaluate the performance of SDN controllers. On the basis of probability, to analyze which queuing model is more preferable. Moreover, the resulting highlight if the length of system and queue is nearly 8.9% and 7.2% in M/M/1: ∞/∞ model but in M/M/1: N/∞ is 3.8%and 3.03%. Similarly, the waiting time of packet in system and queue is 1.011% and 0.9% in M/M/1: ∞/∞ but in M/M/1: N/∞ model is 0.508% and 0.39%. On the probability basis, these parameter shows the M/M/1: N/∞ model is more preferable than another model.

**Key words:** Load Balancing Multiple Controllers, Queuing Model, SDN

## 1.INTRODUCTION

Today in the Network Communication technologies involves fast and swift development and innovations. The Software Defined Networking is considered the most promise technology in the networking field. Because the SDN brings revolutionary in the network industry by offering programmability, flexibility, easy management, adjustable and dynamic reconfiguration of the network devices. This is happened due to the separation between the control plane and the data plane. Thus, all the control logic is transfer to the controller.

But in traditional networks, there is a lack of programmability facility in the network's elements. The main reason behind this is a strong bound exists between the functional components. The new paradigm networking decouples the control plane from data plane which is known as the Software Defined Networks (SDN). Due to this separation, SDN offers programmability, adjustable, and dynamic reconfiguration of the networking devices. The SDN increases the efficiency of the network by improving the network control which enables the network providers to respond to the changing or varying the business requirements swiftly [1-8]. Currently, several industries are supporting the Software Defined Networks paradigm like Microsoft, Google, Cisco, Facebook, HP, IBM, Samsung, VMware, Juniper, etc.

The SDN architecture consists of three different planes which are interacting through well-defined API's (or interfaces) in Figure 1. In the data plane, all forwarding elements exist, which is control or manage by the controller. The controller lies in the control plane whose responsibility to configure or reconfigure the forwarding devices by customizing their policies in a dynamic manner [1-5, 14,22]. The control plane act as an intermediate between the application and data plane. In application plane is responsible for various network applications and services which are implemented to control the logic of the network domain. These applications always run on the top of the controller.The communication between the application and control plane is possible through the northbound APIs such as the REST (REpresentational State Transfer) API's and the communication between the data plane and the control plane is possible through the southbound APIs such as the OpenFlow protocol. Thus, Software Defined Networks enhance the innovation in the network field which copes up the requirements of the network user on demand.

## 2. SIGNIFICANCE OF SDN CONTROLLER AND ITS ROLES

Moreover, the controller acts as a prominent module of the SDN paradigm because it has the potential to perform all manipulations and implementation in the network. Moreover, it also provides a centralized logical view of an entire network. But simultaneously it increases the maximum chance of a failure in the network due to a single controller [9]. As a consequence, it collapses the entire network.

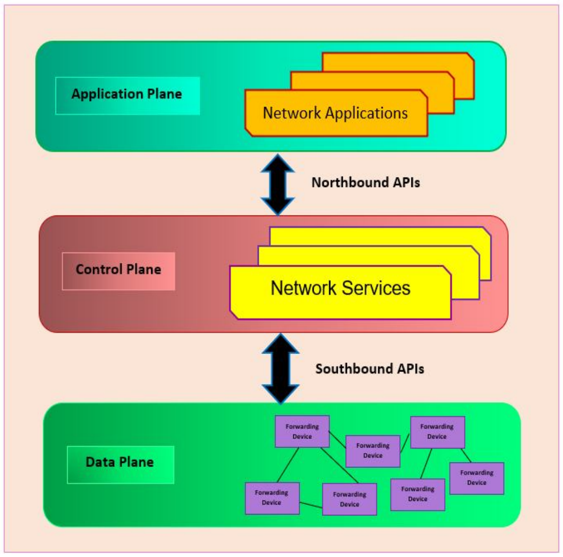
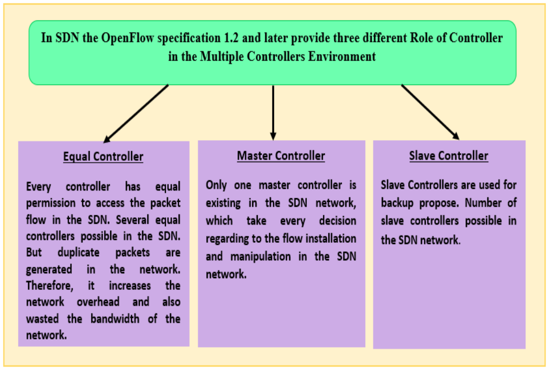


**Figure 1:** Software Defined Network Architecture

To overcome this situation, a fault tolerance mechanism is required which using the multiple controllers in the network. Therefore, the significance of multiple controllers increases the scalability, reliability, and high availability of services in the network. The various roles of multiple controllers in the Software Defined Networks (SDN) are Equal, Master, and Slave controller [9-10,15-19]. The OpenFlow specification supports multiple controllers' environment, in which the controller has anyone role of the following is (in Figure 2):

- **Equal Role**: In the equal role all the controllers configured in the switch have full control to update or modify the flows. The switch must send the PACKET_IN message to all the controllers and also switch process the PACKET_OUT, FLOW_MOD, etc. from all the controllers.
- **Master Role**: The master controller has the responsibility for managing the switches of the data plane. These switches will send the control message to the master controller only.
- **Slave Role**: The slave controller plays the backup role for the master controller. It also receives the HELLO and KEEPALIVE messages. But it cannot send and receive the control messages.

That is why, the master-slave relationship is more preferable than equal role of controller in the multiple controller's environment. But when multiple controllers are used in SDN networks, it introduces or bring some challenges which is related to the load balancing.


In SDN the OpenFlow specification 1.2 and later provide three different Role of Controller in the Multiple Controllers Environment

Equal Controller

Every controller has equal permission to access the packet flow in the SDN. Several equal controllers possible in the SDN. But duplicate packets are generated in the network. Therefore, it increases the network overhead and also wasted the bandwidth of the network.

Master Controller

Only one master controller is existing in the SDN network, which take every decision regarding to the flow installation and manipulation in the SDN network.

Slave Controller

Slave Controllers are used for backup propose. Number of slave controllers possible in the SDN network.

**Figure 2:** Various Role of SDN Controllers

## 3. NEED OF LOAD BALANCING IN SDN CONTROLLERS

Load balancing is a vital issue in the multiple controllers for optimal utilization of available resources of the network and also prevent to reduce the overheads in the control plane [16]. To achieve the load balancing between the multiple controllers in the network, need to distribute the load of the overloaded controller to the other controllers of the network. The load imbalance between the controllers also reduces the overall resource utilization of the network [16,17]. Because when multiple controllers become imbalance then some controllers reach their performance bottleneck due to an increase the response delay whereas other controllers are underloaded or idle state in the network [19].

In SDN, the controller is a very crucial component because it manages all traffic of the network. Thus, the controller is responsible for taking all routing decision of the network while data plane act as a simple forwarding device. Thus, when a controller handles more traffic than its capacity [11-19]. Then these problems are occurring such as controller failure, controller overload when it exceeded its threshold value and cascaded failure of controllers is happened in the network [19].

## 4. PROPOSED DESIGN OF ALGORITHM FOR LOAD BALANCING

In this paper to resolve these issues to design an adaptive algorithm for load balancing in SDN controllers by using the queuing theory techniques, which help to maintain the load fluctuation between the multiple controllers. The motive of this paper is to evaluate a load of multiple SDN controllers by using the queuing model. Because the fault tolerance and load balancing are a complicated issue in the SDN for the multiple controllers. In the proposed model using the Markov and the Queuing Theory Model to design an adaptive load balancing algorithm for multiple controllers.

The Markov process is a simple stochastic process, in which the distribution of future state depends only on the present state and not on how it arrived in the

present state. Therefore, these processes hold the memoryless property [20,21]. According to the memoryless property, the prediction of the future event (state) depends on the basis of the present event only and not upon any previous states.
Therefore, the probability of the future state (Xt+1) at time instant (t+1) depends on the present state (Xt) at time instant t is defined as:

$$P [ X_{t+1} | X_{t,} X_{t-1,} \ldots, X_{2,} X_{1,} X_0] = P [ X_{t+1} | X_t]$$

where Xt is the present state, Xt-1 is the immediate past state and Xt+1 is the future state of the Markov process respectively. Thus, the memoryless property is also applicable in the queuing model and it is also known as the network queuing model.

In the queuing model, there are two types of events that occur either arrival rate or service rate. The arrival rate and time between the arrivals are followed by the Poisson Distribution and Exponential Service Distribution respectively [12,13,20-23].

The arrival rate of a packet in the system at time instant t is represented as λ and service rate of a packet in the system (or processing of packet) at time instant t is represented as μ. The traffic intensity or utilization factor of the system is represented as □. To calculate the value of the traffic intensity is □=λ/μ. The idle time of the system is represented by $P_0$=1-□. In Figure 3, shows how the queuing technique is applicable in SDN Controller. The outline of the load balancing algorithm shown in Figure 4 and flow-chart in Figure 5.

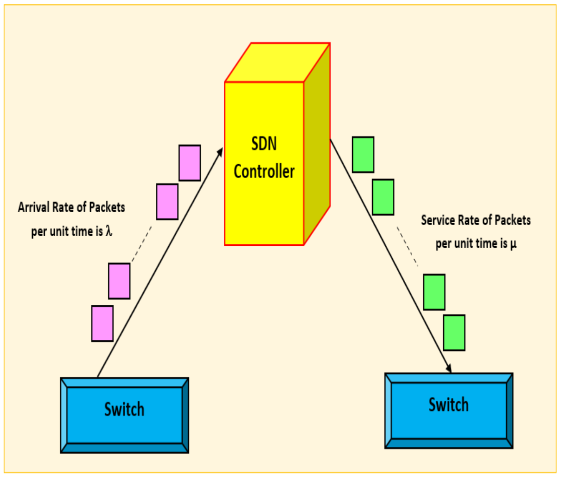

SDN Controller
Arrival Rate of Packets per unit time is λ
Service Rate of Packets per unit time is μ
Switch
Switch

**Figure 3:** Use of Queuing Technique in SDN Controller

**Algorithm for Load Balancing in SDN for Multiple Controllers by using Queuing Technique**

**Initial Requirement:**
CC is represented for Current Controller; SC is represented for Slave Controllers in the network;
λ = Arrival Rate of the packet; μ = Service Rate of the packet; □= Traffic Intensity of Controller;
Traffic Intensity (□) act as a threshold value of the Controller in the network.
**Result:**
0: No need for load balancing
1: Successfully load balancing perform

```
/* Load Balancing between Multiple Controllers in SDN*/
if Load of CC > □ then
  {
        for (i=0; i<n; i++)
        {
             /* Calculate the traffic intensity value of all slave controllers SCi*/
                 □i=λi/μi
                 Select controller SCi whose has lowest traffic intensity value in the network.
        }
        return 1;
  }
else
 {
   return 0;
 }
```

**Output:** As the consequences, this algorithm helps to resolves all the issues which are related to load balancing in the network such as overloaded controller, controller failure in the network. Moreover, also avoid cascaded failure of controllers in the network due to load unbalancing between controllers.

**Figure 4:** Algorithm for Load Balancing in Multiple Controllers

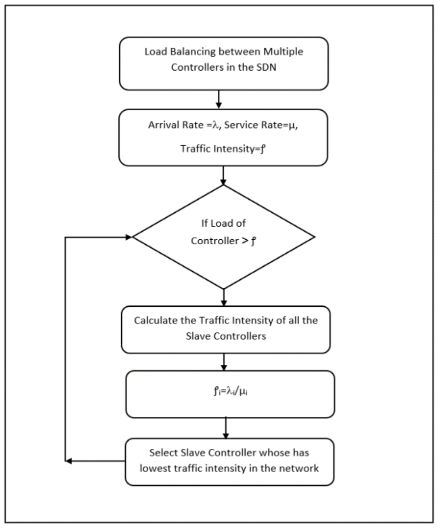

Load Balancing between Multiple Controllers in the SDN
Arrival Rate =λ, Service Rate=μ, Traffic Intensity=ƒ
If Load of Controller > ƒ
Calculate the Traffic Intensity of all the Slave Controllers
ƒi=λi/μi
Select Slave Controller whose has lowest traffic intensity in the network

**Figure 5:** Flow Chart for Load Balancing in SDN Controllers

## 4.1 Queuing Model (M/M/1: ∞/∞) v/s (M/M/1: N/∞)

In the queuing model {M/M/1: ∞/∞}, the average number of arrival rate per unit of time is λ and the average number of service rate per unit of time is µ. The steady state equation of M/M/1: ∞/∞ model is expressed as the sum of three independent compound probabilities [21]. Therefore,

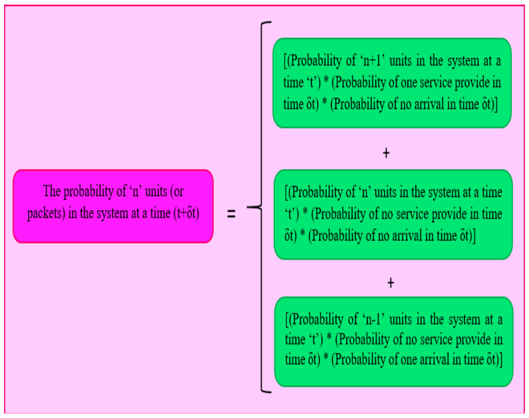

The probability of 'n' units (or packets) in the system at a time (t+δt)
=
[(Probability of 'n+1' units in the system at a time 't') * (Probability of one service provide in time δt) * (Probability of no arrival in time δt)]
+
[(Probability of 'n' units in the system at a time 't') * (Probability of no service provide in time δt) * (Probability of no arrival in time δt)]
+
[(Probability of 'n-1' units in the system at a time 't') * (Probability of no service provide in time δt) * (Probability of one arrival in time δt)]

**Figure 6 (a):** Probability of events occur in time interval (t+δt)

To obtain the steady state of the differential equation of M/M/1: ∞/∞ model is the product of three possibilities events occur [21] are shown in Figure 6 (a) and Figure 6(b). Therefore,

$$\frac{\delta Pn\,(t+\delta t)}{\delta t} = \begin{Bmatrix} -(\lambda+\mu)Pn(t) + \lambda Pn-1(t) + \mu Pn+1(t); \\ for\ n > 0 \end{Bmatrix} \quad (1)$$

$$\frac{\delta P0(t+\delta t)}{\delta t} = \{-\lambda P0(t) + \mu P1(t); \qquad for\ n = 0\} \quad (2)$$

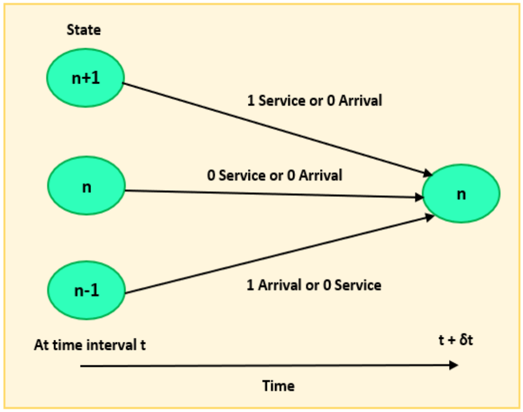


**Figure 6 (b):** Possible of events occur in time interval (t+δt)

To solve the above differential equations of the queuing model M/M/1: ∞/∞, to find the value of $P_1$ from equation (2).

$$P_1=(\lambda/\mu)\ P_0$$

Then put n=1 in equation (1) and get $P_2=(\lambda/\mu)^2\ P_0$ and so on.
Similarly,

$$P_n=□^n P_0 \quad (3)$$

After calculating the probability of $P_n$ of 'n' packets in the system and probability that the queuing system is idle by $P_0$. The queuing system also provides four important properties which are related to each other [21]:

- ➢ To calculate the length of system ($L_s$) by using equation (4).
  $L_s$= □/(1-□)(4)
- ➢ To calculate the length of queue ($L_q$) by using equation (5).
  $L_q$=$L_s$+□ (5)
- ➢ To calculate the average waiting time of packet in the system ($W_s$) by using equation (6).

  $$W_s= L_s/\lambda \quad (6)$$
- ➢ To calculate the average waiting time of packet in the queue ($W_q$) by using equation (7).
  $W_q$=$L_q$/λ(7)

But in M/M/1: ∞/∞ queuing model create infinite queue length which also affects the value parameters like length of queue (Lq), length of system (Ls), waiting time in queue (Wq), and waiting time in system (Ws) in Figure 10.

Thus, it is preferable to use M/M/1: N/∞ queuing model in which the length of the queue is finite. So, the number of arrivals will not exceed the N in any case [21]. Therefore, the capacity of the system is limited to say N.

Let

| | |
|---|---|
| Arrival Rate (λ) = $\lambda_n$ | $\lambda_{n=} \begin{cases} \lambda\ , if\ n = 0,1,2,\ldots..N-1 \\ 0\ , if\ n \geq N \end{cases}$ |
| Service Rate (µ)= $\mu_n$ | $\mu_{n=} \{\mu\ , for\ n = 1,2,3,\ldots.. \}$ |

Similarly, solve differential equations of the queuing model M/M/1: N/∞, and get value of $P_0$ and $P_n$ in equation (8) and (9).

Therefore,

$$P_{0=}\left[\frac{1-□}{1-□^{N+1}}\right](8)$$

$$P_n=□^n P_0 \quad \{for\ n = 0,1,2,3,\ldots.N\}(9)$$

Similarly, evaluate expression of Ls, $L_q$, $W_s$ and $W_q$ in queuing model M/M/1: N/∞ are given below: -

- ➢ To calculate the length of system ($L_s$) by using equation (10).

$$L_s= □/(1-□)\left[\frac{1+N□^{N+1}-(N+1)□^{N}}{1-□^{N+1}}\right](10)$$

- The effective arrival rate ($\lambda_{eff}$ = $\lambda(1-P_N)$)
- To calculate the length of queue ($L_q$) by using equation (11).

$$L_q=L_s - \frac{\lambda_{eff}}{\mu} \qquad (11)$$

- To calculate the average waiting time of packet in the system ($W_s$) by using equation (12).

$$W_s=\frac{L_s}{\lambda_{eff}} \ (12)$$

- To calculate the average waiting time of packet in the queue ($W_q$) by using equation (13).

$$W_q=\frac{L_q}{\lambda_{eff}} \quad (13)$$

## 5. NUMERICAL EVALUATION AND RESULT

When the current controller increases its load from the threshold value then how to distributed the load between other controllers by using the proposed algorithm for the load balancing in the SDN. Suppose in the network (in Figure 7), the current controller whose arrival rate and service rate are 8 and 9 respectively.

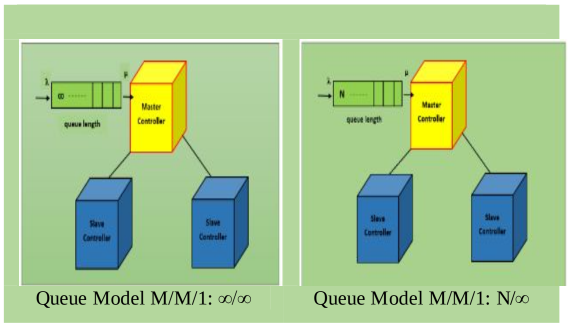


**Figure 7:** Representation of Queuing Models in Network

Calculate the probability of traffic intensity(□), idle time ($P_0$), no queue occurs in the system and also the probability of ten packets in the system ($P_{10}$) by using both queuing model M/M/1: ∞/∞ and M/M/1: N/∞ show in Figure 8 (a and b) respectively. The comparison between both models shown in Figure 9.

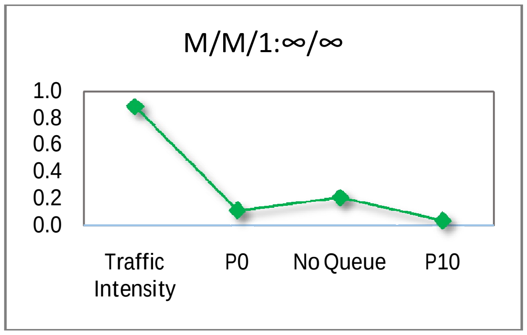


**Figure 8 (a):** M/M/1: ∞/∞ Model

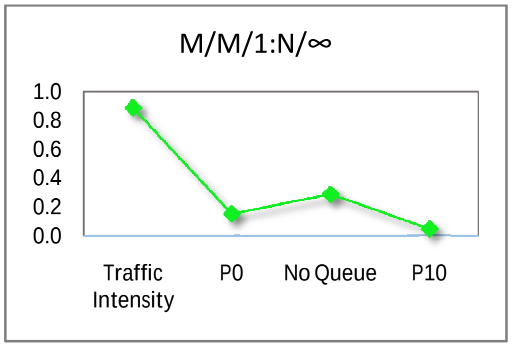


**Figure 8 (b):** M/M/1: N/∞ Model

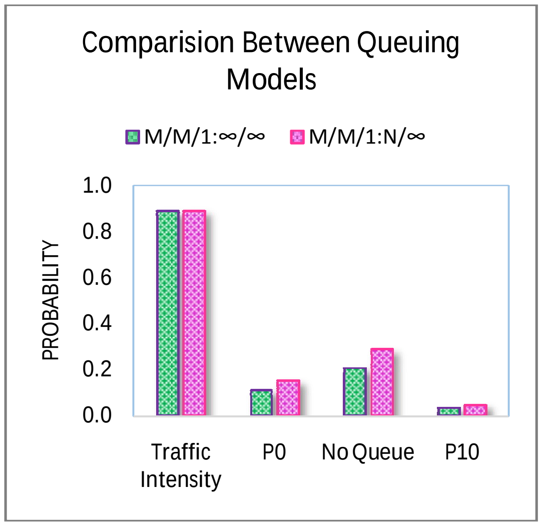


**Figure 9:** Comparison between M/M/1: ∞/∞ and M/M/1: N/∞ Model

And also compare the various parameters like expected length of system ($L_s$), expected length of the queue in the system ($L_q$), expected waiting time of a packet in the system (Ws) and also expected waiting time of a packet in the queue ($W_q$) in both queue models shown in Figure 10.

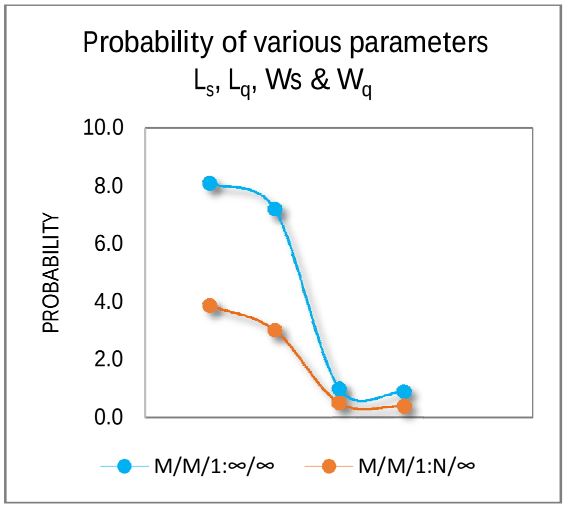


**Figure 10:** Compare both Models w.r.t. $L_s$, $L_q$, Ws&$W_q$ parameter's

In Figure 10, it is more preferable to use the M/M/1: N/∞ queuing model because the probability of these parameters in sequence like (Ls, $L_q$, Ws&$W_q$) are reduced as compare to the M/M/1: ∞/∞ model. The blue curve represents the M/M/1: ∞/∞ and the orange curve represents the M/M/1: N/∞ Model.

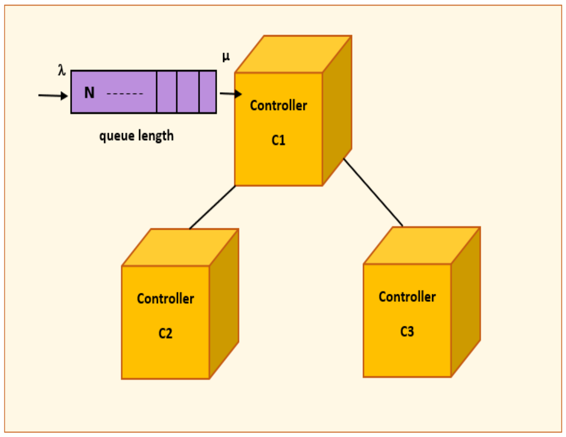


**Figure 11:** Scenario of network usingM/M/1: N/∞ Model for Load balancing in SDN

Suppose in the network (in Figure 11) has three controllers whose arrival and service rate are given in Table 1, by using M/M/1: N/∞ Model in which the finite length of the queue is 10. Then calculate all parameters of these three controllers are shown in graph form in Figure 12, Figure 13, and Figure 14, Figure 15 (a) & (b).

**Table 1**: Arrival and Service Rate of the Controllers in the Network

| Controllers | Arrival Rate | Service Rate |
|---|---|---|
| C1 | 8 | 9 |
| C2 | 4 | 5 |
| C3 | 3 | 6 |

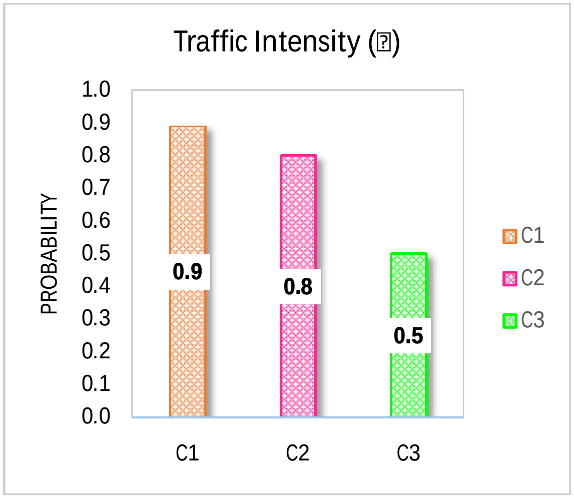


**Figure 12:** Probability of Traffic Intensity of the Controllers

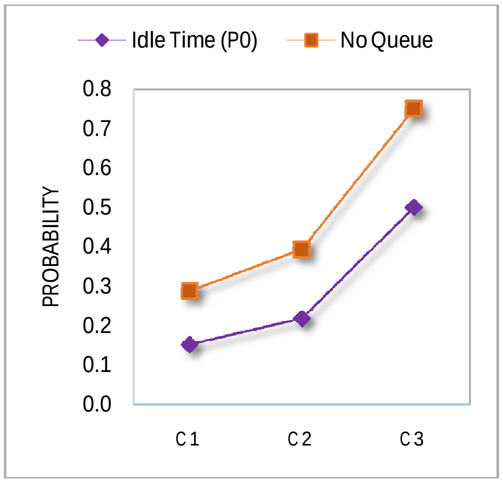


**Figure 13:** Probability w.r.t. $P_0$ and No Queue occur in the System

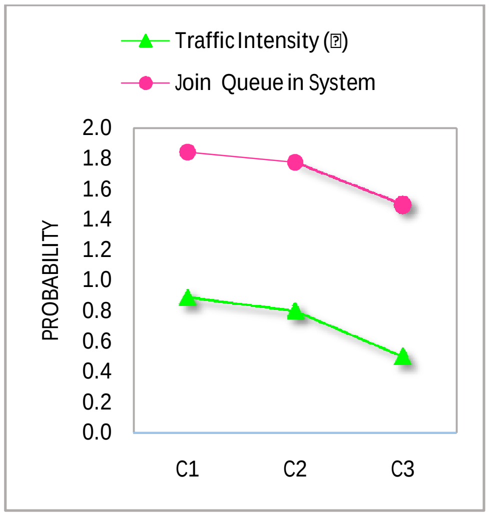
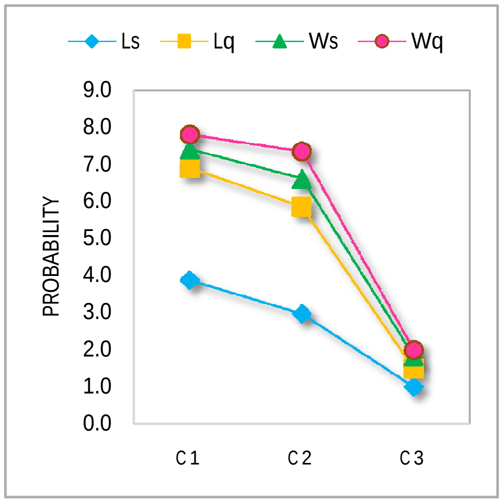


**Figure 14:** Probability w.r.t. Traffic Intensity and Join Queue in the System



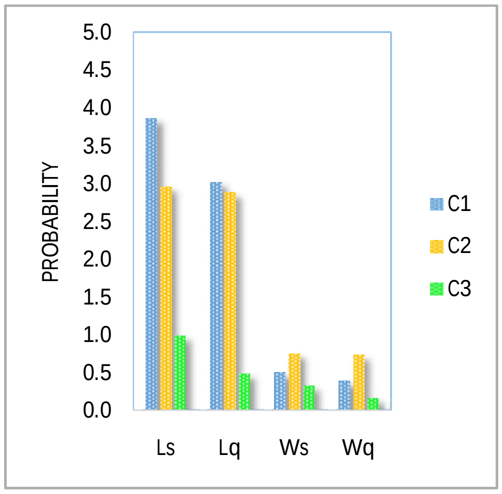


**Figure 15 (a) and (b):** Probability w.r.t. $L_s$, $L_q$, Ws & $W_q$ in the System

In this scenario, controller C3 has the lowest traffic intensity value as compare to controller C2. When the current controller C1 exceeded its threshold value in the network, then the controller C3 is selected to manage the load of the network.

## 6. CONCLUSION

In this paper proposed an algorithm for load balancing in SDN controllers by using the queuing techniques. These techniques help to manages the load between the multiple SDN controllers in the networks. Because the fault tolerance and load balancing are complicated and interrelated issues in the SDN for controllers. The main contribution of this paper is to propose an algorithm for load balancing in the multiple controllers by using the queuing technique. In a Figure 9 and Figure 10 also highlights the comparison between both queuing models respectively. If the probability of idle time in system ($P_0$)is 11% in M/M/1: ∞/∞ model and 15% in M/M/1: N/∞ model. But the probability of other parameters like $L_s$ = 8.09%, $L_q$=7.2%, Ws=1.011% &$W_q$=0.9% approximately in M/M/1: ∞/∞ model. Similarly, the probability of other parameters $L_s$ = 3.87%, $L_q$=3.03%, Ws=0.508% &$W_q$=0.39% in M/M/1: N/∞ model is approximately. The probability of these parameter shows the M/M/1: N/∞ model is more preferable than another model. Because the probability of these parameters ((Ls, $L_q$, Ws&$W_q$) are reduce as compare to another model.

## REFERENCES

1. D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," in Proceedings of the IEEE, vol. 103, no. 1, pp. 14–76, 2015.
2. W. Braun and M. Menth, "Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices," Open Access Future Internet, vol. 6, no. 2, pp. 302-336, 2014. https://doi.org/10.3390/fi6020302
3. Y. Yu, X. Li, X. Leng, L. Song, K. Bu, J. Yang, Y. Chen, L. Zhang, K. Cheng and X. Xiao, "Fault Management in Software-Defined Networking: A Survey," IEEE Communications Surveys & Tutorials, vol. 21, no. 1, pp 349-392, 2018.
4. C. M. Duran, E. A. Leal and J. F. Botero, "Improving fault tolerance in critical networks through OpenFlow," in IEEE Colombian Conference on Communications and Computing (COLCOM), IEEE, 2017.
5. A. Malik, B. Aziz, A. Al-Haj and M. Adda, "Software-Defined Networks: A Walkthrough Guide From Occurrence To Data Plane Fault Tolerance," Open Access, pp. 1-26, 2019.
6. J. Chen, J. Chen, F. Xu, M. Yin and W. Zhang, "When Software Defined Networks Meet Fault Tolerance: A Survey," G. Wang et al. (Eds): International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP), Part III, vol. 9530, pp. 351-368, 2015.
7. M. R. Parsaei, S. H. Khalilian and R. Javidan, "A Comparative Study on Fault Tolerance Methods in IP Networks versus Software Defined Networks," International Academic Journal of Science and Engineering, vol. 3, no. 4, pp. 146-154, 2016.
8. B. Isong, I. Mathebula and N. Dladlu, "SDN-SDWSN Controller Fault Tolerance Framework for Small to Medium Sized Networks," in the 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), IEEE Computer Society, pp. 43-51, 2018. https://doi.org/10.1109/SNPD.2018.8441131
9. L. Sidki, Y. Ben-Shimol and A. Sadovski, "Fault Tolerant Mechanisms for SDN Controllers," in IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), IEEE, pp. 1-6, 2016.
10. Y. Zhang, L. Cui, W. Wang and Y. Zhang, "A Survey on Software Defined Networking with Multiple Controllers," Journal of Network and Computer Applications (Elsevier), pp. 1-58, 2017.
11. I. F. Akyildiz, A. Lee, P. Wang M. Luo and W. Chou, "Research Challenges for traffic Engineering in Software Defined Networks," IEEE Network, 2016.
12. B. Xiong, X. Peng and J. Zhao, "A Concise Queuing Model for Controller Performance in Software-Defined Networks," Journal of Computers, vol. 11, no. 3, pp. 232-237, 2016.
13. T. Issa et al., "Analytical Load Balancing Model in Distributed Open Flow Controller System," Science Research Publishing, vol. 10, pp. 863-875, 2018.
14. O. Blial, M.B. Mamoun and R. Benaini, "An Overview on SDN Architectures in Multiple

Controllers," Journal of Computer Networks and Communications Journal, vol. 2016, pp. 1-8, 2016.
https://doi.org/10.1155/2016/9396525

15. W. H. F. Aly, "Generic Controller Adaptive Load Balancing (GCALB) for SDN Networks," Journal of Computer Networks and Communications, 2019.
16. A. Mahjoubi, O. Zeynalpour, B. Eslami and N. Yazdani, "LBFT: Load Balancing and Fault Tolerance in distributed controllers," in 2019 International Symposium on Networks, Computers and Communications (ISNCC), IEEE, 2019.
17. W. H. F. Aly, "Controller Adaptive Load Balancing for SDN Networks," in 2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN), IEEE, 2019.
18. A. U. Rehman, R. L. Aguiar and J. P. Barraca, "Fault- Tolerance in the Scope of Software-Defined Networking (SDN)," IEEE Access, vol. 7, pp. 1-18, 2019.
19. J. Cui, Q. Lu, H. Zhong, M. Tian and L. Liu, "A Load-Balancing Mechanism for Distributed SDN Control Using Response Time," IEEE Transactions on Network and Service Management", vol. 15, no. 4, pp. 1197-1206, 2018.
20. A. Mondal, S. Misra and I. Maity, "Buffer Size Evaluation of OpenFlow Systems in Software-Defined Networks," IEEE Systems Journal, vol. 13, no. 2, pp. 1359-1366, 2019.
21. L. Kleinrock, Queueing Systems – Volume 1: Theory. Wiley-Interscience, 1975.
22. G. Nencioni, B. E. Helvik, A. J. Gonzalez, P. E. Heegaard and A. Kamisinski, "Availability Modelling of Software-Defined Backbone Networks," in 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W), IEEE, 2016.
23. M. Escheikh and K. Barkaoui, "Scalable Load Balancing Scheme for Distributed Controllers in Software Defined Data Centers," in 2019 Sixth International Conference on Software Defined System (SDS), IEEE, 2019.
https://doi.org/10.1109/SDS.2019.8768708