VOL. 15, NO. 6, MARCH 2020 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

A SIMULATION OF LOAD BALANCING IN SOFTWARE DEFINED NETWORK (SDN) BASED ON ARTIFICIAL NEURAL NETWORKS METHOD

Andika Malraherawan Pradana, Tito Waluyo Purboyo and Roswan Latuconsina
Department of Computer Engineering. Faculty of Electrical Engineering, Telkom University, Bandung, Indonesia
E-Mail: titowaluyo@gmail.com

ABSTRACT

In the development of network technology, where development makes it easier for us to monitor and build computer networks, the increased use of the internet will also result in improved service quality. It is not enough just to be able to connect to the internet, the performance also becomes an important concern. Load balancing is one of the behaviour to divide the load into several servers. The purpose of load balancing is to allocate resources, maximize throughput, minimize response time, and avoid charging resources. This study discusses the analysis of the software defined Network (SDN) for enhancement and optimization applied using the Load Balancing technique. Response time Parameter is an important aspect which is monitored by controlling the Mininet. This system is designed with the queue method and path determinant, namely ANN (Artificial Neural Network) that is able to create data and channel input data into certain categories or predetermined paths with the main role in load balancing in the Software Defined Network to determine the cost which means the ability to process input data without having to have an optimization target.

Keywords: experimental, load balancing, software defined networking, ANN.

INTRODUCTION

The development of internet networks continues to accelerate rapidly; this can be seen from the increasing number of users who are connected to the network, especially the internet network, on the other hand because more and more users are using internet networks to meet the needs of technological developments caused by the number of requests made by users [2]. The general way to deal with it is by adding a server or adding additional hard disks to the database, but this method requires a large amount of money and only a small number of users can provide it. Then there is one solution to overcome these network problems is by load balancing techniques.

Load balancing is one mechanism to divide computational load to several servers. Minimizing response time, maximizing throughput, reducing dealy, and avoiding overloading is an important aspect in this domain [4]. This technique is also often used to distribute traffic loads on connection lines in a balanced manner, and that traffic can be optimal [1]. With the increasingly complex technology of network developers, a new paradigm in network technology appears, namely the Software Defined Network (SDN). By using an SDN network, this final project network programming can be automated and can be changed dynamically as needed. Besides that SDN is a new concept or paradigm in designing, managing and implementing networks [2]. With the flexibility of the SDN, this load balancing program can run optimally by combining using the path determination algorithm, namely the neural network method (ANN).

In load balancing techniques on the network there are several methods and algorithms that can be applied using several queuing methods, namely, Round-Robin [4], Weighted Round-Robin [4], Least Connection [6], Weighted Least Connection [6], and this final project uses one of the path determination algorithms, namely Artificial Neural Network (ANN).

By referring to the formulation of the problem, the purpose of this final project is to distribute data loads, through artificial neural network methods that are sent through the Software Defined Network to several servers with the queue method and path determinants using Load Balancing techniques, which later to minimize Time Response, and avoid overloading on one server and to get performance on CPU utilization.

THEORETICAL BASIS

A. Software Defined Network (SDN)

Software Defined Network (SDN) is a new architecture in the field of computer networks, has characteristics that are dynamic, manageable, cost-effective, and adaptable [2], so it is ideal for the needs of current applications that are dynamic and have high bandwidth [4]. Software Defined Network (SDN) architecture separates network control and forwarding functions, so that network control can be programmed directly [3].

In Software Defined Network (SDN) architecture consists of three layers, such as the following picture:

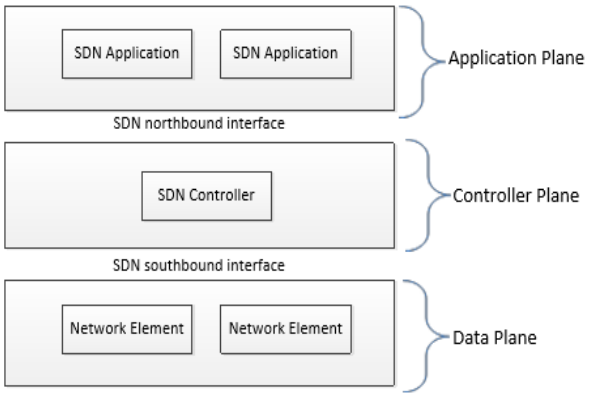


Figure-1. SDN architecture [3].

a) Application Plane is the topmost layer is an application that can directly and explicitly define network requirements and desired network behavior [3]. This layer communicates with the Control Layer through the North Bound Interface (NBI) [3].
b) Controller Plane is to neutralize the needs of various applications with infrastructure by giving instructions in accordance with SDN Data Path [3] and providing information that is relevant and needed by SDN applications.
c) Data Plane, consisting of network elements that can receive instructions from the Control Layer [3]. The interface between Controller Plane and Data Plane is called South Band Interface (SBI).

The Plane Controller is main one in the SDN network, can be run separately from the Data Plane. While the Data Plane is network hardware specifically programmed and fully controlled by the Control Plane [5]. At SDN there is an open interface that allows a software entity or application to control the connectivity of network resources, control network traffic, and modify network traffic. Advantages in Software Defined Network (SDN) include:

a) Easy for device management compared to conventional networks, because of centralized control.
b) Directly Programmable can be programmed directly because the control plane is separate from data plane.
c) Network administrators can dynamically adjust network traffic flow as needed because there is a separation.
d) Network administrators can also quickly configure the network, through an SDN program that is automatic and can also be written alone because the program does not depend on proprietary software or devices.
e) The format of the instructions contained in SDN uses open standards and does not depend on any vendor or specific protocol, so that it can be implemented on any network, without vendor related issues and this greatly facilitates network innovation.

B. Load balancing

Load balancing is one mechanism to divide computational load to multiple servers [8]. Using several computational resources can also reduce the possibility of a service being malfunctioning because each resource can be redundant [10].

Load balance can divide network traffic fairly and minimize the occurrence of overload on one server [9], but disconnections often occur for real time applications due to the gateway shift on each network that leads to the server.

Load balance is also a technique for distributing traffic loads on two or more connection paths in a balanced manner, so that traffic can run optimally, reduce response time and avoid overloading one of the connection lines [7]. And some of the advantages obtained from the Load Balancing technique are as follows:

a) Flexibility, the server is no longer the core of the system and the main resource, but is part of many servers that make up the cluster. This makes the per-unit performance of the cluster not too calculated, but overall cluster performance. Whereas to improve performance and clusters, a new server or unit can be added without replacing the old unit.
b) Scalability, the system does not require redesign because all system architectures can adapt the system when there are changes to system components. For all traffic that passes the Load Balancer, security rules can be implemented easily. With a Private Network used for Real servers, the IP address will not be accessed directly from outside the cluster system [17].
c) High-availability, Load balancer can determine the condition of the Real server in the system automatically, if there is a real server that is dead, it will be deleted from the Real server list, and if the Real server is active then it will be included in the Real server list [7]. Load balancers can also be configured redundant with others Load Balancers

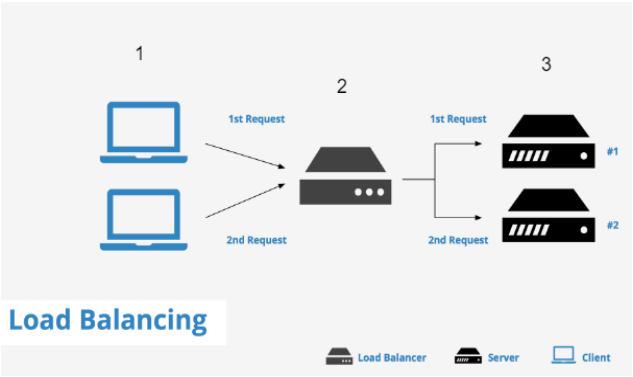


Figure-2. General description of load balancing [4].

Figure-2 above is a computer topology that is combined with Load Balancing. Then divided into several devices, namely:

Client is the user who uses the service on the server.Load Balancer, the technique of dividing traffic

load distribution. Server, the task is to provide services to users, namely a web server. And request 1,2 is path for the acces data network.

C. OpenFlow

OpenFlow is the most important protocol at SDN. Its position is between the Controller and Forwarding (data plane). OpenFlow allows setting routing and packet delivery when via a switch. In a network, each switch only functions to forward packets that pass through a port without being able to distinguish the type of protocol data being sent. OpenFlow makes it possible to access and manipulate Forwarding Plane directly from network devices such as switches and routers both physically and virtually. Nick McKeown *et al* provides a detailed discussion the background of OpenFlow [4]. The basic idea is quite simple, namely exploiting i.e. data switches and the most modern Ethernet routers that have a flow of tables that run at the linerate level to implement firewalls, NAT, QoS, and also to collect statistical data. Flow tables can differ in implementation depending on the vendor of the hardware. However, there are several common set of functions between a variety of switches and vendor-based routers [11]. On the other hand, vendors are not harmed because they do not need to open the internal working mechanism of the product switches or routers made by each vendor. To be able to use OpenFlow, an SDN Controller is needed that supports the running of the OpenFlow protocol. Controller is one of the open source development platforms for SDN applications that support the OpenFlow protocol [12].

D. Route Flow

Route Flow is formed by merging the OpenFlow project with the Quagga routing engine. This system consists of OpenFlow (RFProxy), RFClient and Independent Server (RFServer) controllers [10]. The main purpose of making RouteFlow is to implement IP routing virtualization centrally. This research utilizes RouteFlow as a system that runs on the control plane.

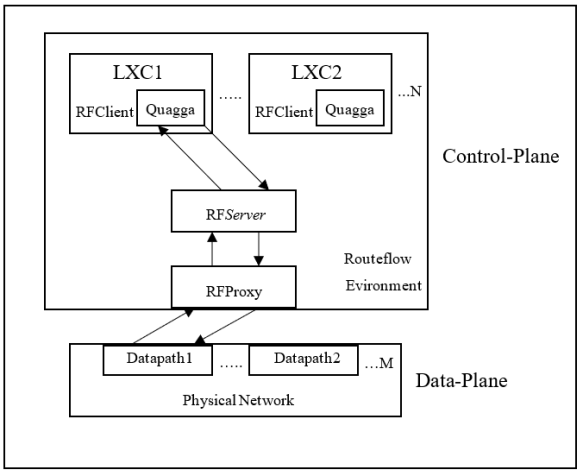


Figure-3. Route flow process [10].

a) RFServer, is a standalone application that holds control of the network control center. RF Server manages a Virtual Machine (VMware) that runs on RFClients and manages logic processes [10].
b) RFProxy is a POX controller whose job is to forward protocol policies (such as update routes, datapath configurations) from RFServer to data-plane.
c) RFClient is a VMware daemon that is tasked with detecting changes in routing information and notifying it to RFServer. The task is communicated with the routing engine (Quagga).

METHOD

The selection of Artificial Neural Network (ANN) method in Load Balancing technique is because the ANN is able to classify and select input data into specified categories and the main role in load balancing in SDN networks is to determine and change weights which mean the ability to process data - input data without having to have an optimization target.

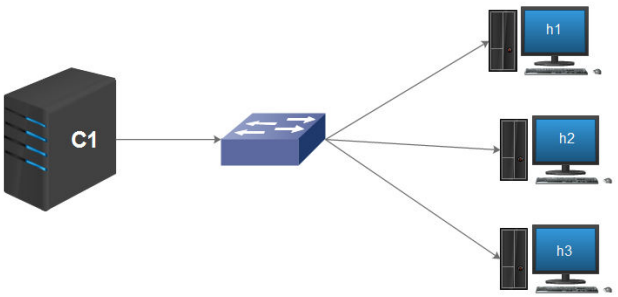


Figure-4. Network topology of Mininet

Mininet is a Command Line Interface (CLI) emulator for making large-scale network prototypes quickly on very limited resources [1]. Mininet was created with the aim of supporting research in the field of SDN. The Mininet emulator itself makes it possible to run a code interactively without having to modify the code [10]. That is, the simulation code is exactly the same as the code in the real network environment. Mininet is also a solution that is considered the most superior in various ways such as ease of use, accuracy and scalability. Mininet is able to provide a reality and easy configuration at a low price [14]. Compared to testbed hardware which tends to be more expensive, it is very difficult to be reconfigured, but more accurate than using the Mininet emulator. In Simulation scenario host is *h,* conttoller is C0, switch is *S*, and server is *SV*.

A. Artificial Neural Network (ANN) method

Artificial neural networks are information processing systems that have the learning ability of data and information received, the ability to model linear functions, parallel computing, and has the nature of tolerating uncertainty (fault tolerance). The application of artificial neural networks is very broad, including in terms of forecasting, data analysis (data analysis), and pattern recognition [4].

$$u_k = \sum_{i=1}^{n} w_{ik} x_i \quad (1)$$

$$y_k = f(u_k + b_k) \quad (2)$$

xi : = servicing heavy neurons k. And if> 0, it shows the situation, and if, it is part of yourself.

n = indicates the number of entries in the vector input.

$W_{ik,}$ = is a linear of vector inputs combination, represented by positive or negative values as neuron values, input from data activation will increase or decrease according to neurons [4]. This is the normalization of neural network

$f(x)$ = the function triggers the translation of logic. output from neurons.

u_k = is defined as the linear combination of input vector

b_k = is denoted as the threshold value of a neuron

y_k = presents the output of neuron.

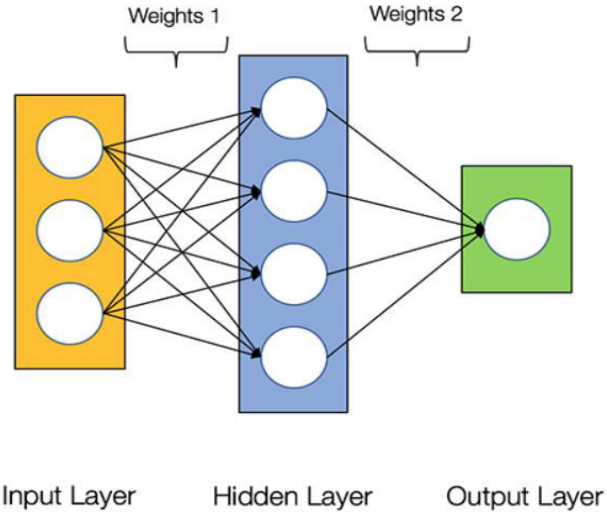


Figure-5. Network topology of ANN [4].

B. Backpropogation ANN

The most popular algorithm is backpropogation, where information travels through the network towards the input layer towards the output layer. The most important task to build artificial neural networks is to choose the right network topology to solve certain problems [4].

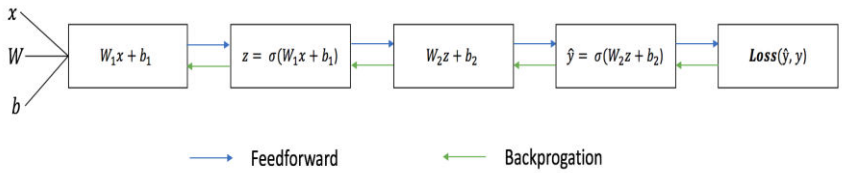


Figure-6. Backpropagation data input process [6].

x =an input layer, an arbitrary amount of hidden layers

ŷ = an output layer

W and b= a set of weights and biases between each layer.

σ = A choice of activation function for each hidden layer.

$\hat{y} = \sigma(W_2\sigma(W_1x + b_1) + b_2)$ = the right values for weights and biases are determines the strength of the predictions. The process of fine-tuning the weights and biases from the input data known as training the Neural Network.

$Loss(y, \hat{y}) = \sum_{i=1}^{n} (y - \hat{y})^2$ = That is, the sum-of-squares error is simply the sum of the difference between each predicted value and the actual value. The difference is squared so that we measure the absolute value of the difference.

Backpropagation involves 3 stages, namely the feedfoward training pattern, error calculation and weight adjustment. After training, network applications only use the first computing step, feedfoward to do testing [6]. Although the training stage is slow, the network can produce output very quickly. The Backpropagation method has been varied and developed to increase the speed of the training process [7]. Although one network layer is very limited in learning, networks with many layers can learn more. More than one hidden layer might be useful for some applications, but one hidden layer is enough.

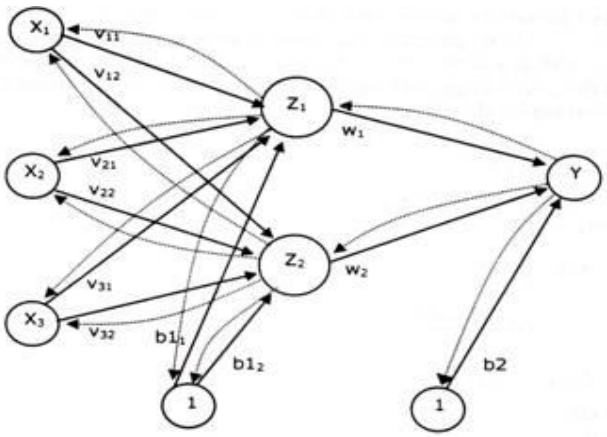


Figure-7. Network topology of ANN backpropagation [7].

In Figure-7, x_1, x_2, x_3, is input data request neuron from the input layer, $Z_1, Z_2$is data procces in hidden layer and *y* is output data procces in output layer.

The Backpropagation algorithm for neural networks is generally applied to multilayer perceptrons [6]. The Perceptron has at least the input part, the output part and several layers that are between the input and output. The center layer, also known as the hidden layers, can be one, two, three and so on [6].

EXPERIMENT SETUP

The simulation tool and another simulation environment can be seen in Table-1 as below.

Table-1. Simulation setup.

Simulation Tool	Mininet
Platform	Windows 10 on VMware
OS	Ubuntu 16 VM
CPU	Intel core i7 Nvidia GTX 850 (3.4Ghz)
RAM	8Gb
SDN Controller	Pox
Supporting Tools	Miniedit,Spyder 3.0, Wireshark,Ab apache, curl, OpenFlow, npstat

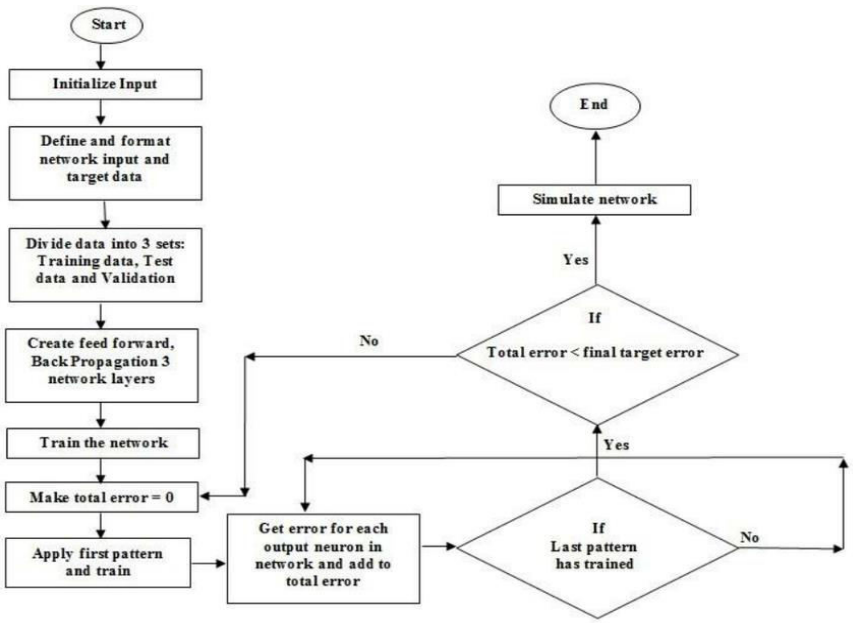


Figure-8. Flowchart system implementation of ANN.

Figure-8 represents the flowchart of ANN. The process used for training the network is called a learning algorithm, whose work is to change the junction weights of the network to obtain the desired objective presented in [7]. Neural networks are typically organized in layers. Layers are drawing up of a number of interrelated 'nodes' which include an 'activation function'. Patterns are represented to the network through the 'input layer', which transfer to one or more 'hidden layers' where the definite action is done through a system of weighted 'connections'. The hidden layers then connect to an 'output layer' where the response is output.

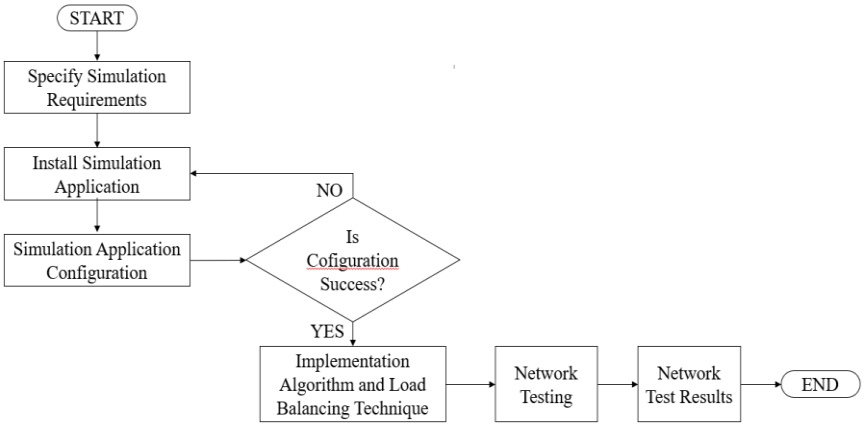


Figure-9. Flowchart simulation.

Figure-9 is a design flowchart and has the following information as below.

a) Start is at the beginning of the design system will be executed.
b) Determination of simulation requirements is software or hardware that is used to support simulation needs.
c) Installation of simulation tools and applications is to start installing the software and applications needed in the simulation.
d) The successful configuration tool is to show if the configuration tool from the previous installation was successful or not.
e) The ANN technique is the goal if the configuration is successful according to what is expected, then the first execution process is entered into the ANN by means of the network to be executed to enter the greatest weight.
f) Load balancing purpose after going through the execution process by ANN then the network enters the second execution process [16], namely the Load Balancing technique for the execution runs evenly.
g) Neteork testing is when running a program in topology with a set scenario
h) Data from network testing is taking analysis after carrying out the execution processes that have been done before.
i) END is the process at the end after all the executions and analyzes are complete.

Test scenario

In the realization system, testing will be carried out with parameters and scenarios to be tested on computer networks with the scenarios and implementations that have been designed. The number of servers is 3 Tests.

Testing of Response time and CPU Utilities in Artificial Neural Network and Round Robin Algorithms is divided into 3 tests, namely:

a) Testing Scenario I
   3 servers and 10 clients with 100 requests
b) Testing Scenario II
   5 servers and 10 clients with 100 requests
c) Testing Scenarios III
   10 servers and 50 clients with 100 requests

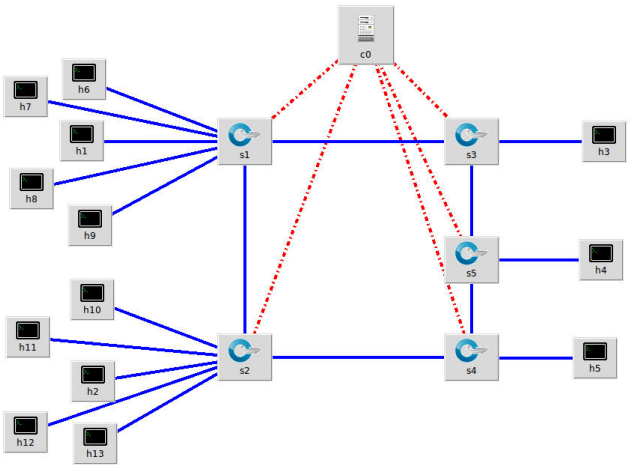


Figure-10. Experiment using 10 client and3 servers (Scenario I).

Figure-10 depicts the implementation by connecting 1 Controller and 5 switches to 3 Servers that get messages from 10 Clients.

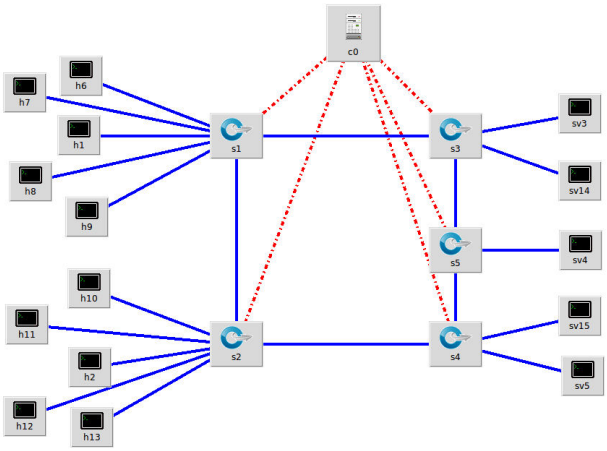


Figure-11. Experiment using 10 client dan 5 servers (Scenario II).

In Figure-11 has a description by connecting between 1 Controller and 5 switches to 5 servers that get messages from 10 clients.

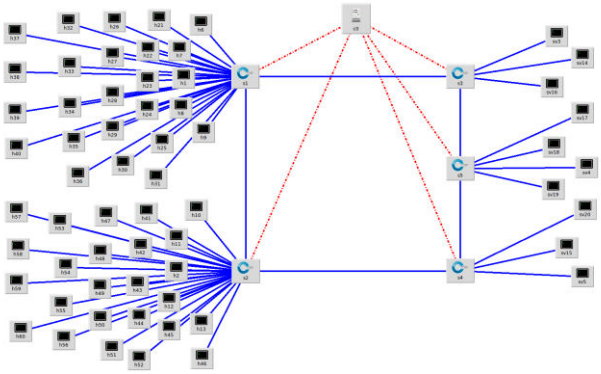


Figure-12. Experiment using 50 client dan 10 servers (Scenario III).

In Figure-12 has a description: by connecting between 1 Controller and 15 switches to 10 Servers that get messages from 50 Clients.

RESULT AND DISCUSSIONS

In the realization of this system, the parameters and scenarios to be elaborated will be carried out. For this testing phase, the testing will be carried out:

a) Bandwidth
   An internet speed of the network
b) Response time
   Overall data requestion
c) CPU Utilization
   CPU presentation of idle

As a result of the scenarios that have been tested, the artificial neural network (ANN) algorithm will be compared with the round-robin (RR) algorithm. in the elaboration of these results, there are several terms namely host as h,conttoller is C0, switch is S, and server is SV/h.

This test aims to retrieve the request data displayed from Client to Server analyzed through Path, whether it is divided according to the criteria of the ANN Algorithm. In this simulation the server will be given an http server compliment on Mininet via Xterm (terminal emulator from Mininet) where in this application the server will serve HTTP packet requests via port 80. With 3 test parameters, namely: bandwidth, response time, and cpu utilization

Scenario Result I

A. Bandwidth

Based on topology on Figure-10, we find out the total bandwidth between h6, h7, h8, h9, h10, h11, h12, h13 using wget. The server represented by h1 using the default port, default bandwidth size (100 mb/ sec), and the interval is 1 second, and server h3, h4, and h5 test the consecutive bandwidth to host and the result can be seen in Figure-10 as below.

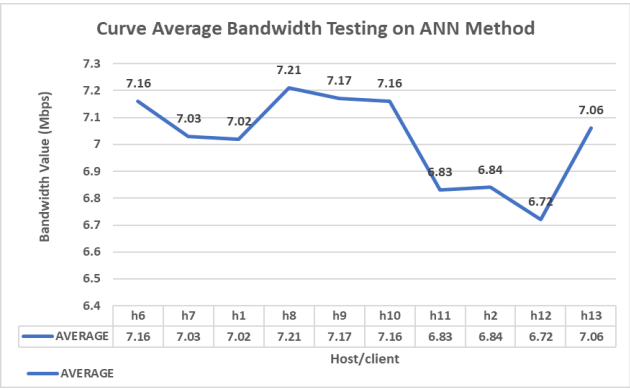


Figure-13. Curve of bandwidth result average testing for ANN method.

In Figure-13, Testing In the graph above shows that when a request is given to the server it gives a fairly stable average bandwidth value at the beginning until mid-testing, and drops at the end of the host but is quite stable. In that chart the smallest bandwidth data is located on host 12 with 6.72 Mbps traffic, while the largest bandwidth value is located at host 8 with 7.21 Mbps traffic. The greater the bandwidth value, the better it is to generate Response time.

B. Response time

The following comparison results of testing on response time with average values in the testing scenario I as below.

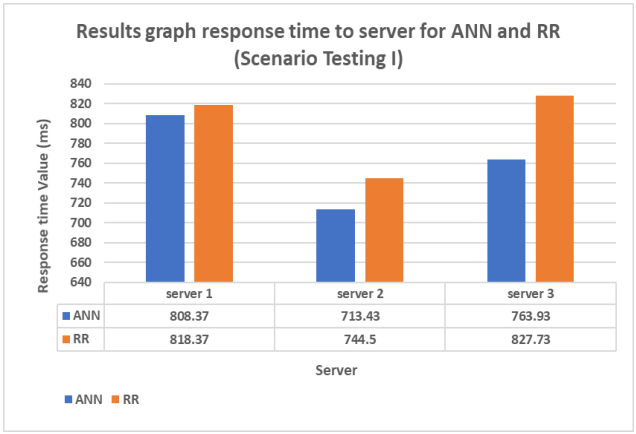


Figure-14. Graph of response time to server for ANN and RR result.

In Figure-14, the comparison test results on response time with an average value in the I test scenario, the analysis that can be obtained is that in the first test the response time per server value in ANN is smaller than the RR value, and with a comparatively stable difference, then in the scenario testing of the ANN algorithm is better.

C. CPU utilization

The following comparison results of testing on CPU Utilization with average values in the testing scenario I as below.

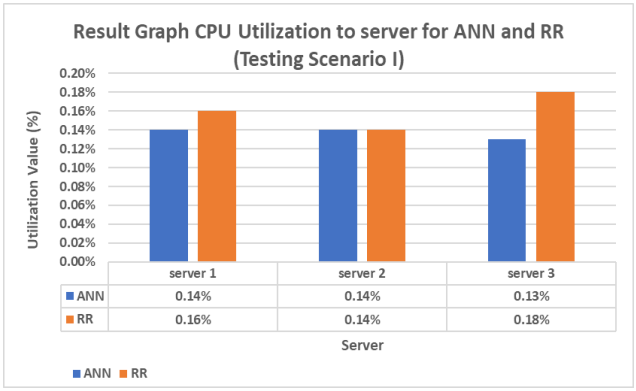


Figure-15. Graph of CPU utilization to server for ANN and RR result.

In Figure-15, the results of comparison testing on CPU utilization with the average value in the Test Scenario I, the analysis that can be obtained is that in the first test the value of per-server scpu utilization in ANN is smaller than the RR value, and with a comparatively stable difference, then in the scenario testing of the ANN algorithm is better.

Scenario Result II

A. Bandwidth

Based on topology on Figure-11, we find out the total bandwidth between h6, h7, h8, h9, h10, h11, h12, h13 using wget. The server represented by h1 using the default port, default bandwidth size (100 mb/ sec), and the interval is 1 second, and server h3, h4, h5, h14, h15 test the consecutive bandwidth to host and the result can be seen in Figure-11 below.

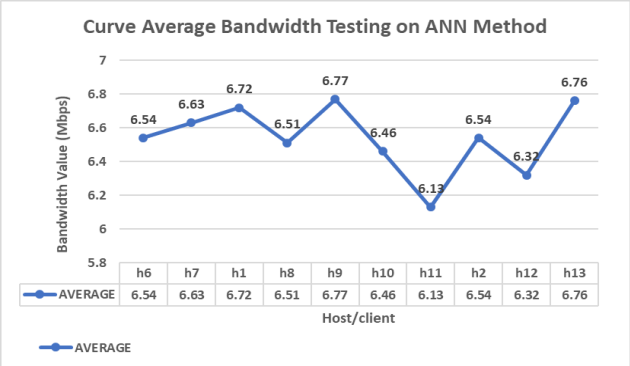


Figure-16. Curve of bandwidth result average testing for ANN method.

In Figure-16, Testing In the graph above shows that when a request is given to the server it gives an average bandwidth value that is unstable at the beginning until mid-test, but the test chart rises to the end of the test. In the graph, the smallest bandwidth data is located on host 11 with traffic of 6.13 Mbps, while the largest bandwidth value is located at host 9 with traffic 6.77 Mbps. The greater the bandwidth value, the better it is to generate Response time.

B. Response time

The following comparison results of testing on response time with average values in the testing scenario II as below.

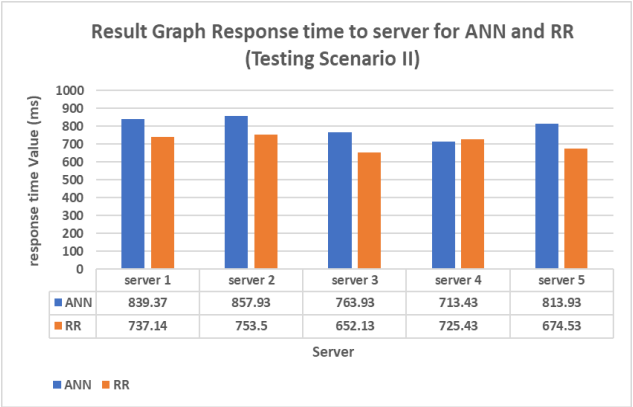


Figure-17. Graph of response time to server for ANN and RR result.

In Figure-17, the comparison results of testing on response time with average values in the testing scenario II, the analysis that can be obtained is that in this test the value of sufficiently stable testing II value of response time per server on RR is smaller than the ANN value, and with a sufficient difference in comparison stable, then in the testing scenario II the RR algorithm is better.

C. CPU utilization

The following comparison results of testing on CPU Utilization with average values in the testing scenario IIas below.

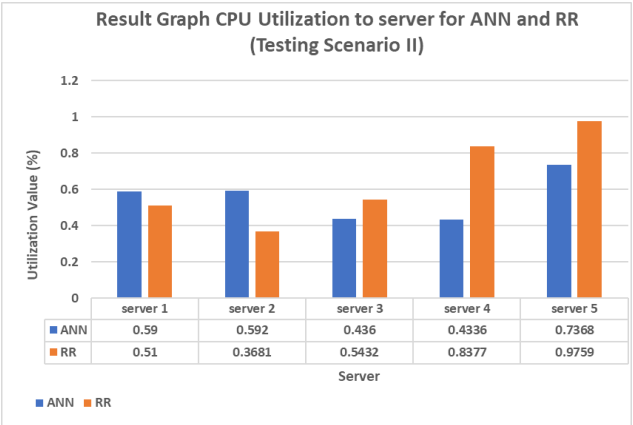


Figure-18. Graph of CPU utilization to server for ANN and RR result.

In Figure-18, the third scenario testing uses 10 server scenarios and 50 clients by giving bandwidth weights on the traffic path of 100 Mbps, delay on each switch 1 (sec) and Host 2 (sec). The request number is 100 requests where 1 request contains data of 10485760bit or 10MB. The initial stage to get the value of Response Time data results is needed Bandwidth value first to see the results of time of client request. After getting the Bandwidth results, the response time (ms) calculation can be done directly using the Ab apache tool and Curl Mininet.

Scenario Result III

A. Bandwidth

Based on topology on Figure-12, we find out the total bandwidth between h6, h1, h9, h22, h26, h28, h30, h32, h34, h35, h36, h37, h38, h38, h40, h11, h12, h41, h43, h45, h47, h49, h51, h53, h55, h57, h59using wget. The server represented by h host using the default port, default bandwidth size (100 mb/ sec), and the interval is 1 second. Server h3, h4, h5, h14, h15, h16, h18, h19, h20, and h2 test the consecutive bandwidth to host and the result can be seen in Figure-12 as below.

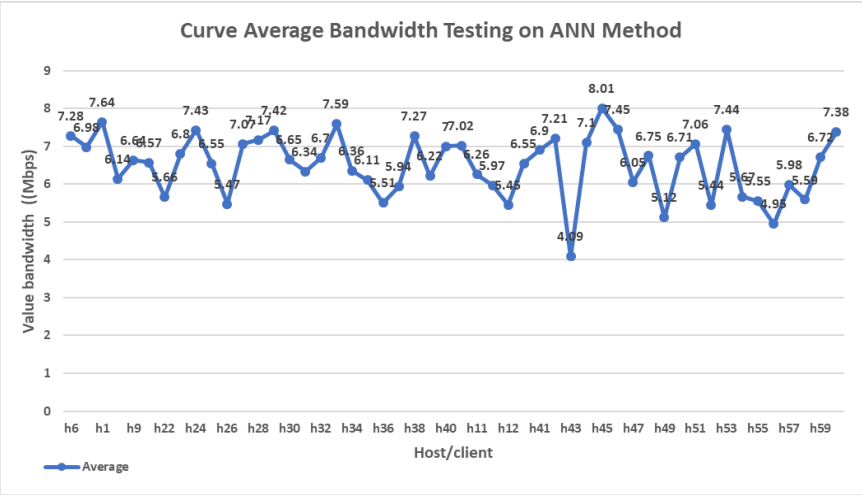
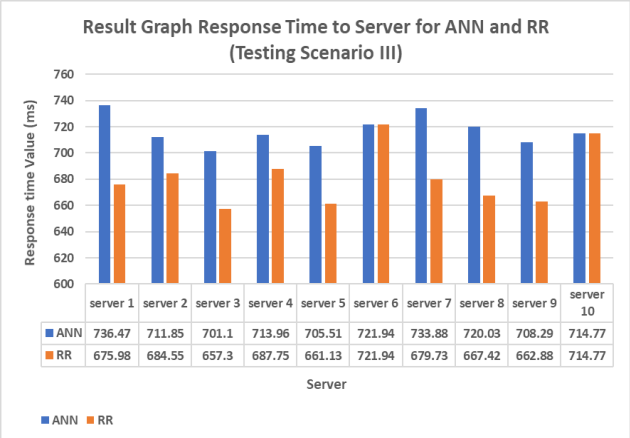


Figure-19. Curve of bandwidth result average testing for ANN method.

In Figure-19, testing in the graph above shows that when a request is given to the server it gives an average bandwidth value that is unstable at the beginning until mid-test, but the test chart rises to the end of the test. In this graph, the smallest bandwidth data is located on Host 43 with 4.09 Mbps of traffic, while the largest bandwidth value is located at Host 54 with 8.01 Mbps traffic. The greater the bandwidth value, the better it is to generate Response time.

B. Response time

The following comparison results of testing on Response time with average values in the testing scenario III as below.



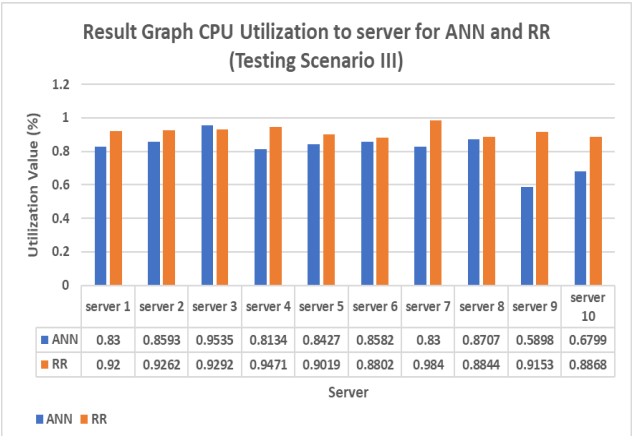
	server 1	server 2	server 3	server 4	server 5	server 6	server 7	server 8	server 9	server 10
ANN	736.47	711.85	701.1	713.96	705.51	721.94	733.88	720.03	708.29	714.77
RR	675.98	684.55	657.3	687.75	661.13	721.94	679.73	667.42	662.88	714.77

Figure-20. Graph of response time to servcer for ANN and RR result.

In Figure-20, the comparison results of testing on response time with average values in the testing scenario II, the analysis that can be obtained is that in this test the value of sufficiently stable testing II value of response time per server on RR is smaller than the ANN value, and with a sufficient difference in comparison stable, then in the testing scenario II the RR algorithm is better.

C. CPU utilization

The following comparison results of testing on CPU Utilization with average values in the testing scenario III as below:



	server 1	server 2	server 3	server 4	server 5	server 6	server 7	server 8	server 9	server 10
ANN	0.83	0.8593	0.9535	0.8134	0.8427	0.8582	0.83	0.8707	0.5898	0.6799
RR	0.92	0.9262	0.9292	0.9471	0.9019	0.8802	0.984	0.8844	0.9153	0.8868

Figure-21. Graph of CPU utilization to server for ANN and RR result.

In Figure-21, the third scenario testing uses 10 server scenarios and 50 clients by giving bandwidth weights on the traffic path of 100 Mbps, delay on each

switch 1 (sec) and Host 2 (sec). The request number is 100 requests where 1 request contains data of 10485760bit or 10MB. The initial stage to get the value of Response Time data results is needed Bandwidth value first to see the results of time of client request. After getting the Bandwidth results, the response time (ms) calculation can be done directly using the Ab apache tool and Curl Mininet.

Value comparison

Comparison of results values Compared with comparing and looking for differences between the two algorithms used are Artificial Neural Networks and Round Robin algorithms as below:

Table-2. Value comparison response time.

Comparison Value of Average Response Time (ms)		
Result	**Round - robin**	**Artificial Neural Network**
Testing Scenario 1	796.87	761.91
Testing Scenario 2	708.55	797.72
Testing Scenario 3	681.35	716.78

In Table-2, explained that data collection in both algorithms by comparing each test in a scenario that is by comparing the value of the Round-robin algorithm with the artificial neural network algorithm, the value that appears after the average.

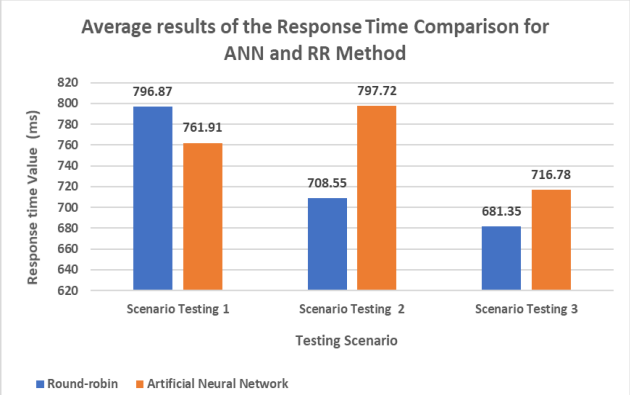


Figure-22. Graph of comparison value of response time.

In Table-2 and Figure-22, it can be seen that the graph has a fairly stable decline from the start of scenario I to scenario III the graph goes down with a fairly stable value, and can be analyzed that the topology has greater bandwidth for each host that enters the server, client request to server in each scenario, the value of the Response time will be smaller.

Table-3. Value difference of CPU utilization.

Ratio Value of CPU Utilization (%)		
Result	**Round-robin**	**Artificial Neural Network**
Testing Scenario 1	0.16	0.13
Testing Scenario 2	2.21	1.6
Testing Scenario 3	6.467	7.335

In Table-3, explained that data retrieval in both algorithms by comparing each test in a scenario that is by taking the value ratio of the Round-robin algorithm with an artificial neural network algorithm, the value that appears after the average of the Ratio.

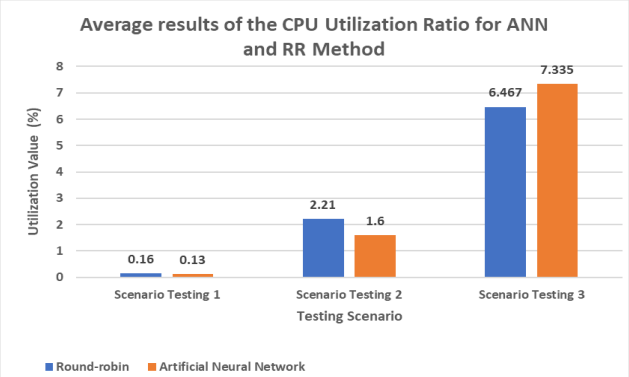


Figure-23. Graph of ratio value CPU utilization.

In Table-3 and Figure-23, the graph shows a significant increase from the start of scenario I to scenario III the graph rises with a fairly stable value, it can also be analyzed in an increasingly heavy topology, namely the client's request to the server in each scenario, the higher the value of utilization the CPU.

CONCLUSION AND FUTURE WORK

In this paper, Artificial neural network algorithm can be applied and simulated in Software Defined Network with load balancing mechanism on the path using POX Controller for parameter testing, then this final assignment research can be summarized as below.

Response Time on a server with a variety of testing scenarios only experiences a slight difference between one server and another server.

From the results of testing scenarios I, II, and III, the smaller the value of response time, the better the test results,

From the results of testing scenarios I, II, and III, the greater the bandwidth value of each host / client, the smaller the response time value on the server.

The more servers, the smaller the value of Response Time

In the I test scenario, the response time test results of the ANN algorithm are better than the RR algorithm, but in scenarios II and III, the Response time of the Round-robin algorithm gets a smaller result which

means the results are better than the Artificial Neural algorithm Network.

The Artificial neural network algorithm produces lower CPU utility values than the Round-robin algorithm, but is more stable as a result of the Round-robin algorithm.

REFERENCES

[1] S. Sakunthala, R. Kiranmayi, P. Nagaraju Mandadi. 2017. A Review on Artificial Intelligence Techniques in Electrical Drives on Neural Networks, Fuzzy logic, and Genetic Algorithm. JNTUACEK - Kalikiri, JNTUA University, Ananthapuram International Conference. Smart Technology for Smart Nation. IEEE. 2017.

[2] Pan Zhu1, Jiangxing Zhang. 2017. Load Balancing Algorithm for Web Server Based on Weighted Minimal Connections. Journal of Web Systems and Applications (2017) Vol. 1, No. 1 Clausius Scientific Press, Canada

[3] Nada M. Al Sallami, Ali Al daoud, Sarmad A. Al Alousi. 2013. Load Balancing with Neural Network. (IJACSA) International Journal of Advanced Computer Science and Applications. 4(10). Jordan.

[4] Cui Chen-xiao and Xu Ya-bin. 2016. Research on Load Balance Method in SDN. School of Computer, Beijing Information Science and Technology University. International Journal of Grid and Distributed Computing. 9(1): 25-36

[5] Daphne Tuncer, Marinos Charalambides, Stuart Clayman and George Pavlou. 2014. Adaptive Resource Management and Control in Software Defined Networks. International Paper, Department of Electronic and Electrical Engineering at University College London, UK.

[6] Shavan Askar. 2016. Adaptive Load Balancing Scheme For Data Center Networks Using Software Defined Network. Journal of University of Zakho. 4(A) (2): 2016.

[7] Al-Khanjari Z., Alani A. 2014. Testability of Information Leak in the Source Code for Independent Test Organization by Using Back Propagation Algorithm. Department of Computer Science, College of Science, Sultan Qaboos University, Muscat, Oman. 2nd Annual International Interdisciplinary Conference, AIIC, 8-12 July, Azores, Portugal Proceedings. Vol. 3.

[8] S. Mondal, A. Bandyopadhyay. 2014. Performance Prediction of Software Defined Network Using an Artificial Neural Network. Chinese Control Decis. Conf. (2): 1-4, 2013.

[9] Fortinet. 2015. The Basics of Server Load Balancing and the Evolution to Application Delivery Controllers. E-book White Paper. Vol. 5.

[10] Hao and K. Bao. 2014. A Survey on Software Defined Network and Open Flow: From Concept to Implementation. IEEE Communications Surveys & Tutorials. 16(4): 2181-2206.

[11] Kreutz, F. Ramos, P. Verissimo, C. Rothenberg, S. Azodolmolky and S. Uhlig. 2015. Software-Defined Networking: A Comprehensive Survey. Proceedings of the IEEE. 14-76.

[12] S. Shin, L. Xu, S. Hong and G. Gu. 2016. Enhancing Network Security through Software Defined Network (SDN). IEEE, 2016.

[13] Faris Putra Perdana, BudhiIrawan, Roswan Latuconsina. 2017. Load Balancing Performance Analysis Based On Weigthed Round Robin Algorithm In Software Defined Network (Sdn). e-Proceeding of Engineering. 4(3): 4161.

[14] Rahmadani Hadianto, Tito Waluyo Purboyo, 2018. A Simulation Study Of Sdn Defense Against Botnet Attack Based On Network Traffic Detection. ARPN Journal of Engineering and Applied Sciences.