A Survey Paper on Dynamic Load Balancing in Software Defined Networking

Gaurav Tiwari, Aditi Rai, V Deeban Chakaravarthy, Ruturaj Kadikar

Abstract- Computer networking has scaled great heights in past few years. And traditional networks had failed the expectations of many of the people. People want quality of service and hence various network applications provide quality of service to customers. In this paper we try to balance load using software defined networking with the use of RYU controller using mininet tool. Now a days a lot of data is send by various people using various networks. There should be no data loss and nd there should be no delay in time taken for data transmission. We develop new mechanism for guaranteed latency requirements of those applications who give guaranteed delivery of data.

Keyworks: controller, data transmission,, RYU mininet, software defined networking.

I. INTRODUCTION

Data communication forms essential part of our lives.With the onset of new applications such as mobile. it has gained speed and also has recovery models. These applications have much more requirements and have large bandwidth and latency is also very good. Initially traditional networks have low rate of file data transferor used the technique of remote login. This was impossible when it came the use of data transfer on large area and it used the TCP protocol that is transfer control protocol. Now a days we use many protocols and they have a good security features and good rate of data transfer. The recent applications such as video on demand, cloud computing, big data etc. use various protocols and have more requirements for time and latency. They present better quality of service and take less time. This research algorithm takes into account the advantages and features of SDN, which can sense the state of each of the elements on the network in order to act consequently. In order to describe the algorithm, first it is needed to disclose the different data structures involved on it. Such structures characterize the different elements that have been taken in account in order to achieve an efficient load balancing, at the same time that to reduce as much as possible the computational cost and time.

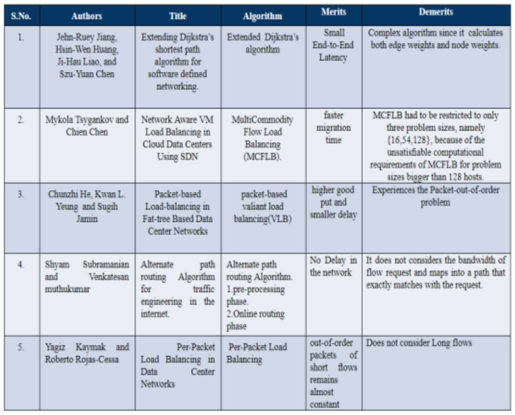
Video streaming requires sufficient bandwidth so that the receiver does not face the problem of buffering. They may have other requirements such as low jitter or low packet loss ratio.But this is not the case of big data transfer files. They don’t have a deadline for data transfer. Traditional networks don’t provide quality of service for data transfer and communication. If we provide total capacity to fulfill the network applications then also they won’t be able to give the quality of service to the networks and to the receivers. Many scholars are working on this in order to provide quality of service in the traditional networks. But they are not able to achieve it. There are many reasons for this. Most prominent of all of these are proprietary protocols in the networks are fixed by the vendors and they can’t be changed by the receivers. They do this because they don’t want to expose the internal structure to public and they want to keep it close thus it makes the program difficult to manage and also the network is very difficult to manage. The network applications cant change their style to meet the requirements of end users. Here the problem arises Software defined network has proposed the new paradigm for computer networks. The other problem that we deal is that of latency. Data should be sensed and signals should be transmitted within the given frame of time so that the system works effectively. There are some factors that these packets have to be in a queue and at the same time there is a lot of traffic. The higher priority queue and switches are are to be allowed to go first. We can understand it in other words such as if bandwidth can be used by others if it is not used by the reserved traffic. But the priorities of the reserved traffic is high and hence there is no delay in control and sensor traffic in the congested network and routers and thus we are able to keep end-to-end latency in limit. New routing paths can be explored and these applications can use the concept of SDN. There are new transmission data for the transfer of data in big data transfer such as cloud computing etc. We don’t use one TCP protocol for the transfer of data but we use multiple TCP protocols to transfer the data effectively. We have a good knowledge about network topology and We have SDN controller that may be RYU or Pox . Thus we can develop algorithm and hence can improve the overall throughput of the system and make the system more effective. Here we use few mechanisms to overcome the problems that have arisen due to traditional network system and provide better quality of service to selected networks. The main works of our project is

Revised Manuscript Received on 30 March 2019.
* Correspondence Author

Aditi Rai, Department of Computer Science And Engineering, SRM University Tamil Nadu, India.
Gaurav Tiwari, Department of Computer Science and Engineering, SRM University Tamil Nadu, India.
V Deeban Chakaravarthy Department of Computer Science and Engineering, SRM University Tamil Nadu, India.
Ruturaj Kadikar Department of Computer Science and Engineering, SRM University Tamil Nadu, India.

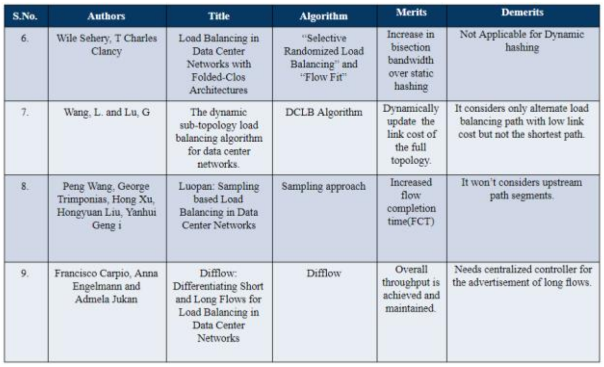
- Improve the quality of service to improve the demands that are associated with the bandwidth. We use SDN and monitor the status of the network and and then set the path for bandwidth demanding traffic flows.
- Provide quality of service to the devices that are sensitive to latency. Different class of traffic flows are adjusted and accommodated with different levels of trafficrequirements.

II. LITERATURE SURVEY



S.No.	Authors	Title	Algorithm	Merits	Demerits
1.	Jehn-Ruey Jiang, Hsin-Wen Huang, Ji-Hau Liao, and Szu-Yuan Chen	Extending Dijkstra's shortest path algorithm for software defined networking.	Extended Dijkstra's algorithm	Small End-to-End Latency	Complex algorithm since it calculates both edge weights and node weights.
2.	Mykola Tsygankov and Chien Chen	Network Aware VM Load Balancing in Cloud Data Centers Using SDN	MultiCommodity Flow Load Balancing (MCFLB).	faster migration time	MCFLB had to be restricted to only three problem sizes, namely {16,54,128}, because of the unsatisfiable computational requirements of MCFLB for problem sizes bigger than 128 hosts.
3.	Chunzhi He, Kwan L. Yeung and Sugih Jamin	Packet-based Load-balancing in Fat-tree Based Data Center Networks	packet-based valiant load balancing(VLB)	higher good put and smaller delay	Experiences the Packet-out-of-order problem
4.	Shyam Subramanian and Venkatesan muthukumar	Alternate path routing Algorithm for traffic engineering in the internet.	Alternate path routing Algorithm. 1.pre-processing phase. 2.Online routing phase	No Delay in the network	It does not considers the bandwidth of flow request and maps into a path that exactly matches with the request.
5.	Yagiz Kaymak and Roberto Rojas-Cessa	Per-Packet Load Balancing in Data Center Networks	Per-Packet Load Balancing	out-of-order packets of short flows remains almost constant	Does not consider Long flows

Figure 1. Literature survey_1



S.No.	Authors	Title	Algorithm	Merits	Demerits
6.	Wile Sehery, T Charles Clancy	Load Balancing in Data Center Networks with Folded-Clos Architectures	"Selective Randomized Load Balancing" and "Flow Fit"	Increase in bisection bandwidth over static hashing	Not Applicable for Dynamic hashing
7.	Wang, L. and Lu, G	The dynamic sub-topology load balancing algorithm for data center networks.	DCLB Algorithm	Dynamically update the link cost of the full topology.	It considers only alternate load balancing path with low link cost but not the shortest path.
8.	Peng Wang, George Trimponias, Hong Xu, Hongyuan Liu, Yanhui Geng i	Luopan: Sampling based Load Balancing in Data Center Networks	Sampling approach	Increased flow completion time(FCT)	It won't considers upstream path segments.
9.	Francisco Carpio, Anna Engelmann and Admela Jukan	Difflow: Differentiating Short and Long Flows for Load Balancing in Data Center Networks	Difflow	Overall throughput is achieved and maintained.	Needs centralized controller for the advertisement of long flows.

Figure 2. Literature survey_2

III. ARCHITECTURE

Multipath TCP and SDN is used in order to maximize throughput.

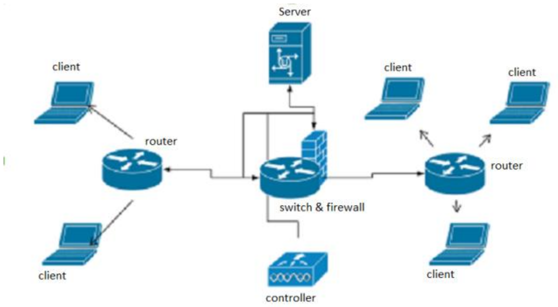


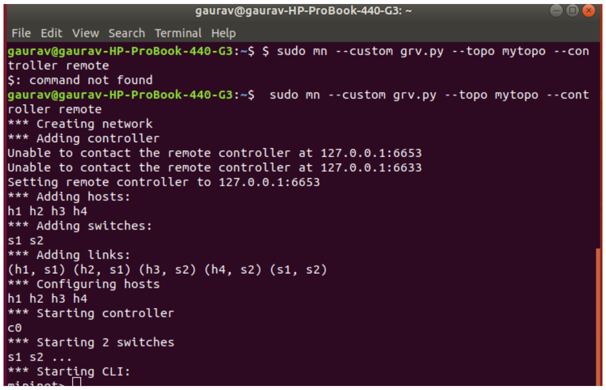
Figure 3. Architecture

An SDN controller is a manages flow control and thus improves network management and application performance.

This platform typically runs on a server and it tells switches the destination of packet transfer using various protocols. In this research python is used in minitet tools and a fat tree topology has been defined. This is also been used to write the load balancing algorithm Photon is an interpreter, object-oriented language suitable for many purposes. It has a clear, intuitive syntax, powerful high-level data structures, and a flexible dynamic type system. Python can be used interactively, in stand-alone scripts, for large programs, or as an extension language for existing applications. The language runs on Linux, Ubuntu, Macintosh, and Windows machines. There are also a number of system-specific extensions. A large library of standard modules written in python also exists. Compared to C, python programs are much shorter, and consequently written much faster.

IV. IMPLEMENTATION

Here we have used a terminal so that we can issue commands to the operating system and in our case it is ubuntu. Terminal is much more powerful than graphical interface.

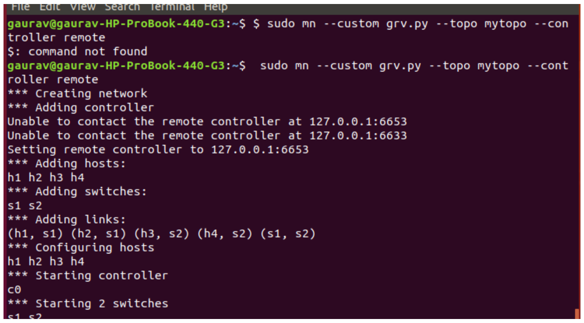

gaurav@gaurav-HP-ProBook-440-G3: ~
File Edit View Search Terminal Help

```
gaurav@gaurav-HP-ProBook-440-G3:~$ $ sudo mn --custom grv.py --topo mytopo --con
troller remote
$: command not found
gaurav@gaurav-HP-ProBook-440-G3:~$  sudo mn --custom grv.py --topo mytopo --cont
roller remote
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6653
Unable to contact the remote controller at 127.0.0.1:6633
Setting remote controller to 127.0.0.1:6653
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2
*** Adding links:
(h1, s1) (h2, s1) (h3, s2) (h4, s2) (s1, s2)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...
*** Starting CLI:
mininet>
```

Figure 4. Starting mininet controller

It is also called bash shell and supports a set of commands and utilities and uses its own programming language for writing shell scripts.

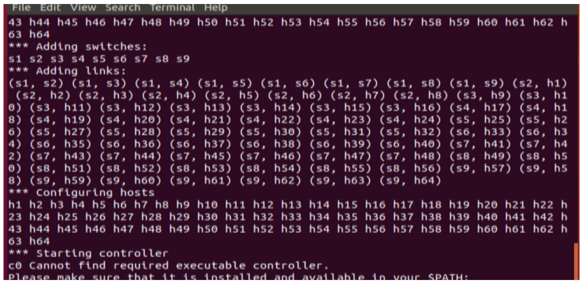
A. Mininet Installation



```
gaurav@gaurav-HP-ProBook-440-G3:~$ $ sudo mn --custom grv.py --topo mytopo --con
troller remote
$: command not found
gaurav@gaurav-HP-ProBook-440-G3:~$  sudo mn --custom grv.py --topo mytopo --cont
roller remote
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6653
Unable to contact the remote controller at 127.0.0.1:6633
Setting remote controller to 127.0.0.1:6653
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2
*** Adding links:
(h1, s1) (h2, s1) (h3, s2) (h4, s2) (s1, s2)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 2 switches
s1 s2
```

Figure 5. Mininet

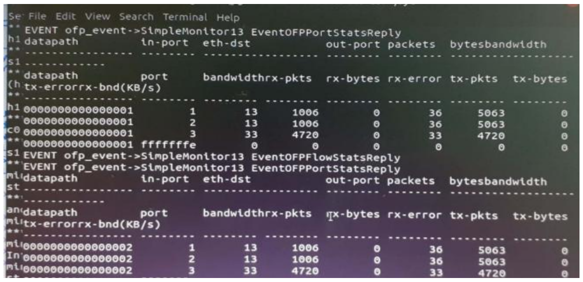
Mininet tool is installed by running the commands in the terminal such as sudo apt get install mininet. The other commands are -sudo mn—custom/mininet.py.



```
File  Edit  View  Search  Terminal  Help
43 h44 h45 h46 h47 h48 h49 h50 h51 h52 h53 h54 h55 h56 h57 h58 h59 h60 h61 h62 h
63 h64
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7 s8 s9
*** Adding links:
(s1, s2) (s1, s3) (s1, s4) (s1, s5) (s1, s6) (s1, s7) (s1, s8) (s1, s9) (s2, h1)
 (s2, h2) (s2, h3) (s2, h4) (s2, h5) (s2, h6) (s2, h7) (s2, h8) (s3, h9) (s3, h1
0) (s3, h11) (s3, h12) (s3, h13) (s3, h14) (s3, h15) (s3, h16) (s4, h17) (s4, h1
8) (s4, h19) (s4, h20) (s4, h21) (s4, h22) (s4, h23) (s4, h24) (s5, h25) (s5, h2
6) (s5, h27) (s5, h28) (s5, h29) (s5, h30) (s5, h31) (s5, h32) (s6, h33) (s6, h3
4) (s6, h35) (s6, h36) (s6, h37) (s6, h38) (s6, h39) (s6, h40) (s7, h41) (s7, h4
2) (s7, h43) (s7, h44) (s7, h45) (s7, h46) (s7, h47) (s7, h48) (s8, h49) (s8, h5
0) (s8, h51) (s8, h52) (s8, h53) (s8, h54) (s8, h55) (s8, h56) (s9, h57) (s9, h5
8) (s9, h59) (s9, h60) (s9, h61) (s9, h62) (s9, h63) (s9, h64)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16 h17 h18 h19 h20 h21 h22 h
23 h24 h25 h26 h27 h28 h29 h30 h31 h32 h33 h34 h35 h36 h37 h38 h39 h40 h41 h42 h
43 h44 h45 h46 h47 h48 h49 h50 h51 h52 h53 h54 h55 h56 h57 h58 h59 h60 h61 h62 h
63 h64
*** Starting controller
c0 Cannot find required executable controller.
Please make sure that it is installed and available in your $PATH:
```

Figure 6. Mininet topology

Providing quality of service was not the aim of traditional networks but as the time passed by they became a crucial demand that can't be ignored . Many people have tried to put their efforts to improve the quality of service and at times they have succeeded in doing so but a lot still needs to be done. These requirements are given in Service Level Agreement that are provided by the clients to their customers.



```
Se File  Edit  View  Search  Terminal  Help
** EVENT ofp_event->SimpleMonitor13 EventOFPPortStatsReply
h1 datapath         in-port  eth-dst            out-port packets  bytesbandwidth
** ---------------- -------- ------------------ -------- -------- -----------------
s1 -------------
** datapath         port     bandwidthrx-pkts   rx-bytes rx-error tx-pkts  tx-bytes
(h tx-errorrx-bnd(KB/s)
** ---------------- -------- -------- -------- -------- -------- -------- --------
h1 0000000000000001        1       13     1006        0       36     5063        0
** 0000000000000001        2       13     1006        0       36     5063        0
c0 0000000000000001        3       33     4720        0       33     4720        0
** 0000000000000001 fffffffe        0        0        0        0        0        0
s1 EVENT ofp_event->SimpleMonitor13 EventOFPFlowStatsReply
** EVENT ofp_event->SimpleMonitor13 EventOFPPortStatsReply
mi datapath         in-port  eth-dst            out-port packets  bytesbandwidth
st ---------------- -------- ------------------ -------- -------- -----------------
** -------------
an datapath         port     bandwidthrx-pkts   rx-bytes rx-error tx-pkts  tx-bytes
mi tx-errorrx-bnd(KB/s)
** ---------------- -------- -------- -------- -------- -------- -------- --------
mi 0000000000000002        1       13     1006        0       36     5063        0
In 0000000000000002        2       13     1006        0       36     5063        0
mi 0000000000000002        3       33     4720        0       33     4720        0
```

Figure 7. Ryu & mininet

The performance of a network is measured by various parameters they are

- The bandwidth should be minimum.There is network congestion and link capacity in traffic streams. If here we give minimum bandwidth then the receiver will receive the required data

- Latency should be maximum. It is the total time taken for the packet to travel from the source to destination. There can be various delays such as transmission delay, propagation delay, processing delay and delay in queuing. Thus these delay should be minimized .

- Packet loss ratio should be maximum. When there is any type of network congestion then the packets will be delivered with a delay. This type of delay mostly happens when buffering in any networking device reaches its maximum limit. We can rely on TCP for this however the packets that are dropped affect the congestion signal and hence result in delay of other packets.

- Jitter should be maximum. Jitter is the variance in the latency. Sometimes packet with different latency go in same traffic flow. This happens when network conditions change. This is not good and hence can affect the quality of the product.

V. CONCLUSION

We have analysed the traffic and have suggested various methods to overcome those . mininet tool is used with RYU controller. Bandwidth has been minimized and latency has been maximized . jitter is also maximized. The network has been analysed through wireshark and graph has been plotted.

REFERENCES

1. Nunes B., Marc Mendonca X. Nguyen, Katia Obraczka and Thierry Turletti. 2014. A survey of software defined networking: Past, present, and future of programmable networks. 1-18.
2. Kreutz D., Ramos F.M., Verissimo P.E., Rothenberg C.E., Azodolmolky S. and Uhlig S. 2015. Software Defined networking: A comprehensive survey. Proceedings of the IEEE. 103(1): 14-76.
3. Zhou W., Li L., Luo M. and Chou W. 2014, May. REST API design patterns for SDN northbound API. In Advanced Information Networking and Applications Workshops (WAINA), 2014 28th International Conference on (pp. 358-365). IEEE.
4. Kaur S., Kumar K., Singh J. and Ghumman N.S. 2015, March. Round-robin based load balancing in Software Defined Networking. In Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on (pp. 2136-2139). IEEE.
5. Yahya W., Basuki A. and Jiang J.R. 2015. The Extended Dijkstra's-based Load Balancing for OpenFlow Network. International Journal of Electrical and Computer Engineering. 5(2): 289.
6. Wang L. and Lu G. 2016, January. The dynamic sub topology load balancing algorithm for data center networks. In Information Networking (ICOIN), 2016 International Conference on (pp. 268-273). IEEE.
7. ONOS:https://wiki.onosproject.org/display/ONOS/Wiki+Ho me.
8. Mininet: http://mininet.org/.
9. Yang Liu, Jogesh K. Muppala, Malathi Veeraraghavan. 2014. A Survey of Data Center Network Architectures.

AUTHORS PROFILE

Aditi Rai, Btech Department of Computer Science And Engineering, SRM UNIVERSITY

Gaurav Tiwari, Btech Department of Computer Science and Engineering, SRM UNIVERSITY

Department of Computer Science and Engineering, SRM University