An Open Flow based Hybrid Control Plane in a Software Defined Network Environment

Girish C
PG Scholar
Dept of CSE, Vemana IT
Bengaluru, Karnataka, India

K.N. Narasimha Murthy, PhD
Professor
Dept of CSE, Vemana IT
Bengaluru, Karnataka, India

ABSTRACT
An OpenFlow control plane architecture characteristic of SDN limit the flow based scalability. To address the scalability problem, two control plane architectures such as flat and hierarchical methods for the improvement of the scalability of the SDN. However, two control plane architectures have issues: the Flat Architecture (FCPA) does not solve the time computing growth of the north bound (control plane), SDN network grows bigger in size; the Hierarchical Architecture (HCPA) problem with route path span. To address the two issues, propose a model named and known as (Orion) hybrid network model, Hybrid Control Plane Architecture (HYCPA) for scale software defined network environment. Hybrid Network model combines the control plane architectures of flat and hierarchical methods. In hybrid network model reduces the computing time and route span of north bound by constructing Abstract Hybrid Network View (AHNV). An Abstract Hybrid Routing Method (AHRM) is introduced to calculate the best route path. A hybrid Fast Reroute Method (HFRM) is constructed to achieve backup path for link removal and restore the previous original shortest path. Hybrid network model is implemented to verify the performance evaluation by calculating computing tine and route span with respect to number of links via the number of areas, flow rate and delay with respect to number of areas.

General Terms
Control Plane Architecture, Hybrid Network Model.

Keywords
SDN, Routing, Scalability.

1. INTRODUCTION
Software Defined Network (SDN) is a virtualized network environment architecture paradigm decouples north bound known as control plane (decision making) from south bound known as data plane (packet forwarder) entities for network application development. It runs and works on general purpose hardware to control an entire network with respect to north bound software. It enables programmable and flexible networks by supporting flow based management based on OpenFlow Application Protocol Interface (API).

An OpenFlow is an open interface x86 instruction set for the SDN is based on Ethernet switches with an internal flow tables and a standard interface for adding and removing flow entries. It constituently divides the bounds with respect to north and south plane. The techniques in OpenFlow control components are: (i) Maestro runs parallel in a single thread control component; (ii) Beacon works on multiple thread strategy to increase scalability of a single control component (controller). The two techniques limit the communication gap via the multiple control components.

The techniques in flow based SDN are: (i) DevoFlow handle flows with respect to SDN control component and create more loads on the controller and nodes (switches). The north bound maintains complete visibility without unnecessary work load and costs. (ii) DIFANE techniques consider the north bound authority switches to share work and load via storing rules and policies. The techniques require modification in the OpenFlow switches or nodes.

The different structure of north bound designs conducts different north bound processing ability. The control plane architecture is categorized into two: (i) Flat (FCPA); (ii) Hierarchical (HCPA).

The control plane methods with respect to flat architecture are: (a) HyperFlow is a sharing based north bound in an OpenFlow occurrence; (b) Onix implemented is a sharing model runs on group of working servers; (c) ONOS is an open source sharing operating system in a SDN environment which is scalable and achieve the complete network view with respect to the information of components.

The control plane methods with respect to hierarchical architecture are: (a) Logical X Bar is an iterative constructing blocks to implement abstract hierarchic layers of networks. (b) Kandoo is two level controller hierarchies, the bottom level works on local applications based on bottom view and the top level runs top most applications based on top network-wide view.

In order to address the two issues of computing time complexity and route span, propose Orion hybrid network model – a Hybrid Control Plane Architecture (HYCPA) for scalable, relative and combines the features of flat and hierarchical control plane, and addresses the two issues.

First design (Orion) hybrid network model, a Hybrid Control Plane Architecture which reduces the computing time growth and route path span by constructing a method named as Abstract Hybrid Network View (AHNV). Second design a method named as Abstract Hybrid Routing Method (AHRM) to address the problem of best path by calculating all intra-area links with respect to area controller and inter-area links with respect to domain controller. Third design a method named as Hybrid Fast Reroute Method (HFRM) is to demonstrate and achieve fast reroute for backup and restore original path in the proposed hybrid network model. Finally implement hybrid network model to verify the efficiency and its effectiveness the performance evaluation.

This paper structure is organized as follows: Section II gives a brief description about related work, section III briefs about proposed model, section IV contains implementation, evaluation is described in section V, conclusion is discussed in section VI, future scope is described in section VII.

2. RELATED WORK

Scalability problem is one of the most challenging problems in the Software Defined Network Environment. The OpenFlow based Control Plane Architecture in SDN has attracted attention of the research community in the last few years. Many researchers around the years of research have come up with new ways of facing this challenge.

In [2] the authors proposed a HyperFlow OpenFlow control component – a flow based sharing occurrence based on flat architecture (FCPA) run on Nox application. This method provides network control centralization. It localizes decision making to each single controller, thus minimizing the control plane response time to data plane requests.

In DevoFlow method [3] categorized in flow based SDN – need changes in the OpenFlow Controller model which divides coupling between north bound and south bound plane which maintains a useful sum of visibility without extra costs. It reduce node internal communication via the north bound and south bound plane by transfer the setups and invoke the flow statistics.

In Logical X Bar [4] the authors proposed hierarchical architecture with respect to control plane for large scale of networks by building iterative blocks. The structure combines ability to control and scale overall network and continue function with respect to local sites or links which are failed.

In the flat architecture – authors proposed Onix [5] a platform in a network north bound plane developed as a sharing structure. The important challenges in developing and release evaluation in a production quality north bound platform are general, simple, scalable north bound plane performance.

In ONOS categorized under the flat architecture [6], the authors proposed an experimental test sharing SDN north bound platform motivates improving performance, scaling, and available requirements of large operational networks. This method proposes a two version of flat architecture (FCPA): (i) the first version implemented main features – a sharing, but central component, overall flat network view, scalable and fault tolerance; (ii) the second version focused on improves performance release evaluation. The challenging requirements are larger throughput which is measured in million requests per second, lesser latency which is measured in milliseconds; domain size is measured with respect to data in terabytes, and higher service available which is measured in percentage.

Kandoo [7] proposed a hierarchy framework of north bound plane architecture for preserve the need of scalable in north bound without changing the nodes.

Maestro [9] is a scalable OpenFlow Controller designed and works on Nox application with simple pseudo structure constructs with multi thread design goals such as: distribute work evenly, minimize overhead and memory consumption. It is effective and efficient in handle process of messages of scalable OpenFlow interface in a domain.

Beacon [10] is a java based free OpenFlow Controller which is achieved by the following construction goals: (i) to increase production of control components; (ii) To provide higher performance; (iii) Give existing and new applications to start and stop with respect to runtime ability.

DIFANE [11] is a scalable and effective and efficient solution which keeps traffic in the south bound plane in selecting rules of directed packets via the nodes and store information. The rule functions in the north bound of the DIFANE SDN control component are: (i) Compute rules before itself; (ii) Utilize rules to separate the capacity; (iii) Split – up algorithm is used to divide the rules with respect to the north bound plane; (iv) Constitute divide a small group of split –up rules; (v) Replicate important rules for reducing span and fail retrieval state.

ElastiCon [12] in this author proposed an elastic based distributed control component in this method the group is dynamic expansion and shrinks with respect to conditions of SDN traffic. This method focusing explores migration API to guarantee adapt and balance load which require buffer messages from the nodes.

WA – SDN [13] with multiple Controllers is a framework for software defined domain solution with a vertical structure design and is a free model of communication.

3. PROPOSED MODEL

The design structure of hybrid network model, a Hybrid Control Plane Architecture (HYCPA) combines the existing methods such as flat and hierarchical model which focus on calculating the edges of the network, calculate shortest path of inter area and intra area by routing method and keep a backup path for routing method whenever a link failure happens. The Orion model (See Figure 1.) is divided into two parts and the description is given below

(1) The bottom level with respect to hybrid model is an area control component which contains areas. The bottom level interacts with the domain controller in which responsible for grouping areas, switches and hosts information. In this controller manage the intra – area structure information, process routing with respective to the switches and their respective links. It gives view of bottom level information to upper domain controller level.

(2) The top level with respect to hybrid network model is domain control component level in which tends bottom level controller represents sub networks / areas, operate the view of the complete network with respect to database. This layer manages inter and intra – area structure by process inter and intra – area route invocation and up gradations. In top level view of domain control component information is from the admin controller.

The bottom level demonstrates the group information for switches, links and hosts in each area with their respective information. The top layer view is of the abstract area. In this level, every area is considered as a sub network, information or data for switches and links in the area are considered as the ports of the sub network.

3.1 Module Description

There are four modules in the proposed system in which are categorized with respect to control components as admin, domain, link discovery and area are described below.

3.1.1 Admin Controller

The admin controller is accountable for operating complete network information. In this module physical switches and their links are created in the control plane controller by using the emulation tool named mininet.

The node information, MAC address, port information of the physical topology network is given as input in the domain controller. The python script is written in the mininet to export the physical topology in the admin controller. The admin controller is an OpenDayLight Controller where the physical topology of network is represented.

3.1.2 Domain Controller

Domain controller is a java based application which provides inter – area host information and manage the global edge switches information and physical topology of the admin controller. The module keeps track of the hosts, switches and links information for every area with respect to inter – area information. A NoSQL database is used store global link information, flow information and port information. The routing method will add flow routing information when the area controller information is interacted among the domain controller.

In the domain controller component demonstrate the upper complete domain topology information and determines inter – area route path with information of the route path of intra area obtained from the area controller. Then Domain ID is used in the area controller which calculates the combination of links switches and hosts for route path. Every domain control component flow entries are interacted with the area control components with respect to the Domain ID.

They are routing modules in the domain control component module. They are as follows:

1) Inter Routing: The domain control component module compute the route information of inter –area path with respect to admin controller network demonstration view.

2) Fast Inter Routing: The domain controller interact with every area control component information considers area with sub network port information for bringing out the restore path for the outside sub network route. The domain fast inter restore route which represent the computing the outside the area backup route information.

3.1.3 Area Controller

The module is a java based application below the domain control module manages the edge information of the switches and hosts calculated in the link discovery block. In this area control component for areas information of switches, links and hosts.

This module connects through the Domain ID. When each area, hosts and switches information is interacted with domain control component and the intra – area route information is calculated for shortest path for each area from the area controller.

They are routing modules in the area control component module. They are as follows:

(1) Intra Routing: The area control routing component module is implemented for computing route information of intra model information with respect to the domain control component view.

(2) Fast Intra Routing: Keep a backup path with respect to sub network path, the area control component input the area information to the domain controller which compute best route path for switches-links-hosts combination for every area with respect to the root route path, earlier compute next nearest shortest route path for each switches-links-hosts pair by its area. The restore route module is computing the backup area information.

In the link discovery block in the area control component keeps track of every area change route with respect to domain controller module. Finally, the restore route module setup new flow information on the nodes with respect to their links in the new path entry.

3.1.4 Link Discovery

The link discovery module calculates the edges of a network with respect to the domain controller module for inter area route computation and area controller module for intra area route calculation. The edges include the link, port, flow information which is stored in the MongoDB with respect to the domain controller. This module keeps track of the link table, port table and flow table information. This module creates edges for switches, hosts and links with their bandwidth information.

The link, port and flow information is acquired for calculating inter and intra area information in the area and domain controller and also for calculating the backup path for each area and domain controller.

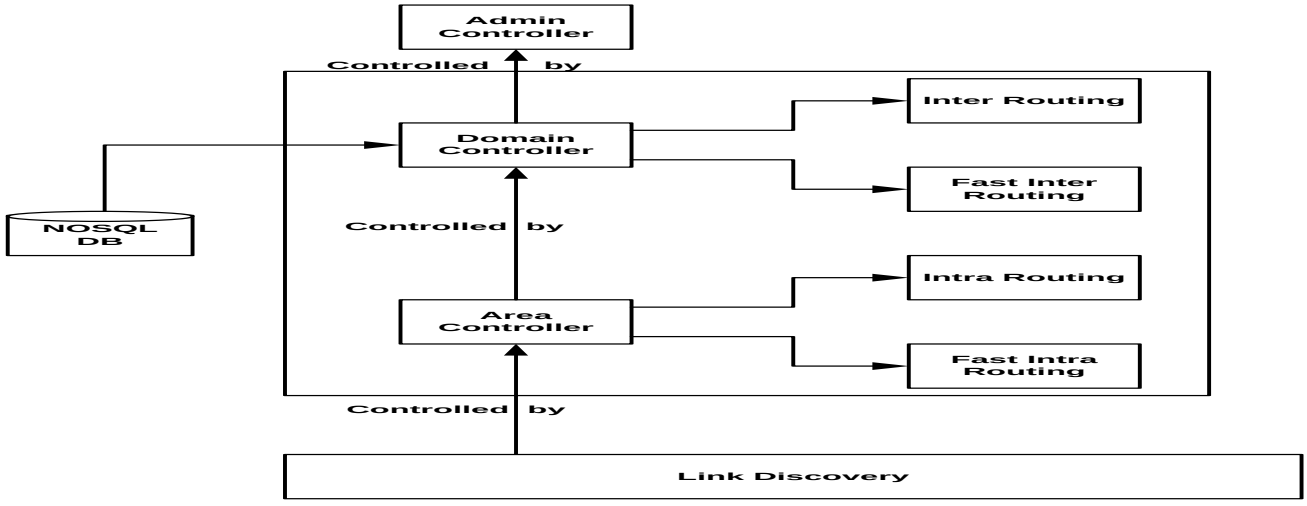

Fig 1: System Architecture

4. IMPLEMENTATION

The implementation of hybrid network model as three methods such as Abstract Hybrid Network View (AHNV) which deals with the view of network, Abstract Hybrid Routing Method (AHRM) deals with computing the inner and outer network route information and Hybrid Fast Reroute Method (HYRM) deals with restore and remove link.

4.1 Abstract Hybrid Network View

In this method the physical topology of the network is emulated through Mininet software in the OpenDayLight Controller. The python script is written in the Mininet for creating hosts, areas, links with bandwidth and the sample pseudo code for hosts, areas, and links is shown below:

#Hosts

Host name = self.addHost ('host name')

Area

Switch name = self.addSwitch ('switch name')

Area links

Self.addLink (Host name, Switch name, cls=TCLink, bw =10, delay='1ms', max_queue_size=20)

After the code written in the Mininet software for a custom topology and execute by using the following command: sudo mn –controller = remote, ip = [address], port = 6633, --custom hybridsdn.py –topo mytopo. The physical topology of the custom network is represented in the OpenDayLight Controller (See Figure 2.).

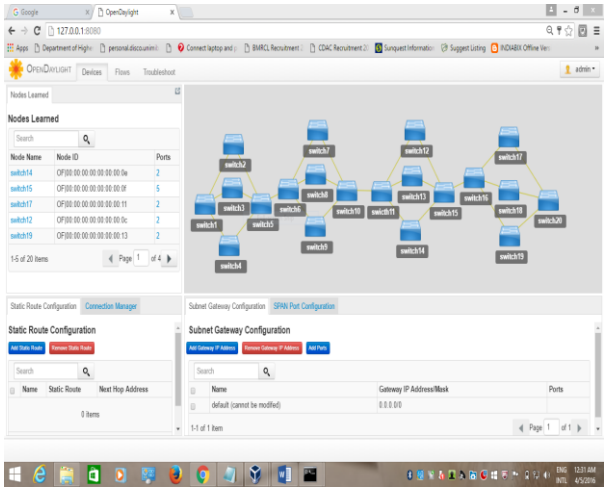

Fig 2: Custom Topology in ODL

In the OpenDayLight Controller the node ID, MAC address, port information is noted and the information of switch ID, Ethernet port, software port from the Controller and the source ID, destination ID, input port, output port and bandwidth from the Mininet is noted. In the Domain and Area Controller java application the edges of the custom topology network is created which include the links and their respective bandwidth information.

In the Abstract Hybrid Network View (AHNV) method use the link discovery module to calculate the edges of the custom physical network with respect to the domain and area controller

4.2 Abstract Hybrid Routing Method

In the Domain and Area Controller the routing method is used to calculate the edges of the custom network with their bandwidth information and use Dijkstra' algorithm to calculate the shortest path for inter area and intra area network. The intra area routing is calculating shortest path for each individual areas by installing flows for each areas in the Area Controller and remove flows after routing operation.

Firstly configure the area controller (See Figure 3.) by input the domain ID, area name, switch ID and host ID, which connects to domain controller for intra routing (See Figure 4.).

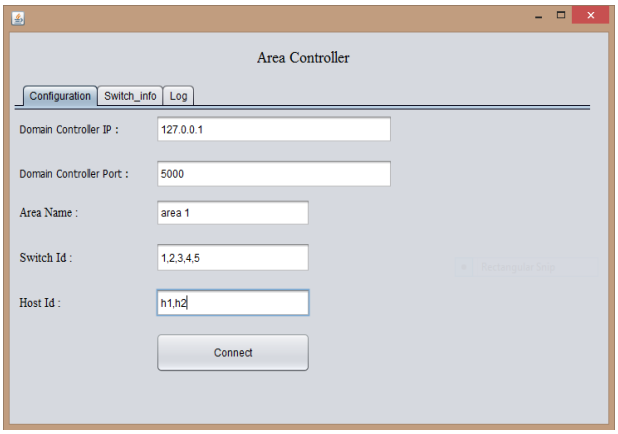

Fig 3: Configure Area Controller

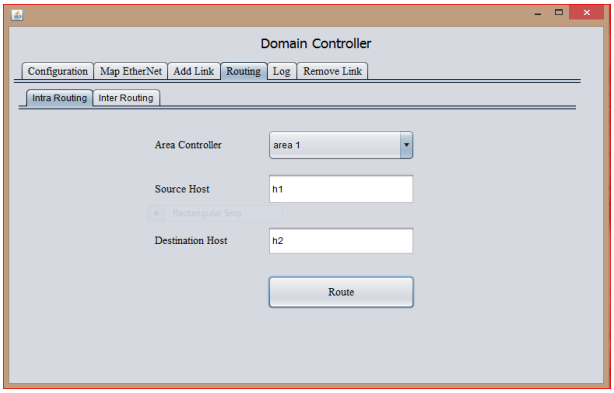

Fig 4: Intra Routing

The inter area routing is calculating shortest path for complete custom network or between the areas in the Domain Controller (See Figure 5.).

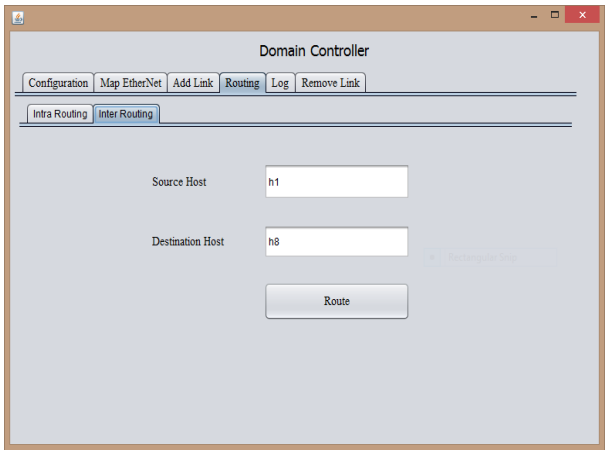

Fig 5: Inter Routing

In the Abstract Hybrid Routing (AHRM) Method uses the domain and area controller module with respect to link discovery module for the computed links, ports and flow information for inter route computation for domain controller and intra route computation for area controller.

4.3 Hybrid Fast Reroute Method

In this method whenever the link is removed it takes a backup path for calculating the shortest path and the link can be restored and it will update the previous shortest path (See Figure 6.). When the link is removed the inter and intra area routing can be done for calculating the shortest path and restore the link of previous existing shortest path for inter and intra area routing.

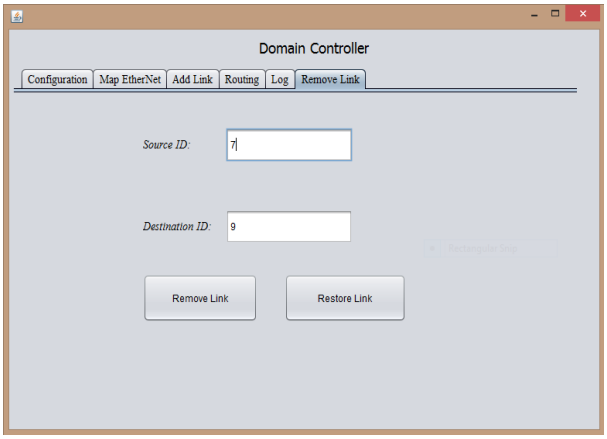

Fig 6: Remove and Restore Link

In the Hybrid Fast Reroute (HFRM) Method uses the link discovery for computed edges, domain controller for the inter fast reroute and area controller for intra fast reroute.

5. EVALUATION

The performance evaluation of the hybrid network model is show in four different graphs, the network is separated as areas where each area as a number of switches, links and hosts. The calculation and computation of each area is done for different parameters.

After the input of the switch ID, source ID, Destination ID, bandwidth, input port and output port the number of links in the domain controller is added based on the custom network topology with respective different areas for calculating route compute time and route span.

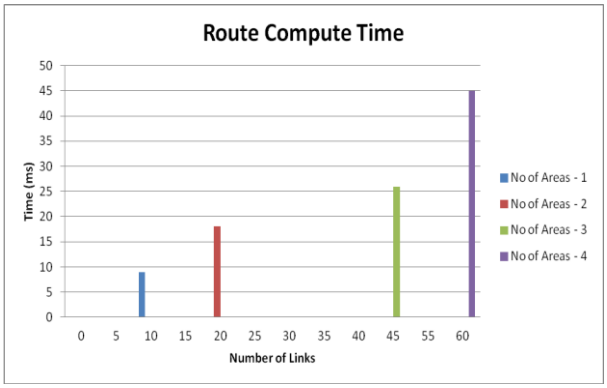

Fig 7: Route Compute Time with different no of areas

In the route computation time graph of different areas such as 1,2,3,4 where each area as different number of switches, hosts with respect to their links, where the links varies based on the number of areas (See Figure 7.).

In the route span graph of different areas such as 1,2,3,4 where each area as different number of switches, hosts with respect to their links, where the links varies based on the number of areas and the span is the time count of stretch of areas with respect to links where the odd number of areas such as 1 and 3 as a odd span and even number of area such as 2 and 4 as a even span (See Figure 8.).

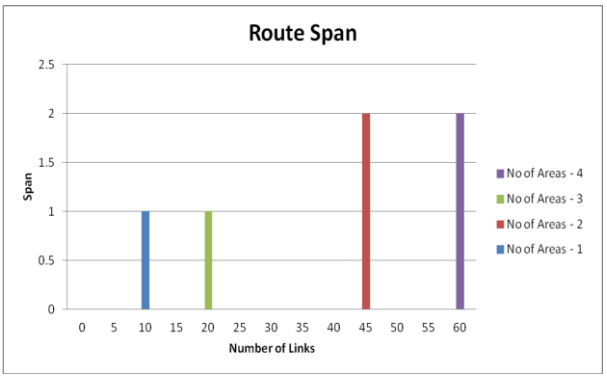

Fig 8: Route Span with different no of areas

After the computation of inter routing and intra routing in the domain controller and area controller the custom network is evaluated with respect to the flow rate and delay of each areas.

In the flow rate graph of each areas is calculated with respect to the area numbers where the unit is bytes per second and the areas increase from 1 to 4 the flow rate increases from low to high which indicates the as the number of areas increases the flow rate increases the performance (See Figure 9.).

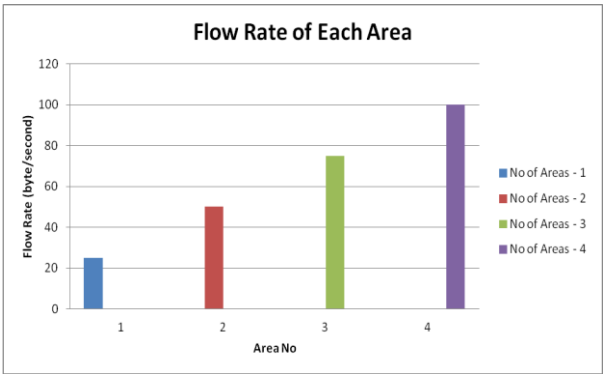

Fig 9: Flow Rate of each area

In the delay graph of each areas is calculated with respect to the area numbers in which the delay reduces as the area number increases (See Figure 10.).

In the existing system the issue related to computing time and span is to take larger time or count but in the present system it is minimal with respect to restricted four number of areas.

The flow rate and delay of each different number of areas are showing good performance with respect to the domain. The four performance evaluations graphs are generated in java based jfree charts and converted them to Microsoft excel charts.

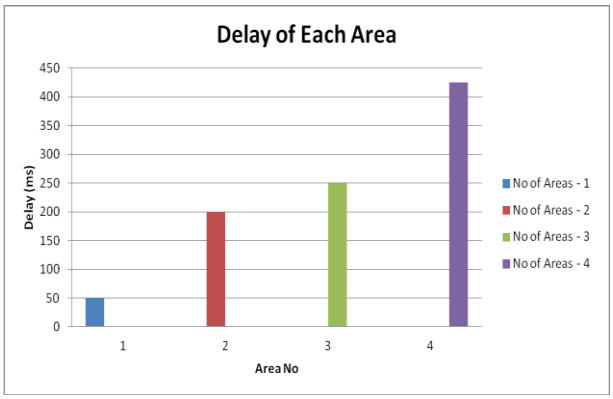

Fig 10: Delay of each areas

6. CONCLUSION

The proposed Hybrid Control Plane Architecture (Orion model) has more advantages than other existing methods. The hybrid network model reduces the computing time and problem of path span. By using Dijkstra's algorithm the proposed Hybrid Control Plane Architecture compute layered network views, routing for inter and intra area controllers and fast reroute for inter and intra area controllers. By implementing hybrid network model verify the performance evaluation of the hybrid approach by calculating the flow rate, delay of the areas.

7. FUTURE SCOPE

From the existing systems some features are implemented and therefore it overcomes the limitations in the proposed method and the project requires future improvement with respect to increase in number of switches, hosts and links and by extracting these network information in the OpenDayLight (ODL) controller itself but in the present system the java based domain and area controller are created and make the hybrid model and in the future scope, design the domain and area controller by using the bundles or plug-ins in the ODL controller itself and perform the computation of time, span, routing of the areas and backup of path with respect to the domain network.

8. ACKNOWLEDGMENTS

My sincere thanks to Dr. K.N. Narasimha Murthy, Professor and Head (PG), Dept of CSE, Vemana IT, mentorship and guidance in this project work with respect to his valued advice, co-operation and to management for their constant support. My special thanks to the BMSIT management for conducting "National Conference on Recent Trends in Information Technology (NCRTIT – 2016)".

9. REFERENCES

[1] N.McKeown, T.Anderson, H.Balakrishnan, G.Parulkar, L.Peterson, J.Rexford, S.Shenker and J.Turner, "OpenFlow: Enabling innovation in campus networks," ACM SIGCOMM Comput. Commun. Rev., vol. 38, no. 2, pp. 69-74, Apr.2008.

[2] A.Tootoocian and Y.Ganjali, "HyperFlow: A distributed control plane for OpenFlow," in Proc. ACM INM/WREN, pp. 1-6, 2010.

[3] A. R.Curtis, J.C.Mogul, J.Tourrilhes and P.Yalagandula, "DevoFlow: Scaling flow management for high performance networks," in Proc. ACM SIGCOMM, pp. 254-265, 2011.

[4] J.McCauley, A.Panda, M.Casado, T.Koponen, S.Shenker, "Extending SDN to large – scale networks," in Proc. ONS, pp. 1-2, 2013.

[5] T.Koponen, M.Casado, N.Gude, J.Stribling, L.Poutievski, M.Zhu, R.Ramanathan, Y.Iwata, H.Inoue, T.Hama and S.Shenker, "Onix: A distributed control platform for large – scale production networks," in Proc. OSDI, pp. 1-6, 2010.

[6] P.Berde, M.Gerole, J.Hart, Y.Higuchi, M.Kobayashi, T.Koide, B.Lantz, B.O'Connor, P.Radoslavov, W.Snow and G.Parulkar, "ONOS: Towards an open, distributed SDN OS," in Proc. ACM SIGCOMM HotSDN, pp. 1-6, 2014.

[7] S.H.Yeganeh and Y.Ganjali, "Kandoo: A framework for efficient and scalable offloading of control applications," in Proc. ACM SIGCOMM HotSDN, pp. 19-24, 2012.

[8] M.P.Fernandez, "Evaluating OpenFlow controller paradigms," in Proc. 12th ICN, pp.151-155, 2013.

[9] Z.Cai, A.L.Cox, and T.S.E.Ng, "Maestro: A system for scalable OpenFlow Control," Rice Univ., Houston, TX, USA, Tech. Rep., 2010.

[10] D.Erickson, "The Beacon OpenFlow controller," in Proc. ACM SIGCOMM HotSDN, pp. 13-18, 2013.

[11] M.Yu, J.Rexford, M.J.Freedman, and J.Wang, "Scalable flow – based networking with DIFANE," in Proc. ACM SIGCOMM, pp. 351-362, 2010.

[12] A.Dixit, F.Hao, S.Mukherjee, T.V.Lakshman, and R.Kompella, "Towards an elastic distributed SDN controller," in Proc. ACM SIGCOMM HotSDN, pp. 7-12, 2013.

[13] R.Ahmed and R.Boutaba, "Design considerations for managing wide area software defined networks," IEEE Commun. Mag., vol. 52, no. 7, pp. 116-123, July. 2014.