Optimal placement of Controllers for the Adoption of Software Defined Networking: In the case of ethiotelecom

BY: ZEMENE MARKOS
ADVISER: EPHREM TESHALE (PHD)

A Thesis submitted to
School of Electrical and Computer Engineering
Addis Ababa Institute of Technology

in Partial Fulfillment of the Requirements for the Degree of Master of Science
(Telecommunication Engineering)

Addis Ababa University
Addis Ababa, Ethiopia
November 14, 2018

Declaration

I, the undersigned, declare that the thesis comprises my own work in compliance with internationally accepted practices; I have fully acknowledged and referred all materials used in this thesis work.

Zemene Markos

_____ _____

Name Signature

Addis Ababa University
Addis Ababa Institute of Technology
School of Electrical and Computer Engineering

This is to certify that the thesis prepared by **Zemene Markos**, entitled *Optimal placement of Controllers for the Adoption of Software Defined Networking: In the case of ethiotelecom* and submitted in partial fulfillment of the requirements for the degree of Master of Science (Telecommunication Engineering) complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

Internal Examiner	_____	Signature	_____	Date	_____
External Examiner	_____	Signature	_____	Date	_____
Adviser	Ephrem Teshale (PhD)	Signature	_____	Date	_____
Co-Adviser	_____	Signature	_____	Date	_____

_____

Dean, School of Electrical and Computer Engineering

ABSTRACT

The tremendous increase of data traffic has been a key reason for the upgrading of traditional networks. One of the new models that has been developed for redesigning and managing communication networks is software-defined networking (SDN). The main idea behind SDN is the decoupling of the control and data planes, which enables the centralization of the control plane and the programmability of the data plane. The decoupling of control and data planes in SDN brings benefits in terms of logically centralized control and application programming. But, the single point of management in physically centralized SDN architectures is a potential point of failure and a bottleneck that compromises network reliability and performance.

To avoid such concerns, SDN control architectures are usually designed as physically distributed systems. This raises practical challenges about the best approach to decentralize the control plane while maintaining the logically centralized network view. In particular, determining the number of controllers and locating them in a wide area network (WAN) is a challenging task that should be addressed appropriately.

The work presented in this thesis addresses the above challenges through clustering and heuristic based optimization models. ethiotelecom WAN was considered as a case study; the network topology is encoded, examined, and the selected algorithms for determining the minimum number of controllers and their optimal location in the WAN, were applied with the goal of enhancing and minimizing global latency and considering the node to controller assignment imbalance.

The optimization is done in two scenarios, controller placement using K Medoid method and Pareto simulated annealing (PSA) method. First, the controllers placement of K-Medoid and Pareto simulated annealing algorithm for different numbers of controllers K is found and investigated. Followed by, a comparison of the results found using k-Medoids clustering model with the Pareto simulated annealing heuristic model is conducted. From the results it is found that the minimum number of controllers required to adopt SDN in ethiotelecom WAN is eight controllers and the k medoid clustering based model is better than the Pareto simulated annealing method in the optimal placement of controllers in the WAN topology for larger number of network elements (Nodes).

KEYWORDS

Software-Defined Networking (SDN), OpenFlow, Wide Area Networks (WAN), Controller placement, Optimization, Clustering, Control plane, Data plane.

ACKNOWLEDGMENTS

First and foremost, I would like to give special thanks and glory to the Omnipotent, Omnipresent, and the Omniscient Almighty God who is the wonderful source of my strength. He gave me the grace, wisdom, good health and guided me in all the ways of my life.

I would like to express my deep and sincere gratitude to my advisor Dr.Ephrem Teshale for his academically essential guidance, hints, motivation, ideas, support and valuable comments over the course of my thesis research. Throughout my thesis work, I have received so much help from him and completing this thesis work would have been all the most difficult were it not for his support. My special thanks also goes to Mr.Yiheyis Takele, ethiotelecom transmission engineering section manager, for giving me valuable resources on ethiotelecoms wide area network topology data.

I am forever indebted to my devoted Mother, Almaz G.Aregay for her endless support, tremendous wisdom, guidance, integrity, selflessness, faithful prayers, abounding and undying love. She has the most impact in my life and I never would have made it here without her endless support and encouragement. My special thanks also goes to my wife for the love, care and encouragement you have bestowed upon me, through thick and thin.

Finally, I would like to thank all my friends (too many to list here but you know who you are!) whom I respect and love dearly. Thank you for the constant encouragement during all these years.

Thank you ALL!!!

Addis Ababa, 15th October 2018

Zemene Markos

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

AI	Artificial Intelligence
ARP	Address Resolution Protocol
ATM	Asynchronous Transfer Mode
BFD	Bidirectional Forwarding Detection
BGP	the Border Gateway Protocol
CAGR	Compound Annual Growth Rate
CLI	Command Line Interface
CPP	Controller Placement Problem
DCPP	Dynamic Controller Provisioning Problem
DHCP	Dynamic Host Configuration Protocol
DSL	Digital Subscriber Line
DUs	Data Units
DWDM	Dense Wavelength Division Multiplexing
EB	Exabyte
ILP	Integer linear program
IoT	Internet of Things
ISDN	Integrated Services Digital Network
LLDP	Link Layer Discovery Protocol
MOCO	Multi-Objective Combinatorial Optimization
MPLS	Multiprotocol Label Switching
NOS	Network operating system
ONF	Open Networking Foundation
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
PSA	Pareto Simulated Annealing
PSTN	Public switched telephone network

SDH	Synchronous Digital Hierarchy
SDN	Software defined networking
SD-WAN	Software defined wide area network
TLS	Transport Layer Security
VSAT	Very Small Aperture Terminal
ZB	Zettabyte

1

CHAPTER ONE:INTRODUCTION

1.1 OVERVIEW

In the current telecommunication networks, telecom network operators are responsible for providing a network configuration, which is adequately robust to deal with a wide range of network events and applications [1]. It is difficult to achieve this because:

(i) the state of the networks can change continuously and today's networks do not provide a mechanism to automatically respond to the wide range of events that may occur and

(ii) the static nature of current network devices does not permit detailed control-plane configuration, given that the hardware and software are provided by the manufacturer and cannot be customized.

On the otherhand in traditional networks, both the control and data planes are embedded within the same network node (hardware). The control plane manages the configuration and programming of paths for data flows. This control information is then used for data forwarding. However, this mechanism is not dynamic because once the control information is defined, it cannot be changed unless reconfiguration is done. This constraint makes the traditional networking approach less reactive towards changing traffic dynamics.

Networks need to be manageable. As the complexity of network deployments increases, the number of devices deployed increases as well. Manual configuration of such devices is error-prone and time-consuming. As a consequence, one expectation of future networks is to provide mechanisms to the operators to simplify network management, reduce the amount of time needed to configure devices and minimize configuration errors.

Networks of the future need to be flexible and dynamic to accommodate the expected demand. For example, as per Cisco visual networking index, the overall IP traffic will grow at a Compound Annual Growth Rate (CAGR) of 24% from 2016 to 2021 [2]. Busy-hour Internet traffic will increase by a factor of 4.6 between 2016 and 2021, while average Internet traffic will increase by a factor of 3.2 and Smartphone traffic will exceed PC traffic by 2021. Where the data load could reach 3.3 ZB per year (ZB; 1000 Exabyte [EB]), or 278 EB per month. It clearly shows that, networks need to provide sufficient bandwidth to transfer such huge amounts of data. However, they also need to be dynamic and flexible so that data transfers can be initiated when needed.

Software-defined Networking (SDN) is an emerging networking paradigm with a great potential to nurture innovation through programmable networks. SDN networks are characterized by the separation of the control and data planes wherein a logically centralized controller performs routing decisions on behalf of forwarding elements. Flexibility and dynamic control are the features that are the base for the emergence of software defined networks [3]. A centralized controller takes over the control from the individual nodes. Network operating system (NOS) collects the information, makes an abstract model of the topology, and facilitates for the controller that hosts the applications. The controller defines a set of rules that govern the actions of the switching node. This complete knowledge of the network helps the controller in flow management and dynamic provision of resources.

A decade ago, OpenFlow was introduced as a protocol for communication between the control plane (controller) and data plane (network devices). Since then, there have been many evolutionary improvements in this field. Today, SDN is mature enough to be deployed in operational networks. It has gained a lot of attention from network operators, equipment vendors and over-the-top application service providers.

Even though SDN is mature enough to be deployed in operational network, adoption of SDN to the existing networks raise many questions related to reliability, scalability, performance, and security. One of the common problems associated with Software defined wide area network (SD-WAN) is the placement of the controllers. This thesis focuses on the adoption of SDN controllers in the wide area networks and it will evaluate ethiotelecom wide area transport network and recommend the minimum number of controllers required and their placement to adopt SDN to this network.

1.2 STATEMENT OF THE PROBLEM

Traditionally, the data and the control planes in the Ethernet networking devices (and most of the communication equipments) have been tied together. This means, the prevailing operating system and its features with the provided hardware are implemented in a single device. Therefore, network devices, such as switches, routers, firewalls, etc., are built with the intelligence of handling traffic relative to the adjacent devices. This makes the intelligence distributed and scattered in the network. In addition, most of the network devices are Command Line Interface (CLI) based and configuration is done separately per device, making configuration slow and prone to errors. This prevents the networking industry of responding quickly to feature requests or innovate new management abilities [4].

SDN concept was the first time introduced in 2010 [5], as the new networking paradigm which aims to ease the control and the management of a computer network environment. It is a new evolutionary concept for network architecture, which separates the control plane from the data plane. The separation helps in better management of the network with efficient handling of the network traffic

on different planes of the SDNs architecture.

The single SDN controller within a network has many benefits, one of them is that the controller has a single view of the whole network and algorithms such as routing are performed using that view, but disadvantages still exist. The problem with a single controller is that it is the only controller in the network, has problems with redundancy, load balancing, network traffics are mostly known to be bursty and a controller may get overloaded. Even a single controller sometimes is difficult to place in a network because one needs to know where to place it. This placement problem particularly holds when the network includes multiple geographical regions such as wide area transport networks.

A solution to the single controller problem is to use multiple controllers. Multiple controllers can reduce the latency, increase the scalability and fault tolerance, and provide availability in SDN deployment. Even though Having multiple controllers solves the above mentioned problems, it also comes with its own challenges. One of the problem is, the placement of such distributed multiple SDN controllers. The research question then becomes:

- How many controllers are needed?
- Where in the topology should they be placed optimally?

The answer to this question relies on the network characteristics themselves and what is chosen to measure when deciding where to place the controllers.

Adopting SDN technology in SD-WAN is a new trend in the networking field. It provides a globally central view of the current state of the network which can be used to make optimal decisions for resource allocation, unlike legacy WAN which is based on local decisions made by each network node.

1.3 OBJECTIVES

1.3.1 *General Objective*

The general objective of this research is to analyze the performance of clustering based controller placement optimization models using network parameters in a realistic telecom operator network of ethiotelecom and recommend the minimum number and optimal placement of controllers on the network topology.

1.3.2 *Specific Objectives*

The specific aims of the research are:

- To identify appropriate clustering algorithm and tools for controller placement problems.
- To prepare the WAN network topology data and encode it into suitable format for the selected algorithms

- To study the performance of different clustering algorithms and from results of the models recommend the minimum number of controllers required and its placement on the network topology.

1.4 SCOPE AND LIMITATION

1.4.1 *Scope of the Study*

It is clear nowadays that the digital world is moving towards cloud-based everything, from applications to network communication. With the increasing demand for high-bandwidth network traffic that is expected from technologies like Artificial Intelligence (AI), Internet of Things (IoT)and 5G, enterprises will need huge amount of interconnection bandwidth to function properly. This amount of traffic will require a more scalable, reliable and dynamic network model. SD-WAN is designed to allow enterprises not only to connect their branch offices but also to provide fast connectivity to all of the necessary applications.

Adopting SDN technology in SD-WAN is a new trend in networking field. It provides a globally central view of the current state of the network which can be used to make optimal decisions for resource allocation, unlike legacy WAN which is based on local decisions made by each network node.

This study lies within this context, and is dedicated to the optimization of controller placement in WAN in order to minimize the total number of controllers to be used and with out the degradation of quality of services. ethio telecom WAN has been taken as a case study.

1.4.2 *Limitation of the Study*

The system model in this thesis considered the main contributors like node to controller delay, and controller to controller delay and nodes to controller assignment load imbalance. Assuming that all controllers are identical, Queuing at the controllers and controller capacity constraints are not considered. The node to controller and controller to controller delay is computed considering geographical distance rather than cable length. However we are confident that taking exact value of delay will not alter the research findings significantly.

1.5 CONTRIBUTIONS OF THE RESEARCH

At the end of this research, the result will have both practical and theoretical significance. The increase of new computing technologies such as big data, Internet of Things (IoT) and cloud services fundamentally change the way we store, acquire and transfer information and data. Accompanied with these new computing trends, telecom operators are facing challenges in network upgrade and management, resource utilization, and high-speed transmission. To tackle these challenges, telecom operators like ethiotelecom need to deploy SDN which is envisioned as

a promising paradigm for the next-generation networking. The research enables ethio telecom to have an insight of how the company can adapt SDN on its wide area transport network.

It also indicates and explores the suitable algorithm and model that can optimize the SDN controller placement problem to adopt software defined networking with minimum number of controllers. The accuracy level is determined by the type of algorithm and techniques used during the research.

This research also could be the opening point to conduct researches in ethiotelecom Wide area transport network. It could be used as a reference for further researches in the area and make it available for academic reference.

1.6 LITERATURE REVIEW

This section contains a brief discussion of the relevant studies related to the SDN controller placement problem. The primary emphasis of the work presented in this research was on the optimization of average latency in large-scale SDN networks.

Since the first introduction of the SDN controller, Controller placement in SDN is one of the critical problems and has attracted extensive attention in the literature.The controller placement aims to find out the best k locations in an SDN-enabled network to place controllers in order to enhance the efficiency of network management [6]. Heller et al. [7] motivated the SDN Controller Placement Problem (CPP) and studied how many controllers are needed and where in the network they should be placed. They argued that in most topologies one controller is enough for meeting latency requirements but insufficient for achieving resilience. They treated the placement of k controllers as a variant of the facility location problem. Their work was extended by [8] to include resilience aspects.

Aoki et al.[9] examined the fundamental issues related to SDN controller placement and presented a new way of addressing controller problems by first dividing the SDN network into different domains and then locating each controller in an appropriate domain. The authors proposed a greedy algorithm for linking each domain to a controller and then demonstrated the most commonly used metric for optimizing a controller in multiple domains based on a determination of the shortest path between a switch and a controller in each domain.

Hock et al.[8] introduced the resilient Pareto-based Optimal Controller placement (POCO) a framework that generates Pareto optimal CPP solutions with various trade-offs between quality (latency) and resilience.While the first version of POCO was intended for small to medium sized networks, a subsequent version [10] comprised a heuristic-based Multi-Objective Combinatorial Optimization (MOCO) approach called Pareto Simulated Annealing (PSA)for large-scale networks. When evaluating that approach on real topologies from the Internet Topology Zoo where the network size ranges from 5 to 50 nodes, the authors only stress the diameter

aspect of large-scale networks and do not assess their scalability in terms of an increased number of nodes.

In [11], Bari et al. proposed a new framework for deploying multiple controllers, which they called the Dynamic Controller Provisioning Problem (DCPP). DCPP entails dynamically adapting the number of controllers and links that are active while maintaining consideration of the state of the network. The researchers formulated DCPP as an integer linear program (ILP). Because a greedy knapsack algorithm can generate overhead for the existing configuration between switches and controllers, they defined two heuristic algorithms (DCP-GK and DCP-SA) to deal with the framework they developed in order to obtain the best possible performance and accuracy.

Similarly, Ahmadi et al.[12] formulated the CPP in large scale networks as a MOCO problem and used a multi objective heuristic called the Non-dominated Sorting Genetic Algorithm $(NSGA-II)$ to find good and diverse approximate Pareto Optimal solutions with respect to competing criteria. Their work provided an extensive analysis of the trade offs between many combinations of reliability and performance metrics.

1.7 METHODOLOGY

1.7.1 *General Approach*

In this study, one of the unsupervised learning approaches called the clustering analysis techniques are selected to reach to the desired knowledge. The clustering algorithm used is a K-medoid algorithm. This algorithm is selected based on previously made researches recommendations on facility location problem which is the parent for controller placement problem. By applying the above algorithm, a controller placement problem model that can find the required number and placement of controller for the adoption of SDN in WAN is experimented. For these study matlab based Pareto optimal controller placement tool is selected.

1.7.1.1 *Literature Review*

Literatures such as books, journals, magazines, Internet sources and others are referred regarding the concept and the researches made on software defined Networking. Also other papers that have direct or indirect relation to this work are reviewed. The research gap is identified and formulated a refined research goal. The knowledge is grasped about the study area of the topic which is going to be researched and learned from previous works done by other researchers in the area. In addition to this we gained better insight of our own research in relation to what has already been done.

1.7.1.2 *Network topology and other Data collection*

In order to get the data from ethio telecom, formal supporting letter is written from Addis Ababa Institute of Technology (AAiT) and deliver it to Chief Human Resource officer of ethio. Finally, this letter is directed to Network division and concerned departments and sections. Then ethiotelcom transport network topology data is collected. To use this data as an input for the research the topology is encoded in to a Graphical markup language (Graphml) format using XML.

1.7.1.3 *Modeling and Experimental Techniques*

For this study Matlab tool is used to optimize the network topology data and different algorithms are compared to select the best one.

1.7.1.4 *Test and Evaluation Techniques*

Test and evaluation is appropriate and vital in order to assess the output of the study. Analysis and interpretation of the results and outcomes of the project was stated.

1.8 ORGANIZATION OF THE THESIS

This thesis paper contains six chapters. Chapter one deals with introduction of the whole thesis. It clearly cites statement of the problem, objective of the study, related works, scopes and limitation of the thesis and the methods how this paper will achieve its objective.

Chapter 2 covers the general structure of traditional networks and software defined network. Chapter 3 presents SDN controller placement metrics and Wide area network elements. Fourth chapter presents parameters that are used in optimal controller placement,algorithms and optimization model in detail. Every performance indexes that are used for analysis are presented here in detail. The fifth chapter focuses on optimization of the models using Matlab based POCO tool. This chapter also presents analysis and result parts. The last chapter covered conclusions and recommendations from this study.

2

CHAPTER TWO:SOFTWARE DEFINED NETWORKING

The vast majority of multivendor legacy systems that are still operational around the world lacks fast and reliable remote identification, fault resolution, and troubleshooting capabilities. The absence of these features, which is considered a major issue facing international telecom service providers [13], makes networks difficult to manage, automate, customize, and optimize. The development of SDN in the first decade of this century by a group of researchers at Stanford University was predicated on the tackling of problems associated with legacy network designs.

Software Defined Networking represents a significant departure from traditional network architecture. In particular, SDN proposes an open architecture, in which the network control and forwarding functions are decoupled. This architecture enables automated network control and management to be programmed on standard servers through a uniform interface to physical network elements. Since network control is no longer included in each network element, SDN introduces a new component: the centralized SDN controller.

In this chapter, we review traditional network architecture, present SDN architecture, and introduce key aspects of the OpenFlow protocol, which enables controller to switch interaction through a standardized protocol.

2.1 TRADITIONAL NETWORKS

A traditional network mainly is a mix of routers and switches that facilitate the transfer of packets from one part of the network to another. The routers use routing protocols to move these packets across the network efficiently. Packets contain sets of data transferred over a physical link (the network cables). At the transmitter side, data is divided into chunks based on which data link technique has been used to transfer the packets. On the destination side, the receiver reassembles the data, and the control plane in each router and switch determines the final destination of each packet based on the routing protocols deployed on the network. As shown in Fig (2.1), The Open Systems Interconnection model (OSI model) is a standard for defining the communication functions in telecommunications or computing systems.

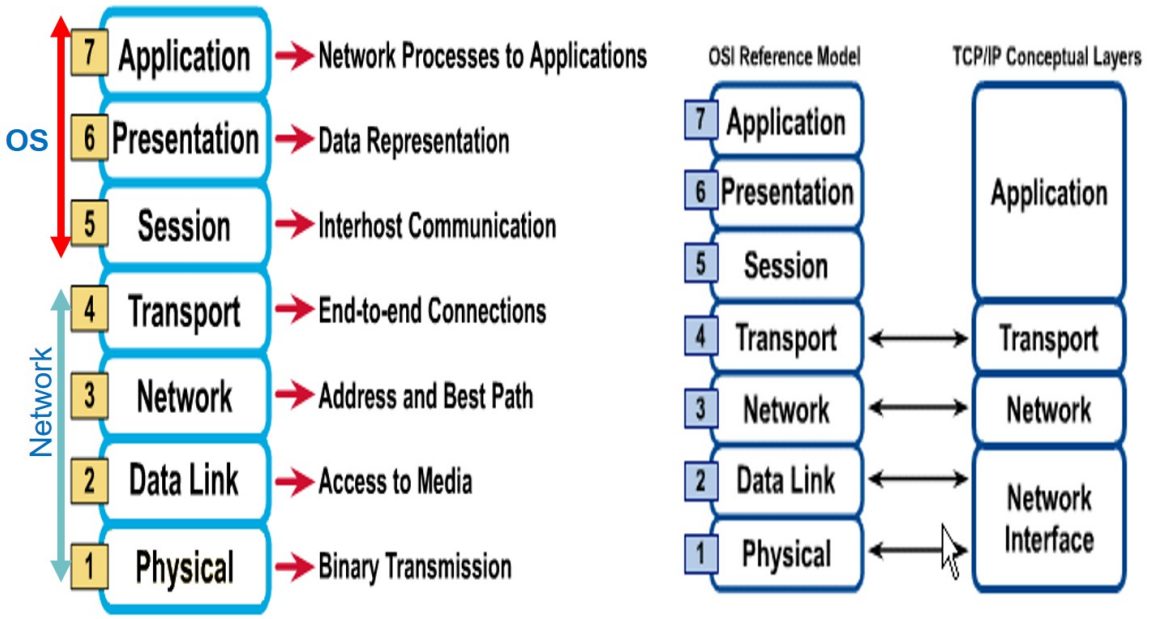


Figure 2.1: OSI and TCP/IP Models [14]

IP network devices today have to decide what to do with packets received from another device in the network whether those devices are servers, a computer, or any other type of IP device connected to the network. In the extensive IP-based networks of today, that have thousands of devices connected to them, there is a need to move from local decisions made on the device itself, which may lead to some delays, to a centralized programmable network for processing the incoming packets. The method described above is how the Internet works today where every IP device on the network is self-sufficient.

Traditional IP network devices like routers and switches have three integrated planes on each network device. As shown in figure (2.2). These are:

- **The management plane:** is the plane that stores the software services that are responsible for managing and configuring control plane functionality of the network devices in the data plane. Network managers are responsible for configuring manually the network policies in the control plane. They transform the high level-policies into low-level configuration commands, in response to a wide range of network events and application requirements.

- **The control plane:** this plane is defined by the set of protocols and algorithms such as Open Shortest Path First (OSPF) or the Border Gateway Protocol (BGP), that are used to compute and populate the routes programmed in the forwarding tables of network devices included in the data plane. These network mechanisms (protocols and algorithms) are implemented on the network devices, defining the network behavior, that is, what and how data is forwarded on the network.

- **The data plane:** also called the forwarding plane. It corresponds to the set of physical and virtual network devices in the underlying network infrastructure that may include any forwarding devices such as routers, switches, virtual switches, wireless access points, among others. These network devices are responsible for forwarding data based on the information in their forwarding tables that are programmed by distributed routing protocols, such as BGP and OSPF.
  In traditional networks, the introduction of network policies takes place in the management plane, the control plane executes these policies, and the data plane transfers data based on those policies [1].

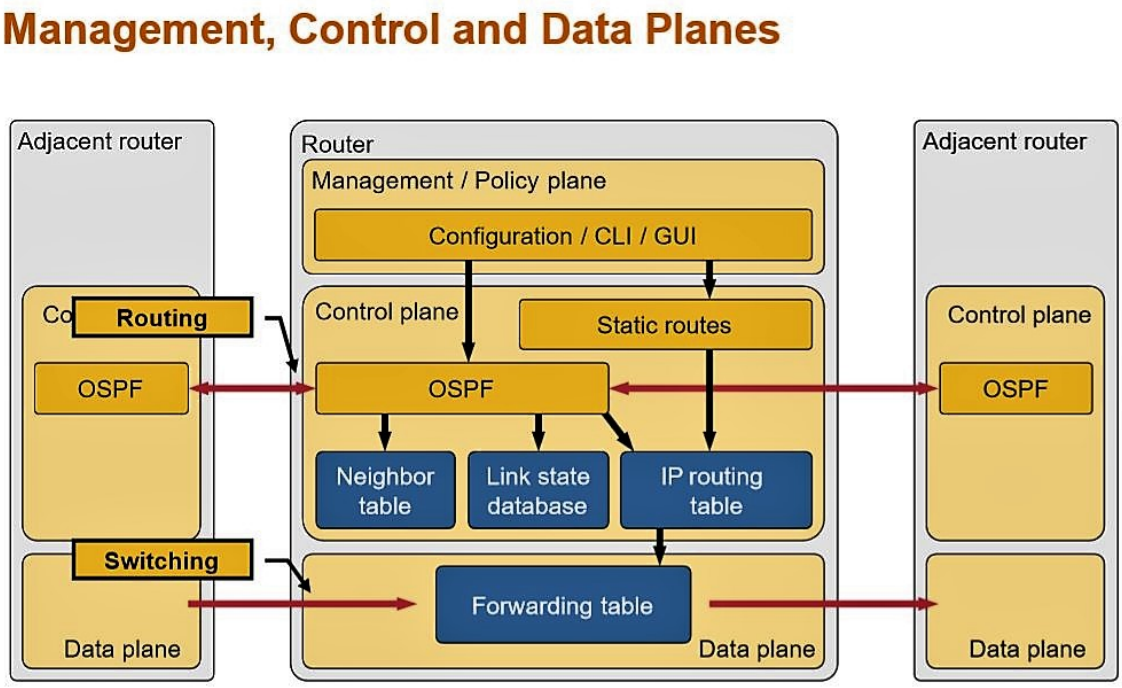


Figure 2.2: Management, Control and Data Planes in Traditional Network Devices and Systems [15]

2.2 SOFTWARE DEFINED NETWORKING (SDN)

SDN is a new way to implement an Internet network that is manageable, dynamic, and cost effective. The power of this new technology is driving big technology companies like Facebook, Microsoft, and Google to fund the Open Networking Foundation (ONF). The ONF is an organization that was established to accelerate and solve the issues that are faced by traditionally architectured IP networks. ONF has defined SDN as a networking technology which allows for a centralized and programmable control planes that permit network operation organizations to directly monitor and manage their own virtualized networks [16]. The basic premise of an SDN architecture is the separation of the two most important elements in IP network devices, the control plane and the data plane. This network architecture has the following characteristics:

- The network control and forwarding functions are decoupled,
- The network control is executed by a centralized external entity,
- The network is programmable,
- Forwarding decisions are flow-based,
- SDN has the capacity to initialize, control, change, and manage network behavior dynamically via open interfaces.
- The underlying infrastructure is abstracted by applications and network services.

2.2.1 *SDN Network elements*

Basically SDN has two main operational components: the controllers and forwarding elements.

The controllers:
Controllers in SDN are considered as the "brain" of the network, since they have a holistic view of the network status and the forwarding logic required to forward data flows properly. This is possible through an open interface to the network devices southbound from the controller. Through this interface the controller can:

(i) communicate and program the network devices,

(ii) execute basic functions, such as monitoring network devices and even gathering network statistics, among other functions.

The forwarding devices:
This refers to all the entities, physical and virtual, that receives packets on its ports and performs one or more network functions on them. In SDN, network devices are often represented as basic forwarding hardware accessible via an open interface. However, different physical network equipment with an interface to the controller can be considered as forwarding devices in SDN.

2.2.2 *SDN Architecture*

An SDN architecture consists of three different planes and a set of interfaces that allows the communication between them. These planes are: control plane, data plane and application plane, as illustrated in Fig.(2.3).

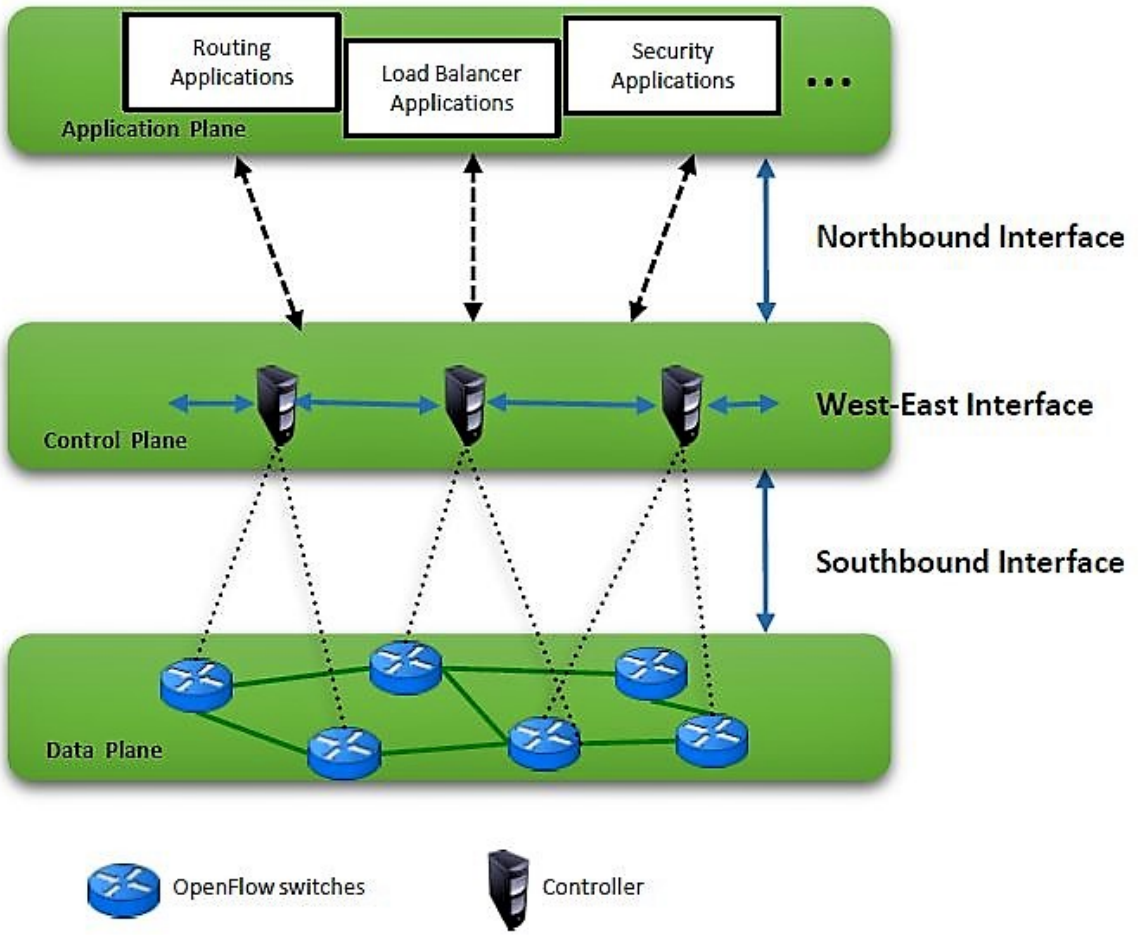


Figure 2.3: SDN network architecture[17]

Control Plane: Responsible for making decisions on how packets should be forwarded by one or more network devices and pushing such decisions down to the network devices for execution [17]. This plane consists of a centralized set of software based SDN controllers that have an abstract view of the whole network infrastructure, enabling the network manager to apply customized policies across the network devices (through the southbound interface), based on the network topology or external service requests. The SDN controller is at the heart of the architecture. It is the intelligent entity that controls resources to deliver services [18]. Control-plane functionalities usually include:

- Topology discovery and maintenance
- Packet route selection and instantiation
- Path failover mechanisms
- Install the forwarding rules on the forwarding tables based on the requested performance from the applications and the network security policy, and
- Collect status information about the forwarding devices.

Data Plane: The data plane consists of physical and virtual forwarding devices that are accessible via the southbound interface through which controller and forwarding devices can communicate with each other. Forwarding devices can support basic functions like forwarding, but also other types of functions,such as: caching, transcoding and monitoring, among others.

Application Plane: The plane where applications and services that define network behavior reside. This plane contains network applications that interact with the controller to achieve a specific network function to fulfill the network operator needs, such as: quality of service, security, virtualization and traffic engineering functions. This plane allows the behaviour of desired control requirements to be specified on the abstract network view. For this purpose,the control plane provides an abstract view from the network to the application plane,while this is shared via a general interface called Northbound. This abstract view does not contain detailed connectivity information, but enough information for the applications to request and maintain connectivity.

In SDN, three interfaces are defined:

- Southbound interface: this is the interface between the control plane and the forwarding plane, through which controller and switches can communicate with each other.
- Northbound interface: it represents the software interface between the software modules of a controller and the network applications.
- East-West interface: through this interface controllers can communicate with each other.

SDN relies on three main abstractions [19], that allow the administrators to run the network as a whole, by way of unified interface:

- Abstraction on the distributed state: network controllers get a global view of the network state, so there is no longer a needed for distributed algorithms on devices with partial network information.
- Abstraction on the forwarding model: forwarding hardware (in switches) is decoupled from the network control plane (handled by network controllers).
- Abstraction on the specification of network operation: this allows a network application to express the desirable network behavior without being responsible for implementing it.

2.3 COMMUNICATION PROTOCOLS

Different communication protocols have been defined in SDN to address the forwarding plane of the network elements in an SDN. One of the most common protocols is OpenFlow [20]. OpenFlow protocol considers the separation of the control plane from the data plane and standardizes information exchange between the control and data planes.

2.3.1 *OpenFlow Protocols*

OpenFlow is a communication protocol that enables the controller-to-switch communication in SDNs through the southbound interface [20],[21]. This protocol also defines a set of basic forwarding and management functions. The forwarding functions let programmers address the network operation by routing packets. The set of management functions can be used to control network features. For instance, switches can inform the controller when links go down or when receiving a packet for which there is no forwarding instruction.

This controller-to-switch protocol runs over either Transport Layer Security (TLS) or an unprotected TCP connection. OpenFlow provides software based access to the forwarding tables that instruct forwarding devices how packets are forwarded. OpenFlow defines the following network components for its operation: OpenFlow switches, OpenFlow controllers and control channels.

OpenFlow switch: OpenFlow switches or forwarding devices have one or more flow tables and a group table. Forwarding devices perform packet look-ups and take forwarding decisions based on flow tables and group tables. These Flow tables contain a list of flow entries, each of which determine how packets belonging to a flow will be forwarded. Flow entries consist of: a match field, a counter and actions.

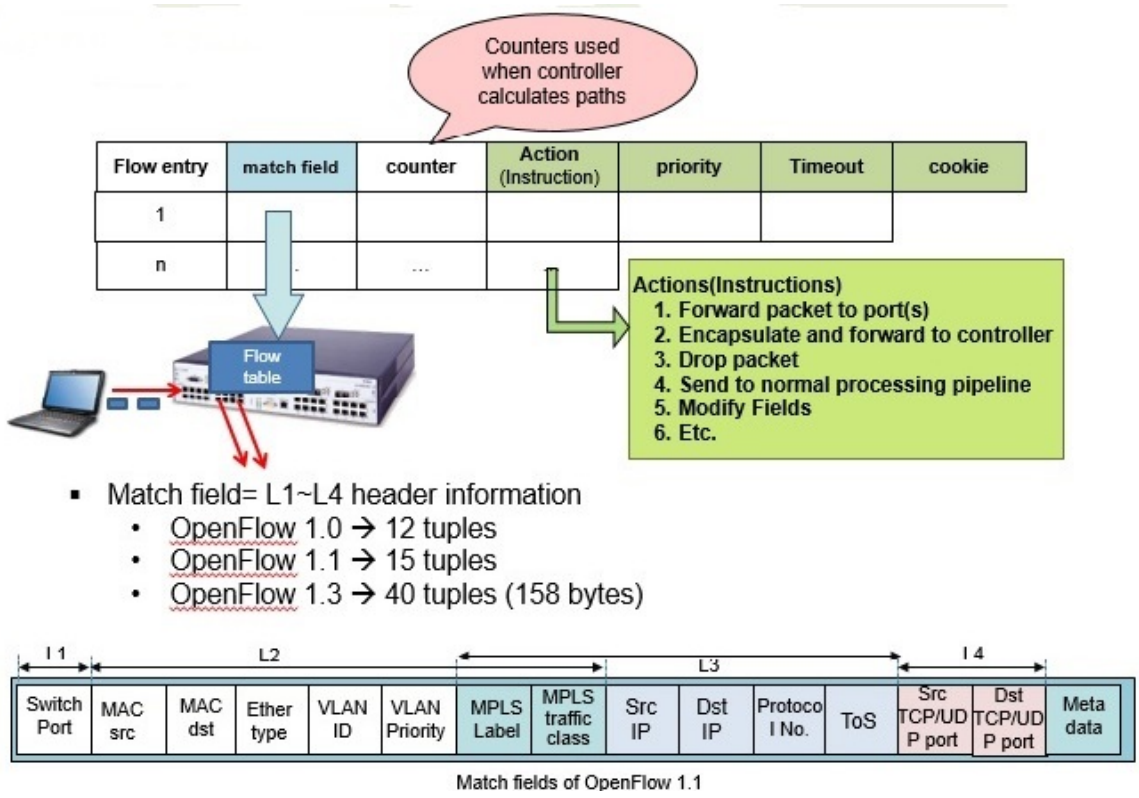


Figure 2.4: Flow Table[21]

- **Match field:** this is used to match incoming packets, entries in the match field contain either a specific value against which the corresponding parameter in the incoming packet is compared or a value indicating that the entry is not included in this flow's parameter set.
- **Counters:** these are used to collect statistics for a particular flow, such as the number of received packets and the duration of the flow.
- **Actions:** these are a set of instructions to be applied upon a match, which define how to handle matching packets. These actions describe packet forwarding, packet modification, and group table processing operations. For instance, actions can specify that the packet will be forwarded through a specified port.

Upon receiving a packet, a switch extracts the packet header field of the flow table entries. If a match is found, the switch applies the set of actions associated with the matched flow entry. If there is not a match, the action taken by the switches depends on the instructions defined in the table-miss flow entry. This table specifies a set of actions to be performed when a match is not found for a packet, such as: dropping the packet or re-forwarding the packet to the controller.

OpenFlow Controller: this is defined as an entity or server software that interacts with the OpenFlow switch using the OpenFlow protocol. This protocol connects controller software to network devices so that controllers can configure network devices and inform where to forward packets.

The OpenFlow channel: the control channel is the interface that connects each OpenFlow switch to a controller. Through this interface, the controllers configure and manage the switches, receive events from the switches, and send packets out the switches (to add, update, and delete flow entries in flow tables). The OpenFlow channel is usually instantiated as a single network connection. This may be encrypted using TLS, but may be run directly over TCP.

Controller: In OpenFlow, three controller roles are defined; equal, slave and master. The default role of a controller is equal, in this role the controllers have access to the switches and can modify their state. A controller can request a change in its role from slave to master. In a slave role, the controllers only have access to read the switches and can only process port status messages. In the master role, the controller has complete access to the switches; switches ensure that a maximum of one controller can be in the master state. Switches may be simultaneously connected to multiple controllers in equal state, multiple controllers in slave state, and at most one controller in master state, for reliability purposes. Following a master controller failure, a slave controller becomes master controller. This role change mechanism supports multiple controllers for failover, allowing the switches to be able to continue operating if their master controller fails.

Control channel: The OpenFlow channel may be composed of multiple network connections. In addition, OpenFlow specifies two simpler modes to deal with the

loss of connectivity with the controller. In fail secure mode,the switches continue operating in OpenFlow mode until it reconnects to a controller. In fail standalone mode, the switches revert to using normal processing. In addition, in OpenFlow is also considered that a switch can have multiple control paths to the same or different controllers.

2.4 NETWORK PROTOCOLS

Two protocols used in traditional networks to detect network changes and discover the network topology are briefly described below.

2.4.1 *Link Layer Discovery Protocol*

Link Layer Discovery Protocol (LLDP) was standardized by the IEEE. This protocol runs over any data link layer network that allows the network devices (switches and routers):

(i) identify capabilities and their adjacent neighbors, and

(ii) listen to their neighbor devices.

LLDP information can only be sent to and received from devices that are directly connected to each other through the same link. That is, announced information is not forwarded to other devices on the network. Network devices discover the status of their neighbors as they continually broadcast and listen to LLDP messages on each connection, discovering when a new device is added or removed. In this way, each network device can maintain its local network topology information updated.

Network devices identify neighbors according to the information (such as the MAC address) they receive in LLDP messages. This information is sent in packets called LLD Protocol Data Units (LLDPDUs). The data sent and received via LLDPDUs is useful as network devices discover:

(i) the neighbor nodes of each device and

(ii) through which ports they connect to each other.

LLDP defines the following aspects:

- A set of common advertisement messages (Type, Length, Values),
- A protocol for transmitting and receiving advertisements,
- A method for storing the information that is contained within received advertisements.

LLDP is unidirectional, operating only in an advertising mode. Therefore, LLDP does not solicit information or monitor state changes between LLDP network devices.

2.4.2 *The Bidirectional Forwarding Detection*

The Bidirectional Forwarding Detection (BFD) protocol detects failures in forwarding paths, operating on top of any data protocol (network plane, link plane, tunnels, etc.).

To detect failures BFD implements a control and echo message mechanism. This is a simple Hello protocol that transmits BFD packets periodically over a path between two pre-configured end-points to detect its status. If an end-point stops receiving BFD packets for long enough, it assumes that a network device in the monitored path has failed. Each network device transmits control messages with the current state of the monitored link or path. That is, a node receiving a control message, replies with an echo message containing its respective session status. A session is built up with a three-way handshake, after which frequent control messages confirm absence of a failure between the session end-points. Under some conditions, systems may negotiate not to send periodic BFD packets in order to reduce overhead.

2.5 TOPOLOGY DISCOVERY

Discovering the complete physical network topology is crucial in SDN networks as SDN applications require this information to make optimal routing decisions. OpenFlow messages combined with the LLDP can be used to construct the physical topology of pure SDN networks. Where they use the Dynamic Host Configuration Protocol (DHCP), Address Resolution Protocol (ARP) and LLDP protocols with OpenFlow to bootstrap a pure SDN network [22].

3

CHAPTER THREE: CONTROLLER PLACEMENT AND WIDE AREA NETWORKS

Although the decoupling of control functions provides network flexibility, it also imposes several challenges in the control plane related to the control plane scalability, resilience and data consistency. The scalability and resilience of the control plane have been tackled from the point of view of the controller placement. In terms of scalability, the controller placements in an SDN network can be selected to guarantee that the control paths as well as the controllers have enough capacity to handle the traffic coming from the switches they manage, avoiding they become a bottleneck [23].

In this chapter, the different metrics used to select the controller placements will be described. This information is used to create a classification of the controller placement approaches found in the literature and finally wide area network technologies will be discussed.

3.1 COMPUTING THE OPTIMAL CONTROLLER PLACEMENTS

On a network of many switches (nodes), finding the best location to install a single controller is not an easy task. The same question is relevant when finding the best placement locations for multiple controllers. This section presents the different objectives, strategies and the metrics that can be defined for selecting the controller placements in SDNs.

3.1.1 *Main controller placement objectives*

The controller placement problem consists of discovering both the number of controllers required and their placement in a network to satisfy a specific objective that can be formulated in terms of:

Metrics to provide QoS:

The controller placements can be selected according to a set of quality of service requirements defined by the service provider. These QoS requirements are related to the switch-to-controller delay [24],[7], inter-controller delay [15], controller utility [25], among others. The location of the controllers in the network, can be also selected based on specific QoS metrics that guarantee the flow setup time and the response time of a request [11]. For instance, critical services as video or voice over IP have to satisfy specific QoS requirements to not be seriously affected.

Economical profit of the controller placement:

From the point of view of network providers, a natural objective to select the controller placement would be to minimize the capital and operational expenditure. This objective is directly proportional to reduce the number of controllers in the network and use the controller resources in an efficient way. The number of controllers implemented in a network, directly impacts the capital and operational expenditures (CAPEX and OPEX). Therefore, it becomes necessary to find the optimal number of controllers in a network and the switch-to-controller assignment. In order to reach this goal, controller placement approaches should try to assign the maximum number of switches(nodes) to each controller while satisfying a set of network requirements [8].

For instance, in [23] the objective is to minimize the cost of assigning a switch-to-controller, and in [26]the objective function is to minimize a given cost function, that seeks to maximize the resilience.

Resilience of the control plane:

In terms of the control plane, resilience is related to find the controller placement that: i) minimize the data loss, by reducing the number of control paths affected by a network failure as in[27],[28] or ii) maximize the number of backup paths per each control path or at least, find the number of backup paths that satisfy defined resilience requirements[29],[30] and [31].

3.1.2 *Optimization strategies*

The controller placement problem is (Non-deterministic polynomial-time) NP-hard. Therefore, for large networks the time to find the optimal solution becomes very high. In this case, different strategies have been used to solve the controller placement problem in an efficient way.

Exact solutions

To find an optimal control placement solution, the problem has to be formulated by means of Integer Linear Programming (ILP). When using this strategy, network resources can be limited (e.g., controller, switches and links resources). In addition, other specific requirements such as resilience, switch-to-controller delay and inter-controller delay can be defined. However, the disadvantage of this strategy is the time it takes to find a solution, finding the controller placements in large networks can take a long time, as shown in[8] and [10].

Heuristic solutions

This strategy allows the controller placement problem to be solved in a more efficient time in comparison with ILP. Heuristic strategy does not find an optimal solution, but it finds a solution near the optimal, compromising optimality for short execution time. In general, the controller placement approaches are formulated through a heuristic algorithm.

Clustering solutions

Clustering is used in many disciplines and applications, it is an important tool that seeks to identify homogeneous groups of objects based on the values of their attributes. In the controller placement problem, clustering can be used with heuristic and exact solutions. In this case, the clustering approach is responsible for reducing the search space where a controller placement is sought, defining the set of feasible controller placements. Then, a heuristic or exact strategy can be used to select the controller placement.

3.2 CONTROLLER PLACEMENT METRICS

Different metrics have been considered to select the controller placement. These can be classified according to the main network objective to select the controller placements, as described in Section 3.1.

3.2.1 *Metric related to the network performance*

These metrics evaluate the performance of the resulting control planes, which are related to:the switch-to-controller path and the switch allocation between controllers. These metrics can also be used to compare different controller placement approaches.

Stretch

This metric compares the path length of each switch to the controller placement found (control path) with the shortest path from the switch to the controller. A high stress might result in some control paths with an additional delay because the resources of the switches/links can be shared by multiple control paths. The more control paths use the same switch or link, the higher the impact regarding possible side effects. For instance, if multiples control paths use the same switch/link, when it fails all the control paths including this switch are disconnected from the controller.

Path length

The path length metric measures the number of hops between the controller and each switch that is managed by it. The longer a control path, the more resources

are used and the node dependence increases (number of down stream nodes). This means that the path length has a direct impact on the cost of the resulting control plane. Depending on the links traffic, switches that have longer control paths to their controller can take a long communication delay, affecting the control plane performance. In long paths, when a network failure happens, all the downstream nodes to the failure node are disconnected from the controller.

Control path delay

This is the time it takes to a switch to communicate with its controller and vice versa. Therefore, this delay influences:

(i) The time switches announce events to their controller and

(ii) The time it takes to the controllers to configure their switches.

Given the importance of this metric in the network performance, most of the approaches take into account this metric in the selection of the controller placement.

3.2.2 *Metric related to control plane scalability*

One of the most important reasons to distribute the network control is based on the fact that one controller alone may not have enough capacity to manage the whole network, and therefore it could become a bottleneck in terms of processing power, memory, or input output bandwidth [32]. As explained in [33], in a centralized and reactive SDN network, scalability problems can be caused by flow initiation overhead or resiliency to failures. In large networks with a distributed control plane, these scalability problems may also arise, since controllers not only have to process requests coming from switches it is responsible for, but also requests sent from other controllers.

Therefore, in large scale WANs, the control plane topology can limit the availability, response time and convergence time on the network [25]. The reason is that, control applications require the ability to reprogram data plane state at very fine time-scales to satisfy network objectives in SDN. Therefore, selecting the controller placements to keep the flow setup time as low as possible is fundamental for an efficient network operation.

Link capacity

In SDNs, data and control paths share network resources. Therefore, the controller placements should be selected so that:

(i) All switches are managed,

(ii) The switch-to controller control paths have enough capacity and,

(iii) The controllers have enough available resources.

A congested link that is used by one or more control paths can add a switch to controller communication delay that affects the response time of a request and consequently the flow setup time [34].

Controller utilization

The controller utilization is measured by the number of flows managed by each controller. Each controller has an upper limit on the number of control messages and flows it can handle at any time. Therefore, there is a limited number of switches that can be managed by a controller. In [7], it was shown that one controller alone is capable of managing a complete network, in terms of controller capacity. However, the switch to controller delay can be very high, affecting the application performance. In large SDN networks, data center or enterprise networks that can manage a high traffic load, only one controller is not enough. The placement of controllers should try to minimize the propagation latency, while the load of each controller should not exceed its capacity.

Number of controllers

The number of controllers in a control plane not only depends on the traffic managed by the network, but also on the other requirements that must be satisfied (e.g., control path delay, inter-controller delay). It has also been shown that the number of required controllers is more dependent on the topology than on network size [35].

Increasing the number of controllers in the network can significantly improve the resilience,as shown in [7]. In [8] and [24], authors shown that there is a specific number of controllers that improve the network resilience, which depends on the network topology. It is also shown that, using more controllers than the necessary does not improve the network performance, but increases the solution cost and can make it difficult for the controller communication.

3.3 WIDE-AREA NETWORKS

Most enterprises today use Local area networks (LANs) for each of their offices or buildings. These LANs usually connect end devices like servers, computers, smart phones, tablets, printers and also the enterprise's network equipment such as firewalls, routers, switches and access points. LANs are usually limited by the office or building in which all of these devices are located. The dominant technology that is used inside the LAN nowadays is Ethernet [36]. But what happens when an enterprise grows and has multiple branch offices across the country, or even the globe?, WANs represent the technologies that are used to connect multiple branch offices and their data centers together and allow for communication between large geographical distances [37]. There can be different types of WAN services metropolitan, regional, national or international. WAN technologies use

the services of Tier 1 or Tier 2 ISPs and telephone, cable or satellite companies in order to create the communication network for the customer [36]. Also WAN technologies operate on the first three layers of the OSI model, the physical, data-link and network layers [38]. A clear view of the separation between LAN and WAN technologies can be seen in figure 3.1.

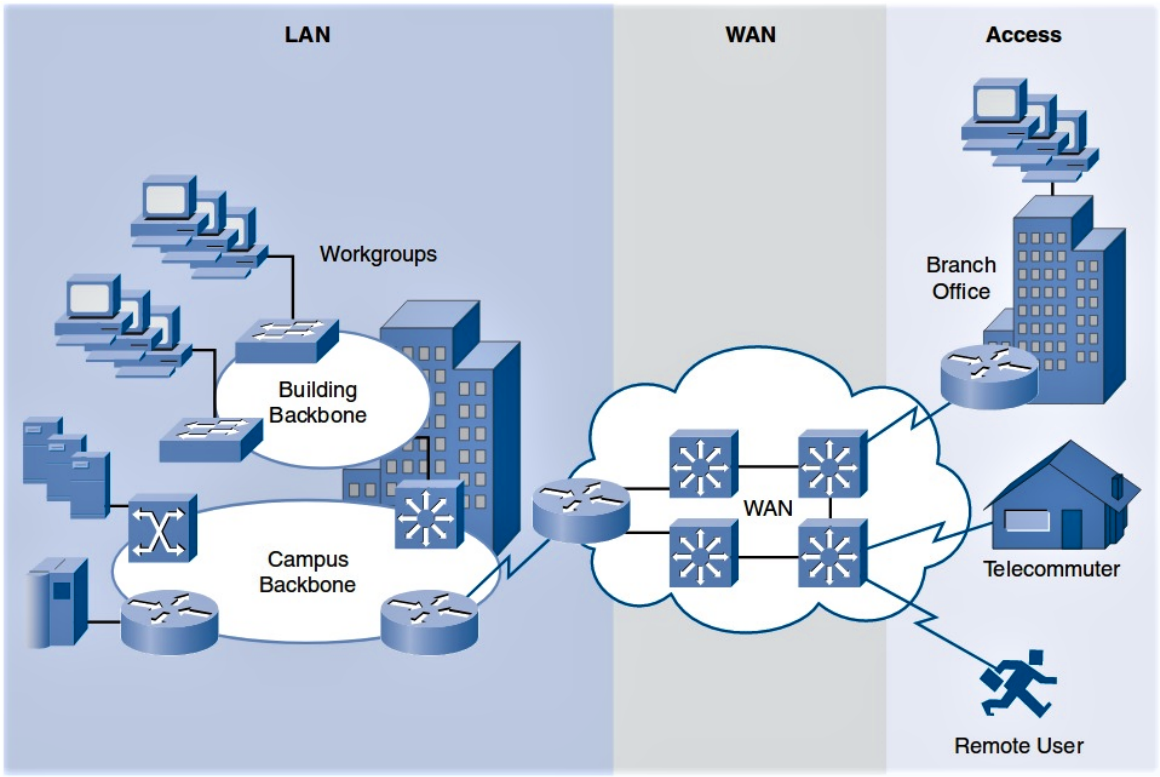


Figure 3.1: Distinction between LAN and WAN [36]

3.3.1 *Types of WAN*

There are plenty of technologies that were designed for WAN communication such as SDH, DWDM, VSAT, ISDN, Frame Relay, ATM, Metro Ethernet, MPLS, DSL. These technologies can be grouped together in two major categories, as shown in figure 3.2.

Private WAN technologies can be further divided into dedicated and switched. An example for dedicated private WAN can be SONET/SDH or DWDM transmission. As for switched private WAN, the circuit-switched technologies like PSTN and ISDN are considered obsolete nowadays and the only focus is on the packet-switched technologies. Currently MPLS is the most widely used private WAN technology in the world.

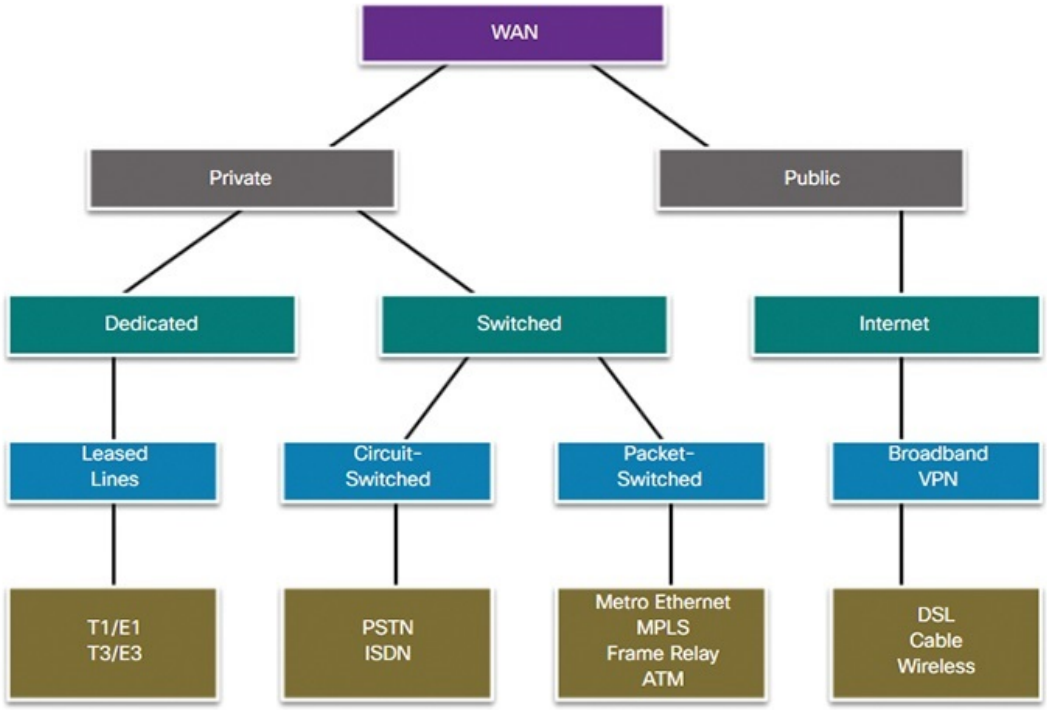


Figure 3.2: WAN technology tree [39]

Public WAN technologies, on the other hand, rely on the Internet. The Internet provides a global network for enterprises, where they can connect all of their offices together and share information among them. There is one catch, however. Since the Internet is a public network, everyone connected to it can see the information that travels through. That is certainly something that enterprises do not want.To prevent this enterprises use virtual private network (VPN) solution which is a means to carry private information across the public network.

Therefore, the choice between private and public WAN technologies can not be defined as straightforward. Where high level of security is needed, the advantage goes to the private WAN. But where low implementation costs and flexibility are more desired, public WAN should be considered. It all depends on the specific requirements of the enterprise. The same applies to the choice between dedicated or switched WAN. If there is a need for high bandwidth sensitive traffic, such as voice or video, between a bank's HQ office and its data center, the cost of leased lines can be justified. If, however, we need to connect multiple branch offices together, the more reasonable choice would be the use of packet-switched WAN [39].

3.4 SOFTWARE-DEFINED WIDE-AREA NETWORKING

Software defined wide area network (SD-WAN) is a technology that has the potential to revolutionize the networking sector in the WAN area. It facilitates a new concept for the network sector application driven networking, where the network is expected to adjust according to application needs [40]. This concept allows SD-

WAN to become a replacement of WAN optimization services, expensive MPLS VPNs and additional network automation and management costs.

3.4.1 *SDN deployment at a WAN scale*

There are significantly less SDN-based deployments at a WAN scale. Currently, only Google and Microsoft have proposed using SDN to control their WANs [41],[42]. Google has deployed SDN in one of their backbone inter-data center network. The authors describe how SDN allows them to meet requirements such as

- Massive bandwidth requirements deployed to a modest number of sites,
- Elastic traffic demand that seeks to maximize average bandwidth,and
- Full control over the edge servers and network, which enables rate limiting and demand measurement at the edge

That are very specific to Google. Most WANs are designed to run at modest utilization (mostly capped at 30-40% utilization for the busiest links), to avoid packet drops and to reserve dedicated backup capacity in the case of failure. As they stated the busiest B4 (Google SDN WAN) edges constantly run at near 100% utilization, while almost all links sustain full utilization during the course of each day and an average of 70% utilization over longtime periods, corresponding to 2-3x efficiency improvements relative to standard practice. Google claims their network tolerate high utilization by differentiating among different traffic classes. In this scenario, Google controls the WAN as well as the applications running on the end hosts. Having an end-to-end control of the network simplifies the administration of the WAN.

Microsoft has experimented deploying SWAN (SDN driven WAN) on a testing environment that simulates their WAN. They describe the limitations of MPLS-TE to maximize bandwidth utilization and they show how using SDN can highly increase the bandwidth usage. Services communicate to the SDN controller how much data they need to send and how sensitive they are to congestion. The controller computes new paths that guarantee that no congestion occurs for those services that are sensitive to it. According to the authors, simulations show that SWAN carries 60% more traffic than the current production network.

3.4.2 *ethiotelecom WAN*

Ethiotelecom transport wide area network is a DWDM based optical transport network to support IP core, backhaul and international gateway services. The network is based on Automatically Switched Optical Network (ASON) advanced system architecture with a support of 40 Gbps and 100 Gbps bandwidth capacity Optical Transport Network (OTN) structure. It have 67 switching DWDM nodes and 276 links (edges). The network is build with two vendor's equipment and the two

networks are interconnected in the data level using tributary cards. For this connection there is a redundant equipment on the border sites of the two networks. This is due to the lack of inter-operability issue between the two vendors network equipment. The network is recently built to carry the countries telecom traffic demand for the next decade. As per the assessment we have done on the last seminar most of the network equipment recently deployed supports SDN.

The network topology document is collected from ethiotelecom. The collected document have 67 switching nodes, with its site names, latitude and longitude location coordinates and 276 node to node links data. The details of the topology is not included due to the confidentiality of the document.

4

CHAPTER FOUR: OPTIMAL PLACEMENT OF CONTROLLERS IN SDN

SDN advocates a centralized network control with a global view of the network topology. Large scale SDN networks may consist of multiple controllers in one or different controller domains by distributing the network management between them. Each controller has a logically centralized but physically distributed vision of the network. In this context, a key challenge faced by providers is to define a control plane that meet the requirements such as load distribution between controllers, propagation delays among controllers and delay between controller-to-switches that leads to a longer response time of the controllers, affecting their ability to respond rapidly to network events and reducing the reliability of communication.

This chapter presents the methodologies that were used for finding the optimal placement of an SDN controller and for ascertaining the number of controllers needed in ethiotelecom WAN in order to achieve the best quality of service (QoS). The determination of the optimum controller locations for a large-scale SDN is still a significant target in SDN research. Our proposed solution for minimizing global latency and ensuring the best QoS is to use a partitioning clustering approach based on a k-medoid algorithm, a method used for addressing a facility location problem through consideration of the minimum distances between a variety of network locations [43].

4.1 CLUSTERING METHODS

Many clustering algorithms exist in literatures. In general, the major clustering methods can be classified into the following categories depending on their techniques [44].

- Partitioning Method
- Hierarchical Method
- Grid-based Method
- Density-based Method
- Model-based methods

Partitioning methods: Given a database of n objects or data tuples, a partitioning method constructs K partitions of the data, where each partition represents a cluster and $k \leqslant n$. That is, it classifies the data into K groups, which together satisfy the following requirements: (1) each group must contain at least one object, and

(2) each object must belong to exactly one group.

Hierarchical methods: A hierarchical method creates a hierarchical decomposition of the given set of data objects. A hierarchical method can be classified as being either agglomerative or divisive, based on how the hierarchical decomposition is formed. The agglomerative approach, also called the bottom-up approach, starts with each object forming a separate group. It successively merges the objects or groups that are close to one another, until all of the groups are merged into one (the topmost level of the hierarchy), or until a termination condition holds. The divisive approach, also called the top-down approach, starts with all of the objects in the same cluster. In each successive iteration, a cluster is split up into smaller clusters, until eventually each object is in one cluster, or until a termination condition holds.

Density-based methods: Most partitioning methods cluster objects based on the distance between objects. Such methods can find only spherical-shaped clusters and encounter difficulty at discovering clusters of arbitrary shapes.Other clustering methods have been developed based on the notion of density. Their general idea is to continue growing the given cluster as long as the density (number of objects or data points) in the "neighborhood" exceeds some threshold; that is, for each data point within a given cluster, the neighborhood of a given radius has to contain at least a minimum number of points. Such a method can be used to filter out noise (outliers) and discover clusters of arbitrary shape.

Grid-based methods: Grid-based methods quantize the object space into a finite number of cells that form a grid structure. All of the clustering operations are performed on the grid structure (i.e., on the quantized space). The main advantage of this approach is its fast processing time, which is typically independent of the number of data objects and dependent only on the number of cells in each dimension in the quantized space.

Model-based methods: Model-based methods hypothesize a model for each of the clusters and find the best fit of the data to the given model. A model-based algorithm may locate clusters by constructing a density function that reflects the spatial distribution of the data points. It also leads to a way of automatically determining the number of clusters based on standard statistics, taking noise or outliers into account and thus yielding robust clustering methods.

Among all these methods, our main focus is on partitioning based clustering methods which a common practice in clustering network topologies in facility location problems. The two commonly used partitioning methods are K-Means and K-Medoids.

4.2 PARTITIONING BASED CLUSTERING METHODS

Given K, the number of partitions to construct, a partitioning method creates an initial partitioning. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another. The general criterion of a good partitioning is that objects in the same cluster are close or related to each other, whereas objects of different clusters are far apart or very different. There are various kinds of other criteria for judging the quality of partitions.

To achieve global optimality in partitioning-based clustering, it would require the exhaustive enumeration of all of the possible partitions. Instead, most applications adopt one of the most well-known and commonly used partitioning methods a k-means, k-medoids heuristic methods [45]. (1) the k-means algorithm, where each cluster is represented by the mean value of the objects in the cluster, and (2) the k-medoids algorithm, where each cluster is represented by one of the objects located near the center of the cluster.

4.2.1 *K-Means*

The K-Means algorithm is a well-known partitioning method for clustering. K-Means clustering method, groups the data based on their closeness to each other according to the Euclidean distance(the sum of squared Euclidean distances from the center). It takes K as input parameter and partition a set of n object into K clusters. The mean value of the object is taken as similarity parameter to form clusters. The cluster mean or center is formed by the random selection of K object. By comparing most similarity other objects are assigning to the cluster. For each data vector this algorithm calculates the distance between data vector and each cluster centroid. The centroid is recalculated each time respectively after addition of data point in cluster.

Input: K: the number of clusters, **D**: a data set containing n object.
Output: A set of **K** clusters.

algorithm:
1. Input the data set and value of K
2. If f K == 1 then Exit.
3. Else
4. Choose k objects from D randomly as the initial cluster centers.
5. For every data point in the cluster reissue and define every object into the cluster where the object matches, based on the object's mean value in the cluster.
6. Update cluster means; after that for each cluster calculate the object's mean value.
7. Repeat from step 4 until no data point was assigned otherwise stop. The satisfying criteria can be either number of iteration or change of position of centroid in consecutive iterations.

Table 4.1: K-Mean Algorithm. Arora and Varshney

4.2.2 *K-Medoids*

The K-Medoids algorithm is used to find Medoids in a cluster which is centre located point of a cluster. K-Medoids is more robust as compared to K-Means as in K-Medoids we find K as representative object to minimize the sum of dissimilarities (distance) of data objects whereas, K-Means used sum of squared Euclidean distances for data objects. And this distance metric reduces noise and outliers.

Medoids is the data object of cluster which is most centrally located. Medoids are selected randomly from the K data objects to form K cluster and other remaining data objects are placed near to Medoids in a cluster. Then process all data objects of cluster to find new Medoids in repeated fashion to represent new cluster in a better way. After finding the new Medoids bind all the data objects to the cluster. Location of Medoids change accordingly with each iteration. So K clusters are formed representing n data objects.

Input: K: the number of clusters, **D:** a data set containing n objects.
Output: A set of k clusters.

algorithm:
1. Randomly select K as the Medoids for n data points.
2. Find the closest Medoids by calculating the distance between data points n and Medoids K and map data objects to that.
3. For each Medoids m and each data point o associated to m do the following: Swap m and o to compute the total cost of the configuration then Select the Medoids o with the lowest cost of the configuration.
4. If there is no change in the assignments repeat steps 2 and 3 alternatively.

Table 4.2: K-Medoid Algorithm. Arora and Varshney

Drawbacks of K-Means algorithm:

1. It is not effective when used with global cluster.
2. If different initial partitions has been selected then it may vary the result for clusters.
3. Different size and different density cluster is not handled by the algorithm.

Due to the above reasons it is preferable to use K-Medoids algorithm that is based on object representative techniques to reduce the drawbacks of K-Means algorithm.

4.3 NETWORK TOPOLOGY BASED K-MEDOID CLUSTERING

The k-medoid algorithm is a typical clustering algorithm, and like the K-means algorithm, it has been used in the data mining area. Compared to the K-means algorithm,this procedure provides better optimized results regardless of the calculation procedure and initial conditions. The proposed approach evaluates the distance among nodes from the network topology $G(V, E)$, and defines it as a dissimilarity index. Node group clusters are determined to minimize the sum of this index in each group, based on the k- medoid algorithm.

4.3.0.1 *The k-Medoid Mathematical Model*

In optimal plant location problem using k-medoid clustering to construct partitions with a fixed number k of clusters, it is often assumed that there exists a function which measures the quality of different clusterings of the same data set. This method is based on a location model with the following general formulation: Given a finite number of users, whose demands for a given service are known and must be satisfied, and given a finite set of possible locations among which k must be chosen for the location of service centers, select the locations in such a way as to minimize the total distance (or equivalently the average distance) travelled by the users.

In the formulation used, the sets of users and of possible locations coincide and both correspond to the set of objects to be clustered. The location of a center is interpreted as the selection of an object as a representative object (or centro-type or medoid) of a cluster. The distance travelled by a user corresponds to the dissimilarity between an object and the representative object of the cluster to which it belongs.

Problem formulation

In physical SDN network topologies,k-medoid provides a first controller placement approximation to guarantee:

- All nodes in the network topology G are managed by a controller and,
- The controller placements guarantee a minimum delay communication to each node, while reducing, as much as possible, the flow communication time between any pair of end nodes in the network.

In the mathematical formulation of the k-medoid model the following notations are used: Given an undirected network topology denoted by the tuple

$$G = (V;E),$$

where $V = (1, \ldots, n)$ is the set of vertices(nodes) and E is the set of edges,

The distance between nodes V_i and V_j (also called nodes i and j) is denoted by

$$d\ (\ i,j\ ).$$

A solution of the model is determined by two types of decisions:

- The selection of nodes as representative nodes in clusters: Y_i is defined as a binary variable, equal to 1 if and only if node i $[i = 1, 2, \ldots, n]$ is selected as a representative node.
- The assignment of each node j to one of the selected representative nodes: Z_{ij} is a 0-1 variable, equal to 1 if and only if node j is assigned to the cluster of which i is the representative node (and also the medoid).

The corresponding optimization model becomes:

$$\text{Minimize} \quad \sum_{i=1}^{n} \sum_{j=1}^{n} d(i,j) Z_{i;j} \tag{4.1}$$

Subject to:

$$\sum_{i=1}^{n} Z_{i;j} = 1, \quad j= 1,2,\ldots,n. \tag{4.2}$$

$$Z_{i;j} \leqslant Y_i, \qquad i,j= 1,2,\ldots,n. \tag{4.3}$$

$$\sum_{i=1}^{n} Y_i = K, \quad K = \text{number of clusters} \tag{4.4}$$

$$Y_i Z_{i;j} \in 0,1, \quad i,j= 1,2,\ldots,n. \tag{4.5}$$

Constraints (4.2) express that each object j must be assigned to a single representative node. They imply [together with constraints (4.5)] that for a given j, one of the $Z_{i,j}$ is equal to 1 and all others are 0. Constraints (4.3) ensure that a node j can only be assigned to a node i if this last node has been selected as a representative node. Indeed, if this is not the case, then Y_i is 0 and the constraint [together with constraints (4.5)] implies that all $Z_{i,j}$ are 0. If i is a representative object, then all the $Z_{i,j}$ (for this i) can be either 0 or 1. Equation (4.4) expresses that exactly k objects are to be chosen as representative objects. As the clusters are formed by assigning each nodes to the most similar representative node, there will be exactly k nonempty clusters.

Equation (4.2) implies that the dissimilarity (distance) between node j and its representative node is given by:

$$\sum_{j=1}^{n} d(i,j) Z_{i;j}. \tag{4.6}$$

As all nodes must be assigned, the total dissimilarity (distance) is given by

$$\sum_{i=1}^{n} \sum_{j=1}^{n} d(i,j) Z_{i;j} \tag{4.7}$$

which is the function to be minimized in the model.

4.3.1 *K-medoid algorithm*

The initial representative nodes are chosen arbitrarily. The iterative process of replacing representative nodes by non-representative nodes continues as long as the quality of the resulting clustering is improved. This quality is estimated using a cost function that measures the average dissimilarity (distance)between a node and the representative node of its cluster. The most common realization of k-medoids clustering is the Partitioning Around Medoids (PAM) algorithm and is as follows: [44]

Algorithm: A k-medoids algorithm for partitioning based on medoid or central nodes.
Input: k: the number of clusters; A network topology G(V,E) containing n nodes.
Output: A set of k clusters.
Method:

1. Arbitrarily choose k nodes in network topology as the initial representative nodes or seeds;
2. Repeat
3. Assign each remaining node to the cluster with the nearest representative node;
4. Randomly select a non-representative node, V_{random};
5. Compute the total cost,S, of swapping representative nodes,V_j, with V_{random};
6. If S<o then swap V_j, with V_{random} to form the new set of k representative nodes;
7. Until no change;

The algorithm attempts to determine k partition for n nodes. After an initial random selection of k representative nodes, the algorithm repeatedly tries to make a better choice of cluster representatives. All of the possible pairs of nodes are analyzed, where one node in each pair is considered a representative node and the other is not. The quality of the resulting clustering is calculated for each such combination. The set of best nodes for each cluster in one iteration forms the representative nodes for the next iteration. The final set of representative nodes are the respective medoids of the clusters.

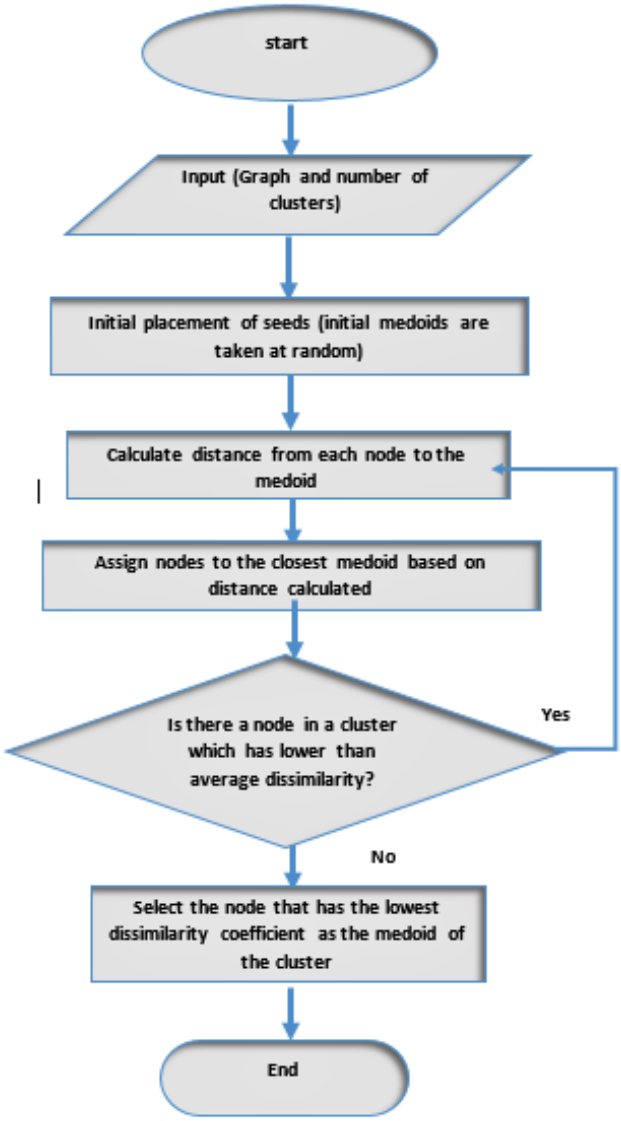


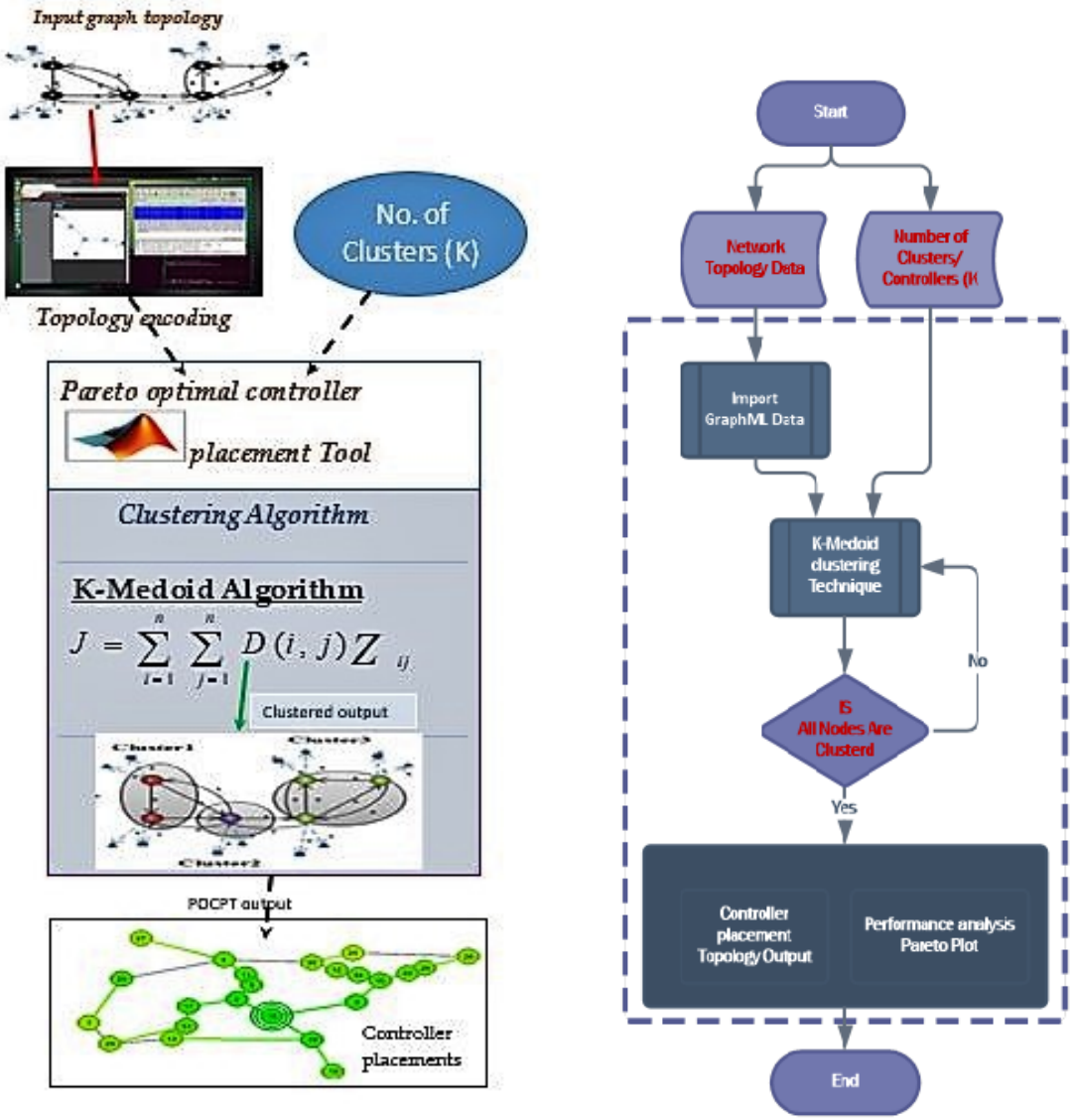
Figure 4.1: K medoid clustering algorithm flowchart.

In the next section, we describe how we propose to address the controller placement problem in WAN using k-medoid algorithm and Pareto optimal controller placement (POCOP) tool.

4.4 OPTIMIZATION SYSTEM MODEL

The research presented in this thesis involved the optimization of wide area transport network to adopt SDN by determining the number and placements of controllers. The goal of POCO is to provide application driven network optimization and finding the number and placement of controllers. To do so, it first relies on import graph algorithm to import the network topology of the WAN and collect attributes of the network. Second, POCO uses the K-medoid clustering and Pareto simulated annealing (PSA) [46] [10], a metaheuristic technique for multiple-objective combinatorial optimization, algorithms to cluster and optimize the net-

work according the given number of controllers (K). Third, POCO uses different distance and delay matrices to optimize and find the optimal placements of controllers in the network topology.



(a) optimization model Architecture. (b) Flow Chart of the optimization model.

Figure 4.2: SDN controller placement optimization model Architecture And Flow Chart.

The POCO framework is a MATLAB based tool which is designed to evaluate the optimum placement of controllers in a variety of network scenarios.The framework includes a scenario which can determine the number of controllers (k) necessary in a given network to guarantee that all nodes are able to communicate with a controller given any two node failures.

In this optimization framework it is used multiple metrics for measuring network performance and all metrics are minimization objective functions:

- Maximum node-to-controller latency:

$$\pi^{maxlatency}(P) = max_{(v \epsilon V)} min_{(p \epsilon P)} d_{v;p} \tag{4.8}$$

- Average node-to-controller latency:

$$\pi^{Avglatency}(P) = \frac{1}{|V|} \sum_{(v \epsilon V)} min_{(p \epsilon P)} d_{v;p} \tag{4.9}$$

- Maximum controller-to-controller latency:

$$\pi^{maxcontroller-latency}(P) = \max_{(p_1,p_2 \epsilon P)} d_{p_1;p_2} \tag{4.10}$$

- Average controller-to-controller latency:

$$\pi^{Avgcontroller-latency}(P) = \frac{1}{|P|} \sum_{(p_1 p_2 \epsilon P)} d_{p_1;p_2} \tag{4.11}$$

- Controller load imbalance (the difference between the controller with the most assigned nodes and that with the least):

$$\pi^{imbalance}(P) = max_{(p \epsilon P)} n_p - min_{(p \epsilon P)} n_p \tag{4.12}$$

where

$G = (V; E)$	Graph with V nodes and E edges
$d_{v,w}$	Distance matrix between all nodes $(v; w) \in V$
$d_{v,p}$	Distance matrix between node and controller.
P	Set of placement of controllers in a network
n_p	the total number of nodes that are assigned to p

In the next section, we describe the optimization setup used to evaluate the proposed framework.

5

CHAPTER FIVE :OPTIMIZATION RESULT AND ANALYSIS

This section explains our approach of finding an optimization solution for an SDN controller placement problem. Our solution is based on Pareto optimal controller placement tool (which is part of a MATLAB Facility location problem tools), a k mediod clustering algorithm and Pareto simulated annealing algorithm in order to verify the optimal SDN controller placement. The goal has been finding how many controllers were needed for the achievement of the minimum delay and to determine the optimal locations for these controllers in ethiotelecom WAN network. POCO tool is used for the minimization of the global latencies and nodes to controller assignment imbalances in ethiotelecom WAN.

5.1 RESULTS OF THE OPTIMIZATION MODELS

Here presented results of the performance evaluation setup outlined above. First, the controllers placement of both algorithms (K-Medoid and Pareto simulated annealing algorithm) for different numbers of controllers K is found and investigated. Followed by, a comparison of the results found using k-Medoids clustering model with the Pareto simulated annealing heuristic model is discussed. The optimization is done in two scenarios, scenario I controller placement using K Medoid method and scenario II controller placement using PSA method. The optimization platform and input parameters are presented.
Optimization plateform:

- Laptop Core i7 3.54G CPU @ 3GHz 3GHz, RAM 8GB, windows 10, 64bit operating system.
- MATLAB R2018a.

Input parameters:

- Number of controllers K = 2....10
- ethiotelecom network Topology.graphML
- number of iteration= 10

5.1.1 *Topology encoding*

ethiotelecom transport WAN has 67 switching nodes and 276 links (edges). For this research work to be able to upload the topology first we encode the network topology in to Grahical markup language (Graphml) format using the extensible markup language (XML). The topology is imported and plotted by the POCO tool as in figure(5.1).

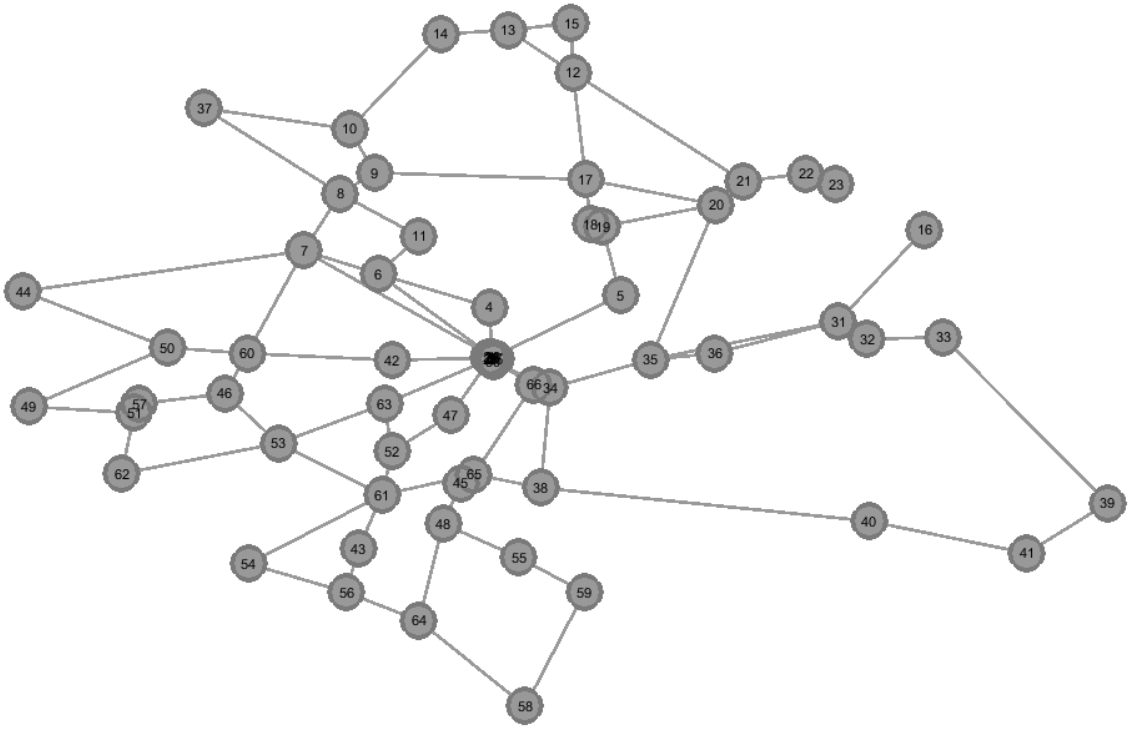
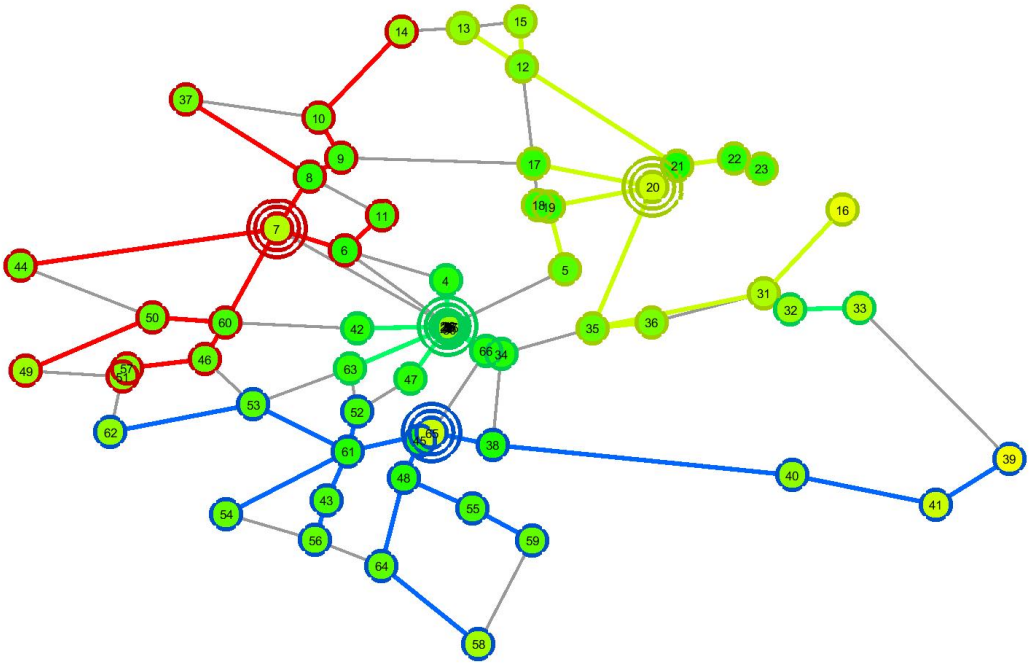


Figure 5.1: ethiotelecom WAN Topology.

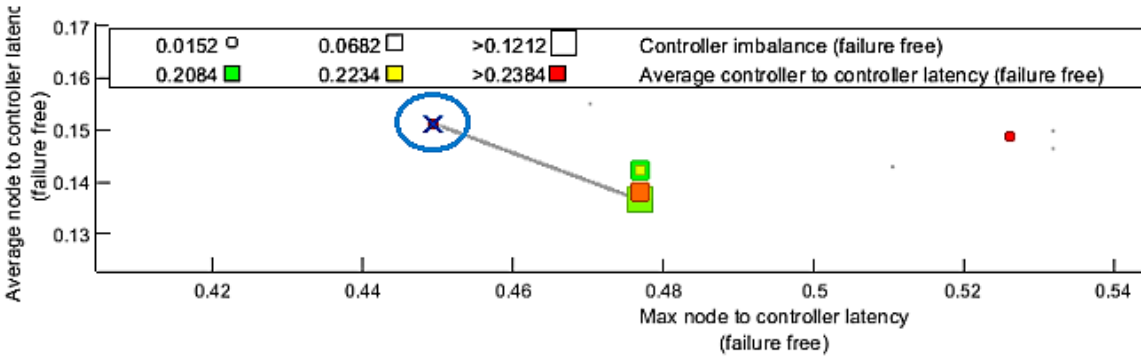
5.1.2 *Scenario I controller placement using K Medoid method*

The POCO based system imports the network topology data and plots the topology in its graphical user interface (GUI). The k Medoid algorithm will start and the number of clusters (controllers) K is inserted with the number of iteration required. The number of iteration for k Medoid graph clustering is adjusted to meet the required clustering quality, for this research ten iteration is taken.

The experiment starts with $K = 2$ and up to $K = 10$, for each number of K the controller placement, node to controller and controller to controller delay and node to controller assignment imbalance have been outputted. For discussion the results of four and eight controllers with their placement and measured Pareto plot are described below.



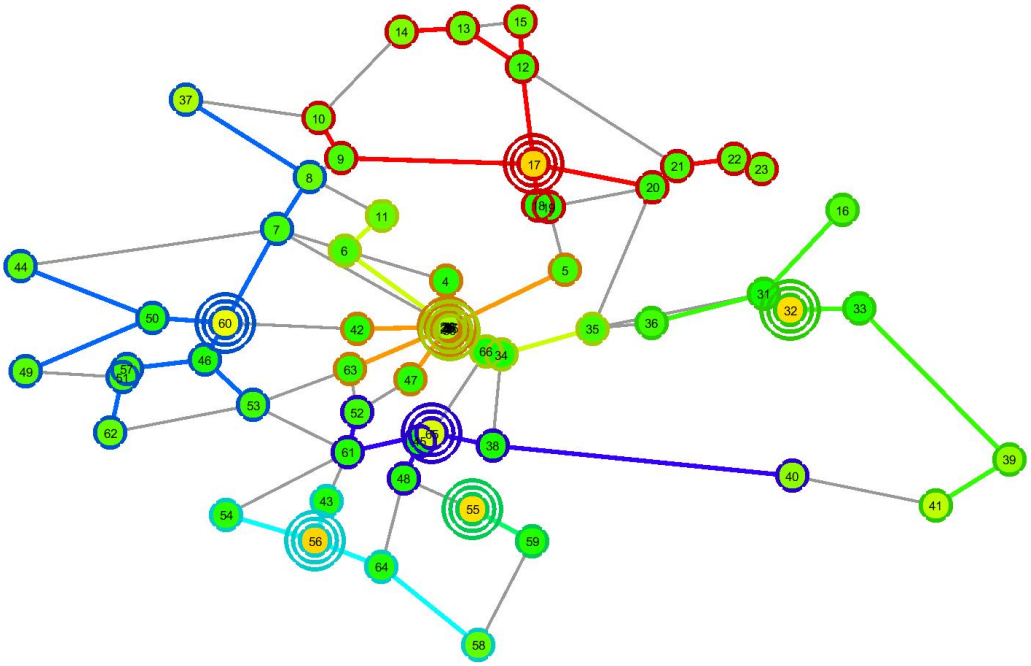
(a) placement for K=4.



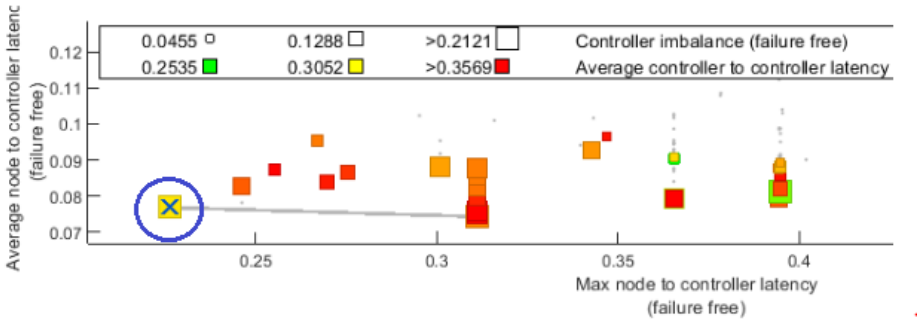
(b) Pareto plot for K=4.

Figure 5.2: Controller placement and pareto plot using K Mediod model for K=4

In the above figure, Figure 5.2a is the placement of controllers and Figure 5.2b is the Pareto plot for this specific placement. As can be seen from the figure (a), the controllers are placed (with triple rings) in Bure, Mile, Addis ababa Microwave and Yabelo sites. The horizontal and vertical axis in figure (b) are the maximum and average node to controller delays respectively and the size and color of the boxes shows the node to controller assignment imbalance and the average latency between controllers respectively. As depicted in the Pareto plot the average node to controller latency is with in the range of the recommended round trip delay (RTT) which is 250ms for WAN with shared meshed network [7] [47] [48] , the node to controller assignment imbalance, which is the difference between the maximum number of nodes assigned controller and the minimum number of nodes assigned controller, is around 20% which is with in the range of acceptable load imbalance and average controller to controller delay is 290ms but the maximum node to controller latency is above the required delay so that it shows there is a need for additional controllers in the network.



(a) placement for K=8.



(b) Pareto plot for K=8.

Figure 5.3: Controller placement and pareto plot using K mediod model for K=8

Figure 5.3 shows that the controller placement of eight controllers in ethiotelecom WAN topology. From the pareto plot we have found that both the Maximum and average node to controller latencies are below 250ms which is the allowable threshold round trip latency for WAN. the node to controller loading imbalance is around 20% and the average controller to controller delay is 320ms. Even though the average controller to controller delay is above the range since it is the measured sum average of the neighboring and faraway controllers it doesn't imply the delay of the nearby protection handover controllers due to this it has no direct impact on the performance of the network.

Figure 5.4: Controller placement using K medoid model for eight controllers.

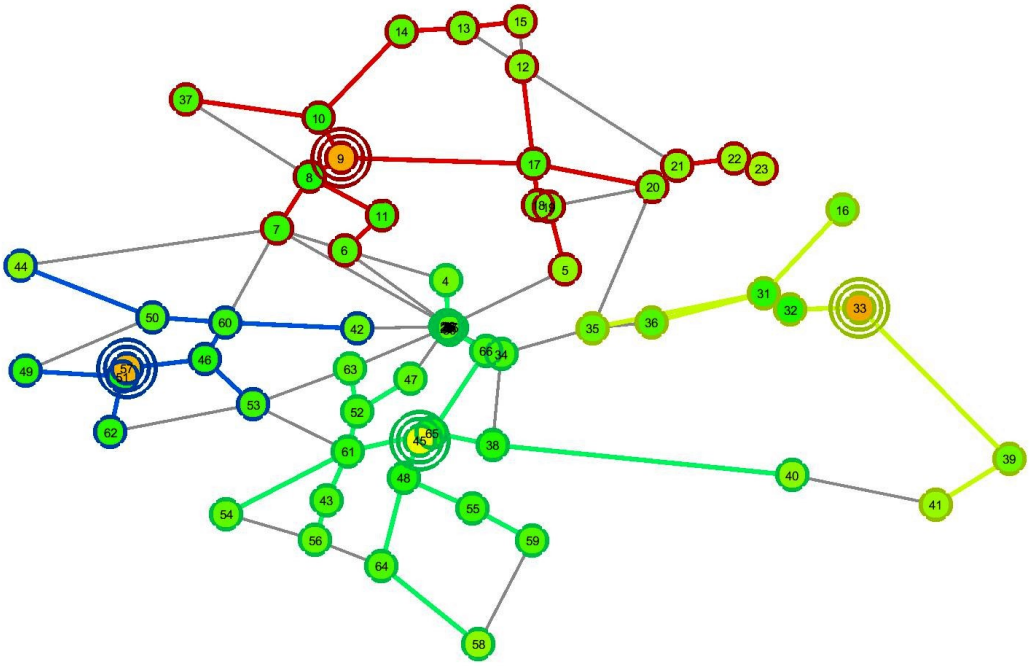
Figure 5.4 shows the placement of eight controllers using K medoid method in Ethiopia map.

5.1.3 *Scenario II controller placement using PSA method*

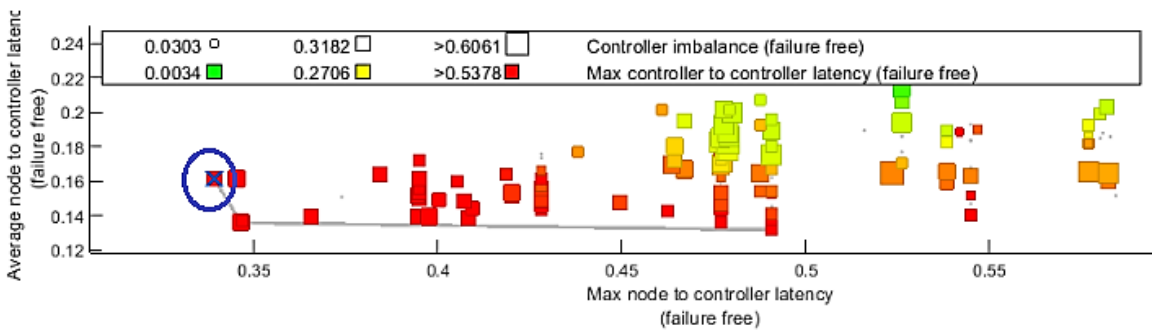
The PSA algorithm begins by first generating an initial, feasible, generally random placement of k controllers. Next, a set of random weight values is assigned to each solution. Then, consider a selection of 'neighboring' solutions; a neighborhood is a selection of possible alternative solutions for controller placements which is some subset of the solution space [10]. On figure 5.5 and 5.6 below the optimal controller

placement and its Pareto plot for four and eight controllers are shown.

From Figure 5.5, we have observed that the controllers are placed (with triple rings) in Konso, Woreta, Ambo and Jigjiga sites. As depicted in the pareto plot the optimal placement for the given number of controllers is ploted taking the maximum and average node to controller latencies and considering the average controller to controller latencies and node to controller assignment load imbalance as a measurement meterics. Even though the average node to controller latency is 160ms,which is with in the range of the recommended round trip latency, but the maximum node to controller latency is 340ms, it is above the recommended latency. It shows that there is a need for additional controllers in the network.



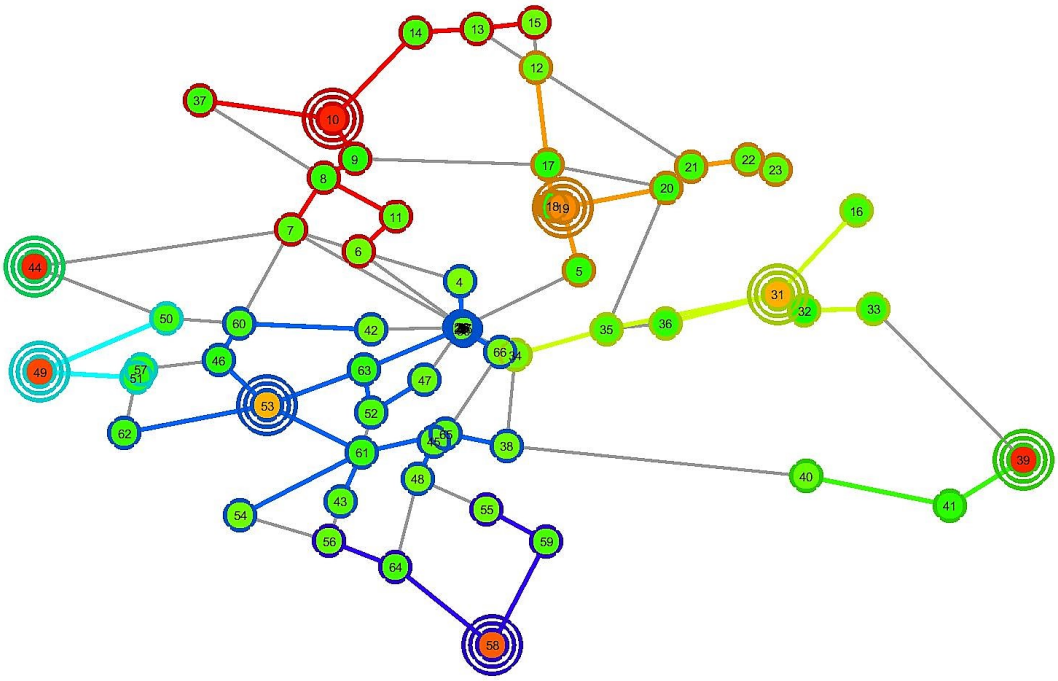
(a) placement for K=4.



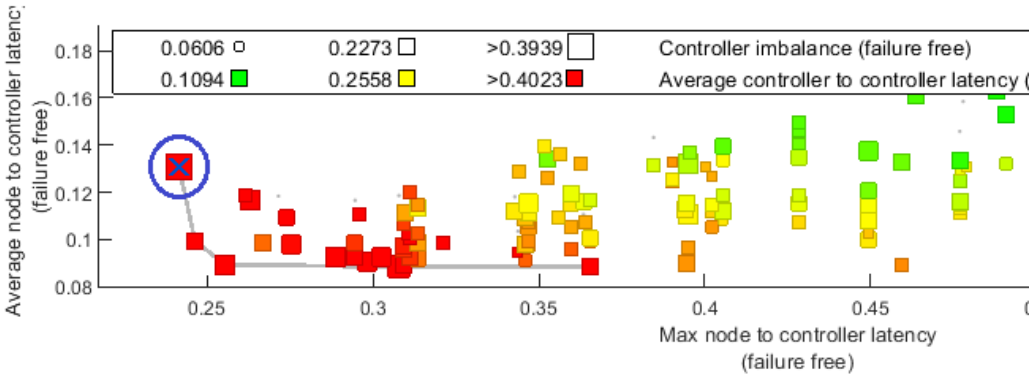
(b) Pareto plot for K=4.

Figure 5.5: Controller placement and pareto plot using PSA model for K=4

Figure 5.6 shows that the controller placement of eight controllers using Pareto simulate annealing model in ethiotelecom WAN topology. From the pareto plot we have found that both the Maximum and average node to controller latencies are 240ms and 130ms respectively,which is with in the allowable threshold round trip latency for WAN. The node to controller loading imbalance is around 39% and the average controller to controller delay is 480 ms.



(a) placement for K=8.



(b) Pareto plot for K=8.

Figure 5.6: Controller placement and pareto plot using PSA model for K=8

Figure 5.7 shows the placement of eight controllers using PSA method in Ethiopia map.

Figure 5.7: Controller placement using PSA model for eight controllers.

5.2 RESULT ANALYSIS AND REFLECTIONS

In this section, we will discuss the findings,outcomes,and reflections of the study.

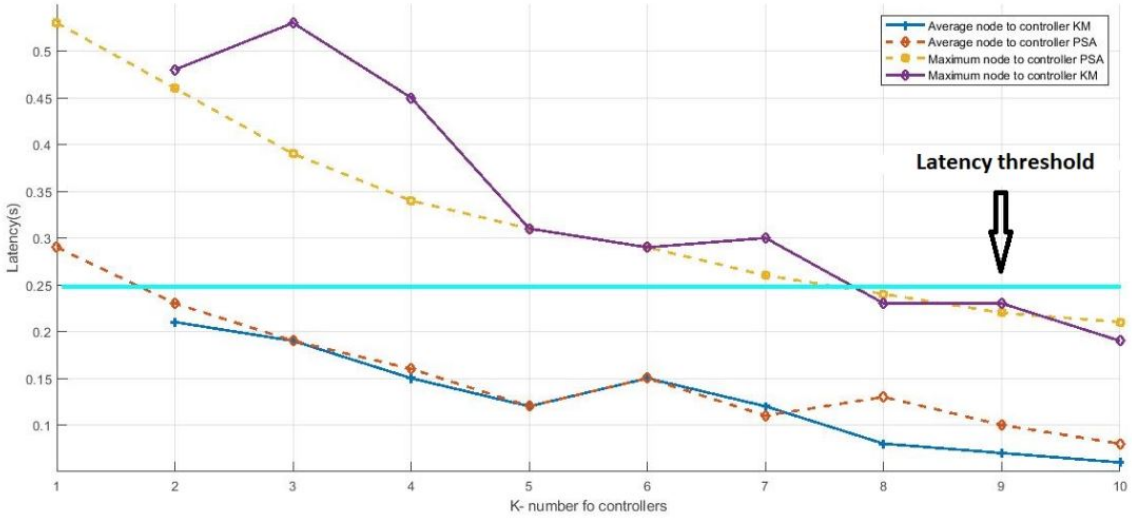


Figure 5.8: The maximum and average latency for SDN network with number of controllers (k).

The horizontal and vertical axis in the above chart are number of controllers K and node to controller latency respectively. It shows the average and maximum controller to node latencies for each number of controller placements using both the K medoid and Pareto simulated annealing models. The broken line represents the PSA where as the solid lines represents the K Medoid average and maximum latencies.

From the average latency perspective for smaller number of controllers even-though the placement selected by the two models are in different sites, the measured latency of both models are nearly equivalent. When the number of controllers increase and reaches above seven the clustering based k Medoid model have a better average latency than the Pareto simulated annealing model. The maximum (worst case) latency between the controller and nodes is above the recommended latency value for both models until the number of controllers reaches eight. Both models meet the allowable latency requirement for WAN when the number of controllers are eight and using above this number of controllers will decrease the latency with in the recommended working zone. For the number of controllers K=5 and 6 the two models placed the controllers in the same sites due to this their average and maximum latencies are equal. Even thought increasing the number of controllers beyond this value will further decrease the latency but it will also increase the deployment cost.

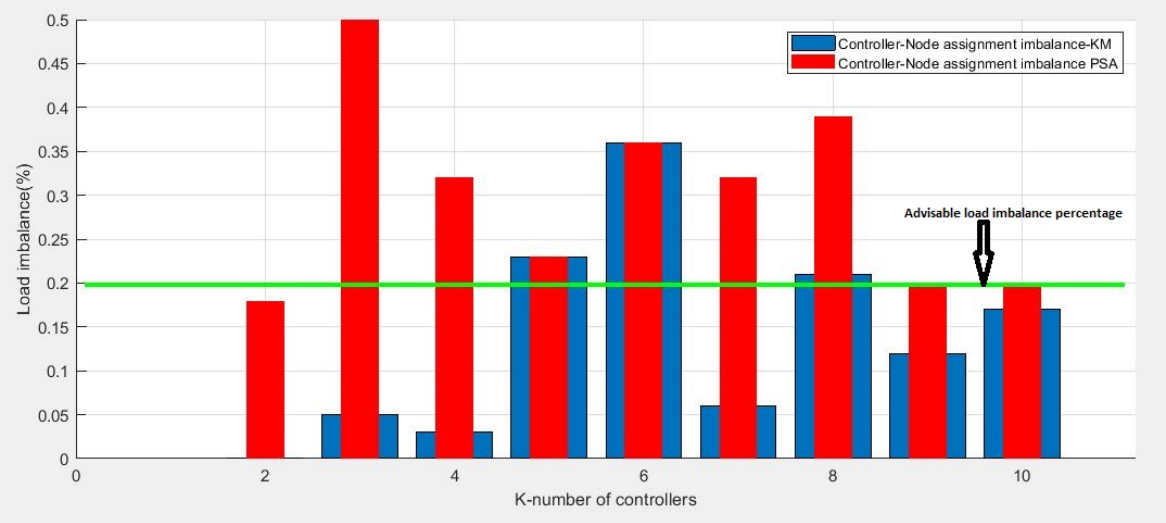


Figure 5.9: Nodes to controller assignment load imbalance with number of controllers (K).

In the above nodes to controller assignment imbalance bar chart the horizontal and vertical axis are the number of controllers K and nodes to controller assignment imbalance respectively. The node assignment difference between the maximum loaded controller and the minimum loaded controllers for each number of controller placement is presented in the plot for both models using the equation (4.12) of chapter four. As it is shown on the plot k Medoid clustering based model is better in evenly distributing nodes to their nearest controller, due to this the difference between the controller with maximum number of nodes and the minimum number of nodes is smaller than the PSA based model except for those K equal to five and six where controllers are placed in the same sites by both models and due to this fact it have the same nodes to controller assignment imbalance.

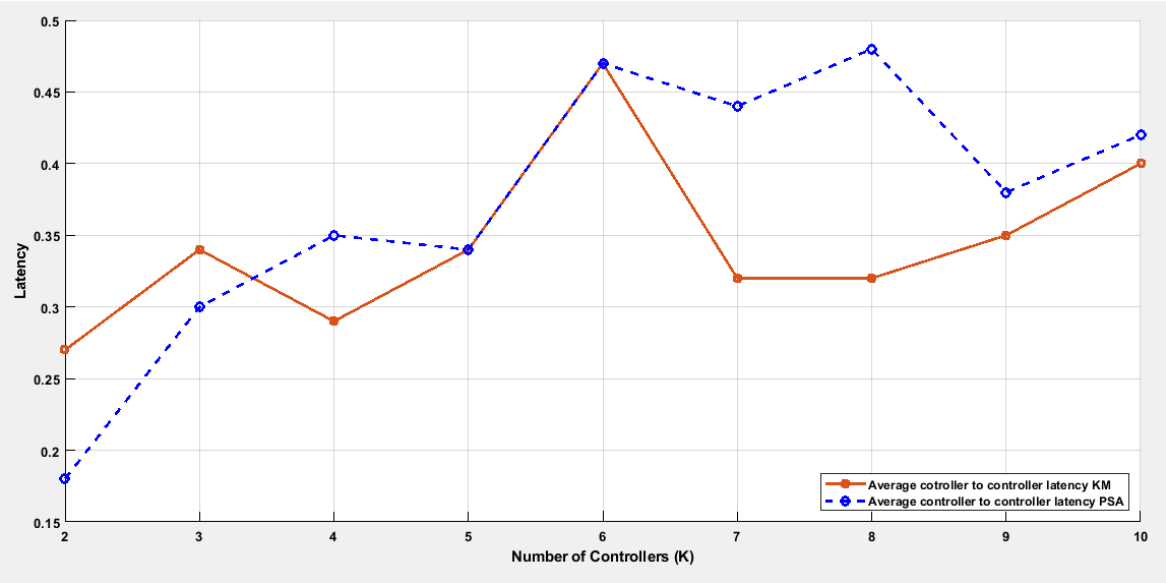


Figure 5.10: The average controller to controller latency with number of controllers (K).

Figure 5.10 shows the average latency between controllers for each number of controller assignment. The horizontal axis shows the number of controllers whereas the vertical axis shows the average latency between controllers. As it was explained in the last chapter the average controller to controller latency is the sum average of the latencies between controllers for each placement. The latency increase as we increase number of controllers until the number of controllers reaches six, then it decreases when the number of controllers reaches seven this is due to the fact that both models put the additional controller near the center of the topology, then it starts again to increase when the number of controllers becomes eight. The separation distance between the far end controllers increases with the increase in number of controllers. As it has been shown from the plot k medoid clustering model controller placement (solid brown line) is much better than that of the PSA model for a bigger number of controllers since it has a lower average controller to controller latency in the network.

The above experiment is done considering $K = 2...10$ controllers for both scenarios. As can be seen from the above results taking more than eight controllers will lower the average and maximum node to controller latencies but on the other hand the average controller to controller latency is increased, it will increase the control signal propagation delay between controllers. Taking this in to consideration and knowing that the maximum node to controller latency requirement is satisfied, we deduce that the minimum number of controllers required to adopt SDN in ethiotelecom WAN is eight controllers and the k medoid clustering based model is better than the Pareto simulated annealing method in the optimal placement of controllers in the WAN topology for larger number of network elements (Nodes).

6

CHAPTER SIX: CONCLUSION AND FUTURE WORKS

6.1 CONCLUSIONS

Since SDN move the decision making into a controller, it is essential to develop methods to find the optimal placement of the controller(s) in a network. This study finds the optimal placement of controllers on a network by suggesting the location and the number of controllers on an existing WAN. The objective is to find the minimum number of controllers and optimal placing of the controllers considering network latency for flow setup and nodes assignment to controller and connectivity between controllers.

The research presented in this thesis involved the optimization of wide area transport network of telecom operators to adopt SDN by determining the minimum number of controllers and their placements in the network topology. It presented a clustering and heuristic algorithm based models for finding the minimum numbers and optimal location for an SDN controller in ethiotelecom WAN. Measurement of the overall latency of this network was based on the delay in propagation among the control and data planes and between controllers. The model is assessed with respect to answering an essential question associated with the optimization of an SDN based WAN network: how many SDN controllers are needed in the network topology. The optimization results were obtained using MATLAB based pareto optimal controller placement tool.

The Network Topology is encoded and a k-Medoid clustering algorithm was employed to find the optimal placement and the minimum number of controllers required. The result found from the proposed method is compared with a Pareto simulated annealing heuristic based model result. As shown in this experiment, Increasing the number of controllers decreases the node to controller latency but it will also increase the inter controller latency, which will in turn have long propagation delay among controllers and affects the ability of the controller to respond to network events in a minimal time. Due to this fact there is always a tradeoff for telecom operators to select the number of controllers required. This work has proposed a method that enables SDN operators to identify the minimum number of SDN controllers, and to optimize the locations for their placement in a way that will achieve the best network performance.

6.2 FUTURE WORK

The assumptions underlying our examination of the optimization of ethiotelecom WAN included only propagation delay. We have not included queuing at the controllers,i.e., arrival and departure waiting times for packets before they are processed. Incorporating queuing delay into the model might improve the results by enhancing accuracy with respect to overall network latency. A further consideration is that Controller capacity can be considered in the model so that it more closely reflects real-life scenarios, it should also be taken into account.

BIBLIOGRAPHY

[1] Y. A.J. Agudelo, "Scalability and robustness of the control plane in software-defined networking (sdn)," PhD thesis, Universitat Politecnica de Catalunya, Barcelona, 2016.

[2] C. V. networking Index, "Forecast and methodology, 2016-2021, white paper," *San Jose, CA, USA*, vol. 1, 2016.

[3] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[4] R. Chua, "Juniper's new sdn strategy and contrail's starring role," *SDN central*, 2013. [Online]. Available: http://www.sdncentral.com/sdn-blog/junipernew-sdn-strategy-contrail-role/2013/01/.

[5] T. Koponen, M. Casado, N. Gude, J. Stribling, L. Poutievski, M. Zhu, R. Ramanathan, Y. Iwata, H. Inoue, T. Hama, *et al.*, "Onix: A distributed control platform for large-scale production networks.," in *OSDI*, vol. 10, 2010, pp. 1–6.

[6] G. Wang, Y. Zhao, J. Huang, and W. Wang, "The controller placement problem in software defined networking: A survey," *IEEE Network*, vol. 31, no. 5, pp. 21–27, 2017.

[7] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the first workshop on Hot topics in software defined networks*, ACM, 2012, pp. 7–12.

[8] D. Hock, M. Hartmann, S. Gebert, M. Jarschel, T. Zinner, and P. Tran-Gia, "Pareto-optimal resilient controller placement in sdn-based core networks," in *Teletraffic Congress (ITC), 2013 25th International*, IEEE, 2013, pp. 1–9.

[9] H. Aoki and N. Shinomiya, "Controller placement problem to enhance performance in multi-domain sdn networks," in *ICN*, vol. 120, 2016, p. 2016.

[10] S. Lange, S. Gebert, T. Zinner, P. Tran-Gia, D. Hock, M. Jarschel, and M. Hoffmann, "Heuristic approaches to the controller placement problem in large scale sdn networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 1, pp. 4–17, 2015.

[11] M. F. Bari, A. R. Roy, S. R. Chowdhury, Q. Zhang, M. F. Zhani, R. Ahmed, and R. Boutaba, "Dynamic controller provisioning in software defined networks," in *Network and Service Management (CNSM), 2013 9th International Conference on*, IEEE, 2013, pp. 18–25.

[12] V. Ahmadi, A. Jalili, S. M. Khorramizadeh, and M. Keshtgari, "A hybrid nsga-ii for solving multiobjective controller placement in sdn," in *Knowledge-Based Engineering and Innovation (KBEI), 2015 2nd International Conference on*, IEEE, 2015, pp. 663–669.

[13] J. Guillermo, "Multilayer data connectivity orchestration exploring the cengn-proof of concept," fujitsu, Tech. Rep., 2016. [Online]. Available: https://www.cengn.ca/multilayer-data-connectivity-orchestration-exploring-the-cengnproof-of-concept-javier-guillermo-fujitsu/.

[14] P. Simoneau, "The osi model: Understanding the seven layers of computer networks," *Expert Reference Series of White Papers, Global Knowledge*, 2006.

[15] I. Pepelnjak, "Management, control and data planes in network devices and systems," 2013. [Online]. Available: https://blog.ipspace.net/2013/08/management-control-and-data-planes-in.html.

[16] O. N. Foundation, "Sdn architecture overview," 2013.

[17] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, and D. Meyer, "Software-defined networking (sdn): Layers and architecture terminology," Tech. Rep., 2015.

[18] ONF, "Sdn architecture,onf tr-521," Open Networking Foundation, Tech. Rep., 2016.

[19] H. Farhady, H. Lee, and A. Nakao, "Software-defined networking: A survey," *Computer Networks*, vol. 81, pp. 79–95, 2015.

[20] 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 Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[21] T. A. Limoncelli, "Openflow: A radical new idea in networking," *Communications of the ACM*, vol. 55, no. 8, pp. 42–47, 2012.

[22] S. Sharma, D. Staessens, D. Colle, M. Pickavet, and P. Demeester, "Automatic bootstrapping of openflow networks," in *Local & Metropolitan Area Networks (LANMAN), 2013 19th IEEE Workshop on*, IEEE, 2013, pp. 1–6.

[23] G. Yao, J. Bi, Y. Li, and L. Guo, "On the capacitated controller placement problem in software defined networks," *IEEE Communications Letters*, vol. 18, no. 8, pp. 1339–1342, 2014.

[24] Y. Jimenez, C. Cervello-Pastor, and A. J. Garcia, "On the controller placement for designing a distributed sdn control layer," in *Networking Conference, 2014 IFIP*, IEEE, 2014, pp. 1–9.

[25] A. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, and R. Sherwood, "On controller performance in software-defined networks.," *Hot-ICE*, vol. 12, pp. 1–6, 2012.

[26] Y. Zhang, N. Beheshti, and M. Tatipamula, "On resilience of split-architecture networks," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, IEEE, 2011, pp. 1–6.

[27] Y. Hu, W. Wendong, X. Gong, X. Que, and C. Shiduan, "Reliability-aware controller placement for software-defined networks," in *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, IEEE, 2013, pp. 672–675.

[28] Y. Hu, W. Wang, X. Gong, X. Que, and S. Cheng, "On reliability-optimized controller placement for software-defined networks," *China Communications*, vol. 11, no. 2, pp. 38–54, 2014.

[29] F. J. Ros and P. M. Ruiz, "On reliable controller placements in software-defined networks," *Computer Communications*, vol. 77, pp. 41–51, 2016.

[30] M. Tatipamula, N. Beheshti-Zavareh, and Y. Zhang, *Controller placement for fast failover in the split architecture*, US Patent 8,804,490, 2014.

[31] N. Beheshti and Y. Zhang, "Fast failover for control traffic in software-defined networks," in *Global Communications Conference (GLOBECOM), 2012 IEEE*, IEEE, 2012, pp. 2665–2670.

[32] V. Yazici, M. O. Sunay, and A. O. Ercan, "Controlling a software-defined network via distributed controllers," *arXiv preprint arXiv:1401.7651*, 2014.

[33] S. H. Yeganeh, A. Tootoonchian, and Y. Ganjali, "On scalability of software-defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 136–141, 2013.

[34] K. Phemius and M. B. Thales, "Openflow: Why latency does matter," in *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, IEEE, 2013, pp. 680–683.

[35] F. J. Ros and P. M. Ruiz, "Five nines of southbound reliability in software-defined networks," in *Proceedings of the third workshop on Hot topics in software defined networking*, ACM, 2014, pp. 31–36.

[36] V. Bob, *Accessing the WAN, CCNA Exploration Companion Guide*. Pearson Education India, 2008.

[37] J. Metzler and A. Metzler, "The 2015 guide to wan architecture and design," 2015.

[38] Cisco, *Introduction to wan technologies*, 2012. [Online]. Available: http://docwiki.cisco.com/wiki/Introduction_to_WAN_Technologies.

[39] ——, *Wan concepts*, 2017. [Online]. Available: http://www.ciscopress.com/articles/article.asp?p=2832405&seqNum=5.

[40] R. Santitoro, "Understanding sd-wan managed services," Metro Ethernet FCisco(MEF), Tech. Rep., 2017. [Online]. Available: http://www.ciscopress.com/articles/article.asp?p=2832405&seqNum=5.

[41] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, *et al.*, "B4: Experience with a globally-deployed software defined wan," in *ACM SIGCOMM Computer Communication Review*, ACM, vol. 43, 2013, pp. 3–14.

[42] C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer, "Achieving high utilization with software-driven wan," in *ACM SIGCOMM Computer Communication Review*, ACM, vol. 43, 2013, pp. 15–26.

[43] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009, vol. 344.

[44] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[45] P. Arora, S. Varshney, *et al.*, "Analysis of k-means and k-medoids algorithm for big data," *Procedia Computer Science*, vol. 78, pp. 507–512, 2016.

[46] P. Czyzak and A. Jaszkiewicz, "Pareto simulated annealing a metaheuristic technique for multiple objective combinatorial optimization," *Journal of Multi Criteria Decision Analysis*, vol. 7, no. 1, pp. 34 –47, 1998.

[47] ANSI, "Technical report on enhanced network survivability performance," Tech. Rep. Technical Report No. 68, 2001.

[48] J. Schallenberg, "Is 50 ms restoration necessary?" In *IEEE Bandwidth Management Workshop*, 2001.

A

APPENDIX

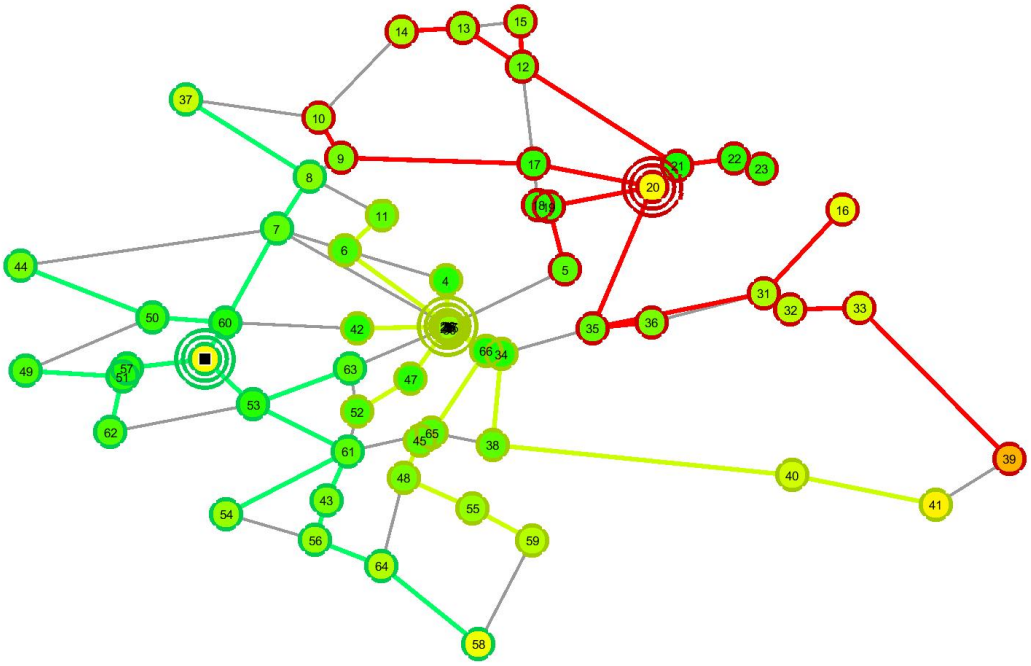
A.1 OPTIMIZATION OUTPUT RESULTS

Table A.1: Output Result Table

K	Max N2c (PSA)	Avg N2C (PSA)	Cont. imbal-ance (PSA)	Max N2C K-M	Avg N2C K-M	Cont. imbal-ance K-M	Avg. Cont. to cont. La-tency PSA	Avg. Cont. to cont. La-tency KM
1	0.53	0.29	0					
2	0.46	0.23	0.18	0.48	0.21	0	0.18	0.27
3	0.39	0.19	0.5	0.53	0.19	0.05	0.3	0.34
4	0.34	0.16	0.32	0.45	0.15	0.03	0.35	0.29
5	0.31	0.12	0.23	0.31	0.12	0.23	0.34	0.34
6	0.29	0.15	0.36	0.29	0.15	0.36	0.47	0.47
7	0.26	0.11	0.32	0.3	0.12	0.06	0.44	0.32
8	0.24	0.13	0.39	0.23	0.08	0.21	0.48	0.32
9	0.22	0.1	0.2	0.23	0.07	0.12	0.38	0.35
10	0.21	0.08	0.2	0.19	0.06	0.17	0.42	0.4

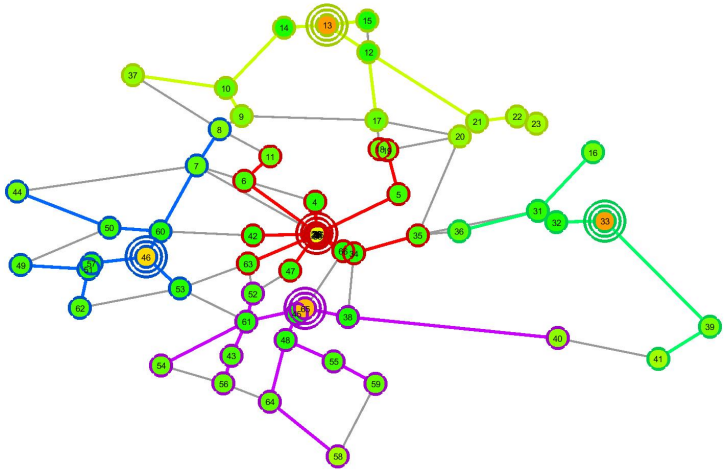
A.2 PLACEMENT RESULTS OF THE OPTIMIZATION MODEL

A.2.1 *K mediod model*



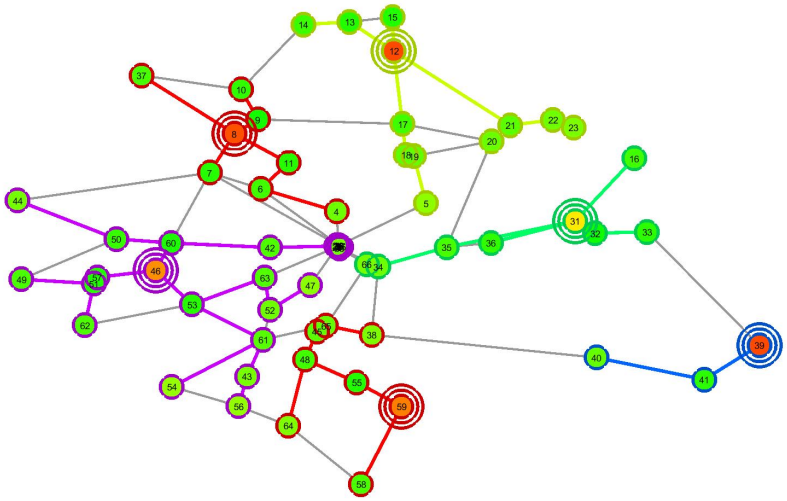
(a) placement for K=3.

Figure A.1: Controller placement using K Mediod model for K=3



(a) placement for K=5.

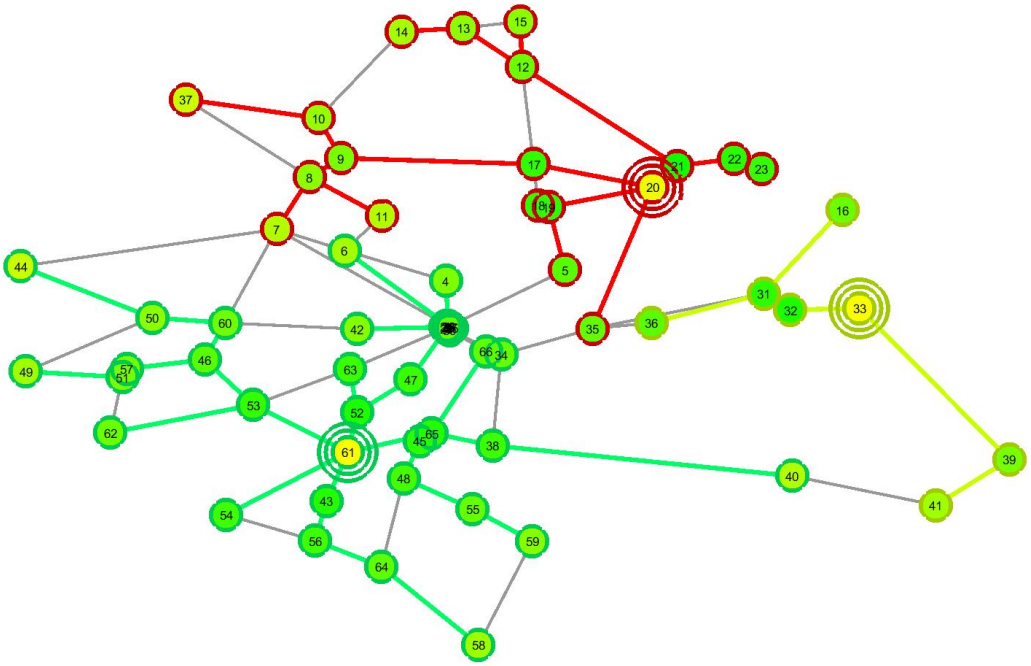
Figure A.2: Controller placement using K Mediod model for K=5



(a) placement for K=6.

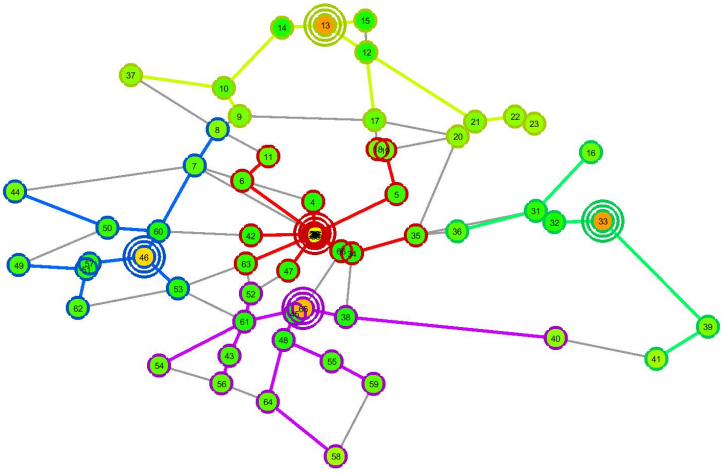
Figure A.3: Controller placement using K Mediod model for K=6

A.2.2 *Pareto simulated annealing model*



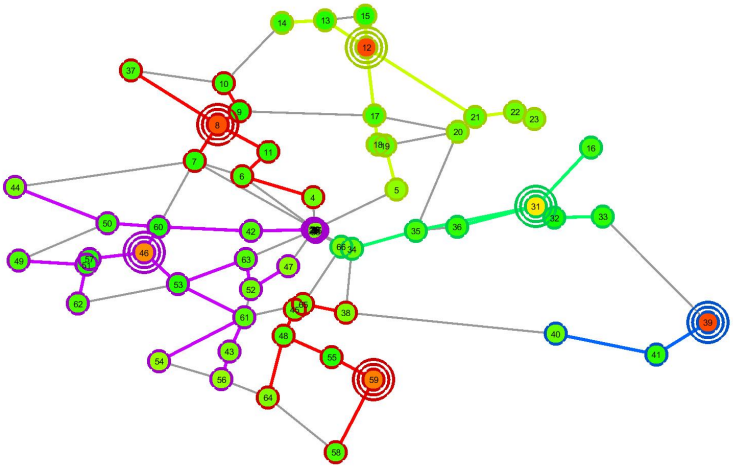
(a) placement for K=3.

Figure A.4: Controller placement using PSA model for K=3



(a) placement for K=5.

Figure A.5: Controller placement using PSA model for K=5



(a) placement for K=6.

Figure A.6: Controller placement using PSA model for K=6

A.3 SAMPLE ETHIOTELECOM NETWORK TOPOLOGY ENCODING

```
<?xml version="1.0" encoding="utf-8"?><graphml xmlns="http://graphml.
    graphdrawing.org/xmlns" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
    instance" xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns http
    ://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
  <key attr.name="id" attr.type="string" for="edge" id="d40" />
  <key attr.name="key" attr.type="int" for="edge" id="d39" />
  <key attr.name="LinkLabel" attr.type="string" for="edge" id="d38" />
  <key attr.name="hyperedge" attr.type="int" for="node" id="d37" />
  <key attr.name="geocode_id" attr.type="string" for="node" id="d36" />
  <key attr.name="label" attr.type="string" for="node" id="d35" />
  <key attr.name="Longitude" attr.type="double" for="node" id="d34" />
  <key attr.name="id" attr.type="int" for="node" id="d33" />
  <key attr.name="type" attr.type="string" for="node" id="d32" />
  <key attr.name="Country" attr.type="string" for="node" id="d31" />
  <key attr.name="Latitude" attr.type="double" for="node" id="d30" />
  <key attr.name="Internal" attr.type="int" for="node" id="d29" />
  <key attr.name="Testbed" attr.type="int" for="graph" id="d28" />
  <key attr.name="LastProcessed" attr.type="string" for="graph" id="d27" />
  <graph edgedefault="undirected">
    <data key="d0">6/03/18</data>
    <data key="d1">Ethiopia</data>
    <data key="d2">Country</data>
    <data key="d3">ethiotelecom</data>
    <data key="d4">Primary</data>
    <data key="d5">Taken from TEP Lot 4 LLD </data>
    <data key="d6">http://www.ces.net/events/2010/cef/p/rusu.pdf</data>
    <data key="d7">1.0</data>
    <data key="d8">REN</data>
    <data key="d9">Historic</data>
    <data key="d10">1</data>
    <data key="d11">0</data>
    <data key="d12">ethiotelecomFibre</data>
    <data key="d13">0.3.34dev-20120328</data>
    <data key="d14">0</data>
    <data key="d15">0</data>
    <data key="d16">e278b1b</data>
    <data key="d17">=</data>
    <data key="d18">09</data>
    <data key="d19">6/03/18</data>
    <data key="d20">0</data>
    <data key="d21">Simplified Fibre</data>
    <data key="d22">Topology Zoo Toolset</data>
    <data key="d23">0</data>
    <data key="d24">0</data>
    <data key="d25">2016_09</data>
    <data key="d26">2016</data>
    <data key="d27">2017_09_01</data>
```

```
      <data key="d28">0</data>
      <node id="0">
        <data key="d29">1</data>
        <data key="d30">09.033463</data>
        <data key="d31">Ethiopia</data>
        <data key="d32">NOC</data>
        <data key="d33">0</data>
        <data key="d34">038.750517</data>
        <data key="d35">Arada</data>
      </node>
      <node id="1">
        <data key="d29">1</data>
        <data key="d30">9.01846</data>
        <data key="d31">Ethiopia</data>
        <data key="d32">Blue dashed node</data>
        <data key="d33">1</data>
        <data key="d34">038.72250</data>
        <data key="d35">Sidist kilo</data>
      </node>
      <node id="2">
        <data key="d29">1</data>
        <data key="d30">09.019652</data>
        <data key="d31">Ethiopia</data>
        <data key="d32">Node</data>
        <data key="d33">2</data>
        <data key="d34">38.74725</data>
        <data key="d35">Yeka</data>
      </node>
      </edge>
      <edge source="5" target="6">
        <data key="d40">e39</data>
        <data key="d39">0</data>
      </edge>
      <edge source="5" target="10">
        <data key="d40">e42</data>
        <data key="d39">0</data>
      </edge>
      <edge source="6" target="0">
        <data key="d40">e43</data>
        <data key="d39">0</data>
      </edge>
      <edge source="6" target="5">
        <data key="d40">e44</data>
        <data key="d39">0</data>
      </edge>
      <edge source="6" target="44">
        <data key="d40">e45</data>
        <data key="d39">0</data>
      </edge>
      <edge source="6" target="60">
```

```
        <data key="d40">e46</data>
        <data key="d39">0</data>
      </edge>
      <edge source="7" target="6">
        <data key="d40">e47</data>
        <data key="d39">0</data>
      </edge>
      <edge source="7" target="8">
        <data key="d40">e48</data>
        <data key="d39">0</data>
      </edge>
      <edge source="7" target="10">
        <data key="d40">e33</data>
        <data key="d39">0</data>
      </edge>
      <edge source="60" target="6">
        <data key="d40">e167</data>
        <data key="d39">0</data>
      </edge>
      <edge source="60" target="42">
        <data key="d40">e168</data>
        <data key="d39">0</data>
      </edge>
      <edge source="60" target="50">
        <data key="d40">e169</data>
        <data key="d39">0</data>
      </edge>
      <edge source="61" target="43">
        <data key="d40">e170</data>
        <data key="d39">0</data>
      </edge>
      <edge source="61" target="52">
        <data key="d40">e171</data>
        <data key="d39">0</data>
      </edge>
      <edge source="61" target="53">
        <data key="d40">e172</data>
        <data key="d39">0</data>
      </edge>
      <edge source="61" target="65">
        <data key="d40">e173</data>
        <data key="d39">0</data>
      </edge>
      <edge source="62" target="51">
        <data key="d40">e174</data>
        <data key="d39">0</data>
      </edge>
    </graph>
  </graphml>
```