Optimum Controller Placements in SD-WAN deployment case of ethio telecom

By: Adugna Getu

Adviser: Ephrem Teshale (Ph.D)

A thesis submitted to the school of Electrical and Computer Engineering

Addis Ababa Institute of Technology

Addis Ababa University

In partial fulfillment of the requirements for the degree of Master of Science in Telecommunication Engineering

Addis Ababa, Ethiopia

October, 2021

Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

This is to certify that thesis entitled by *Optimum Controller Placement in SD-WAN deployment case of ethio telecom* prepared by Adugna Getu and submitted for partial fulfillment of the requirements for the degree of Masters of Science in Telecommunication Engineering complies with the regulations of the university and meets the accepted standards concerning originality and quality.

Signed by Examining committee

Internal Examiner _____Signature _____Date_____

External Examiner _____Signature _____Date_____

Adviser Ephrem Teshale (Ph.D) Signature _____Date_____

Dean, School of Electrical and Computer Engineering

_____

Declaration

I, undersigned declare that this thesis and the work presented in it are my own and have been generated by me as the result of my original research in agreement with accepted practices.

Optimum Controller Placements in SD-WAN deployment case of ethio telecom

. . . . . .

I confirm that I have acknowledged all my resources, paraphrased, and none of this work has been published before submission.

Adugna Getu October 11, 2021

_____

Signature Date

Acknowledgment

Much thanks and glory to my almighty God. This thesis work would not have been possible without the source of my understandings and strengths of the Lord almighty God. Also great thanks to my advisor Dr. Ephrem Teshale who offered his unlimited valuable inputs, guidance, and support.

Finally, my thanks extended to ethio telecom for awarding me to this program with the financial means, and special thanks to my valuable teammates and friends.

Credits to my lovely Kids Bilisa and Bilise! You are So Amazing! I never forget how much I missed you and you missed me during this long time process of work!!!

Thank You Again!

Abstract

The explosive increase of data traffic with the proliferation of devices leads to massive interconnections. As a result, the difficulty of network management operations and configurations, service provisioning, and heterogeneous networks due to continuous network deployments to meet the demand of ubiquitous connections evolved the network programmability concept as a solution.The SDN decoupling of the control plane from the data plane with logically centralized and programmable network came with opportunities and also challenges such as scalability and reliability that addressed with Multiple controller systems to improve network performance.

The number of controllers needed for a given network topology and where it should be placed optimally might vary with different use cases. For these varieties of use cases, different formulations of mathematical models with different optimization models are addressed in many research works. This thesis work focused on optimum controller placement for the WAN network topology of Ethio telecom optimizing propagation latency and controller imbalance with Constraint of controller capacity and the metrics trade-off analysis.

The optimum controller placement to minimize controller number by the constraints of propagation delay, controller capacity, traffic estimates, and load balance. Performance evaluation based on these parameters based optimization model of the heuristic approaches like Simulated Annealing and K-Medoid is performed. In this work, the trial and error way of determining controller number taking as the input parameter is replaced with determining it from controller capacity and node weights (flow requests from nodes). Also introduced the concept of load balance based on Jain's fairness index to measure how load is fairly distributed.

At last, the impact of controller capacity assessment on parameters of the optimum placement investigation showed controller capacity affects the placement metrics. As a result, K-Medoid showed improvement of 32% to 90% in load balance taking the same latency as a reference and 12.5% to 29% node to controller latency taking the same controller imbalance as a reference. Finally, optimum controller placement was identified and shown on google earth.

Keywords—Software-Defined Networking, SD-WAN, OpenFlow, SDN Controller, Heuristic Algorithm, Switches,controller Imbalance, Delay, Jain's Fairness Index

Contents

List of Figures

Acronyms

4G	Fourth Generation Network
5G	Fifth Generation Network
API	Application Programming Interface
OVSDB	Open vSwitch Database Management Protocol
OSI	Open System Interconnection
OS	Operating System
xDSL	Digital Subscriber Line Family
Wi-Fi	Wireless Fidelity
CPU	Central Processing Unit
NIC	Network Interface Card
NAT	Network Address Translation
MAC	Media Access Control
TCP/IP	Transmission Control Protocol/Internet Protocol
DNS	Domain Name Service (DNS)
IGP	Interior Gateway Protocol
ILP	Integer Linear Programming
IoT	Internet of Things
IP	Internet Protocol
IS-IS	Intermediate System-to-Intermediate System
ISO	International Standards Organization

IT	Information Technology
BGP	Border Gateway Protocol
CAM	Content-Addressable Memories
CE	Customer Edge
CPP	Controller Placement Problem
VPN	Virtual Private Network
ForCES	Forwarding and Control Element Separation
MPLS	Multi-Protocol Label Switching
VMM	Virtual Machine Manager
IETF	Internet Engineering Task Force
ISP	Internet Service Provider
VM	Virtual Machine
ISPs	Internet Service Providers
LTE	Long Term Evolution
SD-WAN	Software-Defined -Wide Area Network
KPI	Key Performance Indicator
LSP	Label Switching Path
MPLS	Multi-Protocol Label Switching
NFV	Network Function Virtualization
NOS	Network Operating System
OFConfig	OpenFlow Configuration
ONF	Open Networking Foundation
RAM	Random Access Memory
OSPF	Open Shortest Path First
P	Provider

PE	Provider Edge
QoS	Quality of Service
ARP	Address Resolution Protocol
PSA	Pareto Simulated Annealing
RTT	Round Trip Time
SDN	Software-Defined Networking
POC	Proof of Concept
SLA	Service Level Agreement
TCAM	Ternary Content Addressable Memory
POCO	Pareto-based Optimal Controller-Placement
vCPE	Virtual Customer Premises Equipment
CPE	Customer Premises Equipment

Chapter 1

Introduction

1.1 Background

Internet demand and data traffic explosively increasing with the proliferation of devices and applications within decades. With this amazing internet growth, providing access to the massive internet connections under the hotheaded increase of applications and data traffic volume become operators, service providers, and vendors' business burden and opportunity rise together. To accommodate these demands, continuous deployment of network infrastructure by service providers and telecom operators leads to the network architecture being highly heterogeneous and complex, making network management, difficult. Besides this, having no global view of the network, poor network performance issues and service provisioning becomes their challenging duty.

The promising solutions of Software-Defined Networking (SDN) for the current network infrastructure in service providers and operators emerged with different opportunities and challenges. These implementations of deployment need to be planned and optimized properly to realize the SDN deployment for telecom operators and service providers to reap its benefits. Optimized planning of components of SDN architectures offers significant advantages for cloud data centers, cellular networks, transport networks (Optical), and for service providers in Wide Area Networks (WAN). Simplifying data forwarding, allowing managing a network in a flexible, easy service provisioning, and guaranteeing Quality of Experience/Service in such widely distributed and heterogeneous network devices is important. WAN network is one of the use cases of SDN that consists of the widely distributed and heterogeneous network environments where the aforementioned network issues are usual. The use case of this thesis is Software-Defined Wide Area Network (SD-WAN) as it provides better wireless integration WI-FI and 4G/5G in business applications to Internet of Things (IoT) and other broad network services [1].

The logically centralized single controller scalability (a limited capacity) and reliability (a single

point of failure and vulnerable attacks) are some of the challenges of SDN deployment [2]–[4]. Many researchers in [2], [5], [6] works, proposed distributed multi-controller systems as a solution for such challenges. In distributed controller system, controllers' performance that affects the entire network performance is affected with setup processing time where propagation delay of switches to controllers dominates in WAN [7], [8]. Also, the load imbalance where some controller(s) overloaded increases controller processing time, and synchronization overhead is added to affect the network performance. To obtain a scalable, balanced, and failure-resilient topology i.e., better performance, it is necessary to determine a minimum number of controllers and its optimum location [9]–[11], while ensuring low latency between nodes and associated controllers, as well as between controllers [9]. Simultaneously, it is necessary to keep balanced the processing loads of controllers [10], and within their operating capacity [12]. The other important that needs attention is traffic estimates and analysis of the trade-off between metrics. Therefore, the optimum placement of controllers to address how many controllers are required and where to place them [9], in the perspective of capacities of controllers [12], propagation latency between controllers and nodes, and inter-controllers [9], and load distribution among controllers [10] improves controller performance. The optimum solutions will increase the ability of the network to react to events quickly, leading to a more robust topology, even when failures happen. In this thesis, the optimization models with the mathematical formulation of the controller placement, which minimizes the number of controllers, with respective controller capacities, propagation delay, and traffic estimates was presented. The traffic flow request from each node is varying flows and estimation of the flow request emanate from each node is done. The optimization models for optimum placement of controllers are analyzed and evaluated. In this work of load balancing, the Jain's fairness index was used as a metrics of load balancing in SDN the work of author [13] fairness resource allocation in a shared computer system to minimize the controller imbalance measuring how the load distribution is discriminated.In this work optimal placements of the minimum number of controllers that can accommodate all flow requests arrive at the controller analyzed with generic and heuristics methods.

1.1.1 Wide Area Network

Wide Area Network (WAN) is a telecom network interconnecting multiple network elements over large geographic areas to connect branch offices, data centers and to reach cloud services. With the rapid evolution of telecommunication networking and technologies, consistently high demand of capacity and high quality increases WAN networks. Multi-Protocol Label Switching (MPLS) provides full-mesh connectivity, high reliability, and separation of private traffic flows from the rest of IP traffic. As a result of these advantages, traditional and some of the current

existing WAN networks use MPLS but its high bandwidth costs, configuration overhead and time to upgrade discourages the use of MPLS. In section 2.5.1, WAN network and its architecture was discussed in detail. Adopting Software-Defined Wide Area Network (SD-WAN) for the expensive WAN technologies which provides overlay architecture brings better network performance, automation, and easier management [14], [15].

1.1.2 Software-Defined Wide Area Network

SD-WAN is a software-defined networking architecture that is used to solve WAN's widely distributed and heterogeneous network nature challenges. SD-WAN reduces latency, is higher scalable than VPNs, centralized and simple management, simpler deployment, and maintenance. As Cohn in [1] described WAN network is the foundation of remote connections because of the pandemic Covid-19 SD-WAN adoption speed up, as the need for remote work-forces [16]. It simplifies making connections of widely dispersed and rapidly changing network infrastructures [17]. There are also SD-WAN challenges as described in section 2.5.8.

1.2 Problem Statement

The promising solutions of SDN for the current network infrastructure in telecom operators emerged with different opportunities and challenges. To realize the deployment of SDN and mature the development of SDN the opportunities assessments and challenges of the new feature of the network need to be stated. Moreover, solutions for the challenges need to be proposed and/or addressed with implementations. These implementations deployment need to be properly planned and optimized to realize the SDN deployment benefits for telecom operators [2], [18]–[21].

As the controller is the brain of the SDN network, that generates forwarding rules and populating them to the forwarding devices; indicates that its layout influences the performance of the network. The properly placed controller at its optimum location minimizes the link delay, optimizes load balances, robustness, and reduces energy consumption. Different authors tried to address the optimum controller placement. Most of the authors use propagation delay of the switch to controller and controller to controller, and Load imbalance of switch to controller assignment as evaluating metrics for given network topology. However, to realize the deployment of SDN with optimum solutions should consider parameters like controller capacity [22]–[24], Flow setup time [25], availability [22], [26], traffic estimates, and parameters such as energy and cost that are affected by other parameters. The controller load balance based on the number of the forwarding devices or switches assignment is not visible in the different switches capacity

and different traffic flow requests from each switch. This thesis specifically focused on controller placement with controller capacity and traffic estimates parameters besides the link delay and controller imbalance as the extension work of earlier authors [8], [9], [11], [27] with proper tools of the scenario. Generally, the thesis tries to answer the following questions:

1. What is the minimum number of controllers required to deploy SDN for the case study?
2. Where the controllers should be placed to minimize controller numbers for the constraints of latency, controller capacity, traffic estimates, and controller imbalance?
3. Are the controllers placed with minimum load imbalance within their operating capacity?
4. What is the impact of controller capacity on latency and load distribution, controller imbalance, and load distribution?
5. What is the trade-off between latency and load balance? Controller capacity and load balance?
6. Which heuristic optimization shows better performance in controller placement of our case study that minimizes latency and maximizes load balance while controller capacity is not exceeded?

This thesis work's answer for the aforementioned questions might rely on the network topology and characteristics and the network performance metrics selected for controller placements. Thus choosing the optimal number and placement of the controllers decided by performance evaluation metrics and flow requests from each node. The capacity of the controller analytically affects the metrics. Moreover, it can determine controller number and indirectly impact controller placements.

1.3 Objective

The study aims to contribute to controller placement problems with the perspective of different parameters as a metric of evaluation.

1.3.1 General Objective

The main objective of the research is

- To propose SD-WAN implementations with optimum controller placement with the correlation of traffic estimates and network performance parameters.

1.3.2 Specific Objective

The research specific objective is:

- To determine the proper controller number required to handle the flow request of each node in the SD-WAN deployment for telecom operators.
- To identify optimal required controller number and their optimal placement.
- To identify the impact of controller placement with the correlation of traffic estimates, energy, and controller capacity
- To propose a flow-based load balancing technique for optimum controller placement instead of switches assignment numbers.
- To analyze the proposed implementation approaches of heuristic methods with exhaustive research-based POCO framework with and without controller capacity.
- Investigate the controller capacity impact in the selection of Pareto frontier placements.

Based on the result of the specific objectives, this work summarizes, suggests, and proposes future works. To recap, it gives meaningful insights into controller placements problems in SD-WAN deployments especially for the case study of the work.

1.4 Methodology

In this section, the approaches used in this thesis work are briefly introduced to give hints as depicted the below subsections.

1.4.1 Literature review

The related works and the state of the art of researches, magazines, articles, journals, international conferences, books, and webinars of SDN deployment and controller placements were reviewed. The research gaps were analyzed, identified, and formulated a refined research objective. The gaps formulated on SDN controller placement fulfilled the implementation of the formulated research goal. Moreover, knowledge from the related works reviewed was gained, and modeled and implemented a feasible approach for the deployment and implementation of the controller placement.

1.4.2 Data collection

Every data required for this work is collected with the permission of ethio telecom with the responsibility of keeping its confidentiality. IP core WAN data collected are traffic matrices that are used for estimation of flow requests from each node, site locations with their coordinates, site names, and site link interconnection (edges). The controller capacity data is collected from the product specification of the HP VAN SDN Controller.

1.4.3 Network topology

The network topology of the collected data is transformed to a suitable format. During the study, the data as a network topology that includes the node weights which is the flow request estimation from nodes, controller capacity, topology, and distance matrix. The network topology is created from transformed data with the suitable format as a .topo.mat file was prepared.

1.4.4 Mathematical Formulation Model

The mathematical model of the controller placement problem is formulated as non-deterministic polynomial Integer Linear Programming (ILP) NP-hard [28], [29]. The mathematical model that meets the requirements of the objective with its constraint variables is formulated. In addition, the optimum solution methods of the identified mathematical model were used for the experimentation and simulation.

1.4.5 Modeling and Emulation

The simulation of the model for traffic flow requests estimation is emulated with Mininet emulation for network topology. For the emulation, the Mininet installation was performed on Ubuntu Operating System that is running on VMware but not on the host system. Once the setup of Mininet installation is completed all supporting applications and tools are installed, then the virtual network is created. POX is the SDN controller i.e., the Open Source OpenFlow python based Software is used for traffic flow request estimates as a controller that emanates from each node. POX controller platform is designed is Open source and Mininet built-in is suitable and easy for evaluation setup. On the other hand, heuristic algorithms Simulated Annealing and K-Medoid methods used for SDN controller optimum placement integrated POCO framework Matlab based with used for this thesis.

1.4.6 Performance evaluation

The network topology of existing infrastructure encoded, traffic flow from each node estimated, link delay, controller capacity, edges used as input parameters. From the input parameters, the number of controllers able to handle all flows from the node is determined. The performance evaluation is performed based on an average and maximum node to controller latency, average and maximum inter-controller latency, controller imbalance that measures how the load distribution is discriminated while controller capacity is not exceeding. The optimum placement, controller number are correlated with latency, load balance, energy, and cost.

1.5 Scope and limitation

1.5.1 Scope of the study

The dynamic changing of emerging technologies, services, and applications boosted the traffic demands of service providers. These rise of demands of traffic over wide range of services increased network deployments that increases network complexity. In today's telecom networks almost every service uses the backbone IP core WAN network. Hence the improvements of the IP WAN network directly or indirectly improve the network performances and ease management. Having this notion as input the scope of the study is limited to the IP core WAN network. As the case study only ethio telecom IP Core WAN network is used. These high volumes of IP core traffic need scalable, reliable, and automated management and service provisioning.

The challenges and expensiveness of MPLS in WAN networks might be solved with SD-WAN deployments. During this deployment the number of controllers that can accommodate the flow requests from all nodes and node to controller assignment not exceeding controllers' capacity determined. The assumptions of link bandwidth capacity to the controller Network Interface Card (NIC) is used to estimate the traffic flow and calculated back to the node average flow, and using traffic matrix correlation we estimated it. The study scope is the optimization of these controllers' placement with two heuristic algorithms integrated with combinatorial-based POCO framework exhaustive search performance evaluated and comparative analysis deduced.

1.5.2 Limitation of the study

In this system model, we used the same controller capacity even though our system supports different controller capacities as constraints. The delay computation is computed based on geographical coordinate distances that are shorter than the real link distance. For traffic flow requests we used simple Mininet topology. However, this does not significantly affect the re-

search output as all paths are affected to similar extent.

1.6 Contribution of the research

The final result of the research has significant contributions to academic researches and professional works. The deployment of SD-WAN trends and challenges explored in this work gives a better view of the technologies of the telecom networks provided by operators and service providers. The use cases and planning criteria addressed have contributions for the strategic planning deployments in the real scenario of telecom networks and reveal the ISPs' role ahead of deploying the technology. This work goal is not to push the SD-WAN deployments, but gives better understandings of the technologies for the concerned body and explore how they could attain the benefits the network architecture brings.

For the operators planning to study how they should deploy, how many controllers they need, and where they have to place to maximize the opportunities is the core point addressed in this work. In the controller placement, the controller placement problem formulated, which minimizes the number of controllers, choosing their location, nodes assignments to controllers while satisfying controller capacity, propagation latency, and fair load distribution. As long as our understandings and review of the other works reviewed in the literature we have not come across who used fairness index for load balance of controller placement. This work assumed each node's traffic flow request is different and we estimated it with simple network topology using Mininet to resemble real telecom operators networks. The minimum number of controllers required for the SD-WAN deployment is determined based on the controller capacity and flow requests from all nodes rather than the trial and error way of determining. We also considered the failure case of the controllers with the concept of load sharing and standby pool based rather than as redundant standby. Then keeping the minimum number of controllers based on flow-based fairness index (i.e., load distribution fairness among controllers) and propagation latency. The two heuristic algorithms and capacity-based models that optimize the placement of controllers with the minimum number of controllers are explored and analyzed to indicate the suitable model. The result accuracy level is based on the type of algorithms and techniques used during the research implementations. It could also enhance the researches for telecom operators and even vendors that could be referenced in academic and professional works.

1.7 Literature Review

The research of optimum controller placement is changing with network scale and network applications use cases requirements due to controllers' layout influences the performance of the network. Different authors contributed by starting formulating the controller placement problem mathematically to optimally place the SDN controllers to the constraints of performance parameters and use cases. This section reviews briefly works related to controller placement issues with constraints of latency, controller capacity, load balance, energy, and traffic request demand estimates.

Heller et al. in [8] initiated and formulated the controller placement problem as a general facility location problem. The authors' analysis focused on controller placement based on both average nodes to controller latency and worst-case node to controller latency trade-off observation.

The author in [9] studied the optimum controller placement intending to minimize the propagation latency from nodes to assigned controllers and among controllers without considering the controller capacity and traffic estimates. Markos in [9] addressed the imbalance of load for node assignment but not looked through the tradeoff between the load balance and propagation delay along with load distribution. Similarly, the author [9] and other authors load imbalance method is with the assumptions of every node capacity and link capacity to their controller is the same. For the controller placement optimality, K-Medoid and Pareto Simulating Annealing methods were used. In this thesis, the extension work of the authors [9], [11], [30] continued based on controller capacity and traffic request demand estimates besides the propagation delay. Similarly, Zhao and Wu [27] focuses to reduce the link delay through an integer linear program and heuristic algorithm to provide scalable controller placement.

Hock et al. [31], [32] Offered a new approach for controller placement to improve the resiliency and failure tolerance of software-defined networking performance. In their work different network performance metrics such as inter-controller latency and node to controller latency at average and maximum delay trade-offs were investigated and concluded that there is no single optimal placement solution.

For the optimum controller placements, different techniques and heuristic approaches addressed extending the work of [31], [32] further considering capacity in [11], [30]. Lange et al in their works of [11], [30] heuristic approach for controller placement proposed by extending the POCO tool focus trade-off analysis of resulting accuracy and time of computation investigation. The authors' two objective metrics average latency of node to controller and controller imbalance capacitated based as capacity bound but the load balance concept based on deduction of maximum and minimum number assigned nodes to a controller, and also controller capacity based

flow request not indicated. This method is not viable for varying flow requests traffic volume from different nodes to controllers.

Harned [33] expanded the ideas of [11], [30]–[32] works enabling POCO framework for multi-objective controller placement based on evolutionary algorithm. This thesis used the POCO framework [11], [30], [32] as a tool with modifications on load imbalance based on controllers' load introducing the fairness index (Jian's index) and controller capacity in controller placement using heuristics methods.

The weight of the nodes is not considered and assumed to be the same though it is uneven while Yao et al [12] considered the weight of the nodes when locating the controllers. The first initiated capacitated controller placement problem was addressed in [12] with the objective function of minimizing the maximum propagation latency with the constraint to controllers load not exceeding the controllers capacity. In their work, they considered that the capacities of controllers, load assigned to controllers, and propagation latency between nodes and assigned controllers with the assumption of controller's capacity is limited based on link bandwidth and a limited number of switches assigned to the controller. The authors even though not considered the load balance among the controllers, concluded their method minimized the required number of controllers. Their work revealed also the node assigned to their nearest controller as capacity is not exceeded. Whereas the authors in [34] claim they developed the comprehensive approach for capacitated controller placements to minimize the controller number based on propagation latency, resiliency, and scalability as a multi-objective-based placement. The authors used controller capacity and inter-controller load distribution besides to mentioned metrics.

While He et al [25] controller placement method is used to optimize minimum average flow setup time concerning different traffic conditions to improve network performance. Having the link bandwidth notion addressed in [12], this thesis work, tried to estimate the traffic requests arrive at the controller and from each node using Mininet and the traffic matrix of the nodes.

Hou et al. [35] addressed hierarchical multi-controller deployment for the WAN network that divides control planes into multiple levels. The authors are based on load balancing, reliability, and latency using an improved Louvain algorithm. However, flow-based load balance and controller capacity issues were overlooked.

Khorramizadeh and Ahmadi [23] proposed controller capacity and load-aware controller placement with two phases. Phase one determines the number of controllers that minimizes the overall cost of deployment and in phase two the controller location that balances controller load minimizing inter-controller latency. Two greedy approaches were used for the location of controllers and allocation of controllers' load.

The energy-aware based works of literature [36]–[38] addressed minimizing energy consumption in SDN for controller placement and controller to switches associations while [39]–[41] showed the usage of software-defined networking role in minimizing energy and carbon emission extending to environmental impacts and also depicted methods of reducing energy consumption by SDN itself.

Muller et al [42] use the concept of controller capacity with its backup capacity and switch flow requests of the same capacity with maximization of connectivity controller to switches to enhance controller placement improving SDN survivability. The same flow requests and the same capacity is assumed for controller placements survivability.

Jalili et al improved their work of [43] in their work [44] as multi-criteria-based controller placement such as hop count, propagation latency of node to the controller, and utilization of links used to assign switch to a controller and analyzed the effect of these objectives on quality of service. The trade-offs among the metrics analyzed proposing Genetic Algorithm heuristics for the controller placement. Their result showed that the assignment based on their approach was better efficient regarding the link load balancing problem.

Ahmadi et al in their work of [22], [26], [45] focused on the Load balancing scenario of the Controller placement problem using different techniques like Flow-based controller placement that specifically addressed in [25], [46] and dynamic switch to controller assignment as indicated in authors of [47]–[51].

While Cai et al in [52] addressed the constraints of device hardware indicating that the controller capacity is limited in their work and defining the controller capacity and determining the minimum number of controllers. However, it gradually increases the number of controllers until no overload happens. Their objective is only to meet the load balance of controllers to avoid overload. It resembles our work in their conclusion the load balance avoids the controller overload even though our approach uses latency metrics besides load balance as important metrics for optimum controller placements. However, the load balance the authors used is the difference of the maximum and a minimum number of switches assigned to the controller.
In nutshell, the controller placements optimization using various metrics used. the controller capacity considered for the capacitated controller placements the real product specifications not used. on the other hand the load balance is determined with the difference of the maximum and minimum nodes assigned to controller with the assumptions of nodes traffic flow request to the controller is the same.

This thesis considered the controller capacity and traffic request estimates from each node as the important constraints of the optimum controller placement and controller capacity to keep better controller processing time for network performance. This is because the latency of inter-

controllers or between switches and controllers increases since processing time when controllers are overloaded. The other fact is that the controller capacity should be considered is every network resource limitation because of controllers run on servers with limited resources (CPU, bandwidth, memory). This limited processing capability resulted in the production of different types of controllers with different capacities that affect network performance and further affect also the deployment goal and costs. Considering traffic estimate is also mandatory to select suitable controller capacity that can handle the traffic request demand and to resemble the real network in which request flow from each forwarding device is not the same. Similarly, the load balance is measured with controller imbalance metrics derived from Jain's fairness index.

1.8 Organization of the Thesis

This thesis work consists of six chapters. The first chapter introduces the fundamentals concepts of the entire work of this paper. Also in this chapter, describes the problem statement, aims of the work, how to achieve the goals of the work, scopes, and limitations, the significance of the study, and the related works of the thesis.

In the second chapter the fundamentals and principles of networking, its architectures are briefly discussed to give insight into the concept of traditional networks, software-defined networking, SD-WAN, and WAN architectures with its use cases discussed. The controller placement of SDN controller discussed in chapter three of this paper addressing the basics of flow setup processes, controller selection benchmarking, and placements metrics. In this chapter controller capacity and traffic, demand requests elaborated related to this case study.

In chapter four focused on the optimizations system models, methods used in the model, and generic POCO framework tools optimization and heuristics algorithms used in this work discussed explicitly. The fifth chapter covers the performance evaluation part of the study with subsections of evaluation setup, evaluation methods, and finding and results in analysis. In the last, this work with recommendations and future study was concluded.

Chapter 2

Software-Defined Networking

Telecommunication and Information Technology (IT) systems consist of many complex components and architectures of either separated vendors or developed by various vendors, and delivered to the operators who possess the hardware and ensure several network services and functions. The components can be many different versions and vendors due to continuous deployments in the telecom industry. In telecom services, there are characteristics requirements in each version and vendors' components for communication requirements.

There are significant aspects in telecom services such as resilience (detecting and fixing errors or failures in less time), multivendor (able to synchronize different vendors' artifacts and deployments), and high availability (five nines i.e., 99.999% availability demands). To cope up with these significant aspects and ever-increasing bandwidth demand, current telecom providers networks have evolved to multi-layer infrastructures composed of many IP core networks with switching and routing [53], [54]. In the telecom industry, the most important requirement is availability as addressed in [55] that depends on the speed of how it reacts to the failures and registration to service discovery.

As a result of continuous deployments to cope with rising demands and technological changes, service providers' networks become more complex and heterogeneous. The management, configurations, and service provisioning of these complex networks become a challenging duty.

This nature of networks is highly observed in traditional networks for few companies while it is still an existing network in others.

In traditional networking, devices and services are hardly compatible due to applications of network elements that are vendor and language-specific. This is because control functionalities and data forwarding devices are vertically integrated on the same devices, network control is distributed in these devices. Moreover, difficulty in management, configurations, and service provisioning rises with the scalability of networks increases.

The notion of Software Defined Networking emerged as a solution simplifying the management and configuration. The significant contributions of decoupling control plane from the data plane and open flow architecture changed the network architecture natures and characteristics. It enabled automated and orchestrated network management and control, configurations, service provisioning, and network programmability. As a result, the telecom aspects requirements may be achieved more easily.

In this chapter, the traditional networks architecture and principles, SDN networks with their architectures and protocols, existing WAN networks, SD-WAN as a use case of SDN, SD-WAN deployments, and network performance metrics was discussed briefly.

2.1 Network Management and principles

A set of components of telecom systems computers and telecommunication network technologies interconnected based on physically wired, optical, and wireless radio frequency methods that might be arranged in a variety of network topologies to provide communication services. This group of two or more interconnected computing devices creates a network that may include sub-networks or smaller subdivisions of the networks. Internet service providers (ISPs) can manage their large networks of thousands of network addresses by sub-networking with inter-network connections that take place at the network layer. Network addresses serve for identifying and locating network nodes by communication protocols, for instance, internet protocol (IP).

How these inter-connected devices work is described with different standards that define a framework for implementing telecom and computer network protocols. Open Systems Interconnection (OSI) model and Transmission Control Protocol/Internet Protocol (TCP/IP) standard models for network communications adopted. These standards enabled coordination among various vendors and versions of network communications. The discussion on the standard models-based network layer is briefly addressed in the next section.

2.2 Network Layer

The International Standards Organization (ISO) developed the OSI model that defines the framework for implementing network communications protocols of a hierarchical seven (7) layers architecture. Each of the layers having specific functionality to accomplish. While the US department of defense developed TCP/IP the non-proprietary developed to provide end-to-end data communication specifying how data should be packetized, addressed, transmitted, routed, and received at the destination with four (4) abstraction layers. OSI model is not

implemented and used as reference only but TCP/IP implemented. Each layer's collaboration in data transmission from one user A to the other user B across the globe is described in Figure 2.1 scenarios.

To elaborate on the network layers shown in Figure 2.2, assumed that A sends a message to B in the scenario case.

1. **Application layer**: at this very top layer of OSI, sender A finds network applications that produce that, which has to be transferred over the network. This layer serves as a window for the application services to access the network. E.g. mail services, directory services, and file transfers. Browsers, Gmail, Skype, zoom are some of the applications.

2. **Presentation layer**: the data from the application layer is extracted and manipulated as per the required format to transmit over the network i.e., translates the data. The message (the data) is compressed, encrypted (if any secure data), and converted to bits to be transmitted. At receiver B this application service displays the received information to the receiver decompressing, decrypting, and convert bits back to the message.

3. **Session layer**: when data sent from A arrive at the session layer, connection establishment, authentication, and security ensuring are performed at the session layer. At the receiver, B connection terminated. At this layer, synchronization to identify errors to avoid data loss and dialog controller to start A and B systems communicate with each other in half or full-duplex.

   These three layers (application, presentation, and session layers) are integrated as a single layer in the TCP/IP model as the Application layer and done by network applications themselves.

4. **Transport layer**: transport layer operated by the Operating System (OS) provides services to the application layer and network layer for the End-to-End delivery of the complete message. Data sent from A is formatted in upper layers and segmented at the transport layer. This layer implements also flow and error control to ensure proper data transmission.

Figure 2.1: Network communication

It adds source (A) and destination (B) port numbers (service point address i.e., sender A needs to know it) to segmented data in its header and forwards it to the network layer. At receiver B the transport layer reads the port number from its header and reassemble the message then forwards the received data to the respective application. Services at transport layer connection-oriented (the three-phase process of connection transfer and termination) or connectionless (only data transfer). For connection, less receiver B does not acknowledge data receipt but allows fast communication. In connection-oriented receiver B acknowledgment i.e., reliable service but slow. The seven OSI layers and four TCP/IP layers are illustrated in Figure 2.2.

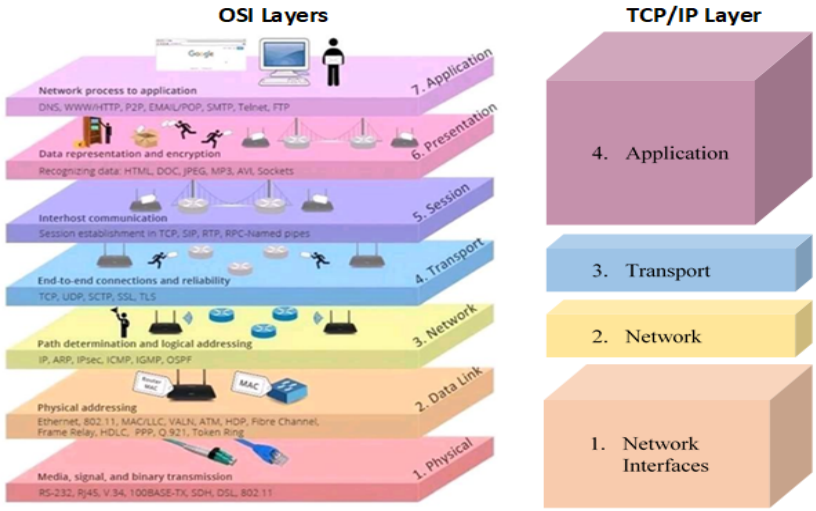


Figure 2.2: Network Layer [56]

5. **Network Layer:** The layer performs the data transmission from host A to the other host B located in a different network including packet routing i.e., shortest path selection. The sender's and receiver's IP addresses are added to the data header to identify each device on an internetwork to form a packet. e.g., routers.

6. **Data-link layer**: data link layer ensures data transfer from node to node is error-free over the physical layer. When a packet arrives in a network it encapsulates the sender and receiver's MAC address (obtained from ARP) in the header forming frames and transmit to the host using its MAC address. Flow control is also performed by this layer for data rate consistency and access control to determine which device is over the channel. Handled by Network Interface Card (NIC). E.g. switch and bridge.

7. **Physical layer**: it is the actual physical connection between sender A and receiver B that transmits individual bits (0s and 1s) from one node A to B. it provides a clock for

synchronization at the bit level, bit rate, physical topologies, and transmission mode. For instance, hub, modem, cables, fiber optics.

Datalink layer and physical layer integrated as a single Network interface layer in TCP/IP model. The network layer and network interface layer are hardware layers.

2.3 Traditional network architecture

Briefly looked at the traditional network architecture to visualize how and why it has evolved to be complex in today's era. The network architectures of routers and switch functions are traditionally separated into data planes, control planes vertically integrated into the same devices and management planes. The planes are capable of horizontal communication with peer elements in adjacent entities in topology, and vertical communication with other categories [57].

1. Data plane: handle the majority of packets by switches with its various ports that are used for transmission and reception of packets and a forwarding table associated with its logic. It is responsible for the header modification, packet scheduling and packet buffering.

2. Control plane: the control plane is involved in many activities with the primary role to keep current information in the forwarding table so that the data plane can independently handle an as high percentage of the traffic as possible. It also processes several various control protocols that may affect the forwarding table, based on the configuration and type of switch. Open the Shortest Path First (OSPF) and Border Gateway Protocol (BGP) are examples of Control planes protocols.

3. Management plane: administrators of the network configure and monitor switches and networks through the management plane. It extracts information from or modifies data in the control and data plane with the appropriate action.

2.4 What is SDN?

Software-Defined Networking is the networking architecture in which a control plane is decoupled from a data plane implemented at layer 2 and layer 3 of the OSI model. This architecture simplifies the management, service provisioning, and complexity of the network. It is a promising solution for today's heterogeneous and complex networks enabling automation and orchestration in data centers, cloud computing, and branches.

For instance, the 5G systems deployments will emerge with challenges of traffic-engineering complexities. The managing of massive changes in operators' bandwidth densities and network slices assisted and offered with SDN [58].

2.4.1 SDN Architecture

Every networking device's functions are categorized into three planes that describe how networking devices work and how network programmability works. SDN split the controller from forwarding devices. To run various types of applications that affords a unified control that contains infrastructure, control, and application planes where control plane and data plane are embedded on the same devices in traditional networks.

A. **Data Plane**

The data plane is the tasks a networking device performs to receive, process, or forward a message (can be data or frame or packet). Data plane is forwarding devices or network infrastructures that include either software or hardware switches and/or other physical devices. SDN uses only programmable switches that are capable to exchange information with the controller using OpenFlow protocol via a secure channel. In the packet-based network OpenFlow, each switch contains flow tables that contain flow entries. Flow entry contains a set of match fields for matching packets, a priority for matching precedence, a set of instructions for actions, and a set of counters to track packets. The header of each flow table entry specifies a flow and an associated action to be taken to an incoming packet matching the respective entry. For all packets, performs header matching and takes specific action based on the matching, then updates the counter. Next, the data plane interfaces and devices was elaborated to give an understanding of how actions are taken on incoming packets.

1. **Southbound interface**

   The southbound interface is the interface through which the controller communicates the forwarding device's data plane. The interface in OpenFlow is a software interface, it's not a physical interface unless noted. The southbound is an OpenFlow protocol specification that includes application programming Interfaces (APIs) with the overall objective of network programmability. API is a method for one application or program to transfer data with another application. The southbound interface allows the controller to program the data plane forwarding tables of the networking infrastructure. Control plane installs flow entry rule (set of instructions) by means of southbound interfaces on forwarding devices. Based on flow rules forwarding devices take actions such as forward packets to

port if match rules available or forward to the controller if no match rules in flow table via packet-in message or modify or drop if expired or no matches. When the controller receives a packet-in message based on an overall view of the network and it replies as a packed-out message via flow-mod then installs flow rules for the packet [59]. Some of the southbound interfaces OpenFlow protocols are OF-config, Open vSwitch Database Management Protocol (OVSDB), OpenFlow Forces, IRIS, and Netconf.

2. **Forwarding Devices**

   The data plane devices are the network infrastructure of forwarding devices. Forwarding devices could be software or hardware switches.

   (a) Software switches

   Software-based SDN forwarding devices implementation is simplest and more portable because the flow tables, flow entries, and match fields involved are easily mapped to general software data structures and different developments behave consistently than hardware. On the other hand, its implementation is slower and less efficient than hardware (no hardware acceleration). As a result, for high-speed network devices such as 100Gbps, only hardware switches are feasible even though processing power and memory size do not issue in software switches.

   (b) Hardware switches

   In performance-sensitive environments such as in data centers and core networks due to hardware, switches operate much faster than software switches it is more applicable. The hardware switches use a specialized hardware accelerator to facilitate inspection of incoming packets and subsequent decisions based on packet matching operation. Hardware switches implemented using content-Addressable Memories (CAMs) and or hardware-based hashing at layer two when there is an exact match such as MAC address while Ternary Content-Addressable Memories (TCAMs) at the layer three forwarding tables associated with complex matching functions used in hardware to check not only for an exact match but also for the third state that uses a mask to treat certain parts of the match field as wild cards. IP destination address matching against networks where the longest prefix match performed uses TCAMs matching. TCAMs matching uses only some header fields of an incoming packet to determine the closest match [57].

B. **Control Plane**

The control plane is decoupled entity from forwarding devices and has an overall sight of the network. The Control plane uses a controller to simplify network management. It is the brain

of SDN architecture and acts as a bond between the forwarding devices and the application plane. It uses the southbound interface to manage flow control in the infrastructure network elements. The control plane consists of an SDN controller and controller interfaces.

1. SDN Controller or Network Operation System

   In the SDN control plane, the controller is the essential component that manages the traffic in core network elements using flows (instruction sets). The controller retains an overall view of the whole network, implements policy decisions, controls all the SDN devices that comprise the network infrastructure, and offers a northbound API for applications (management plane). The policy decisions regarding routing, forwarding, redirecting, load balancing are implemented by both controller and applications. Even though features of every controller might vary, its primary and crucial functionalities are collecting information statistics, topology, notifications, and device management performed similarly through southbound interfaces. Also offers information of network state to the application plane for the development of the application through northbound interfaces [57], [59]. Controllers' architecture could be either a centralized single entity or distributed where each controller communicates for synchronization and to have a global view of the network through East/West interfaces. In Figure 2-3: SDN Architecture elaborates the SDN controller, network infrastructure, Interfaces and APIs, and applications. Some examples of SDN controllers are Open Daylight, ONOS, NOX/POX, RYU, Beacon, floodlight, etc.

2. SDN Controller Interfaces

   The interactions among the SDN planes are performed with Application Programmable Interfaces (APIs) that are used to install configurations data to OpenFlow switches or applications respectively. In Figure 2.3: SDN Architecture different SDN network APIs and their types are discussed as follows.

3. Southbound API

   Southbound API enabler of SDN which provides an OpenFlow protocol for communication of control plane and forwarding devices. The controller uses this API to apply configuration information. Similarly, used to install flow entry rules to forwarding devices providing abstractions function of the network's devices to the control plane. The southbound interfaces challenges of traditional networks vendor-specific network elements, language specifications, and heterogeneity are resolved by providing open and standardized interfaces using OpenFlow protocol.

4. East/West API

The scalability, reliability, and single point of attack challenges of a single centralized controller are solved with a distributed controller. In a distributed controller, each controller possesses its network domain that consists of underlying data plane devices. For a consistent network global view, controllers exchange information with their corresponding domains. Eastbound API is used to exchange information among distributed controllers while westbound APIs enables communication of legacy networks (i.e. Routers, Switches) with controllers.

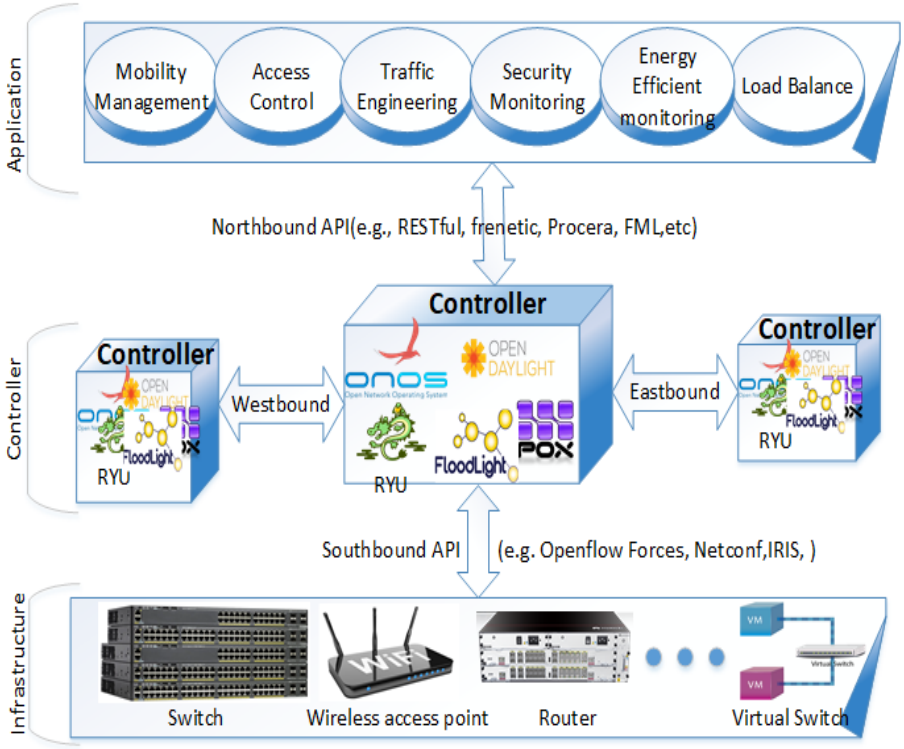


Figure 2.3: SDN Architecture

5. Northbound API

The abundant benefits of SDN are realized through the interaction of the control plane with an application plane. Northbound APIs provide a mutual interface between controller and management plane and play a major role for developers of applications providing the statistics data of underlying devices that makes easy and dynamic SDN control. In some instances, northbound API is a low-level interface, in a case application is aware of individual devices and in other instances, the controller may provide high-level APIs that provide an abstraction of the network itself so the application developer is concerned with the whole network only. The controller informs events that occur in the network to the applications that may pertain to an individual packet that has been received by the controller or some state change in the network topology such as link or node failure. Some of the northbound APIs are RESTful, frenetic,

FML, etc.

C. **Management or Application plane**

The management plane of the SDN network applications is used to orchestrate the whole network over a wide variety of network environments and applications. The orchestration is done specifying the way the operator wishes the network to be run at a high level. SDN applications built on the top of the controller includes routing, policy enforcement, load balancing, traffic engineering, mobility management, security monitoring, energy efficiency, etc. This application can take appropriate actions seamlessly and its implementation uses the SDN management plane.

2.4.2 Network Virtualization and Cloud Computing

Virtualization notion is creating a higher level of abstraction that runs on top of the actual physical infrastructure or devices. The requirements of automation, multitenancy, and multi-pathing are drivers for the network virtualization demand as compute and storage server virtualization growing. Automation refers to single tasks but today's complex network of the large-scale automated arrangement and management, integration, and coordination of automated tasks between complex networks, middleware, and services is labeled as orchestration. Orchestration applications utilize Application Programming Interfaces (APIs).

Network Function Virtualization (NFV) is network architecture based on the virtualization concept to virtualize entire node functions into building blocks that may connect or chain together to create communication services by decoupling network functions. Examples of network functions separated are firewall, Domain Name Service (DNS), Network Address Translation (NAT), and encryption from proprietary hardware devices and moving them on virtual servers. NFV offers a new approach to deploy, manage and design networking infrastructure. The key role of NFV is to control and manage NFV infrastructure computing, storage, and network resources keeping allocation mapping of virtual to physical resources.

Cloud computing is a concept of computer system resource sharing based on demand such as data storage, accessing data, and programs over the internet. The computing power without direct active management by users is possible in cloud services. The cloud relies on resource sharing to achieve coherence of on-demand availability and the pay-as-you-go economy model.

2.4.3 Relations of SDN and NFV

Modern communication networks are highly integrated and coordinated. SDN and NFV are highly complementary but both architectures are independent of each other. NFV applications can be used to control SDN network infrastructures i.e., OpenFlow switches and to support SDN in cloud multi-tenancy requirements. It helps also in virtualizing single SDN controllers for different tenants [54], [60]. NFV operates at the OSI model layer four to layer 7 while SDN operates at layer 2 and layer 3.

2.4.4 SDN for Emerging Technologies

The architectural approach of SDN towards making the network more programmable by decoupling data plane and control plane, simplifies management and service provisioning, increased agility and flexibility with orchestration. These features of SDN helps to solve the challenges of emerging technologies domains.

1. Aspects in 5G networks

   The rapid telecom industry change in the evolvement of cellular networks still needs a more agile, flexible, and speed connection drive to 5G technology. Due to SDN features and capabilities, 5G networks rely on SDN to solve connectivity issues of ubiquitous connections, low latency, and security improving customers' quality of experience across many platforms.

2. Aspects in the Internet of Things (IoT)

   IoT is a network of any physical objects (things), home appliances, sensors, software, and other electronic devices interconnection to exchange data with other devices and systems over the internet or communications networks. The widespread of IoT increases network complexity and heterogeneity that requires flexible network management. SDN network architecture is an adequate mechanism to implement requirements of IoT real-time on forwarding devices with either centralized controller or distributed controller (that supports in optimizing the flow of information and load sharing and load balancing) [59]. The evolution of IoT, tactile internet where the interaction of human beings with electronic devices and real-time communications might be enabled with SDN features.

3. Aspects in Wireless sensor

   Wireless sensor networks deployed in widespread applications from civil to military, and from environmental to health care build large-scale networks with low power devices. One of the challenges of the wireless sensor is that devices have limited power resources

and energy capabilities. The SDN separation of control and data plane moves the most energy-intensive functions (routing) to the controller from underlying devices improves the energy consumption. Also, SDN provides application-specific solutions and high-level programming abstractions, simplicity, and easy configurations that result in the high efficiency of wireless sensor networks. Other issues such as rigidity of policy and security cases were resolved using the SDN management plane.

4. Aspects in industry 4.0

   Industry 4.0 is an intelligent factory of the manufacturing sector, enterprises that produce many customized products and those that produce a few and different of products, to have the opportunity to attain the same production standards and costs. SDN infrastructure improves the overall operations of the network environment, the rate of data transmission, and the quality of services provided in controlling production processes, management and decisions, to optimize production processes. Industry 4.0 integrates cloud, IoT, big data, artificial intelligence to create smart factories, SDN infrastructure feature best suits its requirements [60].

5. Aspects in cloud computing

   The characteristics of SDN architecture bring benefits in cloud computing where dynamic changes and configuration are necessary with its on-demand usage form. SDN provides cost reduction, intelligent global connections, high security, low downtime, the ability to view the entire network to handle problems, and traffic bottlenecks in real-time for cloud network services.

2.5 SD-WAN

SD-WAN is a Software-Defined Networking use case of Wide Area Network (WAN) to enable WAN network agile and flexible, reliable network performance, easy management, configurations, and quick service provisioning. To make clear understandings on SD-WAN in the next subsection, WAN networks and their architectures with techniques and services used in WAN was looked through.

2.5.1 Existing WAN Network

Wide Area Network (WAN) is designed primarily to connect enterprises' branch offices and data centers across long distances. The existing WAN refers to exclusively operators or service

provider services linked via private lines, Multiprotocol Label Switching (MPLS), a virtual private network (VPN), cellular wireless, internet to connect smaller networks in diverse locations into a single, distributed network. The sites could be a few kilometers apart or around the globe in which branch offices connected or even individual remote workers with headquarters or data center to share corporate resources and communications. Figure 2.4: Existing WAN Network architecture illustrates the real scenario of current WAN networks architecture and how it interconnects with branches, data centers, the internet, and clouds. The challenges of the existing WAN network of using MPLS and internet service links are high cost, time to implement new circuits, long time to upgrade the capacity of existing circuits, packets loss, latency, and jitter. These challenges impacted the network performance of WAN networks as networks deployment continued to cope up with traffic demands, emerging applications, and device proliferation [61].

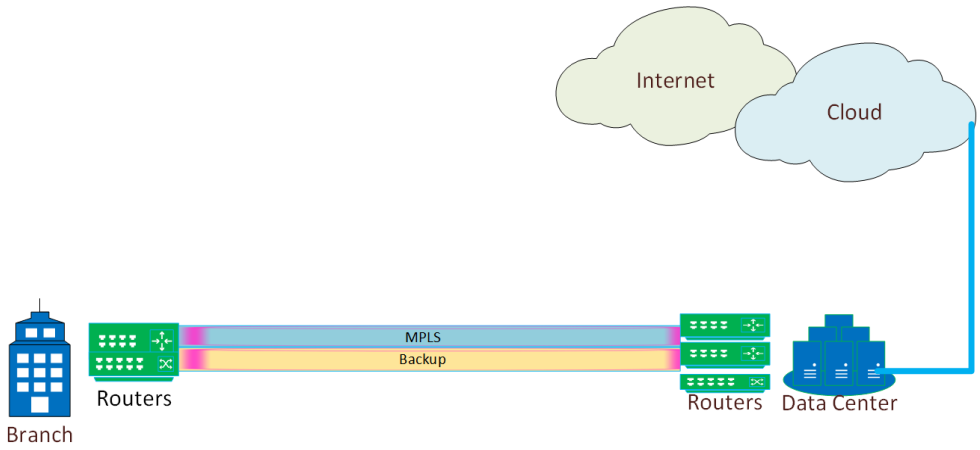


Figure 2.4: Existing WAN Network architecture

2.5.2 Multiprotocol Label Switching (MPLS)

Multiprotocol Label Switching (MPLS) is a technique that delivers anything from IP VPNs and optical services to metro Ethernet services but not a service provided by carriers. Telecom operators build multiprotocol label switching backbones, between layer 2 (data link) and layer 3 (network) headers.

In an old IP network, based on the routing table the routers determine the next hop and forward the packet to it. However, MPLS instead of routing to the next hop it finds the final destination router called label-switching paths (LSPs) that is the pre-determined path to the end router. Then the router applies a label based on that information that will be removed at the final destination router. The router lookup in its MPLS routing table the next hop, with a numeric identifier associated with the LSP [62]–[64].

2.5.3 MPLS Virtual Private Network

To enable users to receive and send data across a public network, the private network extended with the implementation of MPLS virtual private network (MPLS VPN). When MPLS VPN is implemented, constructing a virtual tunnel between endpoints, it performs as users' computing devices were directly connected towards the private networks, while MPLS creates a backbone network to provide VPN services. MPLS VPN contains Provider Edge (PE), Provider (P), Customer Edge (CE), and Customer routers where PE and P are directly connected at layer 3, where in-service providers run MPLS VPN as a service [63], [64].

2.5.4 Why SD-WAN?

The aforementioned challenges of WAN network solutions lead to the emergence of a Software-defined-wide area network (SD-WAN) that leverages key concepts of SDN and NFV to support all connections of WAN edges.

In legacy WAN networks, the deployment of MPLS bandwidth and multipoint VPN has been too expensive and has too long service configurations. SD-WAN addresses this problem allowing elastic traffic demands, dynamic bandwidth allocation, and dynamic path control of business requirements and performance constraints. In the logical separation of SD-WAN, running IP across the Ethernet easily across the MPLS is allowed because of the virtual overlay created by SD-WAN and path selection and routing of traffic performed based on policies defined by the user. The virtual overlay created gathers loss and latency metrics on every connection to the collected metrics with predefined policies to direct traffic flows to another tunnel.

SD-WAN operates across all internet connections like 3G/4G/LTE/5G, xDSL, and cable, private data services such as MPLS and VPN. SD-WAN is also cloud-ready and then public cloud to data centers connectivity is easily performed.

2.5.5 SD-WAN architecture

As different vendors claim, SD-WAN is quickly maturing and widely adopted by enterprises and organizations as a cost-effective way to connect data centers and branches to SaaS and other cloud applications. These are enabled with the architecture of SD-WAN overlay network which creates a virtual network built on top of existing network infrastructure. This approach improves the scalability of the network and supports modularity, multi-tenancy, and network virtualization.

SD-WAN architecture is explained in Figure 2.5 allows service providers and telecom operators

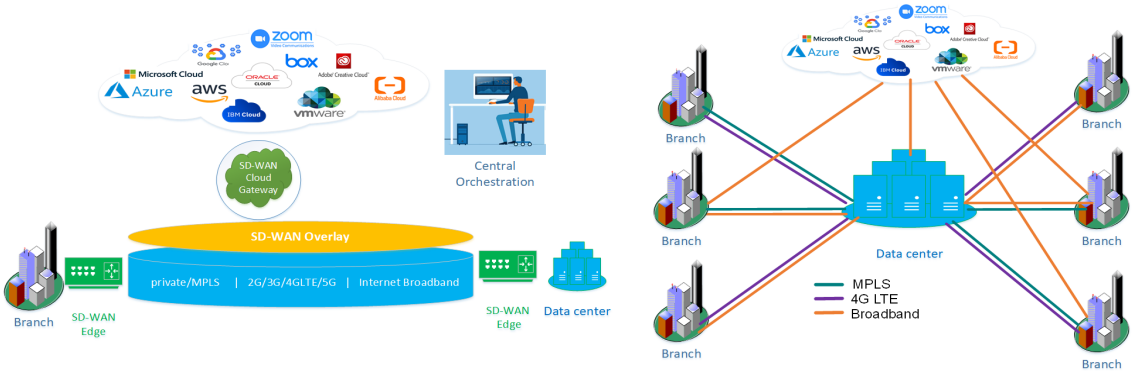


Figure 2.5: SD-WAN Architecture [65]

to leverage secure connection of any mixture of carrier services including MPLS, LTE/5G, and broadband internet services. SD-WAN centralized controller function to intelligently direct and secure traffic across WAN increasing application performance and high-quality user experience delivered.

The cost of back-hauling all traffic in WAN networks in which traffic flows twice that turns around router at headquarter for advanced security inspection no more affects the SD-WAN network as it fully supports applications hosted in on-premise data centers, public or private clouds, and Software as a Service (SaaS) services [65].

The difference between WAN and SD-WAN architecture is that the traditional WAN requires hardware at each end of the network to complete connection whereas the SD-WAN abstracts control of the WAN connection into a software layer, as control and data plane are split into a separate plane. On another hand, each traditional WAN device is configured independently and manually and managed separately, while the SD-WAN controller consolidates and centralizes the configuration and service provisioning and enabled to manage of all WAN endpoints together.

2.5.6 SD-WAN deployment Use cases

In section 2.5.5 SD-WAN architecture components described can be leveraged in several ways in SD-WAN deployment scenario use cases. The deployment scenario could be Hybrid SD-WAN or SD-WAN deployment. Hybrid SD-WAN deployment is the SDN deployment of the SD-WAN overlay on the underlying physical devices of existing legacy WAN networks. SD-WAN is the SD-WAN deployed over the network topology with total migration from legacy WAN to SD-WAN.

Due to the challenges of SD-WAN deployments, the hybrid SD-WAN deployment that integrates the advantages of the existing MPLS and the SD-WAN can be leveraged. There are different use cases for the deployment of SD-WAN as adapted from [66].

1. Service tunneled over the internet and MPLS WANs

   In this use case, the SD-WAN is deployed over the existing internet broadband and MPLS VPN WANs between branches. This use case deployment benefits customers with SD-WAN encryption across the internet services to boost site-to-site bandwidth of MPLS (VPN) and gains higher resilience. The vendors or service providers use at the branches the SD-WAN Edge to deliver the SD-WAN over the existing system. In this use case, the existing and SD-WAN coexist together ensuring the SD-WAN benefits over multiple WANs.

2. Service tunneled over Multiple ISPs

   This use case is the deployment of SD-WAN across several ISPs through multiple WANs such as xDSL-based broadband service, cable internet, dedicated internet access, 3G/4G LTE/5G, and any possible combination. This deployment is conceivable in large SD-WAN service deployment where branches are reached through an internet WAN only. Having multiple ISPs and WANs improves SLA because WAN resilience can be obtained.

3. Services over multiple VNFs via SD-WAN Edge vCPE

   In this use case, the SD-WAN is deployed across several WANs in the branches using SD-WAN edge VNF over SD-WAN edge vCPE. The managed services providers or communications service providers add VNF to deliver virtual network services. The NFV resource orchestrator is required with additional VNFs to SD-WAN edge VNF that ensuring the benefits of virtual network services plus SD-WAN by the VNF on vCPE.

4. Service in the Cloud

   Deploying SD-WAN services across several WANs with Edge and Edge VNF enabled in the cloud environments. In this use case, the SD-WAN service terminates n the physical server or virtual machine (VM) where applications are active (e.g. in a cloud environment) and tunneling happens at the edge which allows establishing secure connectivity.

5. Service interoperates with MPLS VPN

In this scenario, the interconnection between sites is established via SD-WAN Gateway using the internet and sites connected through MPLS VPN. The sites of SD-WAN Edge interconnected with SD-WAN Edge VNF in the public cloud to an SD-WAN gateway that made the scenario easy, economical, and fast connection new sites to the existing network via the internet [66].

2.5.7 Deployed SD-WAN trends

The telecom operators, service providers, and enterprises started deploying SD-WAN at their WAN networks. Several companies deployed SDN in their WAN networks and some of them are briefly described in this section. The global SD-WAN started in Google B4 [67], [68] later upgrade to solve capacity asymmetry in hierarchical topology as in [69] with two-layer hierarchical topology achieving availability of 99.99%. As a result of B4 Google SD-WAN deployment, on average 70% utilization rate over a long period and about 100% of utilization in busiest links and edges gained.

SDN implementation to inter-data center connections such as Facebook expresses backbone [70], [71] for management of bulk data transfers between data centers that are capable of automatically detects paths on the network with available capacity and uses the capacity to transfer the requested data. As a result of the ability to avoid inter-data centers, traffic congestion Facebook was able to increase its network resource utilization by about 90% without packet backlog and automated software repair for core devices failures of 75% achieved [71].

Microsoft SWAN [72] can carry 60% traffic more than MPLS and traffic engineering which is a significant gain and also better load balance and congestion control. In Microsoft, the traditional data centers configuration and unified management experience are obtained from virtual machine managers (VMM) providing virtualized hosts and networks, allocated storage, and library resources. VMM enables users to perform various tasks like load balances, overall view of networks and controllers, virtual network policies define and manages, and directs traffic flows between virtual networks. Moreover, the network controller, RAS gateways, and software load balancing are integrated [73]. Microsoft WAN and SD-WAN for Office 365 as mission-critical Software as a Service (SaaS) minimize latency caused by backhauling traffic through the data center and improve the user experience, frees up various WAN links for other traffic.

KT deployed vCPE with AT&T after AT&T and its rival Verizon deployed vCPE in their network and Open daylight-based T-SDN to create End-to-end for WAN service provisioning and enabled direct management of either the vendor EMS or the actual network element. As a result of T-SDN deployment, the service provisioning time was reduced by 95% [74].

AT&T virtualized 75% of its networks at the end of 2020 including their core network elements running MPLS, and 400-gigabit SDN-enabled optical networks carrying live internet traffic connections under the control of SDN [75] deployed by executing a network transformation initiative that span's from edge to core networks.

ZTE 360 SDN/NFV ZTE commercial deployed also in China Telecom Shanghai's LSN, China Telecom Guangxi's CO re-architecting. China mobile deployed Nokia Nuage SDN product to

expand its public cloud service and Hohhot private Cloud deployed by ZTE 360 SDN/NFV [76], [77].

NTT Communications Corporation (NTT Com) deployed SD-WAN in its service platform that covers more than 190 counties intending to scale operations and bring new services to the market quickly. Similarly, VMware NSX deployed for Tribune media that transferred over 141 applications to an SDN infrastructure over five months. VMware NSX is a virtual networking and security software for the software-defined data center (SDDC) that provides cloud computing that brings agility, flexibility, and security across the environment [73]. Some other SDN deployed companies are Viptela, VeloCloud, Telefonica, Amazon, Verizon, Sprint, etc. According to a survey conducted in [78] only Open Daylight (ODL) controller deployed with more than 100 companies such as Deutsche Telekom, Telefonica, Orange, China Telecom, KT Corporation, China Mobile, Comcast, T-Mobile, TeliaSonera, AT&T, and Globe Telecom.

In the case of the pandemic [16], the author indicated dynamic rises in internet traffic as of lockdowns especially conferencing applications like zoom spike by 700% including YouTube, Netflix, and Amazon services. AT&T's white paper report [75] indicated its SD-WAN deployment in its network-enabled to handle and manage this traffic dynamics without any difficulty.

In nutshell, the SD-WAN deployment meets the requirements of elastic traffic demands, full control over the servers and networks. These come with benefits of cost reduction with the transport independence across MPLS, 3G/4G LTE, broadband internet, and others with increased availability and agility, and improving user experience and efficiency with SaaS and public cloud applications.

2.5.8 SD-WAN Deployment Challenges

The SD-WAN architecture brings tremendous benefits that can handle today's internet traffic demands but without the challenges. The SD-WAN challenge is identifying which underlying service providers are best suitable to a business's locations whether to use a single IP backbone i.e., single autonomous system or multi-ISP strategy. Therefore, the operators should identify the underlying services that best suit their demands. SD-WAN challenges beyond the above mentioned and the technology:

1. Vendor selection

   SD-WAN vendor selection is confusing and the telecom operators or service providers should analyze and document their requirements.

2. Cloud connectivity

The demand for cloud computing in every operator or service provider is critical. The operators require cloud access to public cloud environments. Therefore, evaluating each vendor's strategy for providing the operators' requirements of cloud access need to be analyzed in detail. Different connectivity options might be provided thus, the pros and cons of the connection options must meet the operators' requirements.

3. Management

   The management model options in SD-WAN may vary based on the vendor and service provider requirements. These management options should not be blurred lines and should fit the operators' resources and requirements.

4. Cost reduction

   The cost minimizations in SD-WAN might not always show as quantifiable figures. Detail techno-economic analysis overall the business benefits should be looked at as only the bottom line savings often not quantifiable.

5. Underlay provisioning

   One of the challenges for the operators and service providers in choosing SD-WAN architecture and acquiring the connectivity. The operators should pay attention to the network performance, support and service level agreements, and IP backbones.

   for better SD-WAN deployments that meet the operators' and service providers this thesis work tried to describe under the section 2.8 SDN planning criteria to minimize the challenges and gives insights to the operators.

2.6 ethio telecom WAN

Ethio telecom core networks WAN is based on all IP based of Providers and providers Edges equipment interconnected hierarchically with backbone transport of optical fibers in all its links and MW as a backup redundant link. The IP core network provides various services to the customers connected by the access networks. It constitutes a high capacity of communication facilities that connect primary nodes. The core backbone provides a path for information exchange in different sub-networks.

The edge networks provide information exchanges between core networks and access networks and it's the entry point to the service provider's core network. The access network contains a Radio access network, broadband internet, private MPLS, and others. The core networks interconnected with high-speed interfaces and links of all links 100Gbps bandwidth capacity of

optical transport network (OTN) based structure. The IP core network contains four provider routers (P) and 24 Provider Edges routers (PE) where each P is interconnected to many PE routers. Each PE is connected to the P router directly or via other PE paths equipped with one vendor. The protocols BGP/IGP/OSPF/IS-IS/MPLS are used for the connection within the core network throughout the core establishing, a labeled switch path (LSP).

The network architecture used dual-node Geo-redundancy architecture design for high network availability for every PE node. The network is deployed in 2021 to handle all ethio telecom traffic demands of its various services. The deployment document indicated that SDN-related features are supported.

In this work, the flat topology for PE networks interconnections was considered. All the PE consists of 24 nodes with 47 edges of interconnections. The node(s) or switch(s) terms used in this thesis work represent the PE routers of the IP core networks. The data collected for each PE node consists of the node names, their geographical locations i.e., latitudes and longitudes coordinates, edges, and traffic matrix. All these data and details of the topology are confidential. Hence not included in this document but the output results that may provide basic information included.

2.7 Network performance

Network performance is the evaluation, analysis, and review of collective network statistics, to define the quality of the service offered by the underlying network infrastructure. The network performance evaluation and analysis parameters are commonly used in every network especially in the design and performance characteristics of telecom networks. Parameters of qualitative and quantitative that defines and measures network performance are:

1. Throughput

   Throughput is the average rate of successful data transfer through a communication path that can be affected by bandwidth, packet loss, jitter, and delay.

2. Delay

   Delay is the latency for a bit of data to travel across the network from one network communication endpoint to the other. Delay is usually reported as the maximum and average delay in performance characteristics of the communications network. Delay emerged from:

   (a) Processing delay – is the processing time it takes network devices to process the packet header or requests.

(b) Queuing delay – is the time the packet spends waiting for routing queues until action is taken.

(c) Transmission delay – is the time required to push all the packet's bits onto the link.

(d) Propagation delay – is the time for a signal to propagate through the media across the link.

A minimum level of delay is expected to determine the service performance and delay may increase due to network congestion. In an IP network delays range from a few milliseconds to several hundred milliseconds.

3. Quality of service

   Based on the service requirements, Quality of service (QoS) measures the network performance in telecom services and products. The quality of service is affected by throughput, jitter, packet loss, latency.

4. Network congestion

   When a link or a network node is overloaded i.e., higher data load than its rated capacity, network congestion occurs. The congestion deteriorates the Quality of service of the network affecting queueing delay, packet loss, or the blocking of new connections. Due to networks congested queues become too full, resulting in packets discarded and networks rely on re-transmission.

5. Network resilience Network resilience is the ability of the network to recover from a fault and maintain an acceptable level of service during faults occur and the persistence of normal service operation. It is a capability of the network to withstand and recover from an interruption of service.

Service providers and/or telecom operators evaluate their network performance to deliver the required quality of services and service agreements using these parameters. The next section briefly shows how telecom operators should plan for SDN deployment for their network performance improvements.

2.8 SDN Planning Process

Telecom operators, service providers, and enterprises should be able to identify the problems and/or opportunities they intend to address in SDN deployment. The following lists illustrate the procedures and requirements the operators have to do to plan their SDN deployment [79], [80]. The process diagram is developed as described in Figure [79] in 2.6.

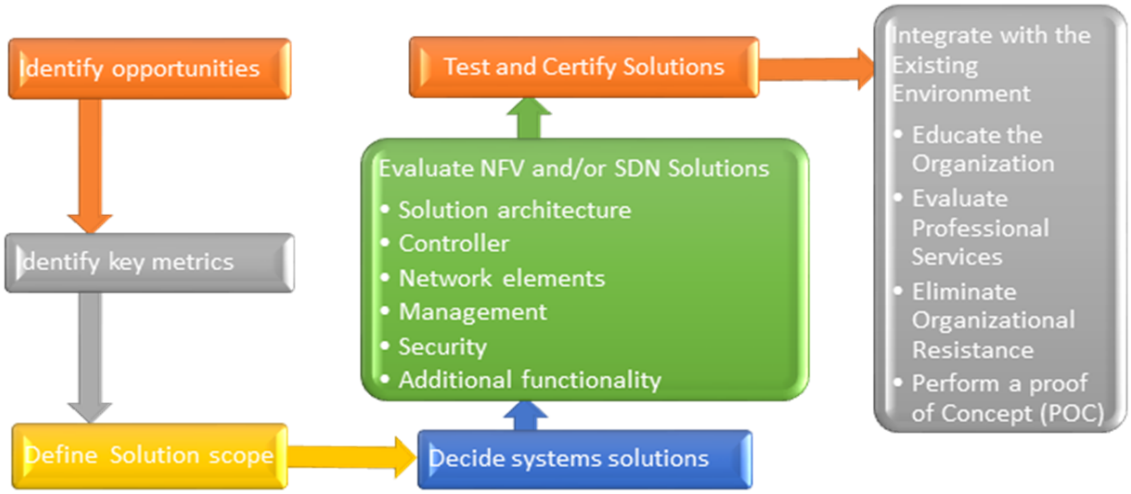


Figure 2.6: SDN Planning process for Telecom Operators

Chapter 3

SD-WAN Controller Placements

In a traditional network where the control and data plane are vertically integrated on the same device, either a device failure or link failure or both cases affects the network performance of both control and data plane. In the network design evaluation, network resilience, availability, and reliability are important factors. On the other hand, in SDN enabled network the network resilience, reliability, and availability of a control plane and data plane is decoupled. However, when the switch is not connected to its control plane, the forwarding device becomes offline subsequently it cannot receives any instructions from the control plane on how to forward the new flows arrived at the switch. In SDN networks different types of failures are listed in the book [81] such as link failure, switches failure, and special case-connectivity loss between switch and controller. This thesis work focuses on the special case failure where the controller placements affect the network performance KPI's such as availability and reliability due to nodes load assignment exceeding the controller's capacity. The overloaded controller may take a longer processing time or may the controller fails or be easily attacked as it might not get enough resources to handle other necessary operations. As a result, the switch cannot reach its controller, the switch discards all arriving packets even if the link is available which might be considered as the switch's failure.

In the SD-WAN network, different path resilience methods are deployed and used. For the reliability of network services and the resilience methods, the controller placements are one of the major factors as it decreases the likelihood of loss connectivity of switch to the controller. Therefore, optimal controller placements are one remedy to the network reliability issues as spotted in different authors' works besides [42], [52], [81]. To grasp understandings on controller placements different related topics how to flow setup processed, how to flow request is estimated and placement metrics elaborated in next section.

3.1 Flow Setup Process

SDN enabled with OpenFlow networks that provide high flexibility using flow-based processing rather than packet-based switching. Controller and forwarding devices communicates through OpenFlow protocol. The controller is responsible for determining forwarding rules in switches. Whenever switches receive new flows, it requires suitable forwarding rules along the desired path to be installed by the controller.

In the OpenFlow network, for flow matching each switch holds a flow table. Packet header information is extracted and matched against the flow entries in the switch when packets arrive at switches. The corresponding action is taken when flow entry matches, but the packet-in message will be forwarded to the controller if no matching entry in switches, then packet-out messages send flow-mod to switches determining the rule to handle the packets [24], [82]. In Figure 3.1: Flow setup process in OpenFlow at switch A and switch D receive packets from the access switches without any flow entry match, both switches send these packets or fraction of these packets to the controller(s) by default through packet-in messages.

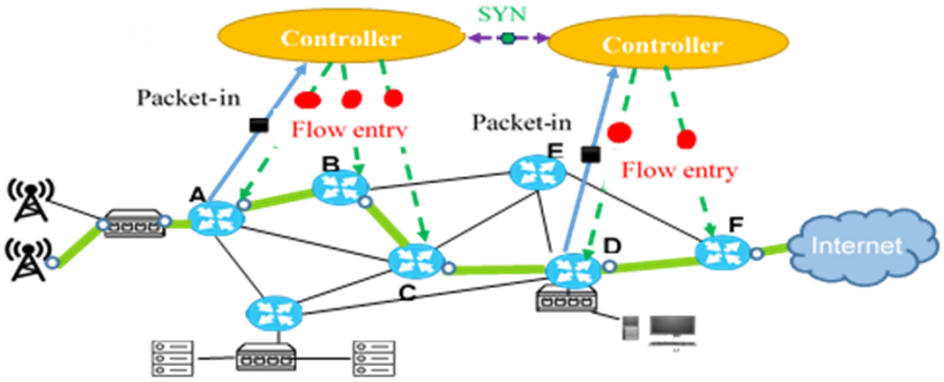


Figure 3.1: Flow setup process in OpenFlow

The controller(s) processes the packet-in message(s) determining the routing paths of these flows and send the corresponding flow entries to the alongside switches A, B, C, D, and F. After flow entries are installed the packets belong to the same flow will go through switches based on routing paths and match rules without requesting the controller(s). In the case of a multi-controller system for consistent operations, synchronize time, and to get a global view of the network, SYN messages are exchanged among the controllers.

Packet_in Message

It is an asynchronous message without a controller soliciting it from a switch. When data arrive at the switch, the switch checks match rules in the flow table and take actions that match the rule. But if no matching rule, it sends a packet_in message to the controller [82] and also can be generated during TTL checking. The packet_in contains only some fraction of the packet

header and a buffer ID if it is configured to buffer packets and switch has sufficient memory unless must send the full packet to controllers as part of the message.

Flow mod Message

It is Flow modification options send by the controller in the installation of flow rules on data paths (OpenFlow instances or devices) option request deleting all flow entries in its flow table to start from the clean state on the switch. It processes buffered packets if not expired automatically after a time.

Packet-out Message

A reply to an OpenFlow request carrying flow entry sends by the controller to switch or to forward packets received through packet_in message. Messages containing a packet from a Packet-In message must be made on the same connection it came in, in sequential order because OpenFlow does not provide an ordering or packet delivery guarantee.

Synchronization Message:

In a multi-controller system that improves network performance and reliability, if one controller or controller connection fails the handover between controllers or to have an overall view of network initiated by themselves. It enables fast recovery from failure and to have an entire network view.

3.1.1 Flow setup time

Flow setup time is the important parameter for performance evaluation in SDN-based networks. The total time to complete the operation of requesting the controller to install appropriate forwarding rules along desired flow path is flow setup time. Flow setup time includes sending of packet-in messages, flow setup service time in a controller, and time of sending packet-out messages [82], [83]. Equation 3.1.1 briefly describes flow setup time.

$$Flow\ setup\ time\ = 2\ *\ (\ propagation\ time) + processing\ time\ in\ controller \tag{3.1.1}$$

Based on the service use case the flow setup time may vary. For instance, for WAN network service the propagation time is very important and dominant as it propagates long distances and controller service time is negligible.

3.2 Controller Selection scheme

Under this section, issues of controller selection techniques whether it is a proactive or reactive controller or passive and active controller, controller selection, and design-related challenges

explored [84]. Controller selection is determined by consumers that need to achieve their objectives and mapping controllers' capability to that objectives. According to the ONF Framework for SDN scope and requirements, the SDN controller must scale to large networks assuring reliability and its distribution components need to be independent of the physical location of the data plane, except there are specific constraints such as latency communication limits. To secure the scalability and reliability challenges, distributed system that spans over a set of heterogeneous processors, internal load balances, clusters, and cascades to gain performance and capacity goals arranged in tree or graph.

In [85] Controller selection process proposed the Analytical Network Process (ANP) that considers qualitative sets of the controller features that influence performance. According the author the network performance results improves the throughput, degrades discovery time, and delay.

In [86] different controller selection criteria were addressed based on classification of deployment architecture, qualitative and quantitative (processing capacity), and use case specifics. Threading techniques used whether single treading or multi-threading can be criteria as multi-threaded controllers are suitable for commercial deployment purposes use cases such as 5G, SDN-WAN, and optical networks. On the other hand, a controller's modularity allows the integration of different applications and functionalities [86]. For advanced level reference, the Internet Engineering Task Force (IETF) defines methodologies for benchmarking controller performance as indicated in [87] and the operators might define their requirements based on their objective and use cases according to standards defined in [87].

3.3 Controller Capacity based Placement

Reasons for taking into account the capacity of controllers during designing a controller placement in SDN deployment even if it is software-based, is that:

i Server capacity is limited: - due to the constraints of the possessor (CPU), memory (RAM and storage), access bandwidth (NIC), and other resources, a commodity server only can manage a limited number of routers or switches. Particularly NIC capacity and accessible bandwidth are also bottlenecks of controllers in SD-WAN [12].

ii Setup processing latency: -The latency of processing messages significantly increases whenever a load of a controller reaches a threshold as indicated in [88]. In such cases, the controller processing latency become non-negligible in total flow setup time.

iii Failure. Reason of a high probability of failure in overloaded controllers is that high likelihood of failures no enough resources to handle different errors, high likelihood of

attack, and even cause failures of another controllers which affects network availability.

The controller load arises from processing Packet-in messages and delivering the events, installing flow entries to the applications, packet-out messages, maintaining the local and an overall networks view.

The node weight of each node may vary seriously but PACKET_IN messages is the greatest load processing part of the entire load [24], [88]. As a huge capacity of PACKET_IN messages reach controller overhead arises because of RAM, CPU and NIC resources bottlenecks. Thus, the time to handle PACKET_IN events decides the networks efficiency and availability. Therefore, for better network performance, controller should provide enough resources for timely processing of the events. Thus, in the trial of flow request estimation of this thesis work, the packet length of PACKET_IN messages send to a controller as the capacity of controllers load was used.

3.3.1 Determining Controller capacity

Controller capacity is an important issue in Software-defined networking that can be defined as number of switches request flows per second the controller can manage from switches (nodes) to the controller in batch arrival process [24], [25]or poison distribution rate.

The traffic load request demand in the core switches from SDN enabled switches to SDN controller is estimated at 4.8 Million flows per second based on concepts of authors [88], [89]. The controller capacity is an important constraint to meet the consumers' requirements and is capable to respond to the request demands from switches. The capacity of the controller is related to hardware performance i.e., RAM, CPU and storage, demands of flow service time (QoS of different traffic demands), and switch buffer size. The controller capacity for this work is estimated at 2.3 Million flows per second based on the product specification of the HP VAN SDN controller [90] and the experiment conducted in [88].

3.3.2 Determining Flow request demand

Whenever packets arrive at the switches, the switches forward the packet if there is a flow match rule in the flow table or sends a fraction or full packet to the controller requesting the controller to install the appropriate flow entry rule as indicated in Figure 3.1: Flow setup process in OpenFlow. As a result of flow setup requests generated by each switch in the network, there will be several packet-in requests originating from switches to controllers.

For network performance analysis and planning this traffic need to be estimated or predicted. In determining flow request demand , limited controller processing capacity and memory, CPU,

bandwidth access, and other resources was assumed.

The bandwidth capacity between the switches and controller is not infinite. As a result, it hinders the rate at which flows are set up, as only hundreds of thousands of flows per second are estimated for core network switches or routers even millions of flows per second for carrier networks [88], [89]. The alternative way for flow estimation is based on a telecom operator that uses bandwidth capacity at link 10Gb/s the request flow estimated in this work. Figure 3.2 briefly shows it. Based on this foundation an average of 213,000 flows per second computed in Equations 3.3.1 and 3.3.2 was estimated.

$$Average\ flows = \frac{Link\ Bandwidth}{Packet-in\ packet\ length * number\ of\ nodes} \tag{3.3.1}$$

$$Average\ flows = \frac{\sum_{i=1}^{n} \lambda_i}{n} \tag{3.3.2}$$

This average flows is estimated for all nodes in the range of 150k-350k flows per second. The Packet_in message packet length was obtained using Mininet emulation depicted in the plot of packet length in Bytes against time sequence in Figure 3.2.

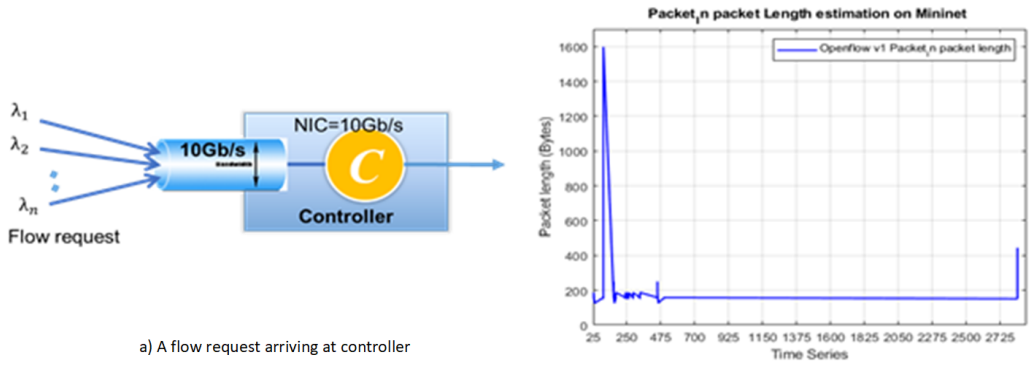


Figure 3.2: Flow Request estimation Model [24]

3.4 Controller Placements Metrics

The scalability and reliability issues of SD-WAN were resolved with multi-controller deployment. The significant challenge is where to place the controllers that provide better network performances. Different metrics affecting network performance of SDN used for effective selection of controller placements. The varieties of the metrics used are based on the goals and network performance requirements (i.e., scalability, delay, reliability, and throughput) of network.

3.4.1 Latency

Latency is the time delay switches sending packet-in messages, the processing time of the messages, and the time of replying flow-mod through packed-out messages as depicted in section 3.1.1. As above-mentioned, this work used propagation latency as it dominates in SD-WAN architecture.

(a) Propagation time

The propagation time is the delay incurred through transmitting packet_in messages across the control paths from switches to controllers and synchronization messages between controllers. It is computed using the speed of optical signal through fiber optics propagation and the shortest path matrix. The link is fiber optics with 67% of the speed of light propagates [91], which is roughly 2 x10^8m/s.

All paths to each node create a distance matrix computed from coordinates by the Harvesine formula which to derive the latency of all shortest paths. The latency of all shortest paths is that of propagation time while other flow setup times are negligible in WAN networks is a suitable metric for controller placement other metrics.

(b) Flow processing time

Flow processing time is the time for controller processing and replying to any flow requests through installing flow rules to determine how packets are handled and establishing policy settings and synchronization flows for global network state updates. Flow processing time is described in detail in section 3.1.1.

The average and maximum propagation latency of node to controller and inter-controller in this research work as latency metrics was used.

3.4.2 Load balance

Load distribution of the flow request among the controllers in a multi-controller system is vital for network resources and improve performance. The overloaded controller introduces high processing time increasing the delay time to reply that degrades network performance. The controller imbalances and fairness index are metrics considered in this work as metrics for load balance.

(i) Fairness index

During controller–switches assignments, balancing load distribution among the controllers improves network resource utilization and performance. Considering different flow re-

quests of switches generates different sizes of traffic flows. The trade-off between the traffic flows and the processing capacity of the controller requires fair load distribution of node requests. Rajendra et al in [13] addressed how measure the fair allocation of resources in traffic flows called Jain's fairness index as a metric of evaluation. This famous index for load balance that defined from Jain's fairness index as described in equation 3.4.1.

$$Fairness\ index = \frac{\left(\sum_{i=1}^{k} C_i\right)^2}{k * \sum_{i=1}^{k} (C_i{}^2)} \tag{3.4.1}$$

, where C_i is the set of all data flows assigned to each controller

The values of the fairness index bounded in (0, 1] range where a value near zero shows the worst fairness that indicates lowest load balanced system. Mininet offers the fairness index as metric of the load balance of controllers' evaluation.

The value of fairness index 1 shows load distribution fairness is 100% that means almost the amount of flows assigned to each controller is matching.

(ii) Controller imbalance

Previous works [9], [12], [31]–[33] and other many works described load imbalance as metrics for load balance computing as the difference of a maximum and a minimum nodes number assignments. In this thesis work the controller imbalance measures how the controller load is discriminately distributed among controllers and defined as 3.4.2 using Jain's fairness index of equation 3.4.1.

$$Controller\ Imbalance = 1 - fairness\ index \tag{3.4.2}$$

If each controller get the identical number of flows, controller imbalance becomes 0 indicating 0% load distribution discrimination. This controller balance is depicted according to equation 3.4.1 that indicates the fair distribution of the load.

Chapter 4

Controller Placement Optimization

The significance of determining the controller places for efficient network performance in different SDN use cases is an interesting area for researchers, telecom operators, service providers, and vendors. Especially the trend in SD-WAN in large-scale network topology indicated that finding the optimal placement of minimum controller number to ensure service quality and availability is paramount.

This section intended to address how determined the controller number is and where to place it optimally. This work customized the POCO framework tool developed by Hock et al. [31], which was later extended in [11], [30] to meet parameters requirements to achieve optimal placements for better network performance. Metrics required in this work are the controller imbalance derived from Jain's fairness index where controller load assigned is not exceeding controller capacity. And average and maximum node to controller latency under the constraints of the controller capacity. Node weights estimation to determine how much flow requests arrive at the controller is performed using Mininet simple topology as described in section 3.3.2.

The minimum controller number determined from controller capacity and flow requests arises from each node. This taking into consideration total flow requests arrive at the controller is not exceeding controllers' capacity. Two scenarios have taken to determine the controller number required. One of the scenarios is assuming 100% controller utilization is available always,while the other assumption is 30 % of backup capacity on each controller reserved for the controller failure case. Moreover, the controller working mode is assumed as a load sharing and standby mode which minimizes extra costs of redundant controller case.

In this chapter the methodologies used, system models proposed, and heuristics methods was discussed.

4.1 System model

This thesis work presented the optimization of controller placements for the minimum number controllers. The controllers' determined should capable to handle the flow requests emanated from each switch of telecom WAN network topology. The system model in Figure 4.1 shows approached used in this work for the final objective of optimum controller placements. In the customization of the POCO framework, various works were performed. Firstly, customization on loading file that contains controller capacity is encoded to enable importing WAN topology graph that has controller capacity because it is one important input parameter as assumed. Secondly, estimating node weights i.e., flow requests from each switches based on link traffic matrix data and packet length of the packet-in message using Mininet emulator. Thirdly, determine the minimum number of controllers that can handle all flow requests from each switch with two different scenarios performed rather than the trial and error method used in earlier works.

The scenarios considered here are when the controller can be utilized maximally without failure issues and the other scenario is when at least one controller fails while controllers are used as load sharing based where the backup capacity of 30% is considered for each controller.

This work solves trial and error methods used in determining controllers' number in network topology and minimizes the controller cost and energy of redundant standby in the failure case

Fourthly, one of the controller placement metrics i.e., controller imbalance determined by how load distribution on each controller is discriminated as discussed in section 3.4.2.

This is proposed because the difference of maximum and the minimum number of assigned switches to the controller is imperceptible in varying switch flow requests arriving at the controller. Lastly, the meta-heuristics used in this work are customized to inherit this capacity impact in their optimization works.

In a nutshell, the optimizing objectives are based on minimizing average and maximum node to controller latency and controller imbalance under controller capacity constraints. Other metrics like maximum and average controller to controller latency used in performance evaluations of controller placements are also considered. For further indication, anyone can look at section 3.4 on controller placements metrics.

4.1.1 Methodology

The controller placements were performed with POCO framework combinatorial-based clustering. The POCO framework performs an exhaustive search to assess the placements of controllers

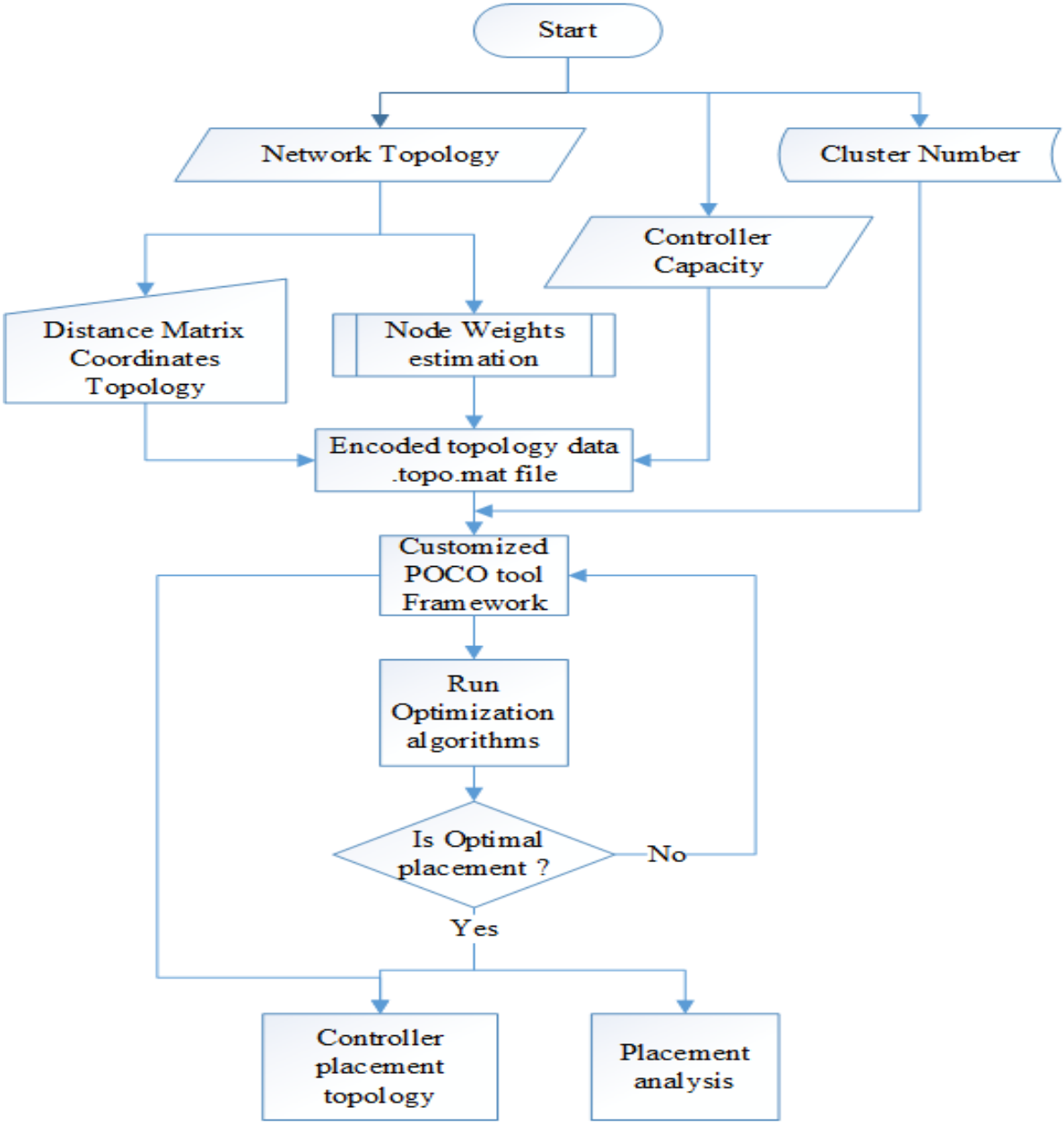


Figure 4.1: Optimization of SDN controller placements system model flow chart

in the whole search space of $\binom{n}{k}$ which requires extremely high memory and time for large-scale networks. To overcome this case the heuristic approaches are required even if the accuracy of the solution decreased [22]. For the allocation of controllers in several ways of placements and positions for k-controllers, the discrete search space contains $\binom{n}{k}$ possible placements. Heuristics may loss accuracy but better computation time especially in large-scale networks solving time is in seconds.

Placements optimization is performed using simulated annealing and K-Medoid-based clustering, methods that are widely used heuristics algorithms to address facility location problems. These algorithms are integrated into the POCO framework tool that works by considering minimum distances between network nodes to the controller. This thesis work focused on con-

troller capacity as the main factor of network performance evaluation input parameter, every node to controller assignments should not exceed this capacity. In another perspective, the load distribution should be fairly distributed for better resource utilization and performance. Therefore, for maximal load-balanced assignments controller Imbalance metrics minimization is important. In this scenario, it becomes multi-objective which is difficult for this framework. Controller imbalance is derived from Jain's fairness index discussed in section 3.4.2, which is directly influenced by the load assigned to the controller that can be limited with the controllers' capacity. Limiting controller imbalance with controller capacity and minimizing average and maximum latency solves the complexity of the tool.

The mathematical formulations, assumptions, and proposed algorithms used in this work are depicted and stated in the next subsection.

4.2 Mathematical formulation

In this work, an SD-WAN of the un-directed graph of G (V, E) where V is the set of switches/routers of n nodes and E is the set of edges of bidirectional interconnections/links between pair of nodes is considered. Graph G (V, E), where V= {1, 2... n} set of nodes and E= {Set of bidirectional Edges} the network contains n nodes that are either switches or routers. The distance matrix D has the shortest path latency between nodes i and j i.e., $d_{i,j}$ normalized with the graph's diameter.

Formalizing the controller placement using Binary Linear Programming (BLP) and defining the Binary decision variables [34], [92] as equations 4.2.1 and 4.2.2 that determine solutions of the model.

$$c_i = \begin{cases} 1, & if\ node\ i \epsilon V\ location\ is\ selected \\ & for\ controller\ placement \\ 0, & Otherwise \end{cases} \tag{4.2.1}$$

$$s_{ij} = \begin{cases} 1,\ if\ node\ j \epsilon V\ is\ controlled\ by \\ controller\ placed\ at\ node\ i \epsilon V \\ 0,\ Otherwise \end{cases} \tag{4.2.2}$$

Equation 4.2.1 determines whether the node is selected as the candidate for controller while Equation 4.2.2 defines the node to controller association sets.

4.2.1 Objective functions

The objective function in the optimum controller placement has several competing objectives to meet the QoS requirements. Minimizing the overall number of controllers that can minimize

the costs in capital investments [34] and operational costs including the energy consumption [36], [38], [40] and maximizes the controller utilization was observed. [92], [93]. In this work minimum number of controllers is determined from controller capacity with maximum controller utilization. But if the operators are interested in SDN deployment to have a backup capacity 30% advance capacity is considered while the controllers work as standby and load sharing principle, minimizing the overall number of controllers during the deployment. The set of objective functions is known and should be minimized means that no other placement is in a search space i.e., determines the Pareto optimal placements was supposed. The goal of this optimization work is to determine the possible locations of k-controllers finding a Pareto optimal set of the whole search space [22]. Pareto frontier is the set of all Pareto optimal solutions. This work focuses on minimizing latency of node to the controller and minimizing controller Imbalance. The optimization work should meet the objective function of maximum load balance and minimum latency based on competing objectives' equation metrics while analyzing trade-off feasibility.

Minimum controller number (k) determined at least to accommodate flow request (R_j) arrived at the controller of capacity Q.

$$k = min\left(\sum_{i\epsilon V} c_i\right) = ceil\left(min\left(\frac{\sum_j R_j\,(1+\alpha)}{Q}\right)\right) \tag{4.2.3}$$

Two scenarios of capacity with backup capacity ($\alpha = 0.3$) and without backup capacity ($\alpha = 0$) in equation 4.2.3 to determine the minimum number of the controller was considered.

Having a combination placement of $P\epsilon 2^V$ and a distance matrix D of shortest path latency between nodes i and j i.e., $d_{i,j}$

$$\text{avg latency} = \frac{1}{|V|}\sum_{v\epsilon V}\left(\min_{p\epsilon P}\ d_{v,p}\right) \tag{4.2.4}$$

The average latency node to controller defined in equation 4.2.6 and controller Imbalance for P placements of C_i, total flow requests assigned to controller i defined in equation 4.2.6. Controller Imbalance measures how the flow requests from node to controller assignment load are discriminated in all P placements. It is determined from Jain's fairness index as shown in equation 4.2.5.

$$Fairness\ \text{index} = \max\left(\frac{\left(\sum_{i=1}^{k} C_i\right)^2}{k*\sum_{i=1}^{k}\left(C_i^{\,2}\right)}\right) \tag{4.2.5}$$

Where fairness Index is discussed in section 3.4.2.

$$\text{coontrollerImbalance} = \min\left(1-\left(\frac{\left(\sum_{i=1}^{k} C_i\right)^2}{k*\sum_{i=1}^{k}\left(C_i^{\,2}\right)}\right)\right) \tag{4.2.6}$$

Even though the optimization criteria are based on average latency and controller imbalance furthermore parameters such as maximum latency of node to the controller for worst-case analysis, and average and maximum controller to controller latency are illustrated in performance evaluation.

4.2.2 Constraints

To increase the ability of the network to react to events quickly that lead to robust topology, even when failures occur, constraints that determine the performance of the SDN networks was considered. These constraints are controller capacity, candidate controller places at nodes, node weights, latency, and controller Imbalance.

1. At least a number of network controllers for SD-WAN should be able to handle all requests;
$$\sum_{i \epsilon V} c_i \ \geq \ \frac{\sum_{j=1}^{n} R_j}{Q} \tag{4.2.7}$$
2. All nodes/switches/forwarding devices should be assigned to controllers while considering:
   (a) Each node is controlled by one and only one controller
   $$\sum_{i \epsilon V} s_{ij} = 1, \ \forall \ j \epsilon \ V \tag{4.2.8}$$
   (b) A node is controlled by an existing controller
   $$s_{ij} \leq \ c_i, \ \ \forall \ i, j \epsilon \ V \tag{4.2.9}$$
   (c) Nodes are always assigned to controllers placed on them
   $$c_i \leq \ s_{ii}, \ \ \forall i \ \epsilon \ V \tag{4.2.10}$$
3. Fair load distribution among all controllers

For load distribution among all controllers the capacity Q_i and the load of the switches as the request demand of the switches/nodesR_i arrive at controller;

i. The load assigned to controller cannot exceed controller capacity Q_i where α is backup capacity.
$$\sum_{j=1}^{n} R_j s_{ij} \leq (1-\alpha) Q_i c_i, \ \forall \ i \epsilon \ V \tag{4.2.11}$$
ii. At least the capacity of the controller should not exceed
$$\sum_{i \epsilon V} \ R_{ij} \ \leq Q_i, \tag{4.2.12}$$

4. Propagation latency should be bounded to service requirements
5. Decision variables are only binary values

4.2.3 Assumptions

The optimum controller placements, addressed in different authors and research works considered different assumptions. In this work, different perspectives was assumed as listed below:

(i) Each node of the switches is the candidate of controllers' placement as it simplifies the deployment and infrastructure costs.

(ii) The controller manages k switches each switch has at least a new arrival flow. When the new flow arrives, switches will generate a "Packet-in" message to the controller, then the controller will send the flow entry to the corresponding core switches.

(iii) The processing time of flow requests is equal to propagation time in WAN.

(iv) The capacity of the controller is assumed to be infinite if not specified.

In a flow-based OpenFlow network, only the first packet of flow needs to be sent to the controller. Therefore, only the new flow set-up requests [24] was focused on.

4.3 Optimization algorithms

In addition to customization of the POCO, the controller placements investigation is main contribution of the research effort. This analysis is based on average latency, maximum latency (worst case in other works), and controller imbalance derived from Jain's fairness index. Based on these parameters the trade-off between average latency and/or and controller imbalance was analyzed with the impact of controller capacity and switch flow request. The influence of controller imbalance on load distribution of the controllers and controller capacity impact on average and maximum latency is investigated. Controller placements with POCO with and without controller capacity are compared with heuristic algorithms integrated into the POCO framework developed in [31] and extended in [11]. Heuristic algorithms parameters were investigated to have understanding of parameter values for various use cases and necessities. The heuristic algorithms used on the ethio telecom PE WAN network topology of this work are briefly described in the next subsection. Controller capacity-based placements evaluation is important for telecom operators, service providers, and ISPs as there is no unlimited capacity of controllers as depicted in the previous section.

4.3.1 Pareto Simulated Annealing

Simulated annealing is an unsupervised learning algorithm, which uses probabilistic technique for approximating the global optimum of an objective function. It exploits the analogy between the annealing process and combinatorial optimization problems where molecules variables in the data structure and energy function is the objective function. The temperature is the real value scalar that controls the degree of randomness of the search where high temperature behaves like random search and low temperature like greedy local search [94]. For this work, simulated

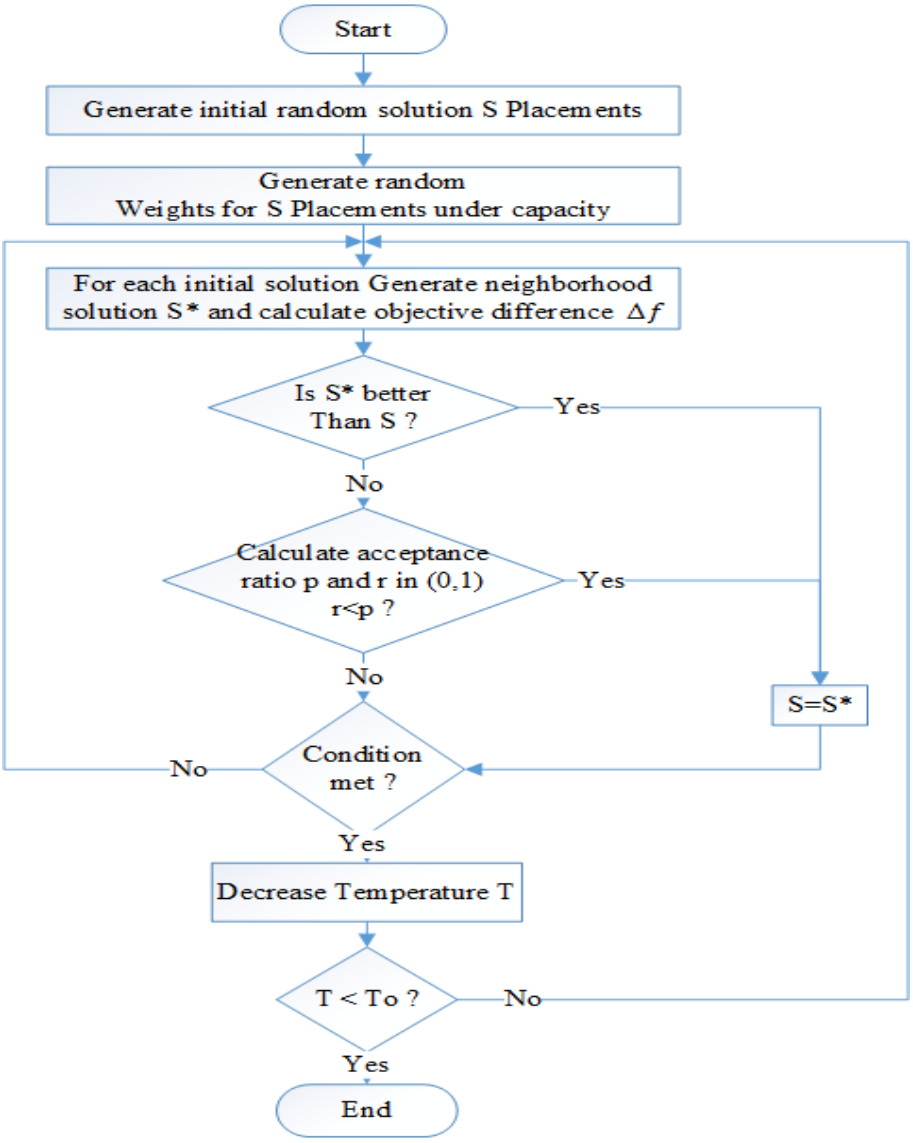


Figure 4.2: Flow Chart Simulated Annealing

annealing implemented clustering the topology taking the internal criterion as minimizing a total within the cluster distance.

In this thesis, the Simulated Annealing system of the structure developed in [11] was used according to the flow chart in Figure 4.2 frameworks. The input parameters to PSA heuristics

are from two sides. One side is network topology graph G (V, E) and the k minimum number of controllers that capable to handle flow requests from n nodes, and the other side are parameters of PSA such as the number of iterations per temperature level m, initial temperature T_0, annealing schedule, and a number of placements to evaluate during each iterations s. initially, a set of S of s random placements of k controller is generated.

The PSA works as, every placements assigned with random weights and then check whether the assigned node weights exceed the controller capacity. In each procedure, the visited M Pareto frontier placements are updated. The temperature T declines using the algorithm, starting with initial temperature T_0 by decaying factor after every m iterations up to T value is less than 1.

Based on the nature of the problem to be optimized, the state of initial energy S random placements of K controllers was assumed. The energy state of the placements system fluctuates around the average energy linked to the temperature. When the equilibrium is reached, the temperature is slightly reduced and again search thermal equilibrium with new temperature. The temperature is reduced until the temperature is less than 1. High initial temperature 50 is selected to reach quickly at the thermal equilibrium and checked trade-off quality of the result and computation time.

4.3.2 K-Mediod Algorithm

The heuristic algorithm proposed to approximate the Pareto frontier sets from two objectives (average latency and controller imbalance). For the clustering-based approach of K-Medoid considering the only node to the controller, latency is not sufficient in a limited capacity of controllers. As a result, the K-Medoid clustering algorithm was upgraded with a controller capacity

bound. Controller capacity restricts the flow requests amount assigned to the controller with controller Imbalance and latency constraints. The controller capacity has a direct effect on the value range of controller imbalance obtained from Jain's fairness index as termed in 3.4.2 section. Flow chart in Figure 4.3 adopted from [30], demonstrates the capacitated K-Medoid method. The algorithm inputs distance Matrix D, network size n, controller number k determined from flow request traffic tm, controller capacity Q, and imbalance constraint introduced via ρ. Instead of node assignment is nearest to controller, minimal latency and maximal fairness index assignment are determined upper bounded by capacity.

The parameter ρ iteratively increased resulting in maximum controller imbalance that satisfies capacity constraints. Normalized ρ controller imbalance to bring the impact of controller capacity to the clustering method used in the K-Medoid algorithm. Controller Imbalance of

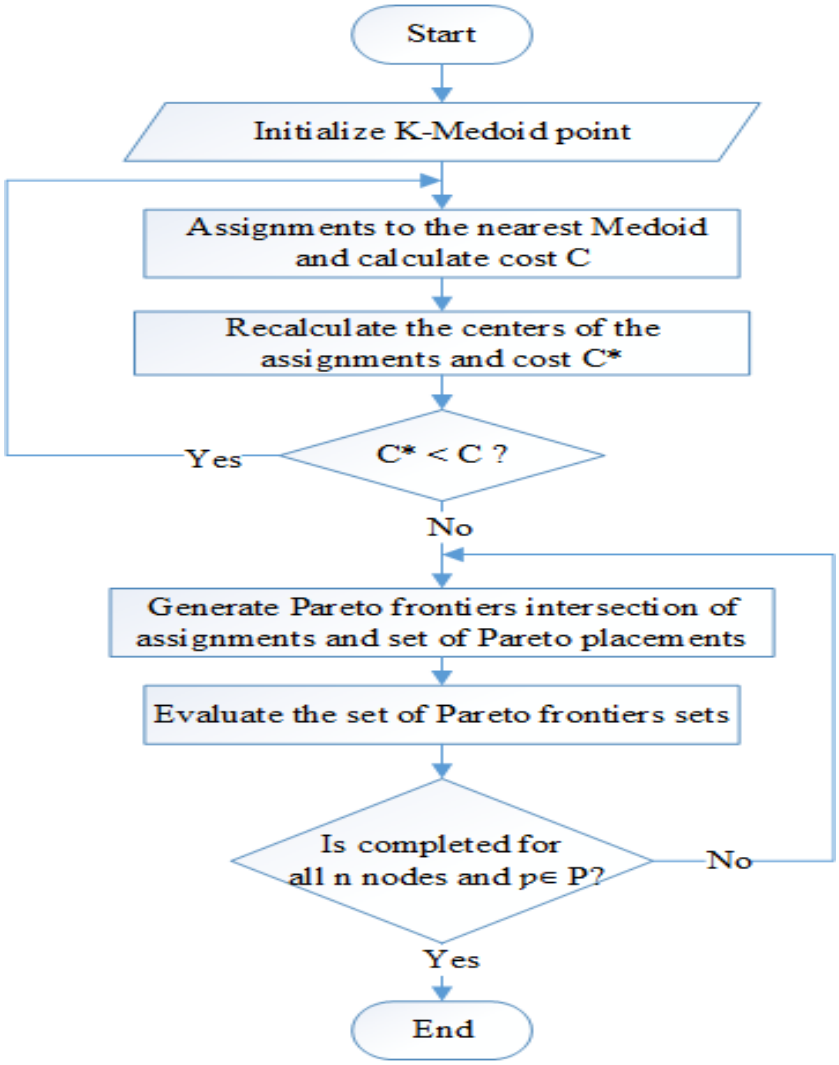


Figure 4.3: Capacitated K-Medoid flow chart

load assigned to each controller not exceeding controller capacity filtered and defined as rhoImbalance that used to normalize ρ as indicated in equation 4.3.1. rhoImbalance is the capacity filtered controller Imbalance placements as indicated in equation 4.4.1.

$$\rho = Imbalance * \frac{200}{k} \tag{4.3.1}$$

A minimum controller Imbalance only may not guarantee the capacity-limited assignment requirements. Based on the capacity requirements load balance assignments matching a cost minimal constructed to enforce the capacity limit of controllers replicating each controller $(\lceil \frac{n}{k} \rceil + \rho)$ times as adapted from [30] in the semantic of controller load discrimination.

The flow chart indicated in the first loop is identical with second loop that is calculating assignments given centers and calculating centers given clusters until the latency convergence is reached. Lastly, results of the cluster C and the assignments of each node to the centers evaluated with C* and its assignments.

For every value of ρ, algorithm calculates placement that minimizes average node to controller

latency respecting imbalances and a capacity constraint imposed by ρ. The goal of this work is to use the customized algorithm in [30] and providing insights and analysis of metrics trade-off comparatively the placements obtained without using heuristics and with both heuristics discussed in this section.

Pareto frontiers obtained from determining Pareto optimal placements assessing the elements in all sets of S random placements meet capacity requirements with respect to the latency (node to controller) and controller imbalance imposed by ρ.

4.3.3 Node request estimation

Simple topology is created on Mininet virtual network and packet length of packet_in message captured with Wireshark. POX controller is used to capturing request flows and 10TB of data sent from host 1 to host 2 for the emulation. This discussion is in section 3.3.2. Determining Flow request demand.

4.4 Evaluation Methods

The models were implemented with the aim of determining the optimum controller placements in which nodes assignment do not exceed controller capacity with the highest fair load distribution and minimum average node to controller latency. In the mathematical model, the minimum number of controllers is determined based on the controller capacity and flow request rise from each switch. While determining controller number two scenarios were considered . The first scenario is 100% controller utilization without failure. This may lead to a longer processing time in controllers and also reliability issues that affect the performance of the controller if its 100% is utilized. As a result, optimizing this case with fair load distribution and latency was considered. Another scenario is if the controller failure occurs, a reserved capacity of 30% is assumed as a backup capacity for the controller to increase reliability and performance. The controllers should be configured as standby load sharing based assumed during deployment for the significance of cost minimization. This is if any of the controllers or controllers' links failures occurred, it performs as load sharing and standby for the failed controller. The performance evaluation was performed for both scenarios that assist Internet service providers (ISP), vendors, and other telecom operators in need of deploying SDN networks for their networks especially WAN networks.

The performance evaluation is based on metrics described under section 3.4. The performance evaluation consists of different placement scenarios without controller capacity, with controller capacity and backup capacity consideration for network availability. The load imbalance role

in controller load distribution and impact of controller capacity consideration and node to controller latency also looked through as indicated in Figure 5.5 and Figure 5.6 shown briefly . The analysis of a set of parameters and the Pareto frontiers achieved and metrics trade-offs for pair of the three below listed cases are performed and their comparatives discussed. Every evaluation is carried out on the real network topology of the ethio telecom WAN IP core network.

(i) Placement without optimizations

In this analysis case, the impact of considering controller capacity in controller placement in a generic POCO framework without using heuristics methods that places randomly guessing placements used as a baseline was investigated. This comparison explores the consequences of controller capacity considerations in controller placements.

(ii) Placements optimization with Simulated Annealing

In this case, simulated annealing heuristics algorithms to comparatively analyze the controller placements in the aforementioned case was considered. Controller capacity case and without controller capacity is considered and analyzed.

(iii) Placements optimization using K-Medoid cases.

K-Medoid clustering heuristics are used to analyze the controller placements optimally based on controller capacity constraints and without it with uniform metrics used for other evaluation methods.

Finally, the three cases summarized and analyzed their Pareto placements obtained in each case comparatively. The suggestion is, the SDN use case deployments based on the investigation observed. The controller Imbalance of each placement of load assignments that meets the capacity requirement is filtered out based on equation 4.4.1.

$$\begin{cases} 1, & where\ C_i - Q > 0 \\ controllerImbalance, & otherwise \end{cases} \tag{4.4.1}$$

$$Where\&, C_i \ \ load\ of\ controller$$
$$\&Q\ controller\ Capacity$$

Chapter 5

Performance Evaluation

In this chapter, the evaluation of the proposed system model and detail results discussion addressed. The evaluation setup and performance evaluation of the approach finding optimal solutions of Pareto optimal placements stated. In the result analysis, the generic POCO approach and heuristics methods investigated comparatively in detail. In each comparative cases, the result analysis considered two approaches. the first approach is without controller capacity while the other approach is based controller capacity as a constraint to limit maximum flow request arrive at controller.

To reveal the impacts of the controller capacity on evaluation metrics assessment investigated. Before the evaluation of optimization results is discussed, the controller capacity impact in minimizing controller imbalance is observed and discussed. Also, the controller imbalance and latency impacts on controller load assignments are investigated. In this investigation load distribution among the controllers was measured using the Jain's fairness index that resulted in controller imbalance. The node to controller average latency used in the thesis is, the node to controller average distance of graph topology normalized to diameter. Similarly, the maximum latency is the node to controller maximum distance of graph topology normalized to diameter of the topology.

In this work, the goal of optimizing controller placement with only minimizing the latency might also leads to high load assignment to controller as investigated in shown in the graph latency impact on load distribution. As a result, a controller might be assigned over its capacity which results in high processing time due to overloaded requests and low network availability with poor network performance. At last, not only do controller imbalances and global latency minimization guarantee optimal controllers placements but also load assigned to the controller should not exceed its capacity.

5.1 Evaluation Setup

In this section, how the optimization platform is established to meet the objective discussed under different sub-subsections.

5.1.1 Topology

The research work uses the ethio telecom IP Core WAN topology data. Based on the topology data as input and prepared in graphical descriptions of network topologies as .topo.mat file. The .topo. mat file is shown on Matlab graphically as shown in Figure 5.1.

The graphical description provides sufficient and necessary information to build up testbed networks to emulate real-world topologies.

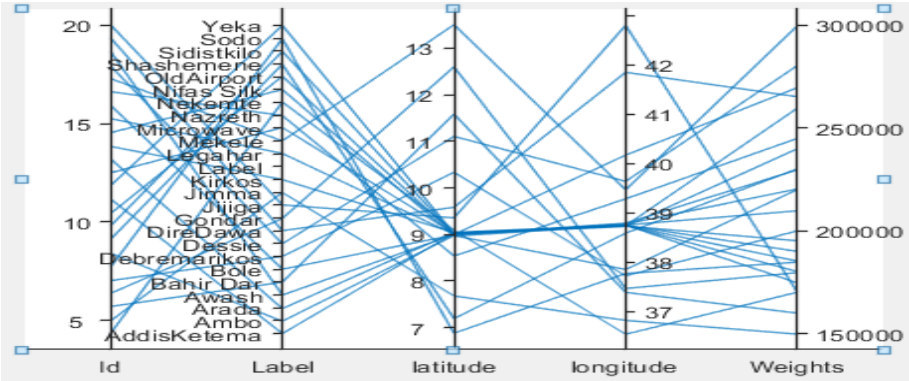


Figure 5.1: Topology Data of IP Core WAN Network

The network topology contain twenty-four provider's edges (PE) and 47 connected edges in flat topology. Fig below shows the google earth network topology map with all connected links to nodes. Google earth network topology is created with *kml* script as indicated in Figure 5.2.

5.1.2 Emulation Setup for flow request estimation

Mininet version 2.3.0 installed on Ubuntu 20.04.2 LTS on VMware 15.05 pro virtual network.

Simple default Mininet topology is used to estimate flow requests using the POX controller.

5.1.3 POCO Framework Matlab based tool

POCO framework performs combinatorically i.e. Placements of all $\binom{n}{k}$ possible with an exhaustive computation. It is suitable tool for small and medium-sized networks. In large-sized networks, the size of search space rises very quickly and high execution time. Different heuristic approaches are suggested in the case of large-scale network topology to fully evaluate in a

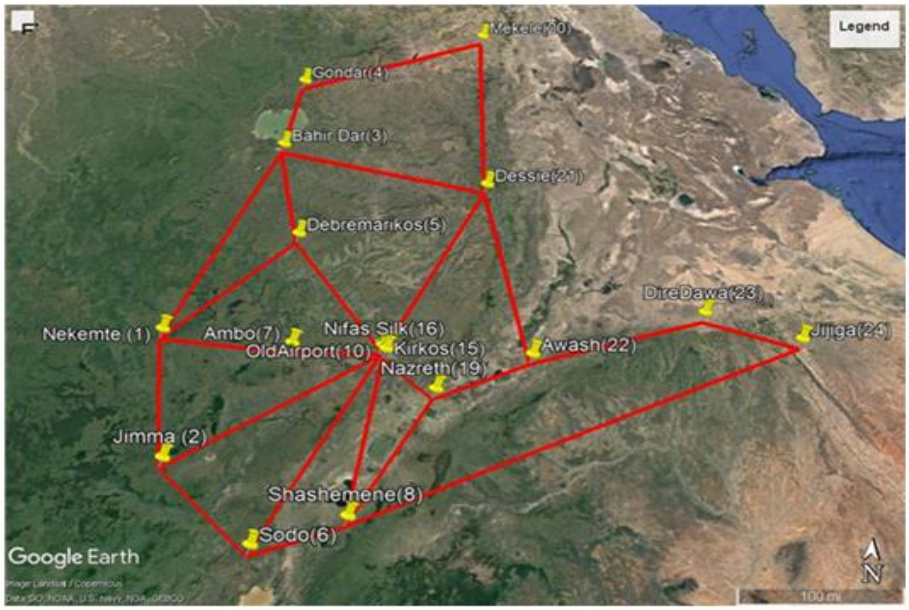


Figure 5.2: Topology of IP Core WAN network on google earth

practical time frame [11].

All algorithms and POCO framework run on Matlab 2020b version on Core i7 Intel Processor, 8GB RAM, 500GB Storage on Windows 10 Operating System, 64-bit Laptop installed with simulation tools indicated in Figure 5.3.

Figure 5.3: Simulation Tools

5.2 Result Discussion and Analysis

In this section, the result of the performance evaluation setup and method aforesaid is discussed. In this result discussion and analysis, It is presented in two sub-sections. The first part is the impact of controller capacity on controller placement metrics, where the various values of the metrics with and without considering controller capacity is discussed. The second part is the finding techniques of optimum controller placements with heuristics methods integrated with

generic POCO framework tool with and without controller capacity is analyzed in detail. In the second part, two scenarios of deployment cases of failure-free and failure cases is considered and analyzed.

5.2.1 Controller capacity effect analysis on placement metrics

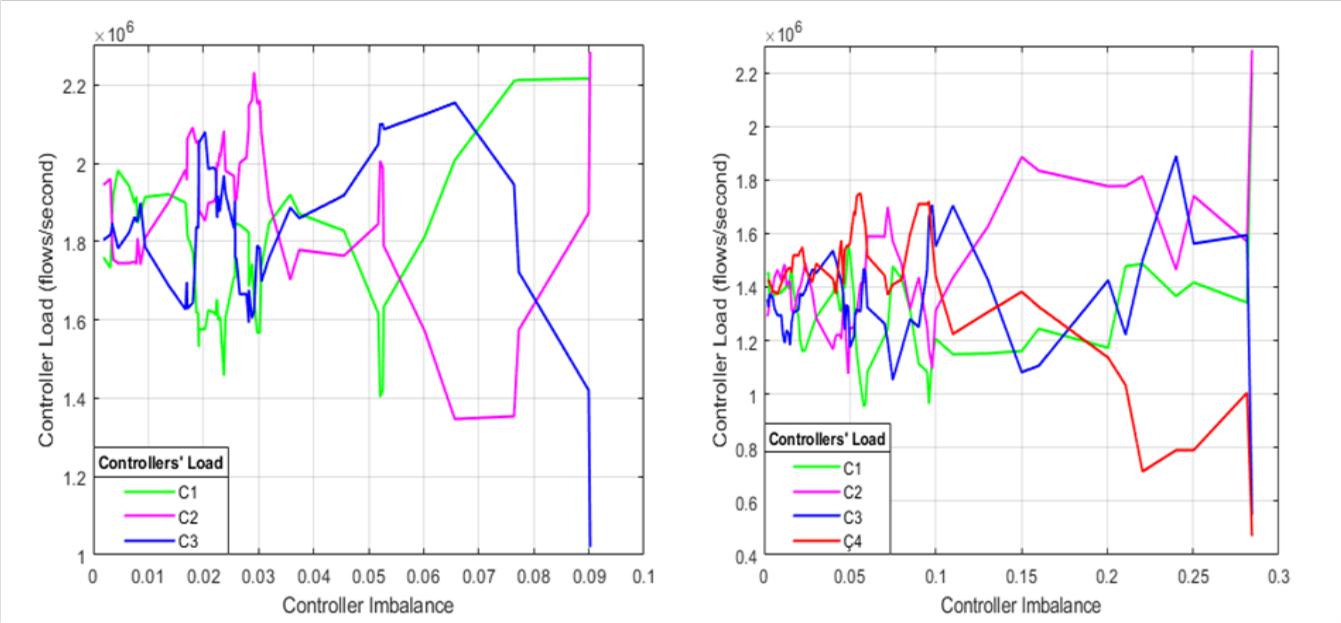
In this sub- section the controller capacity impacts on the controller placement metrics investigation and observation is carried out. Specifically, how the controller imbalance values are affected with the controller capacity consideration in controller placement and how the load distribution is affected with the controller imbalance values is investigated. Similarly, the controller capacity impact on latency and the Load distribution vs latency trade-off is observed.

(A) Controller Imbalance impact on Controller load distribution

The Controller Load on each controllers required to accommodate the flow request from nodes shown in the Y-axis while the Controller Imbalance in the x-axis shown in figure 5.4 that intended to show how the load is distributed along the controller imbalance. The first figure shows when the controller placement in failure free case 5.4(a) whereas the right side 5.4(b) is the case of backup capacity is considered for failure case. As shown in the legend of the graph, minimum number of controllers is computed by dividing total flow request to controller capacity and adding backup capacity if it is likely to be considered according to equation 4.2.3. The controller capacity impact on controller imbalance is that it filters the lower imbalances that meet capacity requirements. The controller capacity caused by the capacity bound implicitly limits the range of the values controller imbalance can attain as indicated in Figure 5.4. In the scenario of failure free (no backup capacity) the minimum number of controller is three as indicated in Figure 5.4(a) showing load distribution among the three controllers and at controller imbalance of 0.09 or 9% the load on C2 (Controller 2) surpass the controller capacity limit i.e., 2.3 Mflows/s while C3 (Controller 3) goes below 1 Mflows/s. Likewise in the case of backup capacity consideration shown in 5.4(b) the minimum number of controllers required determined is four (4) and the load distribution among the controllers shown. As indicated in (b) of the 5.4 the C2 (Controller 2) load assignment surpass the controller capacity limit at about 0.28 (28%) while the other C4(Controller 4) goes below 0.5 M flows/s. The fair distribution of load among controllers increased as depicted in Figure 5.4 in lower values of controller imbalance in both scenarios. This could proves also that the controller imbalance derived from Jain's fairness index is a good measurement for load balance.

As can be seen from the graph load distribution among the controllers, at lower controller imbalance highly fairly distributed which increases the controller utilization efficiency . As

a result better performance and resource utilization were obtained with minimum controller number with controller imbalance minimization.



a) No backup capacity case k=3

b) Backup capacity case k=4

Figure 5.4: Controller Imbalance effect on Load distribution

As indicated in Figure 5.4: Controller Imbalance effect on Load distribution, minimizing the load imbalance can avoid controllers overload assignments. This approach minimizes the number of controllers required for the network topology through load balance of distributing the load fairly to each controller.

(B) Controller Capacity Impact on Latency

This part addressed how the average latency and maximum latency node to controller is affected in controller placement with the constraint of controller capacity. In the graph 5.5 the y-axis shows the change of latency while the x-axis controller Imbalance. The controller Imbalance range that satisfy the controller capacity limit is determined in previous section. As observed in the controller capacity impact on latency graph shown in 5.5 at lower controller Imbalance less than 0.09 (9%) both average latency and maximum latency never affected. This shows that for load distribution on each controller never surpassed controller capacity, no latency change observed and for any load distribution surpassing controller capacity the change of latency observed. The red dotted distribution shows the average latency change while the blue indicates the maximum latency change. The positive change shows the improvement in latency mean that the latency value decreased while the negative values indicates the latency value increased.

Controller capacity consideration will improve the maximum latency node to the controller with about in average 8.5 % or 9.32 milliseconds in a total maximum latency of all nodes to controller. However, the average latency changes distribution is both positive and negative and improves very slightly with 0.33% or 0.36 milliseconds in average in case k is 3 as shown in Figure 5.7 in total nodes to controller latency. These are because of reassignment of nodes (adding or removing of nodes) in the case of overloaded and under-loaded controllers. The difference of latency change distribution indicates that lower latency is preferred in node assignments (it is clear average latency has a lower value than maximum). No influence investigated of the load assignment within the controller capacity constraint on both the average and maximum controller to controller latency. This indicated that better latency can be obtained using controller capacity.

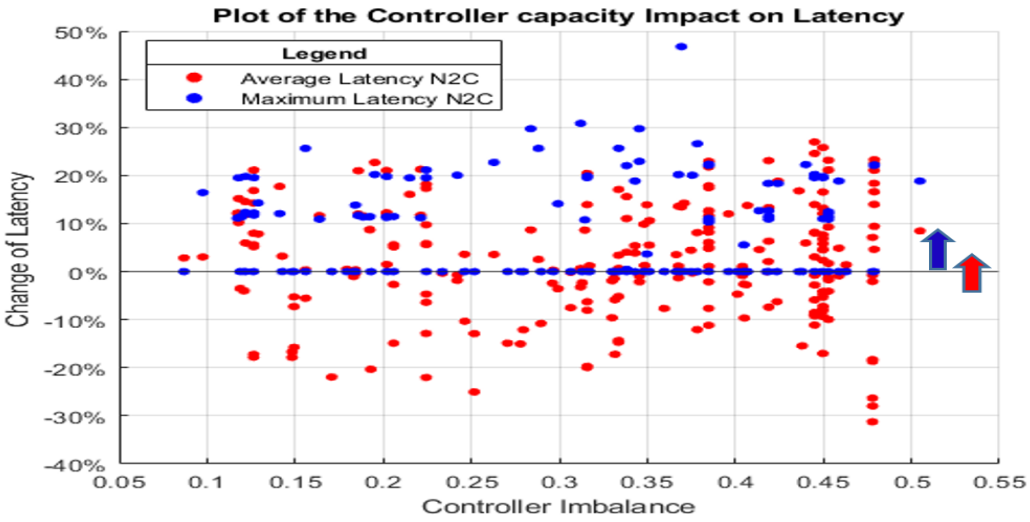


Figure 5.5: Controller Imbalance Versus change of latency

Latency impact on load distribution

This section of the work wanted to address whether the latency could affect the load distribution among the controllers. The x-axis shows the controller load assigned to each controller while the y-axis shows the latency values each load is assigned to controller. As observed in Figure 5.6, at very low and very high average and maximum latency of a node to a controller, imbalances increase i.e., the controller capacity limitation might not meet. This investigation leads us to conclude that only minimization of latency might not result in the optimal placements for controllers under any evaluation circumstances as discussed in Evaluation Methods. This thesis investigated that focusing only on latency minimization might overload some controller(s) where nodes are densely populated like a big city and other nodes are underutilized especially where nodes are sparsely distributed. As a result load imbalances increases which negatively affect the performance and reliability of the network. In both maximum and average node to controller

latency, similar conclusions can be derived. The network topology used in this case study is a good example of such topologies where nodes in Addis Ababa City are within hundreds of meters to a few kilometers.

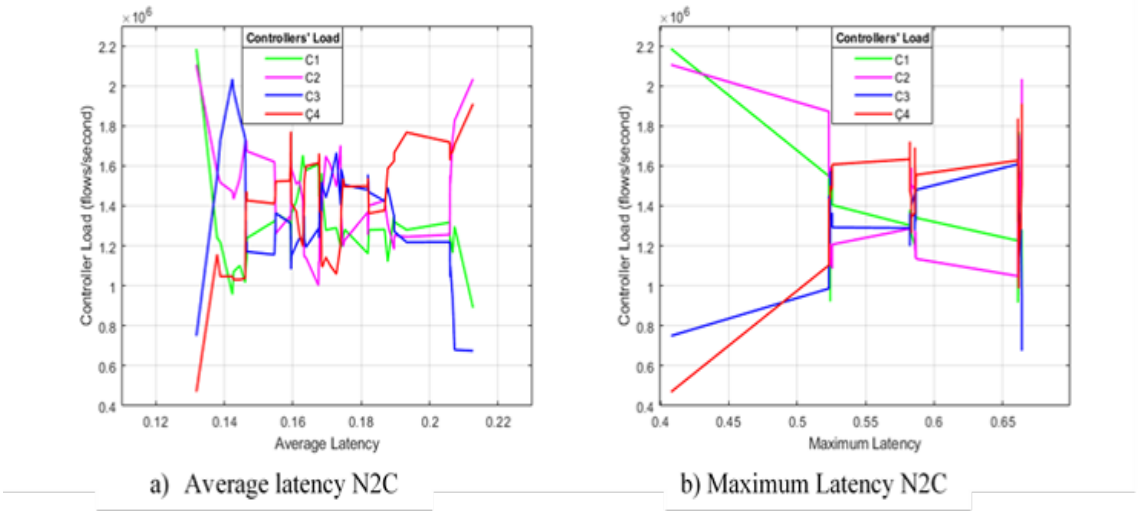


Figure 5.6: Average and Maximum node to controller Latency effect on load distribution

Having the impact of controller capacity, in the next section optimizing controller placements discussed.

5.2.2 Placement Optimization results Analysis

As discussed in the evaluation methods section, the optimization evaluation is performed with different cases, where controller backup capacity is considered and did not considered. Based on the two scenarios of Figure 5.7 and 5.9shows the number of controller and its placements. The two cases analysis considered in this work as explained in detail in next part of this section.

Case 1: A 100% controller utilization with failure-free For the determination of controller number, while assuming failure-free case and 100% controller utilization, three (3) controllers was obtained based on equation 4.2.3. The numbers shown in the circle is the node ID to represent the node names and coordinates. All possible placement options shown in figure 5.7 indicated controllers are placed at nodes ID 12, 13 and 17 with indicating multiple line circles and listing on top right side of the figure at the position indicated with the red point shown in the lower graph with latency and controller imbalance values. These placements are based on the node weights and controller capacity as a constraint and controller imbalance and maximum node to controller latency. The line and circle color indicated on the figure shows that nodes assigned to the similar color controller. In this case, optimum controller placement in keeping the controller number at the minimum number that can handle all node requests by minimizing node to controller latency and controller imbalance was performed. Minimizing

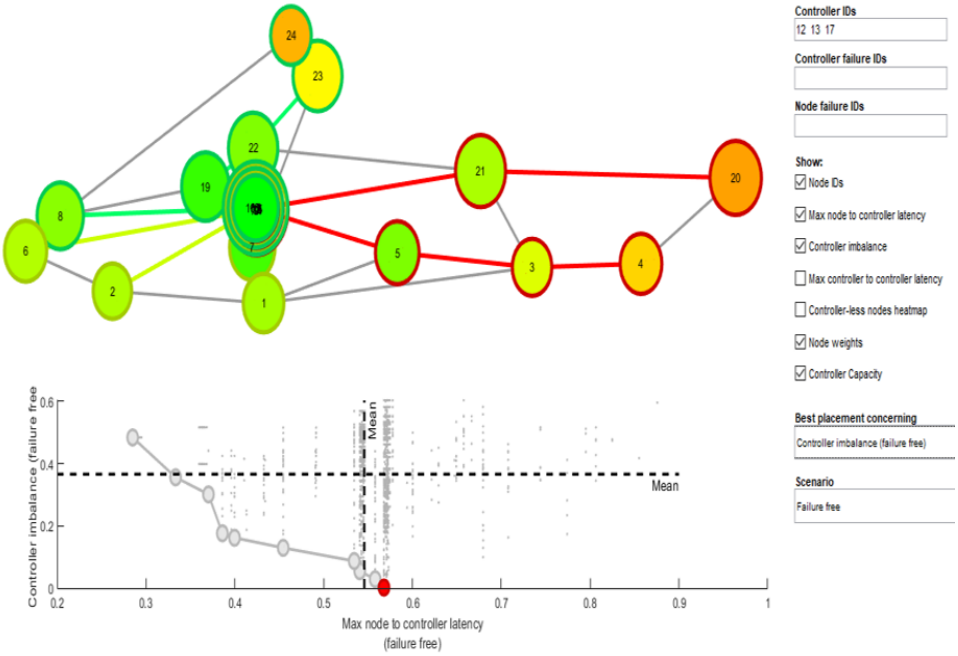


Figure 5.7: Controller Placements scenario without backup capacity

these metrics under the constraints of controller capacity and without controller capacity is investigated in this work. According to the result analysis, considering controller capacity as a constraint showed a great difference in better load balance for the same average latency and maximum latency placements. As discussed earlier the performance for average latency is better than maximum latency because of the POCO framework is based on load assignments that minimizes latency. Even though the capacity and with no capacity case converges to the same point, capacity-based results shows better-optimized placements in load balance and node to controller latency. The controller placements in a graph 5.8 with controller capacity observed is smaller in value (shorter line) for all possible placements because the controller capacity limits controller imbalance values range according description in evaluation method of equation 4.4.1.

The trade-off between load balance and latency is inversely proportional as observed in Figure 5.8 and Figure 5.10. The figures showed the trade-off analysis of controller Pareto frontier placement sets using three different cases with the capacity-based scenario and no capacity scenario. The heuristic methods used for optimizing placements were analyzed similarly. In this analysis K-Medoid generated better Pareto frontier sets in general though at some points Pareto Simulated method showed better placements in average latency. Another interesting is that capacity-based controller placement without heuristics showed better performance than heuristics methods optimization of with no capacity consideration case. The result investigations of this case is shown in Figure 5.8: Pareto frontier sets Placements scenario of no backup capacity.

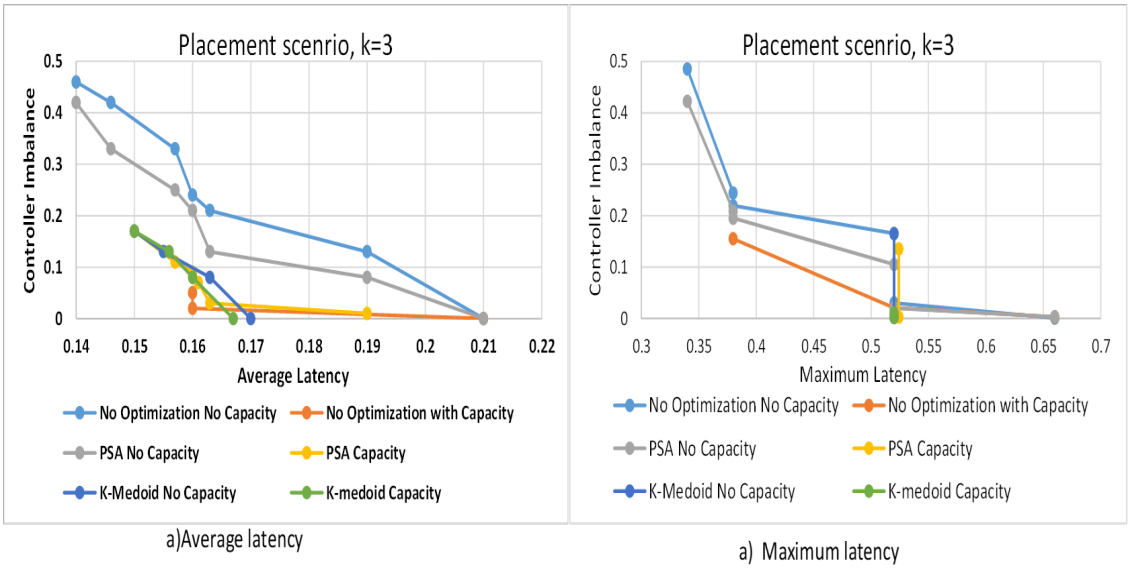


Figure 5.8: Pareto frontier sets Placements scenario of with out backup capacity

Case 2:A backup capacity of 30% for each controller where the controllers are operating in load sharing and standby pool mode The controller number determined while assuming 30% backup

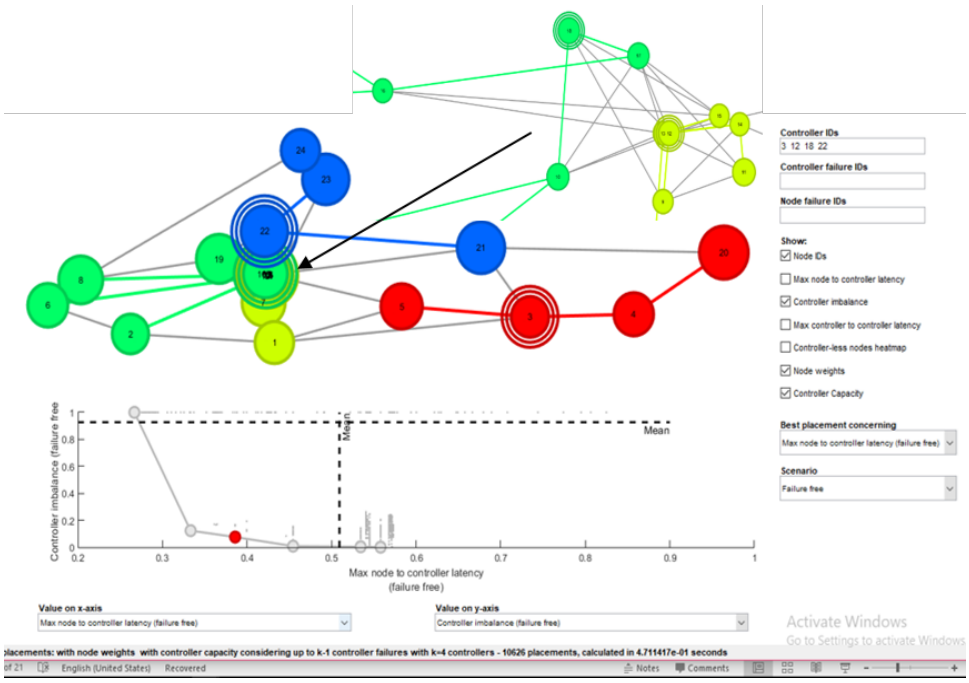
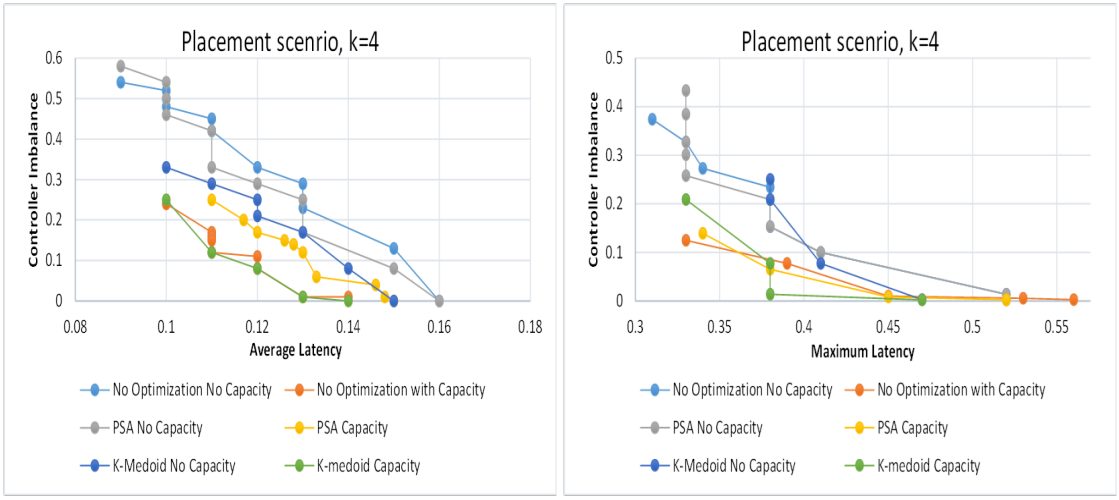


Figure 5.9: Controller Placements scenario with backup capacity

capacity is four (4) controllers based on equation 4.2.3.The numbers shown in the circle is the node ID to represent the node names and coordinates. All possible placement options shown in figure 5.9 indicated controllers are placed at nodes ID 3,12, 18 and 22. The multiple line circles and listing on top right side of the figure indicates controllers placement at the position indicated with the red point shown in the lower graph with latency and controller imbalance values.

The part of graph indicated using arrow is zoomed view of the topology the arrow pointed to the crowded nodes due to their coordinates are near to each other (Addis Ababa City nodes). These placements are based on the node weights and controller capacity as a constraint and controller imbalance and maximum node to controller latency. The line and circle color indicated on the figure shows that nodes assigned to the similar color controller. In this case,the goal is keeping the controller number at the minimum that can handle all node requests by minimizing node to controller latency and controller imbalance . Minimizing these metrics under the constraints of controller capacity and without controller capacity is investigated in this work. According to the result analysis considering controller capacity as a constraint showed a



a) Average latency b) Maximum Latency

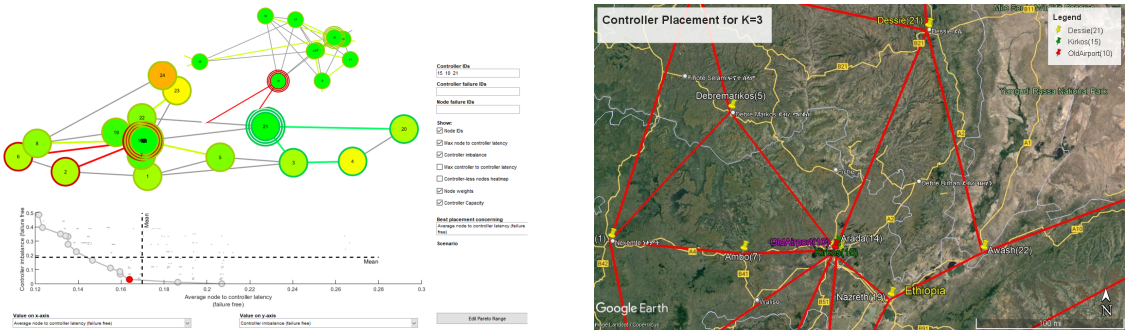
Figure 5.10: Pareto frontier sets Placements scenario of with backup capacity

great difference improvement in load balance for the same average and maximum latency sets of placements. Similarly, the two heuristic approaches used with capacity constraints showed better fair load distribution than heuristics without capacity consideration. capacity-based results better-optimized placements in load balance at the same node to controller latency and overall network performance. The trade-off between load balance and latency is inversely proportional as observed in Figure 5.8 and Figure 5.10. The figures showed the trade-off analysis of controller Pareto frontier placement sets using three different cases with the capacity-based scenario and no capacity scenario. The heuristic methods used for optimizing placements investigation indicated K-Medoid generated better Pareto frontier sets in general though at some points Pareto Simulated method showed better placements.

The capacity-based generic POCO framework optimization showed better network performance metrics than the heuristic methods used without capacity-based. Figure 5.10 showed the comparative analysis of controller placements using average latency and maximum latency with

controller imbalance. The PSA heuristics optimization results in a very slight improvement than capacity-based without heuristics in maximum latency case while it results in poor performance than capacity-based without heuristics. This is because the accuracy of the heuristic methods lower than the generic exhaustive method but shows better computation time.

To recap, in the topology case study and works, the K-Medoid results showed better performance in all cases and scenarios. Specifically this study result showed the improvement of 32% to 90% in load balance taking the same latency as a reference while 12.5% to 29% node to controller latency taking the same controller imbalance as a reference. Generally, a significant improvement in overall network performance is gained as shown in two the scenarios described above. The optimum placement could be decided by the service providers and telecom operators from a set of Pareto frontier placements that meet their minimum service requirements and Quality of Service (QoS). However,it is also possible to determine the optimum controller placement. For this case study of Ethio telecom WAN IP Core topology, the optum controller placement identified as shown in figure 5.11 for the scenario of no backup controller capacity case and figure 5.12 for backup controller capacity case. The figure shown in 5.11 a) is the placement generated



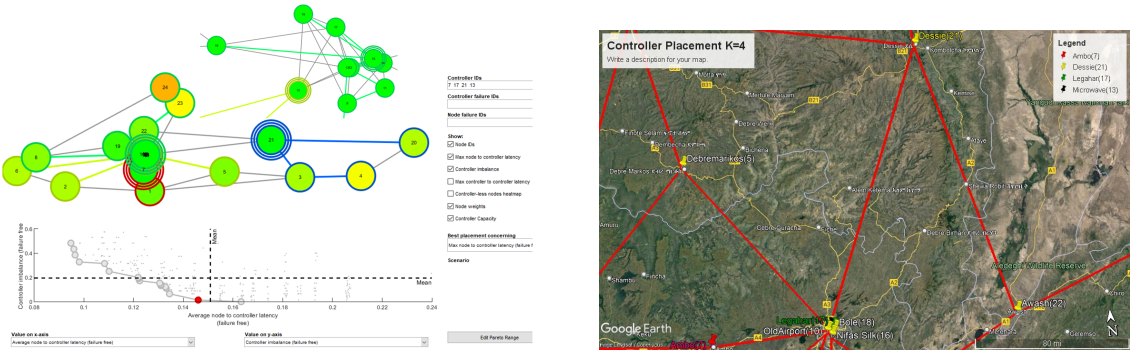
(a) Topology from POCO

(b) Topology on Google Earth

Figure 5.11: Optimum controller placement for k=3

using POCO framework Matlab based while b) is the optimum controller placement location on google earth. The node ID is the number in the parenthesis of location names as indicated on google earth place labels in both placement scenario. The node ID is used for convenience on POCO framework. Node ID on the POCO framework can be referred from google earth to identify the exact location of controller placement.The controller placement location shown on the google earth identified with different color that are described in the legend shown on the top right side of 5.11 (b) and 5.12 (b). The controller placement location shown on the POCO framework with triple circle lines while the node to controller assignment is shown with line color for link and node circle color of the same color with the color of controller placement circled on node ID of controller candidate. The optimum controller placement obtained with

K-Medoid heuristics is at node name (ID) are Kirkos(15), Old Airport(10) and Dessie1(21) for no backup capacity with 0.58ms average node to controller latency and 0.01 (1%) controller Imbalance as shown in figure 5.11. Similarly, for the case of backup capacity controllers placed at node names (IDs) are Ambo (7), Legehar (17),Dessie (21), and Microwave (13) with 0.53ms average node to controller latency and 0.0023(0.23%) Controller Imbalance as shown in figure 5.12. The node ID can be referred from the google earth network topology.



(a) Topology from POCO

(b) Topology on Google Earth

Figure 5.12: Optimum controller placement for k=4

Chapter 6

Conclusion and Future Work

6.1 Conclusion

The SD-WAN benefits are reaped if controller placement of the required number of controllers is properly addressed with respect to different vital parameters and constraints. These parameters crucially are based on their influences on the network performances. The node to controller latency and controller imbalance is indicated in this work they are important optimizing parameters as indicated in parameters impact assessment. It also revealed that, using controller capacity as a constraint significantly improved the network performance.In varying flow traffic requests from node to controller, using controller imbalance derived from Jain's fairness index is a good metrics. Nodes flow requests estimated to determine the minimum number of controller required for a specified controller product processing capacity and also to measure the load distribution fairness.

The number of the controller is determined from the capacity of the controller and flow requests arriving at the controller from each node.This method solved the trial and error way of determining controllers number for a specified network topology. For this case study network topology,in failure free case three and in failure case four controller numbers determined. In this work, the placement of controllers' issues is addressed with metrics such as average and maximum node to controller latency and controller imbalances derived from the fairness index with scenarios of controller capacity with and without backup capacity. The metrics trade-offs investigated as competing with one another to decide the optimized placements. This work also revealed that only minimization of node to controller latency might not result to the optimal placements of controller. At lower values of controller imbalance better load distribution indicating Jain's fairness index is good metric for measuring load balance in the circumstances of varying flow request traffic from nodes.

The analysis is based on the generic POCO framework Matlab-based tool and heuristics meth-

ods with and without controller capacity considerations. The placements for the determined controller based on capacity showed better improvements as indicated in network performance evaluation. The heuristics methods showed better results even though simulated annealing with capacity-based showed worse than capacity-based without heuristics. The K-Medoid result is better in convergence and performance evaluation.

To evaluate the time and accuracy of heuristics and the generic POCO framework a variety of network topology scales should be considered. Furthermore, controller capacity considerations improves the controller placements sets filtered with capacity constraint from all possible combinatorial placements. filtering controller placements that not exceeding controller capacity brings a quick convergence to optimal placements and lower algorithms execution time. The Pareto optimal solutions result in the POCO framework with and without controller capacity considerations and the heuristics solutions of simulated annealing and k-Medoid clustering analyzed showed that the load balance improvements for all methods gained from 32% to 90%. Based on the study, capacity constraint helped to converge rapidly to optimality. Controller placements without controller capacity limitation show higher latency and higher low load distribution, while capacity-based placement showed better load distribution fairness even better average latency from node to controller as shown in section .

6.2 Future Work

Anyone interested in the aims of this work can further work on it resembling the node request estimation on real topology. It is also interesting to extend the work considering flow processing time,queuing time, link utilization's, energy consumption. to meet the dynamics of network traffic, dynamic node assignments could be another interesting area in varying traffic flow requests. This work can be extended with verifying the network performance of the heuristics using Mininet emulation, and extend to other SDN use cases and network functions virtualization.

References

[1] L. Doyle. "7 key sd-wan trends to evaluate in 2021." (2021 (Accessed on August 15,2021)), [Online]. Available: https://searchnetworking.techtarget.com/tip/7-key-SD-WAN-trends-to-evaluate/.

[2] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[3] S. Khorsandroo, A. G. Sanchez, A. S. Tosun, J. M. A. Rodrıguez, and R. Doriguzzi-Corin, "Hybrid sdn evolution: A comprehensive survey of the state-of-the-art," *Computer Networks*, p. 107 981, 2021.

[4] J. C. C. Chica, J. C. Imbachi, and J. F. B. Vega, "Security in sdn: A comprehensive survey," *Journal of Network and Computer Applications*, vol. 159, p. 102 595, 2020.

[5] S. Scott-Hayward, "Design and deployment of secure, robust, and resilient sdn controllers," in *Proceedings of the 2015 1st IEEE conference on network Softwarization (NetSoft)*, IEEE, 2015, pp. 1–5.

[6] S. Ahmad and A. H. Mir, "Scalability, consistency, reliability and security in sdn controllers: A survey of diverse sdn controllers," *Journal of Network and Systems Management*, vol. 29, no. 1, pp. 1–59, 2021.

[7] J. H. Shaffer, *The effects of high bandwidth networks on wide-area distributed systems*. University of Pennsylvania, 1996.

[8] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," *ACM SIGCOMM Computer Communication Review*, vol. 42, pp. 473–478, 2012.

[9] Z. Markos, "Optimal placement of controllers for the adoption of software defined networking: In the case of ethiotelecom," Addis Ababa Unversity, Tech. Rep., Nov. 2018, An optional note.

[10] Y. Hu, T. Luo, W. Wang, and C. Deng, "On the load balanced controller placement problem in software defined networks," in *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, 2016 2nd IEEE International Conference on Computer and Communications (ICCC), 2016, pp. 2430–2434.

[11] S. Lange, S. Gebert, T. Zinner, *et al.*, "Heuristic approaches to the controller placement problem in large scale sdn networks," *IEEE Transactions on Network and Service Management*, vol. 12, pp. 4–17, 2015. DOI: 10.1109/TNSM.2015.2402432.

[12] G. Yao, J. Bi, Y. Li, and L. Guo, "On the capacitated controller placement problem in software defined networks," *IEEE Communications Letters*, vol. 18, pp. 1339–1342, 2014.

[13] D.-M. W. C. Rajendra K. Jain and W. R. Hawe, "A quantitative measure of fairnesss and discrimination for resource allocation in shared computer system," Eastern Research Lab, Digital Equipment Corporation 77 Reed Road Hudson MA 01749, Tech. Rep. 2, Sep. 1984, Fairness Index of resource allocation defined which could be used for controller allocation.

[14] Z. Yang, Y. Cui, B. Li, Y. Liu, and Y. Xu, "Software-defined wide area network (sd-wan): Architecture, advances and opportunities," in *2019 28th International Conference on Computer Communication and Networks (ICCCN)*, IEEE, 2019, pp. 1–9.

[15] S. Troia, L. M. M. Zorello, A. J. Maralit, and G. Maier, "Sd-wan: An open-source implementation for enterprise networking services," in *2020 22nd International Conference on Transparent Optical Networks (ICTON)*, IEEE, 2020, pp. 1–4.

[16] S. Bendale and J. Prasad, "Preliminary study of software defined network on covid-19 pandemic use cases," *Available at SSRN 3612815*, 2020.

[17] G. Mine, J. Hai, L. Jin, and Z. Huiying, "A design of sd-wan-oriented wide area network access," in *2020 International Conference on Computer Communication and Network Security (CCNS)*, 2020, pp. 174–177. DOI: 10.1109/CCNS50731.2020.00046.

[18] J. Costa-Requena, J. L. Santos, V. F. Guasch, *et al.*, "Sdn and nfv integration in generalized mobile network architecture," in *2015 European conference on networks and communications (EuCNC)*, IEEE, 2015, pp. 154–158.

[19] S. Tomovic, M. Pejanovic-Djurisic, and I. Radusinovic, "Sdn based mobile networks: Concepts and benefits," *Wireless Personal Communications*, vol. 78, no. 3, pp. 1629–1644, 2014.

[20] M. Wichtlhuber, R. Reinecke, and D. Hausheer, "An sdn-based cdn/isp collaboration architecture for managing high-volume flows," *IEEE Transactions on Network and Service Management*, vol. 12, no. 1, pp. 48–60, 2015.

[21] N. Zhang and H. Hämmäinen, "Cost efficiency of sdn in lte-based mobile networks: Case finland," in *2015 international conference and workshops on networked systems (NetSys)*, IEEE, 2015, pp. 1–5.

[22] V. Ahmadi and M. Khorramizadeh, "An adaptive heuristic for multi-objective controller placement in software-defined networks," *Computers Electrical Engineering*, vol. 66, pp. 204–228, 2018.

[23] M. Khorramizadeh and V. Ahmadi, "Capacity and load-aware software-defined network controller placement in heterogeneous environments," *Computer Communications*, vol. 129, pp. 226–247, 2018.

[24] L. Yao, P. Hong, and W. Zhou, "Evaluating the controller capacity in software defined networking," in *2014 23rd International Conference on Computer Communication and Networks (ICCCN)*, IEEE, Aug. 2014. DOI: 10.1109/icccn.2014.6911857.

[25] M. He, A. Basta, A. Blenk, and W. Kellerer, "Modeling flow setup time for controller placement in sdn: Evaluation for dynamic flows," in *2017 IEEE International Conference on Communications (ICC)*, 2017 IEEE International Conference on Communications (ICC), 2017, pp. 1–7. DOI: 10.1109/ICC.2017.7996654.

[26] V. Ahmadi, A. Jalili, and M. Khorramizadeh, "Multi-objective controller placement problem: Issues and solution by heuristics," *International Journal of Computer Science and Information Security*, vol. 14, no. 8, p. 543, 2016.

[27] B. Zhang, X. Wang, and M. Huang, "Multi-objective optimization controller placement problem in internet-oriented software defined network," *Computer Communications*, vol. 123, pp. 24–35, 2018, ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2018.04.008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366416307241.

[28] A. K. Singh and S. Srivastava, "A survey and classification of controller placement problem in sdn," *International Journal of Network Management*, vol. 28, no. 3, e2018, 2018.

[29] J. Hollinghurst, A. Ganesh, and T. Baugé, "Controller placement methods analysis," in *2016 6th International Conference on Information Communication and Management (ICICM)*, IEEE, 2016, pp. 239–244.

[30] S. Lange, S. Gebert, J. Spoerhase, *et al.*, "Specialized heuristics for the controller placement problem in large scale sdn networks," in *2015 27th International Teletraffic Congress*, IEEE, 2015, pp. 210–218.

[31] 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 *Proceedings of the 2013 25th International Teletraffic Congress (ITC)*, 2013, pp. 1–9. DOI: 10.1109/ITC.2013.6662939.

[32] D. Hock, S. Gebert, M. Hartmann, T. Zinner, and P. Tran-Gia, "Poco-framework for pareto-optimal resilient controller placement in sdn-based core networks," in *2014 IEEE Network Operations and Management Symposium (NOMS)*, IEEE, 2014, pp. 1–2.

[33] S. I. Harned, "Poco-moea: Using evolutionary algorithms to solve the controller placement problem," 2016.

[34] G. Schütz and J. Martins, "A comprehensive approach for optimizing controller placement in software-defined networks," *Computer Communications*, 2020.

[35] X. Hou, W. Muqing, L. Bo, and L. Yifeng, "Multi-controller deployment algorithm in hierarchical architecture for sdwan," *IEEE Access*, vol. 7, pp. 65 839–65 851, 2019. DOI: 10.1109/ACCESS.2019.2917027.

[36] Y. Hu, T. Luo, N. C. Beaulieu, and C. Deng, "The energy-aware controller placement problem in software defined networks," *IEEE Communications Letters*, vol. 21, no. 4, pp. 741–744, 2017. DOI: 10.1109/LCOMM.2016.2645558.

[37] A. A. Neghabi, N. J. Navimipour, M. Hosseinzadeh, and A. Rezaee, "Energy-aware dynamic-link load balancing method for a software-defined network using a multi-objective artificial bee colony algorithm and genetic operators," *IET Communications*, vol. 14, no. 18, pp. 3284–3293, 2020. DOI: 10.1049/iet-com.2019.1300.

[38] A. Ruiz-Rivera, K.-W. Chin, and S. Soh, "GreCo: An energy aware controller association algorithm for software defined networks," *IEEE Communications Letters*, vol. 19, no. 4, pp. 541–544, Apr. 2015. DOI: 10.1109/lcomm.2015.2394457.

[39] K. Kurroliya, S. Mohanty, B. Sahoo, and K. Kanodia, "Minimizing energy consumption in software defined networks," in *2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, 2020, pp. 885–890. DOI: 10.1109/SPIN48934.2020.9070985.

[40] J. Galán-Jiménez, "Minimization of energy consumption in ip/sdn hybrid networks using genetic algorithms," in *2017 Sustainable Internet and ICT for Sustainability (SustainIT)*, 2017, pp. 1–5. DOI: 10.23919/SustainIT.2017.8379802.

[41] D. B. Rawat and C. Bajracharya, "Software defined networking for reducing energy consumption and carbon emission," in *SoutheastCon 2016*, IEEE, 2016, pp. 1–2.

[42] L. F. Muller, R. R. Oliveira, M. C. Luizelli, L. P. Gaspary, and M. P. Barcellos, "Survivor: An enhanced controller placement strategy for improving SDN survivability," in *2014 IEEE Global Communications Conference*, IEEE, Dec. 2014. DOI: 10.1109/glocom.2014.7037087.

[43] A. Jalili, V. Ahmadi, M. Keshtgari, and M. Kazemi, "Controller placement in software-defined wan using multi objective genetic algorithm," in *2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, 2015, pp. 656–662. DOI: 10.1109/KBEI.2015.7436121.

[44] A. Jalili, M. Keshtgari, R. Akbari, and R. Javidan, "Multi criteria analysis of controller placement problem in software defined networks," *Computer Communications*, vol. 133, pp. 115–128, 2019.

[45] V. Ahmadi, A. Jalili, S. M. Khorramizadeh, and M. Keshtgari, "A hybrid nsga-ii for solving multiobjective controller placement in sdn," in *2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, 2015, pp. 663–669. DOI: 10.1109/KBEI.2015.7436122.

[46] S. Auroux, "Flow processing-aware control application placement," Ph.D. dissertation, Paderborn, Universität Paderborn, 2017.

[47] Z. Li, Y. Hu, T. Hu, and P. Wei, "Dynamic sdn controller association mechanism based on flow characteristics," *IEEE Access*, vol. 7, pp. 92 661–92 671, 2019.

[48] H. Xue, K. T. Kim, and H. Y. Youn, "Dynamic load balancing of software-defined networking based on genetic-ant colony optimization," *Sensors*, vol. 19, no. 2, p. 311, 2019.

[49] X. Huang, S. Bian, Z. Shao, and H. Xu, "Dynamic switch-controller association and control devolution for sdn systems," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1–6. DOI: 10.1109/ICC.2017.7997427.

[50] T. Yuan, X. Huang, M. Ma, and J. Yuan, "Balance-based sdn controller placement and assignment with minimum weight matching," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–6. DOI: 10.1109/ICC.2018.8422637.

[51] H. Ren, X. Li, J. Geng, and J. Yan, "A sdn-based dynamic traffic scheduling algorithm," in *2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, 2016, pp. 514–518. DOI: 10.1109/CyberC.2016.103.

[52] N. Cai, Y. Han, Y. Ben, W. An, and Z. Xu, "An effective load balanced controller placement approach in software-defined WANs," in *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, IEEE, Nov. 2019. DOI: 10.1109/milcom47813.2019.9020804.

[53] A. Martinez, M. Yannuzzi, V. Lopez, *et al.*, "Network management challenges and trends in multi-layer and multi-vendor settings for carrier-grade networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2207–2230, 2014.

[54] M. P. Martinez, T. Laszlo, N. Pataki, C. Rotter, and C. Szalai, "Multivendor deployment integration for future mobile networks," in *International Conference on Current Trends in Theory and Practice of Informatics*, Springer, 2018, pp. 351–364.

[55] C. Rotter, J. Illés, G. Nyıri, L. Farkas, G. Csatári, and G. Huszty, "Telecom strategies for service discovery in microservice environments," in *2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*, IEEE, 2017, pp. 214–218.

[56] Sheldon. "Tcp/ip vs. osi: What's the difference between the two models?" (2017[Accessed on July 10, 2021]), [Online]. Available: https://community.fs.com/blog/tcpip-vs-osi-whats-the-difference-between-the-two-models.html.

[57] P. G. B. Culver, *Software Defined Networks A Comprehensive Approach*, 2nd ed. 50 Hampshire Street, 5th Floor, Cambridge, MA 02139, United States: Todd Green, 2017, ISBN: 978-0-12-804555-8.

[58] ITU. "New itu architecture for sdn control of transport networks." (2018[Accessed on September 15, 2021]), [Online]. Available: https://www.itu.int/en/myitu/News/2020/05/18/12/57/New-ITU-architecture-for-SDN-control-of-transport-networks.

[59] Z. Latif, K. Sharif, F. Li, M. M. Karim, S. Biswas, and Y. Wang, "A comprehensive survey of interface protocols for software defined networks," *Journal of Network and Computer Applications*, vol. 156, p. 102 563, Apr. 2020. DOI: 10.1016/j.jnca.2020.102563.

[60] Y.-W. Ma, Y.-C. Chen, and J.-L. Chen, "SDN-enabled network virtualization for industry 4.0 based on IoTs and cloud computing," in *2017 19th International Conference on Advanced Communication Technology (ICACT)*, IEEE, 2017. DOI: 10.23919/icact.2017.7890083.

[61] D. M. Metzler, *2018 guide to wan architecture and design*, Webtorials Analyst Devision guide, WAN network, 2018.

[62] Cisco. "What is mpls - multiprotocol label switching." (July 30, 2016[Accessed on September 8, 2021]), [Online]. Available: https://www.section.io/engineering-education/introduction-to-mpls-and-mpls-vpn-technology/.

[63] L. D. Ghein. "Mpls fundamentals." (2016 (September on 13, 2021)), [Online]. Available: https://www.ciscopress.com/store/mpls-fundamentals-9780133433272.

[64] S. A. N. "Introduction-to-mpls-and-mpls-vpn-technology." Accessed Aug. 30, 2021. (July 30, 2020[Accessed on August 25, 2021]), [Online]. Available: https://www.section.io/engineering-education/introduction-to-mpls-and-mpls-vpn-technology/.

[65] S. Peak. "Sdn-explained what is sdn ?" (2021[Accessed on August 21, 2021]), [Online]. Available: https://www.silver-peak.com/sd-wan/sd-wan-explained.

[66] D. Boraah. "Sd-wan: 5 deployment scenarios." (2019 (Accessed on August 18,2021)), [Online]. Available: https://lavellenetworks.com/blog/sd-wan-5-deployment-scenarios/.

[67] S. Mandal, "Experience with b4: Google's private SDN backbone," Santa Clara, CA: USENIX Association, Jul. 2015.

[68] S. Jain, A. Kumar, S. Mandal, *et al.*, "B4: Experience with a globally-deployed software defined wan," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 3–14, 2013.

[69] C.-Y. Hong, S. Mandal, M. Al-Fares, *et al.*, "B4 and after," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, ACM, Aug. 2018. DOI: 10.1145/3230543.3230545.

[70] M. Jimenez and H. Kwok. "Building express backbone: Facebook's new long-haul network." (2017 [Accessed on September, 2021]), [Online]. Available: https://engineering.fb.com/data-center-engineering/building-expressbackbone-facebook-s-new-long-haul-network/.

[71] J. Meza, T. Xu, K. Veeraraghavan, and O. Mutlu, "A large scale study of data center network reliability," in *Proceedings of the Internet Measurement Conference 2018*, 2018, pp. 393–407.

[72] C.-Y. Hong, S. Kandula, R. Mahajan, *et al.*, "Achieving high utilization with software-driven wan," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 15–26, Aug. 2013, ISSN: 0146-4833. DOI: 10.1145/2534169.2486012. [Online]. Available: https://doi.org/10.1145/2534169.2486012.

[73] N. Cranford. "Three examples of sdn deployments." (2017[Accessed on August 12, 2021]), [Online]. Available: https://www.rcrwireless.com/20170622/software-defined-networking-sdn/.

[74] OpenDaylight. "Kt-corporation-uses-opendaylight-to-deploy-transport-sdn-wan-network." (2017 [Accessed on September 12, 2021]), [Online]. Available: https://www.opendaylight.org/use-cases/stories/.

[75] ATT. "7 principles of att's network transformation disaggregation, cloud, and intelligent sdn." (Apr. 2020 [White Paper]), [Online]. Available: https://about.att.com/content/dam/snrdocs/7_Tenets_of_ATTs_Network_Transformation_White_Paper.pdf.

[76] ZTE. "Te and china mobile complete industry's first cloudos and commercial sdn system decoupling test in nfv architecture." (2018[Accessed on September 05, 2021]), [Online]. Available: https://www.zte.com.cn/global/about/news/20180904.html.

[77] D. Goovaerts. "Nokia-scores-cloud-win-china-mobile." (2021[Accessed on July 12, 2021]), [Online]. Available: https://www.fiercetelecom.com/telecom/.

[78] J. H. Cox, J. Chung, S. Donovan, *et al.*, "Advancing software-defined networks: A survey," *IEEE Access*, vol. 5, pp. 25 487–25 526, 2017.

[79] M. Ashton and Associates. "How to plan for network virtualization and sdn." (2021 (Accessed on August 21,2021)), [Online]. Available: https://www.necam.com/doclibrary/WhitePaper_How_to_Plan_for_Network_Virtualization_and_SDN.pdf.

[80] J. Metzler. "Planning for sdn." (2013 (Accessed on August 21,2021)), [Online]. Available: https://www.networkworld.com/article/2172058/lan-wan-planning-for-sdn.html.

[81] Y. Zhang, *Network Function Virtualization: Concepts and Applicability in 5G Networks*. John Wiley & Sons, 2018.

[82] OpenNetworkingFoundation. "Openflow switch specification v1.5.1." (Mar. 2015[Accessed on March 2, 2021]), [Online]. Available: https://opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf.

[83] M. F. Bari, A. R. Roy, S. R. Chowdhury, *et al.*, "Dynamic controller provisioning in software defined networks," in *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*, IEEE, 2013, pp. 18–25.

[84] H. Yu, K. Li, and H. Qi, "An active controller selection scheme for minimizing packet-in processing latency in SDN," *Security and Communication Networks*, vol. 2019, pp. 1–11, Oct. 2019. DOI: 10.1155/2019/1949343.

[85] J. Ali, B.-h. Roh, and S. Lee, "QoS improvement with an optimum controller selection for software-defined networks," *PLOS ONE*, vol. 14, no. 5, J. Huang, Ed., e0217631, May 2019. DOI: 10.1371/journal.pone.0217631.

[86] L. Zhu, M. M. Karim, K. Sharif, F. Li, X. Du, and M. Guizani, "Sdn controllers: Benchmarking & performance evaluation," *arXiv preprint arXiv:1902.04491*, 2019.

[87] B. A. Bhuvaneswaran V., M. V. Tassinari M., and S. Banks, *Benchmarking methodology for software-defined networking (sdn) controller performance*, Internet Request For Comments, RFC, Oct. 2018. DOI: 10.17487/RFC8456. [Online]. Available: https://tools.ietf.org/html/rfc8456.

[88] A. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, and R. Sherwood, "On controller performance in software-defined networks," in *2nd {USENIX} Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE 12)*, 2012, pp. 5–9.

[89] R. Khalili, Z. Despotovic, and A. Hecker, "Flow setup latency in SDN networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 12, pp. 2631–2639, Dec. 2018. DOI: 10.1109/jsac.2018.2871291.

[90] H. P. E. D. LP. "Hp van sdn controller software - specifications." (2021[Accessed on June 10, 2021]), [Online]. Available: https://support.hpe.com/hpesc/public/docDisplay?docLocale=en_US&docId=c03967703#N10011.

[91] E. Joseph Coffey senior principal engineer, *Latency in optical fiber systems*, White Paper CommScope, optical fiber speed, 2017.

[92] V. Huang, G. Chen, P. Zhang, *et al.*, "A scalable approach to sdn control plane management: High utilization comes with low latency," *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 682–695, 2020. DOI: 10.1109/TNSM.2020.2973222.

[93] V. Huang, G. Chen, Q. Fu, and E. Wen, "Optimizing controller placement for software-defined networks," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, IEEE, 2019, pp. 224–232.

[94] D. E. BROWN and C. L. HUNTLEY, "A practical application of simulated annealing to clustering," *Institute for Parallel Computation and Department of Systems Engineering*, vol. 25, pp. 401–412, 1992.

Optimum Controller Placement in SD-WAN Deployment case of Ethio Telecom

Adugna Getu[1] and Ephrem Teshale (PhD)[2]

[1]Addis Ababa Institute of Technology, Addis Ababa University
[2]Ethio Telecom

Abstract: In multiple controller systems the scalability and reliability challenges solved but challenging issue left with the performance of the network. The number of controllers needed for a given network topology and where it should be placed optimally is a challenging issue determines network performance. In this work, controller placement for SD-WAN deployment optimization in varying traffic flow request amount arriving at controller that not exceed controller capacity is addressed. The optimum controller placement to minimize controller number through minimization of propagation delay and controller imbalance with constraints of controller capacity and flow requests. In this work, the trivial way of determining controller number taking as the input parameter is replaced with determining it from controller capacity and node weights (flow requests from nodes). Also introduced the concept of load balance based on Jain's fairness index. At last, the impact of controller capacity assessment on parameters of the optimum placement investigation showed controller capacity affects the placement metrics. As a result, K-Medoid showed improvement of 32% to 90% in load balance taking the same latency as a reference and 12.5% to 29% node to controller latency taking the same controller imbalance as a reference. Finally, optimum controller placement was identified and shown on google earth.

Keywords—Software-Defined Networking, SD-WAN, OpenFlow, Controller placement, Heuristic Algorithm, SDN controller, Switches, Jain's Fairness Index

I. Introduction

Internet demand and data traffic explosively increasing with the proliferation of devices and applications within decades. To accommodate these demands, continuous deployment of network infrastructure by service providers and telecom operators leads to the network architecture being highly heterogeneous and complex, making network management, difficult. Besides this, having no global view of the network, poor network performance issues and service provisioning becomes their challenging duty.

The promising solutions of Software-Defined Networking (SDN) for the current network infrastructure in service providers and operators emerged with different opportunities and challenges. Minimizing challenges and maximizing the opportunities i.e., Optimized planning of components of SDN architectures offers significant advantages for cloud data centers, cellular networks, transport networks (Optical), and for service providers in Wide Area Networks (WAN). Simplifying data forwarding, allowing managing a network in a flexible, easy service provisioning, and guaranteeing Quality of Services in such widely distributed and heterogeneous network specially in the WAN network that consists of the widely distributed and heterogeneous network environments is vital. This work focused on Software-Defined Wide Area Network (SD-WAN) as it provides better wireless integration WI-FI and 4G/5G in business applications to Internet of Things (IoT) and other broad network services [1].

The logically centralized single controller scalability (a limited capacity) and reliability (a single point of failure and vulnerable attacks) are some of the challenges of SDN deployment [2]–[4]. Many researchers in [2], [5], [6] works, proposed distributed multi-controller systems as a solution for such challenges. In distributed controller system, controllers' performance that affects the entire network performance is affected with setup processing time where propagation delay of switches to controllers dominates in WAN network [7], [8]. Also, the load imbalance where some controller(s) overloaded increases controller processing time, and synchronization overhead is added to affect the network performance. To obtain a scalable, balanced, and failure-resilient topology i.e., better performance, it is necessary to determine a minimum number of controllers and its optimum location [9]–[11], while ensuring low latency between nodes and associated controllers, as well as between controllers [9]. Simultaneously, it is necessary to keep balanced the processing loads of controllers [10], and within their operating capacity [12]. The other important that needs attention is traffic estimates and analysis of the trade-off between metrics. Therefore, the optimum placement of controllers to address how many controllers are required and where to place them [9], in the perspective of capacities of controllers [12], propagation latency between controllers and nodes, and inter-controllers [9], and load distribution among controllers [10] improves controller performance. The optimum solutions will increase the ability of the network to react to events quickly, leading to a more robust topology, even when failures happen. In this work, This work presented optimization models with the mathematical formulation of the controller placement, which minimizes the number of controllers, with respective to controller capacities, propagation delay, and flow traffic requests. The traffic flow request from

each node is assumed as varying traffic flows and estimation of the flow request emanated from each node is done. The optimization models for optimum placement of controllers are analyzed and evaluated. In this work of load balancing, the fairness index is used as a metrics of load balancing as indicated in the work of author [13] of title "fairness resource allocation in a shared computer system" to minimize the controller imbalance measuring how the load distribution is discriminated.In this work optimal placements of the minimum number of controllers that can accommodate all flow requests arrive at the controller analyzed with generic and heuristic methods.

The paper organization is initiated introducing the topic and goes to reviewing related works in its second section. The methodology and result discussion followed from previous section respectively. The paper closed concluding the work with remarking the future works.

II. Contribution

In the controller placement, the controller placement problem formulated, which minimizes the number of controllers, choosing their location, nodes assignments to controllers while satisfying controller capacity, propagation latency, and fair load distribution. As long as our understandings and review of the other works reviewed in the literature we have not come across who used fairness index for load balance of controller placement. This work assumed each node's traffic flow request is different and estimated with simple network topology using Mininet to resemble real telecom operators networks. The minimum number of controllers required for the SD-WAN deployment is determined based on the controller capacity and flow requests from all nodes rather than the trivial way of determining. Failure case of the controllers with the concept of load sharing and standby pool based is also considered rather than as redundant standby. Then keeping the minimum number of controllers based on flow-based fairness index (i.e., load distribution fairness among controllers) and propagation latency. The two heuristic algorithms and capacity-based models that optimize the placement of controllers with the minimum number of controllers are explored and analyzed to indicate the suitable model. The result accuracy level is based on the type of algorithms and techniques used during the research implementations. It could also enhance the researches for telecom operators and even vendors that could be referenced in academic and professional works.

A. *Why SD-WAN?*

In legacy WAN networks, the deployment of MPLS bandwidth and multipoint VPN has been too expensive and has too long service configurations. SD-WAN addresses this problem allowing elastic traffic demands, dynamic bandwidth allocation, and dynamic path control of business requirements and performance constraints. In the logical separation of SD-WAN, running IP across the Ethernet easily across the MPLS is allowed because of the virtual overlay created by SD-WAN and path selection and routing of traffic performed based on policies defined by the user. The virtual overlay created gathers loss and latency metrics on every connection to the collected metrics with predefined policies to direct traffic flows to another tunnel.

SD-WAN operates across all internet connections like 3G/4G/LTE/5G, xDSL, and cable, private data services such as MPLS and VPN. SD-WAN is also cloud-ready and then public cloud to data centers connectivity is easily performed.

B. *SD-WAN architecture*

As different vendors claim, SD-WAN is quickly maturing and widely adopted by enterprises and organizations as a cost-effective way to connect data centers and branches to SaaS and other cloud applications. These are enabled with the architecture of SD-WAN overlay network which creates a virtual network built on top of existing network infrastructure. This approach improves the scalability of the network and supports modularity, multi-tenancy, and network virtualization.

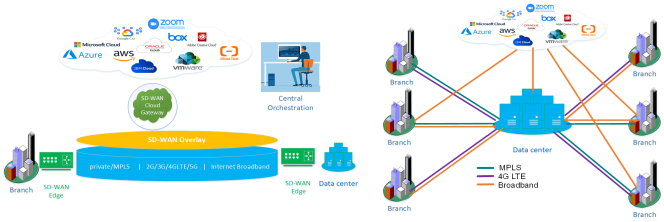


Fig. 1: SD-WAN Architecture [14]

SD-WAN architecture is explained in Figure 1 allows service providers and telecom operators to leverage secure connection of any mixture of carrier services including MPLS, LTE/5G, and broadband internet services. SD-WAN centralized controller function to intelligently direct and secure traffic across WAN increasing application performance and high-quality user experience delivered.

The cost of back-hauling all traffic in WAN networks in which traffic flows twice that turns around router at headquarter for advanced security inspection no more affects the SD-WAN network as it fully supports applications hosted in on-premise data centers, public or private clouds, and Software as a Service (SaaS) services [14].

The difference between WAN and SD-WAN architecture is that the traditional WAN requires hardware at each end of the network to complete connection whereas the SD-WAN abstracts control of the WAN connection into a software layer, as control and data plane are split into a separate plane. On another hand, each traditional WAN device is configured independently and manually and managed separately, while the SD-WAN controller consolidates and centralizes the configuration and service provisioning and enabled to manage of all WAN endpoints together.

III. Literature Review

The research of optimum controller placement is changing with network scale and network applications use cases require-

ments due to controllers' layout influences the performance of the network. This section reviews briefly works related to controller placement issues with constraints of latency, controller capacity, load balance, energy, and traffic request demand estimates.

Heller et al. in [8] initiated and formulated the controller placement problem as a general facility location problem. The authors' analysis focused on controller placement based on both average nodes to controller latency and worst-case node to controller latency trade-off observation.

The author in [9] studied the optimum controller placement intending to minimize the propagation latency from nodes to assigned controllers and among controllers without considering the controller capacity and traffic estimates. Similarly, the authors Zhao and Wu [15] focuses to reduce the link delay through an integer linear program and heuristic algorithm to provide scalable controller placement.

Hock et al. [16], [17] Offered a new approach for controller placement to improve the resiliency and failure tolerance of software-defined networking performance. In their work, different network performance metrics such as inter-controller latency and node to controller latency at average and maximum delay trade-offs were investigated and concluded that there is no single optimal placement solution.

For the optimum controller placements, different techniques and heuristic approaches addressed extending the work of [16], [17] further considering capacity in [11], [18]. Lange et al in their works of [11], [18] heuristic approach for controller placement proposed by extending the POCO tool focus trade-off analysis of resulting accuracy and time of computation investigation. The authors' two objective metrics average latency of node to controller and controller imbalance capacitated based as capacity bound but the load balance concept based on deduction of maximum and minimum number assigned nodes to a controller, and also controller capacity based flow request not indicated. This method is not viable for varying flow requests traffic volume from different nodes to controllers.

Harned [19] expanded the ideas of [11], [16]–[18] works enabling POCO framework for multi-objective controller placement based on evolutionary algorithm.

The weight of the nodes is not considered and assumed to be the same though it is uneven while Yao et al [12] considered the weight of the nodes when locating the controllers. The first initiated capacitated controller placement problem was addressed in [12] with the objective function of minimizing the maximum propagation latency with the constraint to controllers load not exceeding the controllers capacity. In their work, they considered that the capacities of controllers, load assigned to controllers, and propagation latency between nodes and assigned controllers with the assumption of controller's capacity is limited based on link bandwidth and a limited number of switches assigned to the controller. The authors even though not considered the load balance among the controllers, concluded their method minimized the required number of controllers. Their work revealed also the node assigned to their nearest controller as capacity is not exceeded. Whereas the authors in [20] claim they developed the comprehensive approach for capacitated controller placements to minimize the controller number based on propagation latency, resiliency, and scalability as a multi-objective-based placement. The authors used controller capacity and inter-controller load distribution besides to mentioned metrics.

While He et al [21] controller placement method is used to optimize minimum average flow setup time concerning different traffic conditions to improve network performance.

Hou et al. [22] addressed hierarchical multi-controller deployment for the WAN network that divides control planes into multiple levels. The authors are based on load balancing, reliability, and latency using an improved Louvain algorithm. However, flow-based load balance and controller capacity issues were overlooked.

Khorramizadeh and Ahmadi [23] proposed controller capacity and load-aware controller placement with two phases. Phase one determines the number of controllers that minimizes the overall cost of deployment and in phase two the controller location that balances controller load minimizing inter-controller latency. Two greedy approaches were used for the location of controllers and allocation of controllers' load.

The energy-aware based works of literature [24]–[26] addressed minimizing energy consumption in SDN for controller placement and controller to switches associations while [27]–[29] showed the usage of software-defined networking role in minimizing energy and carbon emission extending to environmental impacts and also depicted methods of reducing energy consumption by SDN itself.

Muller et al [30] use the concept of controller capacity with its backup capacity and switch flow requests of the same capacity with maximization of connectivity controller to switches to enhance controller placement improving SDN survivability. The same flow requests and the same capacity is assumed for controller placements survivability.

Jalili et al improved their work of [31] in their work [32] as multi-criteria-based controller placement such as hop count, propagation latency of node to the controller, and utilization of links used to assign switch to a controller and analyzed the effect of these objectives on quality of service. The trade-offs among the metrics analyzed proposing Genetic Algorithm heuristics for the controller placement. Their result showed that the assignment based on their approach was better efficient regarding the link load balancing problem.

Ahmadi et al in their work of [33]–[35] focused on the Load balancing scenario of the Controller placement problem using different techniques like Flow-based controller placement that specifically addressed in [21], [36] and dynamic switch to controller assignment as indicated in authors of [37]–[41].

While Cai et al in [42] addressed the constraints of device hardware indicating that the controller capacity is limited in their work and defining the controller capacity and determining the minimum number of controllers. However, it gradually increases the number of controllers until no overload happens. Their objective is only to meet the load balance of controllers to avoid overload.

In nutshell, the controller placements optimization aforementioned in each authors considered different metrics and use cases. The traffic flow request from node to controller is assumed the same while load balance is measured with the value of controller imbalance. Number of controllers required figured based on trivial way of evaluating network performance for each number. On the other hand, the load balance is determined with the difference of the maximum and minimum nodes assigned to controller with the assumptions of nodes traffic flow request to the controller is the same and controller capacity considered is based on traffic arrived on Network Interface Card (NIC) of the controller.

This work considered the controller capacity, having the link bandwidth notion addressed in [12], to estimate the traffic requests arrive at the controller from each node using Mininet and the traffic matrix of the nodes. The traffic request estimates from each node used to limit controller capacity not exceed its capacity during assignment of node for better controller processing time for network performance. Similarly, the work used the POCO framework [11], [17], [18] as a tool with modifications on load imbalance based on controllers' load introducing the fairness index (Jian's index) and controller capacity in controller placement using heuristics methods.

IV. Methodology

The mathematical formulations, assumptions, and proposed algorithms used in this work are depicted and stated in the next subsections.

A. Mathematical formulation

In this work, an SD-WAN of the undirected graph of G (V, E) where V is the set of switches/routers of n nodes and E is the set of edges of bidirectional interconnections/links between pair of nodes is considered. Graph G (V, E), where V= $\{1, 2\ldots$ n$\}$ set of nodes and E= $\{$Set of bidirectional Edges$\}$ the network contains n nodes that are either switches or routers. The distance matrix D has the shortest path latency between nodes i and j i.e., $d_{i,j}$ normalized with the graph's diameter.

Formalizing the controller placement using Binary Linear Programming (BLP) and defining the Binary decision variables [20], [43] as equations 1 and 2 that determine solutions of the model.

$$c_i = \begin{cases} 1, & if\ node\ i \epsilon V\ location\ is\ selected \\ & for\ controller\ placement \\ 0, & Otherwise \end{cases} \quad (1)$$

$$s_{ij} = \begin{cases} 1,\ if\ node\ j \epsilon V\ is\ controlled\ by \\ controller\ placed\ at\ node\ i \epsilon V \\ 0,\ Otherwise \end{cases} \quad (2)$$

Equation 1 determines whether the node is selected as the candidate for controller while Equation 2 defines the node to controller association sets.

B. Objective functions

The objective function in the optimum controller placement has several competing objectives to meet the QoS requirements. Minimizing the overall number of controllers can minimize the costs in capital investments [20] and operational costs including the energy consumption [24], [26], [28] and maximizes the controller utilization [43], [44]. In this work minimum number of controllers is determined from controller capacity with maximum controller utilization. But if the operators are interested in SDN deployment to have a backup capacity 30% advance capacity is considered while the controllers work as standby and load sharing pool principle, minimizing the overall number of controllers during the deployment. This work Supposed that the set of objective functions is known and should be minimized means that no other placement is in a search space i.e., determines the Pareto optimal placements. The goal of this optimization work is to determine the possible locations of k-controllers finding a Pareto optimal set of the whole search space [34]. Pareto frontier is the set of all Pareto optimal solutions. This work focuses on minimizing latency of node to the controller and minimizing controller Imbalance.

Minimum controller number (k) determined at least to accommodate flow request (R_j) arrived at the controller of capacity Q.

$$k = min\left(\sum_{i \epsilon V} c_i\right) = ceil\left(min\left(\frac{\sum_j R_j\,(1+\alpha)}{Q}\right)\right) \quad (3)$$

Two scenarios of capacity with backup capacity ($\alpha = 0.3$) and without backup capacity ($\alpha = 0$) in equation 3 to determine the minimum number of the controller are considered.

A combination placement of $P \epsilon 2^V$ and a distance matrix D of shortest path latency between nodes i and j i.e., $d_{i,j}$ is assumed.

$$\text{avg latency} = \frac{1}{|\mathrm{V}|}\sum_{\mathrm{vV}}\left(\begin{array}{c}\min \\ \mathrm{pP}\end{array}\ \mathrm{d}_{\mathrm{v,p}}\right) \quad (4)$$

The average latency node to controller defined in equation 6 and controller Imbalance for P placements of C_i, total flow requests assigned to controller i defined in equation 6. Controller Imbalance measures how the flow requests from node to controller assignment load are discriminated in all P placements. It is determined from Jain's fairness index as shown in equation 5.

$$Fairness\ \text{index} = \max\left(\frac{\left(\sum_{\mathrm{i}=1}^{\mathrm{k}} \mathrm{C_i}\right)^2}{\mathrm{k}*\sum_{\mathrm{i}=1}^{\mathrm{k}} (\mathrm{C_i}^2)}\right) \quad (5)$$

Where fairness Index is discussed in section **??**.

$$\text{coontrollerImbalance} = \min\left(1 - \left(\frac{\left(\sum_{\mathrm{i}=1}^{\mathrm{k}} \mathrm{C_i}\right)^2}{\mathrm{k}*\sum_{\mathrm{i}=1}^{\mathrm{k}} (\mathrm{C_i}^2)}\right)\right) \quad (6)$$

Even though our optimization criteria are based on average latency and controller imbalance furthermore parameters such

as maximum latency of node to the controller for worst-case analysis, and average and maximum controller to controller latency are illustrated in performance evaluation.

C. Constraints

To increase the ability of the network to react to events quickly that lead to robust topology, even when failures occur constraints that determine the performance of the SDN networks are considered. These constraints are controller capacity, candidate controller places at nodes, node weights, latency, and controller Imbalance.

1) At least a number of network controllers for SD-WAN should be able to handle all requests;

$$\sum_{i \epsilon V} c_i \ \geq \ \frac{\sum_{j=1}^{n} R_j}{Q} \tag{7}$$

2) All nodes/switches/forwarding devices should be assigned to controllers while considering:
   a) Each node is controlled by one and only one controller

$$\sum_{i \epsilon V} s_{ij} = 1, \ \forall \ j \epsilon \ V \tag{8}$$

   b) A node is controlled by an existing controller

$$s_{ij} \leq \ c_i, \quad \forall \ i, j \epsilon \ V \tag{9}$$

   c) Nodes are always assigned to controllers placed on them

$$c_i \leq \ s_{ii}, \quad \forall i \ \epsilon \ V \tag{10}$$

3) Fair load distribution among all controllers

For load distribution among all controllers the capacity Q_i and the load of the switches as the request demand of the switches/nodesR_i arrive at controller;

i) The load assigned to controller cannot exceed controller capacity Q_i where α is backup capacity.

$$\sum_{j=1}^{n} R_j s_{ij} \leq (1 - \alpha) Q_i c_i, \ \forall \ i \epsilon \ V \tag{11}$$

ii) At least the capacity of the controller should not exceed

$$\sum_{i \epsilon V} \ R_{ij} \ \leq Q_i, \tag{12}$$

4. Propagation latency should be bounded to service requirements
5. Decision variables are only binary values

D. Assumptions

The optimum controller placements, addressed in different authors and research works considered different assumptions. In this work, various assumption from different perspectives are considered.

(i) Each node of the switches is the candidate of controllers' placement as it simplifies the deployment and infrastructure costs.
(ii) The controller manages k switches each switch has at least a new arrival flow. When the new flow arrives, switches will generate a "Packet-in" message to the controller, then the controller will send the flow entry to the corresponding core switches.
(iii) The processing time of flow requests is equal to propagation time in WAN.
(iv) The capacity of the controller is assumed to be infinite if not specified.

In a flow-based OpenFlow network, only the first packet of flow needs to be sent to the controller. Therefore, only focusing on the new flow set-up requests [45] is sufficient.

E. Pareto Simulated Annealing

Simulated annealing is an unsupervised learning algorithm, which uses probabilistic technique for approximating the global optimum of an objective function. It exploits the analogy between the annealing process and combinatorial optimization problems where molecules variables in the data structure and energy function is the objective function. The temperature is the real value scalar that controls the degree of randomness of the search where high temperature behaves like random search and low temperature like greedy local search [46]. For this work, simulated annealing implemented clustering the topology taking the internal criterion as minimizing a total within the cluster distance.

In this thesis, the Simulated Annealing system of the structure developed in [11] is adopted according to the flow chart in Figure 2 frameworks. The input parameters to PSA heuristics are from two sides. One side is network topology graph G (V, E) and the k minimum number of controllers that capable to handle flow requests from n nodes, and the other side are parameters of PSA such as the number of iterations per temperature level m, initial temperature T_0, annealing schedule, and a number of placements to evaluate during each iterations s. initially, a set of S of s random placements of k controller is generated.

The PSA works as, every placements assigned with random weights and then check whether the assigned node weights exceed the controller capacity. In each procedure, the visited M Pareto frontier placements are updated. The temperature T declines using the algorithm, starting with initial temperature T_0 by decaying factor after every m iterations up to T value is less than 1.

Based on the nature of the problem to be optimized, the state of initial energy S random placements of K controllers assumed. The energy state of the placements system fluctuates

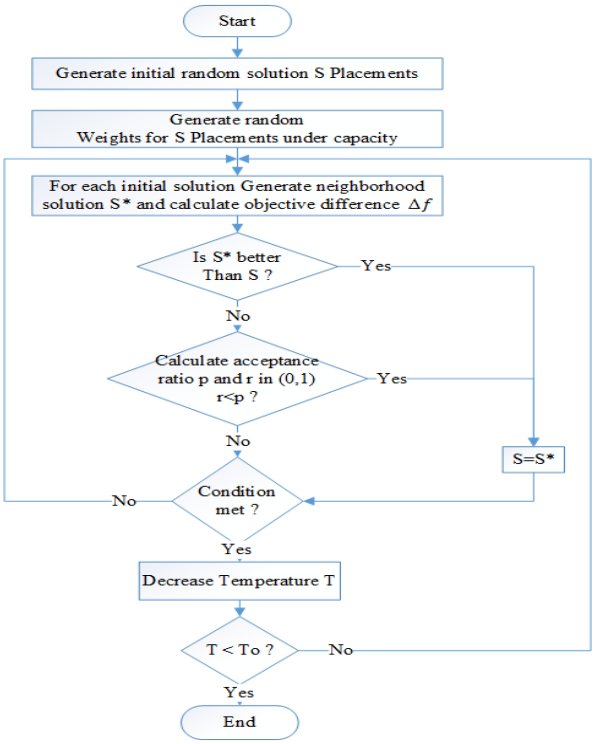


Fig. 2: Flow Chart Simulated Annealing

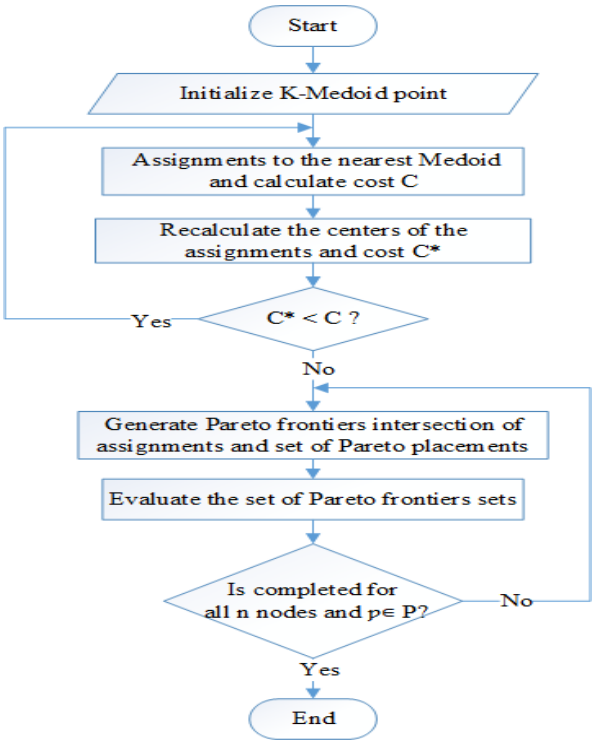


Fig. 3: Capacitated K-Medoid flow chart

around the average energy linked to the temperature. When the equilibrium is reached, the temperature is slightly reduced and again search thermal equilibrium with new temperature. The temperature is reduced until the temperature is less than 1. High initial temperature 50 is selected to reach quickly at the thermal equilibrium and checked trade-off quality of the result and computation time.

F. K-Mediod Algorithm

The heuristic algorithm proposed to approximate the Pareto frontier sets from two objectives (average latency and controller imbalance). For the clustering-based approach of K-Medoid considering the only node to the controller, latency is not sufficient in a limited capacity of controllers. As a result, the K-Medoid clustering algorithm was upgraded with a controller capacity

bound. Controller capacity restricts the flow requests amount assigned to the controller with controller Imbalance and latency constraints. The controller capacity has a direct effect on the value range of controller imbalance obtained from Jain's fairness index as termed in **??** section. Flow chart in Figure 3 adopted from [18], demonstrates the capacitated K-Medoid method. The algorithm inputs distance Matrix D, network size n, controller number k determined from flow request traffic tm, controller capacity Q, and imbalance constraint introduced via ρ. Instead of node assignment is nearest to controller, minimal latency and maximal fairness index assignment are determined upper bounded by capacity.

The parameter ρ iteratively increased resulting in maximum controller imbalance that satisfies capacity constraints. Normalized ρ controller imbalance to bring the impact of controller capacity to the clustering method used in the K-Medoid algorithm. Controller Imbalance of load assigned to each controller not exceeding controller capacity filtered and defined as rhoImbalance that used to normalize ρ as indicated in equation 13. rhoImbalance is the capacity filtered controller Imbalance placements as indicated in equation 16.

$$\rho{=}Imbalance{*}\frac{200}{k} \tag{13}$$

A minimum controller Imbalance only may not guarantee the capacity-limited assignment requirements. Based on the capacity requirements load balance assignments matching a cost minimal constructed to enforce the capacity limit of controllers replicating each controller $\left(\left\lceil\frac{n}{k}\right\rceil+\rho\right)$ times as adapted from [18] in the semantic of controller load discrimination.

The flow chart indicated in the first loop is identical with second loop that is calculating assignments given centers and calculating centers given clusters until the latency convergence is reached. Lastly, results of the cluster C and the assignments of each node to the centers evaluated with C* and its assignments.

For every value of ρ, algorithm calculates placement that minimizes average node to controller latency respecting imbalances and a capacity constraint imposed by ρ. The goal of this work is to use the customized algorithm in [18] and providing insights and analysis of metrics trade-off comparatively the

placements obtained without using heuristics and with both heuristics discussed in this section.

Pareto frontiers obtained from determining Pareto optimal placements assessing the elements in all sets of S random placements meet capacity requirements with respect to the latency (node to controller) and controller imbalance imposed by ρ.

G. Node request estimation

Simple topology is created on Mininet virtual network and packet length of packet_in message captured with Wireshark. POX controller is used to capturing request flows and 10TB of data sent from host 1 to host 2 for the emulation.

The bandwidth capacity between the switches and controller is not infinite. As a result, it hinders the rate at which flows are set up, as only hundreds of thousands of flows per second are estimated for core network switches or routers even millions of flows per second for carrier networks [47], [48]. The alternative way for flow estimation is based on a telecom operator that uses bandwidth capacity at link 10Gb/s the request flow estimated in this work. Figure 4 briefly shows it. Based on this foundation, an average of 213,000 flows per second computed based on Equations 14 and 15.

$$Total\ Flows = \frac{Link\ Bandwidth}{Packet\ length} \tag{14}$$

$$Average\ flows = \frac{\sum_{i=1}^{n} \lambda_i}{n} \tag{15}$$

This average flows is estimated for all nodes in the range of 150k-350k flows per second. The Packet_in message packet length was obtained using Mininet emulation depicted in the plot of packet length in Bytes against time sequence in Figure 4.

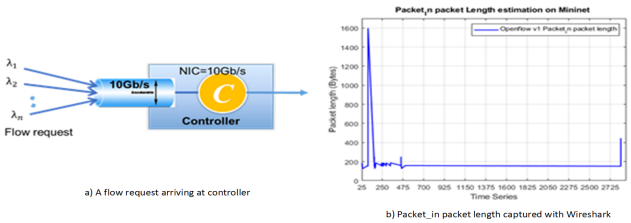


Fig. 4: Flow Request estimation Model [45]

H. Evaluation Methods

The models were implemented with the aim of determining the optimum controller placements in which nodes assignment do not exceed controller capacity with the highest fair load distribution and minimum average node to controller latency. Two scenarios considered in determining controller number. The first scenario failure free case and 100% controller utilization. Second scenario is controller failure case where a reserved capacity of 30% is assumed to increase network reliability and performance.

The performance evaluation consists of different placement scenarios without controller capacity, with controller capacity failure free and backup capacity failure case.Every evaluation is carried out on the real network topology of the ethio telecom WAN IP core network. The parameters trade-offs analysis of the set of Pareto frontiers achieved is performed based on pair of the three below listed cases.

(i) Placement without optimizations
(ii) Placements optimization with Simulated Annealing
(iii) Placements optimization using K-Medoid cases.

Finally, the three cases summarized and analyzed their Pareto placements obtained in each case comparatively. the SDN use case deployments based on the investigation observed is suggested. The controller Imbalance of each placement of load assignments that meets the capacity requirement is filtered out based on equation 16.

$$\begin{cases} 1, & where\ C_i - Q > 0 \\ controller Imbalance, & otherwise \end{cases} \tag{16}$$

Where , C_i $load\ of\ controller$
Q controller Capacity

I. Evaluation Setup

The network topology used in the case study is the IP core WAN network of ethio telecom trasformed to suitable .topo.mat format as shown in Figure 5.

The graphical description provides sufficient and necessary information to build up test-bed networks to emulate real-world topologies.

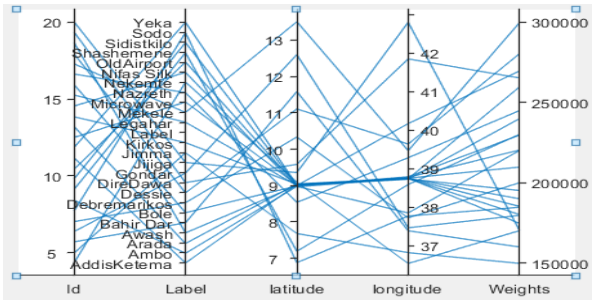


Fig. 5: Topology Data of IP Core WAN Network

The network topology contain twenty-four provider's edges (PE) and 47 connected edges in flat topology.

Mininet version 2.3.0 installed on Ubuntu 20.04.2 LTS on VMware 15.05 pro virtual network used to setup Simple default Mininet topology to estimate flow requests using the POX controller. All algorithms and POCO framework run on Matlab 2020b version on Core i7 Intel Processor, 8GB RAM, 500GB Storage on Windows 10 Operating System, 64-bit Laptop installed with simulation tools indicated in Figure 6.

V. Result Discussion and Analysis

In this section, the result discussion and analysis, presented in two sub-sections. The first part is the impact of controller capacity on controller placement metrics, where the various values of the metrics with and without considering controller

Fig. 6: Simulation Tools

capacity is discussed. In the second part the finding techniques of optimum controller placements with heuristics methods integrated with generic POCO framework tool with and without controller capacity is analyzed in detail.

A. Controller Capacity impact Assessment

In this sub- section the controller capacity impact on the controller placement metrics is investigated. Specifically, how the controller imbalance and latency values are affected with the controller capacity in controller placement. Then indirect impact of controller capacity investigation of how the load distributed among the controllers with the controller imbalance values was done. Similarly, the controller capacity impact on latency and the Load distribution versus latency is observed.

1 Controller Imbalance impact on Controller load distribution

The Controller Load on each controllers required to accommodate the flow request from nodes assigned shown in the Y-axis while the Controller Imbalance in the x-axis shown in figure 7 that intended to show how the load is distributed along the controller imbalance. The figure 7(a) shows when the controller placement in failure free case whereas the right side 7(b) is the case of backup capacity is considered for failure case. As shown in the legend of the graph, minimum number of controllers is computed by dividing total flow request to controller capacity and adding backup capacity if it is likely to be considered according to equation 3. The controller capacity impact on controller imbalance is that it filters the lower imbalances that meet capacity requirements. The controller capacity caused by the capacity bound implicitly limits the range of the values controller imbalance can attain as indicated in Figure 7. In the scenario of failure free (no backup capacity) the minimum number of controller is three as indicated in Figure 7(a) showing load distribution among the three controllers and at controller imbalance of 0.09 or 9% the load on C2 (Controller 2) surpass the controller capacity limit i.e., 2.3 Mflows/s while C3 (Controller 3) goes below 1 Mflows/s. Likewise in the case of backup capacity consideration shown in 7(b) the minimum number of controllers required determined is four (4) and the load distribution among the controllers shown. As indicated in (b) of the 7 the C2 (Controller 2) load assignment surpass the controller capacity limit at about 0.28 (28%) while the other C4(Controller 4) goes below 0.5 M flows/s. The fair distribution of load among controllers increased as depicted in Figure 7 in lower values of controller imbalance in both scenarios. This could proves also that the controller imbalance derived from Jain's fairness index is a good measurement for load balance.

As it can be seen from the graph load distribution among the controllers, at lower controller imbalance highly fairly distributed which increases the controller utilization efficiency . As a result better performance and resource utilization were obtained with minimum controller number with controller imbalance minimization.

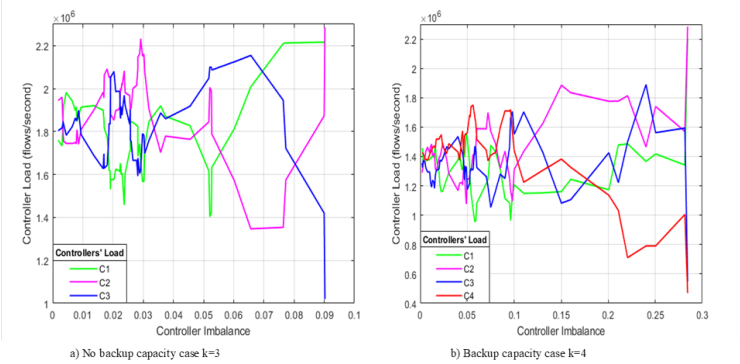


Fig. 7: Controller Imbalance effect on Load distribution

As indicated in Figure 7: Controller Imbalance effect on Load distribution, minimizing the load imbalance can avoid controllers overload assignments. This approach minimizes the number of controllers required for the network topology through load balance by distributing the load fairly to each controller.

2 Controller Capacity Impact on Latency

This part addressed how the average latency and maximum latency node to controller is affected in controller placement with the constraint of controller capacity. In the graph 8 the y-axis shows the change of latency while the x-axis controller Imbalance. The controller Imbalance range that satisfy the controller capacity limit is determined in previous section. The figure is used to show the controller capacity impact on latency graph shown in 8 at lower controller Imbalance less than 0.09 (9%) both average latency and maximum latency never affected. This shows that for load distribution on each controller never surpassed controller capacity, no latency change observed and for any load distribution surpassing controller capacity the change of latency observed. The red dotted distribution shows the average latency change while the blue indicates the maximum latency change. The positive change shows the improvement in latency mean that the latency value decreased while the negative values indicates the latency value increased.

Controller capacity consideration will improve the maximum latency node to the controller with about in average 8.5 % or 9.32 milliseconds in a total maximum latency of nodes. However, the average latency changes distribution is both positive and negative and improves very slightly with 0.33% in average in case k is 3 as shown in Figure 10. These are because

of reassignment of nodes (adding or removing of nodes) in the case of overloaded and under-loaded controllers. The difference of latency change distribution indicates that lower latency is preferred in node assignments (it is clear average latency has a lower value than maximum). No influence investigated of the load assignment within the controller capacity constraint on both the average and maximum controller to controller latency. This indicated that better latency can be obtained using controller capacity.

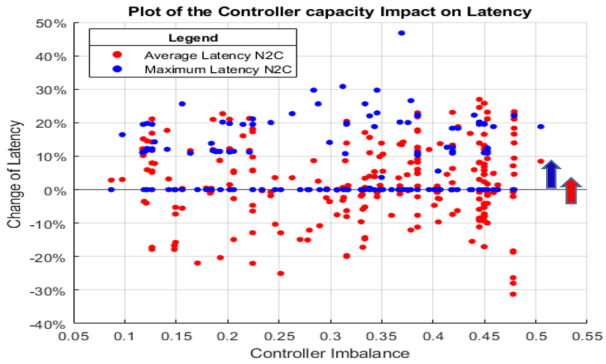


Fig. 8: Controller Imbalance Versus change of latency

1) Latency impact on load distribution

This section of the work wanted to address whether the latency could affect the load distribution among the controllers. The x-axis shows the controller load assigned to each controller while the y-axis shows the latency values each load is assigned to controller. As observed in Figure 9, at very low and very high average and maximum latency of a node to a controller, imbalances increase i.e., the controller capacity limitation might not meet. This investigation leads us to conclude that only minimization of latency might not result in the optimal placements for controllers under any evaluation circumstances as discussed in Evaluation Methods. This thesis investigated that focusing only on latency minimization might overload some controller(s) where nodes are densely populated like a big city and other nodes are underutilized especially where nodes are sparsely distributed. As a result load imbalances increases which negatively affect the performance and reliability of the network. In both maximum and average node to controller latency, similar conclusions can be derived.The network topology used in this case study is a good example of such topologies where nodes in Addis Ababa City are with hundreds of meters to a few kilometers.

B. *Placement Optimization results Analysis*

As discussed in the evaluation methods section, the optimization evaluation is performed with different cases, where controller backup capacity is considered and did not considered. Based on the two scenarios of Figure 10 and 12shows the number of controller and its placements. The two cases analysis considered in this work as explained in detail in next part of this section.

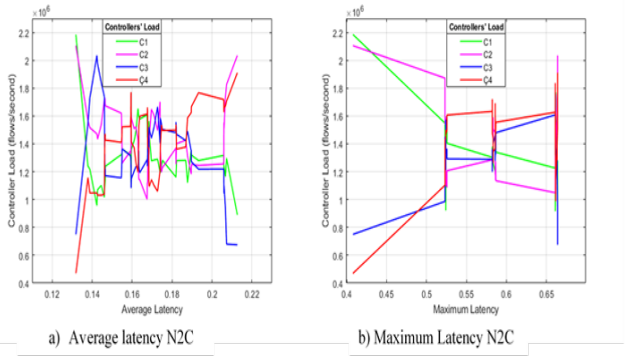


Fig. 9: Average and Maximum node to controller Latency effect on load distribution

Case 1: A 100% controller utilization with failure-free For the determination of controller number, while assuming

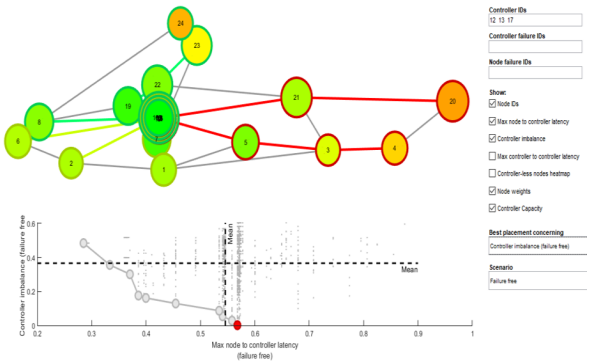
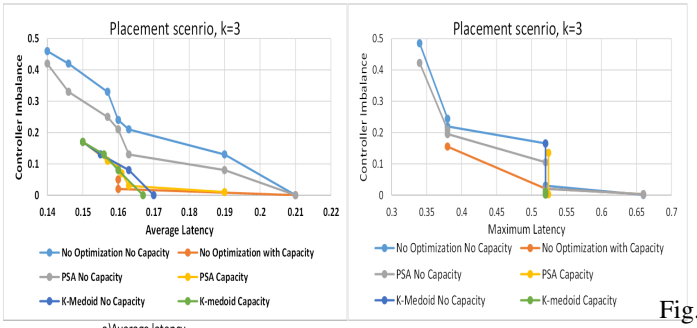


Fig. 10: Controller Placements scenario without backup capacity

failure-free case and 100% controller utilization, 3 controllers obtained based on equation 3. In this case, optimum controller placement keeping the controller number at the minimum that can handle all node requests by minimizing node to controller latency and controller imbalance. Minimizing these metrics under the constraints of controller capacity and without controller capacity is investigated in this work. According to our result analysis considering controller capacity as a constraint showed a great difference in better load balance for the same average latency and maximum latency placements. As discussed earlier the performance for average latency is better than maximum latency because of load assignments in minimizing latency based on the POCO framework. Even though the capacity and with no capacity case converges to the same point, capacity-based results shows better-optimized placements in load balance at the same node to controller latency. The controller placements in a graph 11 with controller capacity observed is smaller in value (shorter line) for all possible placements

The trade-off observed between load balance and latency is inversely proportional as indicated in Figure 11 and Figure 13. The figures showed the trade-off analysis of controller



a)Average latency a) Maximum latency

Fig. 11: Pareto frontier sets Placements scenario of with out backup capacity

Pareto frontier placement sets using three different cases with the capacity-based scenario and no capacity scenario. The heuristic methods used for optimizing placements were analyzed similarly. In this analysis K-Medoid generated better Pareto frontier sets in general though at some points Pareto Simulated method showed better placements in average latency. Another interesting is that capacity-based controller placement without heuristics showed better performance than heuristics methods optimization of no capacity consideration. The result investigations of this case is shown in Figure 11: Pareto frontier sets Placements scenario of no backup capacity.

Case 2: A backup capacity of 30% for each controller where the controllers are operating in load sharing and standby pool mode The controller number while assuming 30% backup

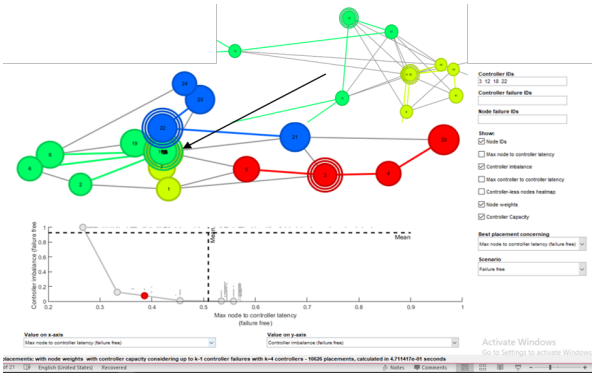
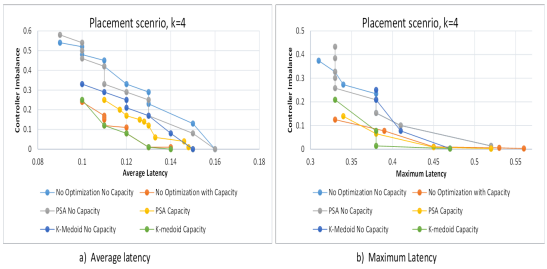


Fig. 12: Controller Placements scenario with backup capacity

capacity, four (4) controllers obtained based on equation 3. In this case, the goal is to keep the controller number at the minimum that can handle all node requests by minimizing node to controller latency and controller imbalance. Minimizing these metrics under the constraints of controller capacity and without controller capacity is investigated in this work. According to our result analysis considering controller capacity as a constraint showed a great difference improvement in load balance for the same average and maximum latency sets of placements. Similarly, the two heuristic approaches used with capacity constraints showed better fair load distribution than heuristics



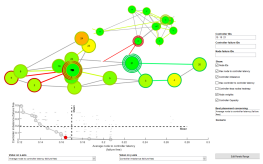
a) Average latency b) Maximum Latency

Fig. 13: Pareto frontier sets Placements scenario of with backup capacity

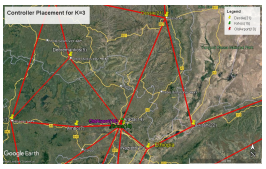
without capacity consideration. Even though the capacity and with no capacity case converges to the same point, capacity-based results better-optimized placements in load balance at the same node to controller latency. The trade-off observed between load balance and latency is inversely proportional as indicated in Figure 11 and Figure 13. The figures showed the trade-off analysis of controller Pareto frontier placement sets using three different cases with the capacity-based scenario and no capacity scenario. The heuristic methods used for optimizing placements investigation indicated K-Medoid generated better Pareto frontier sets in general though at some points Pareto Simulated method showed better placements.

The capacity-based generic POCO framework optimization showed better network performance metrics than the heuristic methods used without capacity-based. Figure 13 showed the comparative analysis of controller placements using average latency and maximum latency with controller imbalance. The PSA heuristics optimization results in a very slight improvement than capacity-based without heuristics in maximum latency case while it results in poor performance than capacity-based without heuristics. This is because the accuracy of the heuristic methods lower than the generic exhaustive method but shows better computation time.

To recap, in our topology case study and works, the K-Medoid results showed better performance in all cases and scenarios. Specifically this study improves 32% to 90% in load balance at the same latency. To recap, in the topology case study and works, the K-Medoid results showed better performance in all cases and scenarios. Specifically this study result showed a significant improvement of 32% to 90% in load balance taking the same latency as a reference while 12.5% to 29% node to controller latency taking the same controller imbalance as a reference. The optimum placement could be decided by the service providers and telecom operators from a set of Pareto frontier placements that meet their minimum service requirements and Quality of Service (QoS). However,it is also possible to determine the optimum controller placement. For this case study of Ethio telecom WAN IP Core topology, the optum controller placement identified as shown in figure 14 for the scenario of no backup controller capacity case and figure 15 for backup controller capacity case. The figure shown in 14 a) is the placement generated using POCO



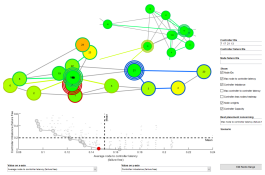
(a) Topology from POCO



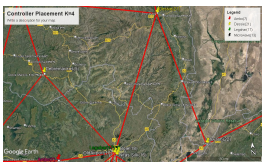
(b) Topology on Google Earth

Fig. 14: Optimum controller placement for k=3

framework Matlab based while b) is the optimum controller placement location on google earth. The node ID is the number in the parenthesis of location names as indicated on google earth place labels in both placement scenario. The node ID is used for convenience on POCO framework. The controller placement location shown on the google earth identified with different color that are described in the legend shown on the top right side of 14 (b) and 15 (b). The controller placement location shown on the POCO framework with triple circle lines while the node to controller assignment is shown with line color for link and node circle color of the same color with the color of controller placement circled on node ID of controller candidate. The optimum controller placement obtained with K-Medoid heuristics is at node name (ID) are Kirkos(15), Old Airport(10) and Dessie1(21) for no backup capacity with 0.58ms average node to controller latency and 0.01 (1%) controller Imbalance as shown in figure 14. Similarly, for the case of backup capacity controllers placed at node names (IDs) are Ambo (7), Legehar (17),Dessie (21), and Microwave (13) with 0.53ms average node to controller latency and 0.0023(0.23%) Controller Imbalance as shown in figure 15.



(a) Topology from POCO



(b) Topology on Google Earth

Fig. 15: Optimum controller placement for k=4

VI. Conclusion

The SD-WAN controller placement optimization affected with flow traffic request and controller capacity. The number of the controller is determined from the capacity of the controller and flow requests arriving at the controller from each node.This method solved the trivial way of determining controllers number for a specified network topology. For this case study,in failure free case three controllers and in failure case four controllers determined. This work also revealed that only minimization of node to controller latency might not result to the optimal placements of controller. At lower values of controller imbalance better load distribution indicating Jain's fairness index is good metric for measuring load balance in the circumstances of varying flow request traffic from nodes.

The placements for the determined controller number based on capacity showed better improvements in network performance evaluation. The heuristics methods showed better results even though simulated annealing with capacity-based showed worse than capacity-based without heuristics. The K-Medoid result is better in convergence time and network performance evaluation of higher fair load distribution. Furthermore, controller capacity considerations improved the controller placements sets filtered from all possible combinatorial placements and with quick execution time. The study result showed that the load balance improvements for all methods gained from 32% to 90% . At last, not only do controller imbalances and global latency minimization guarantee optimal controllers placements but also load assigned to the controller should not exceed its capacity. Anyone interested in the aims of this work can further work on it resembling the node request estimation on real topology, verifying the performance of the heuristics using Mininet emulation, and extend to other SDN use cases and network functions virtualization.

References

[1] L. Doyle. (2021 (Accessed on August 15,2021)) 7 key sd-wan trends to evaluate in 2021. [Online]. Available: https://searchnetworking.techtarget.com/tip/7-key-SD-WAN-trends-to-evaluate/

[2] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[3] S. Khorsandroo, A. G. Sanchez, A. S. Tosun, J. M. A. Rodríguez, and R. Doriguzzi-Corin, "Hybrid sdn evolution: A comprehensive survey of the state-of-the-art," *Computer Networks*, p. 107981, 2021.

[4] J. C. C. Chica, J. C. Imbachi, and J. F. B. Vega, "Security in sdn: A comprehensive survey," *Journal of Network and Computer Applications*, vol. 159, p. 102595, 2020.

[5] S. Scott-Hayward, "Design and deployment of secure, robust, and resilient sdn controllers," in *Proceedings of the 2015 1st IEEE conference on network Softwarization (NetSoft)*. IEEE, 2015, pp. 1–5.

[6] S. Ahmad and A. H. Mir, "Scalability, consistency, reliability and security in sdn controllers: A survey of diverse sdn controllers," *Journal of Network and Systems Management*, vol. 29, no. 1, pp. 1–59, 2021.

[7] J. H. Shaffer, *The effects of high bandwidth networks on wide-area distributed systems*. University of Pennsylvania, 1996.

[8] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," *ACM SIGCOMM Computer Communication Review*, vol. 42, p. 473–478, 2012.

[9] Z. Markos, "Optimal placement of controllers for the adoption of software defined networking: In the case of ethiotelecom," Addis Ababa Unversity, Tech. Rep., Nov. 2018, an optional note.

[10] Y. Hu, T. Luo, W. Wang, and C. Deng, "On the load balanced controller placement problem in software defined networks," in *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*. 2016 2nd IEEE International Conference on Computer and Communications (ICCC), 2016, p. 2430–2434.

[11] 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, p. 4–17, 2015.

[12] G. Yao, J. Bi, Y. Li, and L. Guo, "On the capacitated controller placement problem in software defined networks," *IEEE Communications Letters*, vol. 18, p. 1339–1342, 2014.

[13] D.-M. W. C. Rajendra K. Jain and W. R. Hawe, "A quantitative measure of fairnesss and discrimination for resource allocation in shared computer system," Eastern Research Lab, Digital Equipment Corporation 77 Reed Road Hudson MA 01749, Tech. Rep. 2, 9 1984, fairness Index of resource allocation defined which could be used for controller allocation.

[14] S. Peak. (2021[Accessed on August 21, 2021]) Sdn-explained what is sdn ? [Online]. Available: https://www.silver-peak.com/sd-wan/sd-wan-explained

[15] B. Zhang, X. Wang, and M. Huang, "Multi-objective optimization controller placement problem in internet-oriented software defined network," *Computer Communications*, vol. 123, pp. 24 – 35, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366416307241

[16] 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 *Proceedings of the 2013 25th International Teletraffic Congress (ITC)*, 2013, pp. 1–9.

[17] D. Hock, S. Gebert, M. Hartmann, T. Zinner, and P. Tran-Gia, "Poco-framework for pareto-optimal resilient controller placement in sdn-based core networks," in *2014 IEEE Network Operations and Management Symposium (NOMS)*. IEEE, 2014, pp. 1–2.

[18] S. Lange, S. Gebert, J. Spoerhase, P. Rygielski, T. Zinner, S. Kounev, and P. Tran-Gia, "Specialized heuristics for the controller placement problem in large scale sdn networks," in *2015 27th International Teletraffic Congress*. IEEE, 2015, pp. 210–218.

[19] S. I. Harned, "Poco-moea: Using evolutionary algorithms to solve the controller placement problem," 2016.

[20] G. Schütz and J. Martins, "A comprehensive approach for optimizing controller placement in software-defined networks," *Computer Communications*, 2020.

[21] M. He, A. Basta, A. Blenk, and W. Kellerer, "Modeling flow setup time for controller placement in sdn: Evaluation for dynamic flows," in *2017 IEEE International Conference on Communications (ICC)*. 2017 IEEE International Conference on Communications (ICC), 2017, p. 1–7.

[22] X. Hou, W. Muqing, L. Bo, and L. Yifeng, "Multi-controller deployment algorithm in hierarchical architecture for sdwan," *IEEE Access*, vol. 7, p. 65839–65851, 2019.

[23] M. Khorramizadeh and V. Ahmadi, "Capacity and load-aware software-defined network controller placement in heterogeneous environments," *Computer Communications*, vol. 129, p. 226–247, 2018.

[24] Y. Hu, T. Luo, N. C. Beaulieu, and C. Deng, "The energy-aware controller placement problem in software defined networks," *IEEE Communications Letters*, vol. 21, no. 4, pp. 741–744, 2017.

[25] A. A. Neghabi, N. J. Navimipour, M. Hosseinzadeh, and A. Rezaee, "Energy-aware dynamic-link load balancing method for a software-defined network using a multi-objective artificial bee colony algorithm and genetic operators," *IET Communications*, vol. 14, no. 18, pp. 3284–3293, 2020.

[26] A. Ruiz-Rivera, K.-W. Chin, and S. Soh, "GreCo: An energy aware controller association algorithm for software defined networks," *IEEE Communications Letters*, vol. 19, no. 4, pp. 541–544, apr 2015.

[27] K. Kurroliya, S. Mohanty, B. Sahoo, and K. Kanodia, "Minimizing energy consumption in software defined networks," in *2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, 2020, pp. 885–890.

[28] J. Galán-Jiménez, "Minimization of energy consumption in ip/sdn hybrid networks using genetic algorithms," in *2017 Sustainable Internet and ICT for Sustainability (SustainIT)*, 2017, pp. 1–5.

[29] D. B. Rawat and C. Bajracharya, "Software defined networking for reducing energy consumption and carbon emission," in *SoutheastCon 2016*. IEEE, 2016, pp. 1–2.

[30] L. F. Muller, R. R. Oliveira, M. C. Luizelli, L. P. Gaspary, and M. P. Barcellos, "Survivor: An enhanced controller placement strategy for improving SDN survivability," in *2014 IEEE Global Communications Conference*. IEEE, dec 2014.

[31] A. Jalili, V. Ahmadi, M. Keshtgari, and M. Kazemi, "Controller placement in software-defined wan using multi objective genetic algorithm," in *2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, 2015, pp. 656–662.

[32] A. Jalili, M. Keshtgari, R. Akbari, and R. Javidan, "Multi criteria analysis of controller placement problem in software defined networks," *Computer Communications*, vol. 133, pp. 115–128, 2019.

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

[34] V. Ahmadi and M. Khorramizadeh, "An adaptive heuristic for multi-objective controller placement in software-defined networks," *Computers Electrical Engineering*, vol. 66, p. 204–228, 2018.

[35] V. Ahmadi, A. Jalili, and M. Khorramizadeh, "Multi-objective controller placement problem: issues and solution by heuristics," *International Journal of Computer Science and Information Security*, vol. 14, no. 8, p. 543, 2016.

[36] S. Auroux, "Flow processing-aware control application placement," Ph.D. dissertation, Paderborn, Universität Paderborn, 2017.

[37] Z. Li, Y. Hu, T. Hu, and P. Wei, "Dynamic sdn controller association mechanism based on flow characteristics," *IEEE Access*, vol. 7, pp. 92 661–92 671, 2019.

[38] H. Xue, K. T. Kim, and H. Y. Youn, "Dynamic load balancing of software-defined networking based on genetic-ant colony optimization," *Sensors*, vol. 19, no. 2, p. 311, 2019.

[39] X. Huang, S. Bian, Z. Shao, and H. Xu, "Dynamic switch-controller association and control devolution for sdn systems," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1–6.

[40] T. Yuan, X. Huang, M. Ma, and J. Yuan, "Balance-based sdn controller placement and assignment with minimum weight matching," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–6.

[41] H. Ren, X. Li, J. Geng, and J. Yan, "A sdn-based dynamic traffic scheduling algorithm," in *2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, 2016, pp. 514–518.

[42] N. Cai, Y. Han, Y. Ben, W. An, and Z. Xu, "An effective load balanced controller placement approach in software-defined WANs," in *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*. IEEE, nov 2019.

[43] V. Huang, G. Chen, P. Zhang, H. Li, C. Hu, T. Pan, and Q. Fu, "A scalable approach to sdn control plane management: High utilization comes with low latency," *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 682–695, 2020.

[44] V. Huang, G. Chen, Q. Fu, and E. Wen, "Optimizing controller placement for software-defined networks," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2019, pp. 224–232.

[45] L. Yao, P. Hong, and W. Zhou, "Evaluating the controller capacity in software defined networking," in *2014 23rd International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 08 2014.

[46] D. E. BROWN and C. L. HUNTLEY, "A practical application of simulated annealing to clustering," *Institute for Parallel Computation and Department of Systems Engineering*, vol. 25, pp. 401–412, 1992.

[47] R. Khalili, Z. Despotovic, and A. Hecker, "Flow setup latency in SDN networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 12, pp. 2631–2639, dec 2018.

[48] A. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, and R. Sherwood, "On controller performance in software-defined networks," in *2nd {USENIX} Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE 12)*, 2012, pp. 5–9.