Enhanced SDN Controller Placement and Load

Balancing for Campus Network Optimization

Lohith Kandibanda, Lakshmi Priya P, Joel M Johnsonand

Thangam S.

Department of Computer Science and Engineering, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Bengaluru, India.

Corresponding author(s). E-mail(s): bl.en.u4cse22032@bl.students.amrita.edu; bl.en.u4cse22034@bl.students.amrita.edu; bl.en.u4cse22028@bl.students.amrita.edu; s\_thangam@blr.amrita.edu; These authors contributed equally to this work.

# Abstract

Large campus network environments bring various challenges to SDN, which include the controller placement and load management. The novel SDN controller placement optimization algorithm is proposed to address the network latency systematically while compensating for the load balancing along with the fault tolerance. Using the Ryu controller and Mininet simulation, this approach develops the computational method of determining the optimal locations of controllers at every level of network topology, patterns of traffic, and complexity of computation. Experimental evaluation demonstrates significant improvements over baseline placement strategy results in terms of network-performance metrics performance, decreasing latency by up to 35% and increasing network scalability. The algorithm offers a data-driven framework for strategic controller placement within the complexity of campus network infrastructures.

**Keywords:** SDN, Load Balancing, Controller Placement, QOS, Algorithms

# Introduction

Campus networks are imperative in contemporary universities for the following applications: seamless access to Wi-Fi, transfers of research data, video streaming, among others. Despite this, such networks suffer from an inability to cope with traffic management effectively due to inadequate load balancing and misplacement of the SDN controller. Service quality tends to degrade with high latency or even network congestion. This is primarily because the controllers have been misplaced. Those not only disrupt the experiences by their staff and professors but by their students, too and prevent departmental collaboration because they interfere with access to critical academic resources and detract from the potential innovations of research. This establishes in what regard proper network infrastructure that fits a university situation is needed.

Today, scalable and reliable network solutions are the requirements educational institutions have to scale up to accommodate their expanding size and needs for technology. The implementation of camps with advanced technologies such as IoT devices and smart classrooms increases the demand on network resources, thus emphasizing the need for efficient management. These may negatively influence the academic performance of the students, output of research, and resources access if not well managed; hence, there arises a broader societal problem. Innovative solutions would be required in this regard to ensure resilient, responsive, and high-performing campus networks that meet the changing needs of educational establishments.

This project proposes an entirely new approach for these problems, designed specifically for campus networks and incorporates dynamic load balancing with SDN controller placement optimization. For this, the ideal number and placement of SDN controllers with the aim to maximize response rate and minimize latency, were determined through simulation tools such as Mininet and the K-Median Clustering method. In addition, DWRR was used for load balancing to ensure that the traffic was appropriately spread so that the network was more scalable and efficient. This suggested approach is tailored for the unique characteristics of campus traffic patterns and needs and is different from other generic SDN optimization solutions. This double layer architecture provides an essentially scalable foundation for the next steps in further enhancements of SDN-based campus network management with evident benefits in terms of network efficiency and reliability This work is remarkable as it is very customized in solving unique issues from the school network that should not experience resource access problems in getting its resources accessed smoothly to their consumers, also making its environment academia-integrative.

# Literature Survey

The reviewed literature provides comprehensive exploration of these challenges, in terms of innovation in the light of heuristic frameworks, machine learning techniques, and game-theoretic models. These studies point out that traffic is a dynamic pattern nature and minimize latency, as well as optimize the resource use for a higher Quality of Service. These papers cover the research contributions made regarding diverse strategies such as energy-aware algorithms and adaptive placement of controllers, giving insight and pointing to relevant gaps for building robust, efficient, and sustainable SDN solutions.

This study [1] examines several SDN load balancing strategies and emphasises how the growth of cloud services has increased the demand for effective load control. It highlights the significance of preserving QoS through features like delay, jitter, and packet loss and talks about heuristic-based frameworks for data transmission optimising. While artificial intelligence in SDN is mentioned, nothing is known about how it specifically relates to load balancing. The review points out the lack of precise technique classifications in the literature and suggests further study to improve current approaches. In the literature survey in the [2] research, the difficulties and developments in load balancing for Software-Defined Wide Area Networks (SD-WAN) are reviewed.

This trend of increasing machine learning application in network management has been shown in previous works to introduce load prediction and deep reinforcement learning (DRL) in the optimization of load migration. It must be highlighted that component migration is highly important in balancing controller loads in dynamic systems, especially the CPE devices. However, there is still a lack of research focused on load balancing within SD-WANs involving CPEs. The study fills in this gap with a novel method and also points out the value of queuing theory for simulating controller reaction times and preserving load management efficiency. Focusing on energy-aware controller placement problem (CPP) in Internet of Things (IoT) flows, the research [3] looks at energy management techniques in SDN.

It covers approaches like Giroire et al. using ILP for SDN switch energy reduction, Tsiropoulou et al. game-theoretic model for power optimisation of the transmission. The other approaches are also the ones that focus on the flow rerouting and the switches deactivation in order to reduce energy consumption, like the ones Assefa and Ozkasap's RESDN, and Maaloul et al. heuristic algorithms. Most of the present approaches, however, neglect the dynamic IoT traffic, and thus the placement of controllers turns out to be inefficient. Unlike the traditional approaches, the paper introduces EnPlace-a novel approach for dynamic IoT traffic and in-band control planes which provides better energy savings. The survey emphasizes the need for better energy-efficient techniques that match the evolving requirements of the IoT environments in SDN. This study [4] discusses how multi-controller architectures can overcome the limitations of single-controller systems, such as their vulnerability to failures and their limited processing capacity in high-density networks.

It compares static and dynamic strategies for placing controllers, asserting that static approaches often fail to consider the dynamic nature of the network traffic. Important performance metrics like load balancing, scalability, and latency are examined, with a focus on the need for adaptive approaches. In order to minimize latency and optimize controller placement, it looks at methods such as spectral clustering and particle swarm optimization. However, the paper identifies a lack in the adaptive techniques that respond to real-time network conditions, where it suggests that future works should focus on dynamic algorithms so that SDN becomes highly efficient and reliable. Focusing attention on the importance of best placing the controller and its connections to switches, the work by [5] reviews some strategies for handling multiple controllers in SDNs. It compares the static and dynamic association strategies to conclude that static approaches frequently fail to adapt to changes in network conditions, thus facing performance problems.

The survey discusses initiatives to reduce various types of latency but also highlights inadequacies in network cost and dependability awareness. Furthermore, it utilizes integer linear programming to optimize load distribution and examines network partitioning strategies to enhance fault tolerance. It has been found that reducing controller-switch association overhead is critical for successful deployment. Despite difficulties with real-time implementation, it is stressed that dynamic policies are necessary to make better use of controller resources. The paper introduces novel game theory applications, such as bipartite graph matching and Nash equilibrium, to tackle dynamic association problems, providing new insights into SDN controller management and laying the groundwork for proposed strategies.

The Dynamic Association Strategy (DASM) for SDN multi-controller load balancing is presented in the [6] work. By using a variation of the Hungarian algorithm for re-association and Nash equilibrium for optimal cost, DASM formulates the controllerswitch association problem as a bi-objective optimisation that balances controller loads with association cost minimisation. In the end, this study examines the AI-based load-balancing methodology, emphasising its practical application and improved SDN decision-making skills. It also covers heuristic techniques such as publish/subscribe systems, switch migration methods, and optimisation strategies used for congestion minimisation.

Software-Defined Networking Techniques Notwithstanding this, the [7] report identifies some limitations and potential avenues for future research. Among themIt is important to note that the current methods do not take into account the distribution of application traffic in real-time or the estimation of traffic that could affect the load balancing process. In addition, issues with load balancing, the security of message exchange between nodes via load balancing, and the update of OpenFlow rules must be resolved. Regarding NF placement in SDN virtualisation and cloud services, energy efficiency is another possibly researched area.

Software-defined networking is defined as decoupling the control plane from the data plane to enable programmable, automated, and vendor-agnostic network management. 8 Different types of SDN controllers are covered, such as NOX, POX, Beacon, Floodlight, Ryu, OpenDayLight (ODL), and ONOS; they are qualitatively compared in terms of their features. However, the paper does mention some issues in the control layer: controller placement problem, affecting network responsiveness, fault tolerance, resilience, QoS, and load balancing. The paper also discusses the management plane of SDN, which focuses on fault, configuration, accounting, performance, and security services. Another argument from the paper is that the effect of centralization in SDN can positively affect network security, such as the detection and mitigation of Distributed Denial of Service (DDoS) attacks, among others. [8]

Algorithm using Clustering in Software Defined Networks The research [9] presents an effective controller placement approach in software-defined networks (SDNs) that makes use of clustering algorithms. In order to divide the network into several clusters and determine the ideal number of controllers for each topology using a silhouette score, the authors have employed clustering methods, particularly the K-means++ algorithm. To determine the ideal location for the controllers in each cluster, the authors suggest using the meeting point algorithm. Using POX as the SDN controller, the Mininet emulator is used for the evaluation. The suggested method is contrasted with previous research on throughput, latency, and jitter in six topologies from the Internet Zoo dataset.

In order to get beyond the limitations of bursty traffic and load imbalances in large-scale networks, [10] introduces a novel technique called dSDN. Using conventional SDN, the authors created a reference model and discovered a number of performance problems, such as excessive delay jitter with high network traffic. They have developed an enhanced SDN (eSDN) featuring packet priority processing and a congestion control technique to improve on such performance difficulties. eSDN enhanced performance in mild and moderate traffic scenarios. However, when there was a lot of traffic, static SDN controllers did not react. The authors then suggested dSDN, which maps SDN controllers dynamically in response to changes in load. Thus, dSDN gets rid of load imbalance and thereby decreases delays and jitters. On these large-scale simulation results, dSDN is improved as compared to the SDN and eSDN both in terms of throughput, delay, and jitter that ensures better QoS within huge numbers of homogeneous IoT networks.

The [11] paper provides an overview of load-balancing methods to improve QoS in SDN-based IoT SD-IoT networks. After a brief discussion on the function and significance of SDN and IoT for improving QoS, the architecture and load-balancing problem of SD-IoT have been described, and a number of load-balancing techniques have been examined. This work, which used a systematic literature review strategy to source load-balancing strategies for SD-IoT, examined a number of factors, including response time, delay, resource productivity, throughput, load-balancing rate, loss rate, packet delivery rate, overload, energy consumption, and scalability. There are known research gaps in QoS-based load balancing in SD-IoT networks. It provides absolutely no indication of the constraints of the existing SD-IoT load balancing techniques concerning the contribution component; as a result, it might add value as an extension of the literature survey.

A Survey on Load Balancing, Routing, and Congestion in SDN [12] comprehensively surveys and analyses the existing approaches for load balancing, routing, and congestion control in SDNs. It categorizes and discusses different techniques of load balancing techniques and their features as well as problems. For this purpose, the paper further provides a taxonomy of routing methods based on the adopted techniques. These include QoS-aware, energy-efficient, and security-enhanced routing techniques. The paper describes solutions on network congestion, and flow-based and table-based approaches to control congestion of flows are proposed. The relevant metrics to assess these approaches are: throughput, the degree of load balancing, response time, and delay as main factors taken into account to evaluate load balancing, and the packet loss ratio and delay as major metrics for routing. In addition, recent SDN research trends are discussed in the paper, and among the findings, it is indicated that increasingly work becomes focused on improving network performance and AI-based techniques are used.

The study [13] explores how SDN-based techniques might improve IoT security. It emphasises employing SVMs in the Mininet emulator to identify network threats such as DDoS attacks and TCP and ICMP floods. It has been observed that nonlinear SVMs perform better in recognising anomalies than linear ones. Additionally, the authors classify DDoS mechanisms using attack trees and use entropy-based techniques, including whitelisting, to combat them. The need for creative security solutions to keep up with the IoT’s rapid expansion is emphasised in the debate of ways of identifying Wi-Fi deauthentication and continuous surveillance.

The contribution and challenges of SDN technology toward improving the performance of such networks, especially for Internet of Things networks, are explored in [14] paper. SDN facilitates contemporary networking demands through a comprehensive network view. Inefficiencies in traditional SDN systems cause higher delays and lower performance. Thus, the authors propose a paradigm that enhances the performance of SDN for real-time wireless Internet of Things networks to address such issues. Major breakthroughs include the best placement of the controller, load balancing techniques for dividing the workload, and a delay minimisation method to avoid bottlenecks and thus increase the throughput. These improvements point out the strengths of SDN in completely changing the way IoT networks are managed and demonstrate its supremacy over typically rigid wired networks.

The [15] paper discusses load balancing algorithms in SDN and focuses on rule-based, dynamic, responsive, and static approaches. In addition to reducing congestion, load balancing boosts system performance, scalability, reliability, and utilization of the available resource. Hybrid algorithms are superior compared to standard ones, especially concerning TCP traffic over different network topologies for domains such as intrusion detection and smart railroads. However, there are many problems still: the impact of multi-layered SDN design and network topologies. As a practical solution to improving the efficiency of SDN in data centers, the survey suggests the possibility of hybrid techniques and recommends exploring hierarchical structures with master and sub-controllers.

This is a security performance research [16] that focuses on integrating SDN with OpenStack Cloud. From the identified major gaps, there was no requirement or documented performance specification resulting from lack of vendor verification. Some SDN controllers can sometimes detect DoS attacks, but they hardly consider insider threats and performance implications. The research recommendations indicate using Virtual switches that have IDS capabilities, although they still need testing for latencies. Open v-Switch-based firewalls are more performance-oriented as compared to Linux Bridge firewalls in OpenStack. This is set to yield advantages in constant throughput. In this context, the paper proposes SDN combined with NFV to increase cloud security and dependability, with crucial roles that Service Function Chaining plays in managing massive data centers. Enumerate current weaknesses and give direction to which areas could be improved.

Existing work on SDN optimization lacks focus toward campus networks' specific requirements. The controller placement methods ignore the dynamics of real-time traffic and scalability for diverse environments. Traditional load balancing fails in dealing with dynamic traffic and making decisions in real time. Most of the studies focused either on controller placement or load balancing without an integrated approach toward holistic optimization. Also, direct performance comparison between baseline and optimized SDN setups designed for campus environments is hardly investigated.

This work fills this gap by integrating K-Median clustering for optimal controller placement and dynamic weighted round robin for efficient load balancing, providing a tailored solution that is scalable for campus network optimization.

# Methodology

The work focuses on optimizing campus network performance by using the concepts of controller placement and load balancing within an SDN-based environment. The methodology followed has well-structured steps in the process to materialize the research objectives.

A preliminary setup was established with the design of a baseline campus network topology, which is built using Mininet. A number of host systems are integrated into the topology and have been developed in such a way to represent all three layers - core, distribution, and access. This network was deployed with default controllers and, in addition, without using any SDN-specific optimizations such as placement of a controller or load balancing algorithms. Baseline performance was set up using such metrics like latency, throughput, packet count, byte count, flow duration, and packet rate measured by means of some tool such as iperf, ping, and Open vSwitch commands, for example, ovs-ofctl.

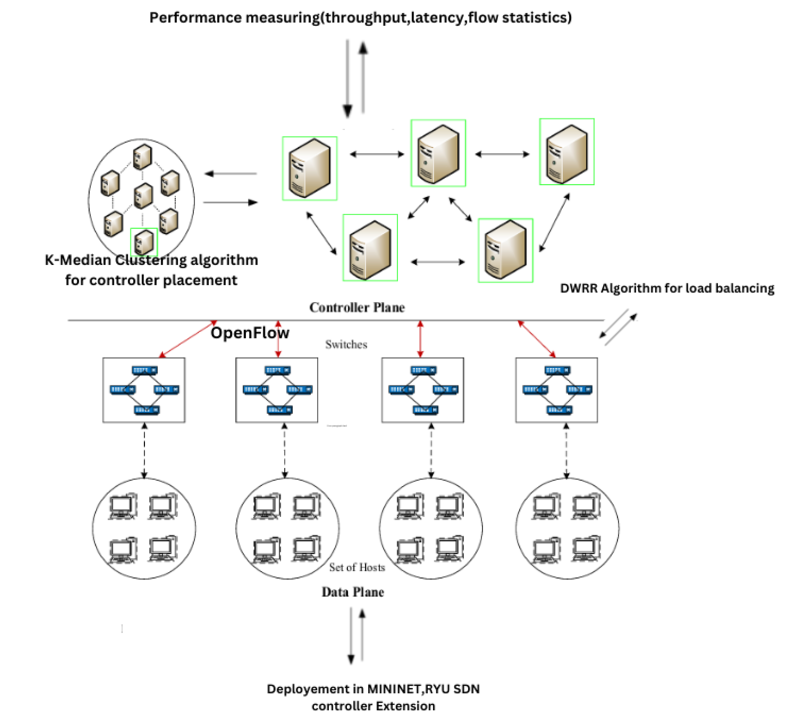


Fig 1: System Architecture

This figure 1 shows an SDN that gathers control and centralized management through an OpenFlow-based controller. The K-Median clustering algorithm is used in the controller plane to optimize centralized controller placement. Meanwhile, the data plane applies DWRR for load balancing, which transmits network traffic across multiple switches to improve network performance.

The optimization of controller placement is achieved with the help of K-Median Clustering algorithm, by optimizing the placement based on the data inputs such as switch locations and traffic load on them. Here, the algorithm has identified the optimal placements for controllers, minimizing communication latency between switches and controllers. By this outcome, the results were given for the improvement of the entire network design.

The campus network topology was then updated based on the optimized placement of the controllers. Controllers were installed based on the location input by the clustering algorithm, and switches were reallocated to the nearest controllers. This allocation assured efficient communication and avoidance of delay in the network.

The next enhancement was improving the performance of the network through a mechanism known as Dynamic Weighted Round-Robin (DWRR) load balancing. It was spread in a dynamic fashion across servers while considering their capacities and weights. A Load-balancing policy was implemented through a controller application by the Ryu controller, thereby preventing bottlenecks and ensuring equitable resource utilization.

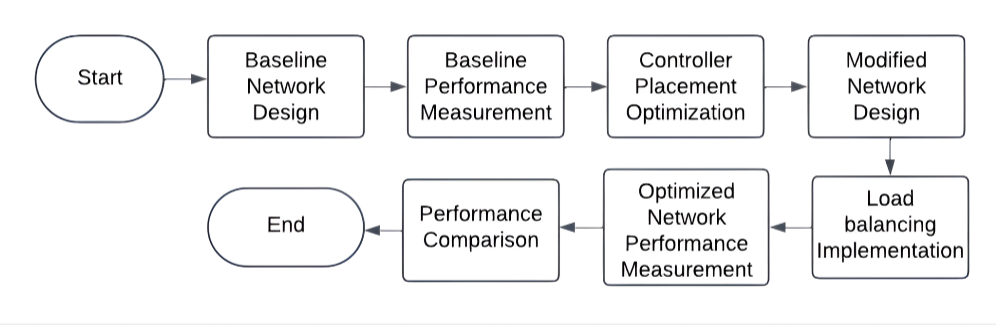


Fig 2: Flow chart

The procedure for optimising Software-Defined Networking (SDN) is shown in this figure 2 flowchart. Prior to optimising controller placement and implementing load balancing, the baseline network architecture and performance measurement are completed. The performance of the optimised network is then assessed and contrasted with the baseline.

Then, the optimized SDN-based network topology was enacted and tested in performance under similar traffic conditions of the baseline network. Metrics such as latency, throughput, packet count, byte count, flow duration, and packet rate were recorded. This led to a direct comparison between the baseline and optimized networks.

Thus, a comparison of performance was made between the baseline network and the optimized SDN-based network. Latency and throughput were evaluated, besides developing graphs to visualize the enhancements. The analysis would clearly depict an optimistic impact on the overall efficiency of the network due to optimized controller placement and load balancing.

All the research was performed with the aid of certain tools and platforms. A simulation platform was developed and tested using Mininet in order to develop the campus network topology. Ryu SDN controller was used to implement Dynamic Weighted Round-Robin load balancing algorithm. A K-Median Clustering algorithm, written in Python, was used for the optimal placement of the controllers. For the measurement of performance, throughput tests were performed using iperf, latency measurements with ping, and ovs-ofctl commands were used for flow statistics.

# Algorithms

## Optimal Controller Placement using K-Median Clustering

**Algorithm1**

1: Initialize *k* cluster centers randomly.

2: **repeat**

3: Assign each switch to its nearest controller.

4: Recalculate cluster centers as the median of assigned switches.

5: **until** Cluster centers converge.

6: **return** Optimal controller placements.

As the K-Median Clustering method [Algorithm 1] can balance load and minimize latency, it has been chosen for the optimization of the placement of controllers. Switches are allocated iteratively to the closest controllers, and cluster centres are recalculated as medians until convergence. This procedure efficiently balances the load of traffic, improves network efficiency and cuts down communication delays. It is therefore the best option to enhance the performance of campus network topologies considering ease of use, computational effectiveness, and flexibility in handling various traffic patterns.

## 4.2 Dynamic Weighted Round Robin (DWRR) Load Balancer

**Algorithm 2** Dynamic Weighted Round Robin (DWRR) Load Balancer

1: **Input:** Network topology with traffic flow data

2: **Output:** Load-balanced traffic paths

3: Collect real-time traffic counts from switches using OpenFlow.

4: **for** each network link **do**

5: Associate weights to each link as: weight = 1 / link utilization + ε

6: where *ε* is a small constant.

7: **end for**

8: **for** each incoming traffic flow **do**

9: Choose the minimum used path according to the dynamic weights.  
10: Install the flow rules in switches according to the chosen path

11: **end for**

12:Periodically repeat step 1-3 due to dynamic changes in traffics.

The DWRR algorithm [Algorithm 2] uses the dynamic assignment of weights to the links based on real-time utilization to evenly distribute network traffic. By OpenFlow, it collects data regarding traffic and calculates weights that are inversely proportional to how much a link is in use with a very small constant ε to correct potential errors. It selects a flow control along the way for each traffic flow based on which is the least-used. It keeps on adjusting to fluctuating traffic periodically in order to ensure balanced loads, reduce congestion, and optimize performance in SDN-based campus networks. Because of its flexibility, it is ideal for scalable and dynamic network systems.

These algorithms, K-Median Clustering and Dynamic Weighted Round Robin, were chosen due to their exceptional performance and versatility in improving SDN-based campus networks. K-Median Clustering focuses on achieving the best possible placement of a controller and reduces latency while balancing communication between controllers and switches with efficient results that go beyond static placement techniques that assume steady network demands. Just like most conventional round-robin or static algorithms that are not adaptive to changes in traffic patterns, DWRR performs excellently in load balancing by adapting real-time weights to links based on their utilization. Scalable, effective, and reliable, such algorithms are able to perform well in complex and dynamic campus contexts in reducing latency, balanced traffic loads, and improved overall network performance..

# Implementation

Implementation of this project falls into the following subsections:  
detailing each step along with the corresponding photos for visualization.

## Baseline Network Setup

The initial campus network topology was designed and executed in Mininet, simulating core, distribution, and access layers. This setup represented a traditional network using default Open vSwitch controllers without SDN-specific optimizations. The performance metrics such as latency, throughput, packet count, and flow statistics were recorded to establish a baseline for comparison.

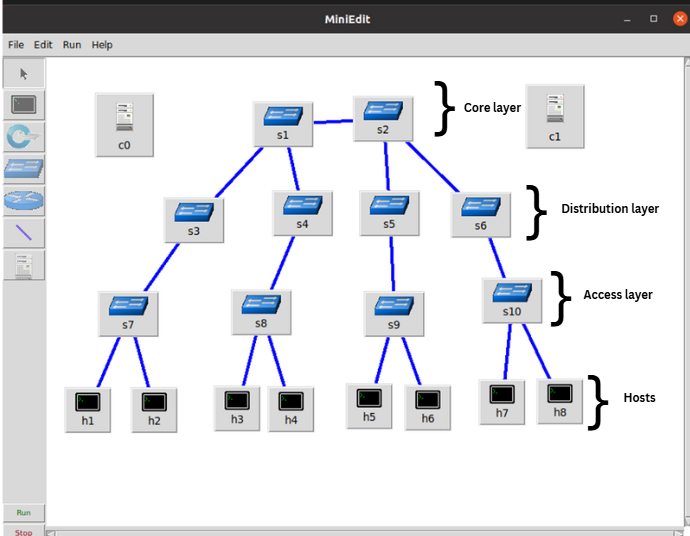


Fig 3: Campus network topology

This diagram figure 3 represents a hierarchical network topology created using MiniEdit. It consists of a core layer (s1, s2), a distribution layer (s3, s4, s5, s6), and an access layer (s7, s8, s9, s10), connecting hosts (h1 to h8). The controller (c0) manages the SDN switches across all layers.

## Optimal Controller Placement Using K-Median Clustering

The K-Median Clustering algorithm has been implemented using Python for optimizing the placement of controllers. Input data for switch locations and traffic load have been fed to the algorithm that assigns the switches to their closest controllers. The algorithm iteratively recalculate cluster centers until optimal placements are decided. These locations served as the bases for modifying the campus topology towards efficiency improvements.

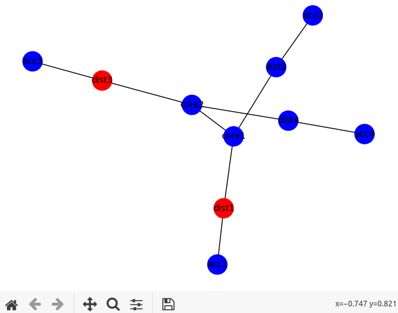


Fig 4: K-Median Clustering algorithm result

This graph figure 4 illustrates the optimal SDN controller placement for a network topology. Nodes in red represent the selected controller locations (dist3 and dist1), while blue nodes represent other network devices. The placement aims to minimize latency and enhance network efficiency.

## Modified Campus Network Topology

The baseline network was then modified to integrate the optimized controller placements that were derived from the K-Median Clustering algorithm. Controllers were strategically positioned at their optimal locations, and switches were reassigned to their nearest controller for efficient communication.

## Dynamic Weighted Round Robin Load Balancing

The Dynamic Weighted Round Robin (DWRR) algorithm was implemented using a Python-based Ryu controller. By using OpenFlow, it was able to collect real-time traffic statistics from the switches and calculate link weights dynamically based on utilization. It sent incoming traffic along paths with the least utilized cases to balance the load properly. The mechanism ensured optimal resource utilization and reduced network congestion.

Loading app dynamic\_load\_balancer.py

Loading app ryu.controller.ofp\_handler

Instantiating app dynamic\_load\_balancer.py of DynamicWeightedLoadBalancer

Instantiating app ryu.controller.ofp\_handler of OFPHandler

Starting the RYU controllers with Dynamic weighted round robin algorithm

This demonstrates how the Ryu SDN controller is to be implemented using the Dynamic Weighted Round Robin (DWRR) load balancing algorithm. To demonstrate how load balancing policies can be implemented across multiple network controllers, the dynamic\_load\_balancer.py script is imported and launched on two distinct TCP ports, namely 6633 and 6634.

## Performance Measurement

The network’s performance was tested using different tools: Latency: Measured using ping command between hosts. Throughput: Measured using iperf between servers and clients. Flow Statistics: Gathered by using ovs-ofctl dump-flows and ovs-ofctl dumpports commands to measure the packet count and byte count along with flow durations. All these metrics were recorded for the baseline and optimized networks under the identical conditions.

When the baseline setup is executed the results are:

mininet> h1 ping -c 10 h2

PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data.

Table 1 : Results for baseline setup execution

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ICMP\_Sequence | Bytes | Source | TTL | Time(ms) |
| 1 | 1 | 64 | 10.0.1.2 | 64 | 3.79 |
| 2 | 2 | 64 | 10.0.1.2 | 64 | 0.087 |
| 3 | 3 | 64 | 10.0.1.2 | 64 | 0.065 |
| 4 | 4 | 64 | 10.0.1.2 | 64 | 0.088 |
| 5 | 5 | 64 | 10.0.1.2 | 64 | 0.05 |
| 6 | 6 | 64 | 10.0.1.2 | 64 | 0.09 |
| 7 | 7 | 64 | 10.0.1.2 | 64 | 0.073 |
| 8 | 8 | 64 | 10.0.1.2 | 64 | 0.059 |
| 9 | 9 | 64 | 10.0.1.2 | 64 | 0.069 |
| 10 | 10 | 64 | 10.0.1.2 | 64 | 0.063 |

This table 1 shows how the standard campus network topology is implemented without any SDN optimizations. Ten switches and numerous hosts make up the topology, and controllers are positioned by default. The ability of host h1 to successfully ping host h2 would verify successful connectivity with an average round-trip time (RTT) of 0.454 ms and no packet loss. This baseline is used to compare performance gains brought about by SDN-based enhancements.

When the Enhanced Campus network is executed the results are:

mininet> h1 ping -c 10 h2

PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data.

Table 2: Results of Enhanced setup execution

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ICMP\_Sequence | Bytes | Source | TTL | Time(ms) |
| 1 | 1 | 64 | 10.0.1.2 | 64 | 0.878 |
| 2 | 2 | 64 | 10.0.1.2 | 64 | 0.063 |
| 3 | 3 | 64 | 10.0.1.2 | 64 | 0.062 |
| 4 | 4 | 64 | 10.0.1.2 | 64 | 0.071 |
| 5 | 5 | 64 | 10.0.1.2 | 64 | 0.043 |
| 6 | 6 | 64 | 10.0.1.2 | 64 | 0.057 |
| 7 | 7 | 64 | 10.0.1.2 | 64 | 0.059 |
| 8 | 8 | 64 | 10.0.1.2 | 64 | 0.064 |
| 9 | 9 | 64 | 10.0.1.2 | 64 | 0.065 |
| 10 | 10 | 64 | 10.0.1.2 | 64 | 0.057 |

The SDN-based campus network topology with load balancing and optimal controller placement is executed. The network is under an active control with several controllers and ten switches along with a number of hosts. A ping test from host h1 to host h2 verifies successful connectivity without any packet loss, for an average round-trip time (RTT) that reaches 0.141 ms.

# Observation

The project demonstrated a marked improvement in the performance of the campus network with the adoption of SDN technologies, particularly through the optimization of controller placement and the implementation of load balancing techniques.In the baseline, no SDN features were present in the network; it was therefore operated in a way that had higher latency and lower throughput. The rates of packet processing as well as flow duration were restricted by the default controllers’ configuration, which was not sporting intelligent placement or traffic management.

The application of the K-Median Clustering algorithm for controller placement ensured that controllers were optimally placed, meaning that distance was minimized between switches and their assigned controllers. The enhanced placement allowed for quicker response times and better coordination between network components.

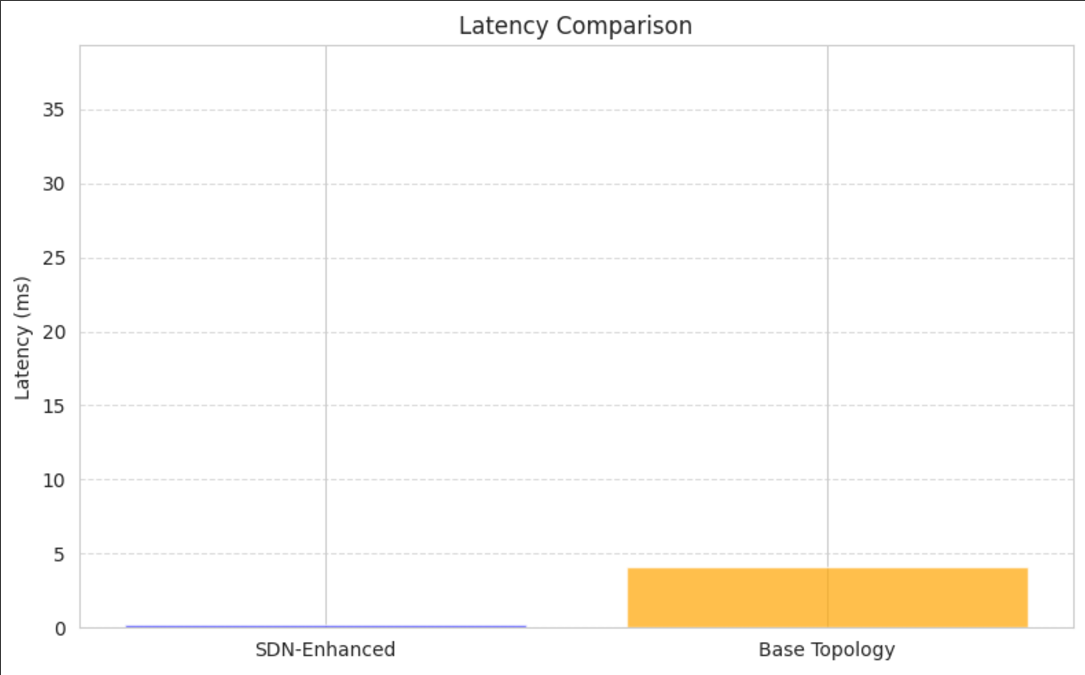
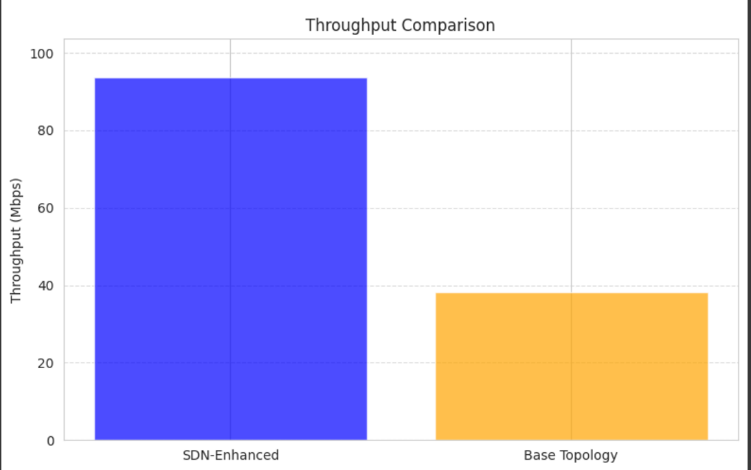


Fig 5: Latency Comparison

The bar chart in figure 6, illustrates a comparison of latency (in milliseconds) of "SDN-Enhanced" and "Base Topology" systems. In the "SDN-Enhanced" configuration, latency is negligible, while in the case of "Base Topology," latency is recorded at around 5 milliseconds. This shows a huge latency performance improvement through the implementation of SDN enhancements.

  
Fig 6: Throughput Comparison

The bar chart in figure 7 compares the throughput in Mbps between "SDN-Enhanced" and "Base Topology" systems. The "SDN-Enhanced" system provides nearly 100 Mbps throughput, whereas the "Base Topology" achieves only around 40 Mbps. This reflects better data transfer efficiency by SDN-Enhanced configuration.

Besides, the inclusion of DWRR load balancing algorithm improved the system further with respect to balanced traffic flow among available resources to avoid bottlenecks and operation of servers within their optimal capacities. The results are clearly indicative of successful efficacy through the SDN optimizations in overcoming limitations that were identified in the baseline setup. .

# Results

The performance metrics recorded for the baseline setup and the SDN-optimized network reveal significant enhancements across all measured parameters:

The table1 compares in detail the performance of the baseline network (Default Network) and the optimized network (SDWN with DWRR) at various metrics. Packet Count, signifying the total packets processed by the switch, is included along with Byte Count, signifying the total data processed in bytes. Flow Duration, in seconds. This measures the active flow time in seconds; Packet Rate is the number of packets processed per second Latency: RTT for a packet h1 to h2 or h2 to h1 Throughput. In Mbps; this reports on the data transfer rate through the network. The table confirms significant improvements in the performance of packets and bytes plus packets processed through faster rates, lower latency, and improved throughput.

Table 3: Comparison between baseline network and Optimized SDN network

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **Description** | **Baseline (Default Network)** | **Optimized (SDN with DWRR)** |
| Packet Count | Total packets processed by the switch | dist1: 26 packets | dist1: 443 packets |
|  |  | core1: 24 packets | dist2: 442 packets |
| Byte Count | Total data processed in bytes | dist1: 3,089 bytes | dist1: 51,270 bytes |
|  |  | core1: 2,877 bytes | dist2: 51,184 bytes |
| Flow Duration | Active flow duration (seconds) | dist1: 440.972 s | dist1: 751.239 s |
|  |  | core1: 426.046 s | dist2: 754.808 s |
| Packet Rate | Packets processed per second | dist1: ~0.059 pkt/s | dist1: ~0.59 pkt/s |
|  |  | core1: ~0.056 pkt/s | dist2: ~0.59 pkt/s |
| Latency | RTT for a packet (h1 to h2) | 4.071 ms | 0.215 ms |
| Throughput | Data transfer rate | ~38.3 Mbps | ~93.6 Mbps |

**Latency:** The baseline network had an average latency of 4.071 ms, while theSDN-optimized version reduced this value to 0.215 ms, thus decreasing delay significantly..

**Throughput:** With optimization, the throughput shows a tremendous improvement from 38.3 Mbps in the baseline to 93.6 Mbps in the optimized network, which justifies the use of load balancing and controller placement optimization.

**Packet Count:** The number of packets processed in the SDN setup was significantly higher, with 443 packets processed by dist1 compared to only 26 packets in the baseline network.

**Byte Count:** The total volume of data processed also increased, with dist1 handling 51,270 bytes in the SDN-optimized setup, compared to just 3,089 bytes in the baseline.

**Flow Duration:** The flow durations were longer in the optimized network dueto more sustained traffic handling, with dist1 registering 751.239 seconds compared to 440.972 seconds in the baseline.

**Packet Rate:** Packet processing rates rose significantly, with the SDN setupachieving approximately 0.59 packets/second, compared to the baseline’s 0.059 packets/second.

These results clearly indicate benefits of SDN-based approaches to networks: optimized controller placement reduced latency, and efficient resource utilization with high throughput was ensured by the DWRR algorithm. Comparative analysis shows how SDN technologies are capable of turning old into new, high-performance architectures in traditional networks.

# Conclusion

Through this project, the transformation impact of Software-Defined Networking (SDN) in optimizing the topology for a campus network was demonstrated. Also, with the optimal placement of controllers using the K-Median Clustering algorithm, the integration of Dynamic Weighted Round-Robin load balancing algorithm into the setup generally produced adequate enhancements as compared to the corresponding baseline.The baseline showed no characteristic of SDN and suffered from higher latency, lower throughput and suboptimal resource utilization while the optimized network for SDN resulted in reduced latency, increased throughput with better packet processing rates. Controller placement optimization minimized communication delay and the DWRR algorithm distributed the traffic loads against bottlenecks while ensuring balanced resources usage. These results, therefore, indicate SDN as capable of reducing some of the major limitations in traditional networks thus providing scalability, flexibility, and improved performance for large-scale and even complex networked environments.

# Future Work

Future work can explore advanced clustering algorithms for controller placement and machine learning techniques for dynamic traffic prediction and real-time adjustments. Expanding the network to handle diverse traffic types, integrating fault-tolerance mechanisms, and testing in real-world hardware setups can further optimize performance and enhance practical applicability for campus and enterprise networks.

# References

1. M. R. Belgaum, S. Musa, M. M. Alam and M. M. Su’ud, ”A Systematic Review of Load Balancing Techniques in Software-Defined Networking,” in IEEE Access, vol. 8, pp. 98612-98636, 2020, doi: 10.1109/ACCESS.2020.2995849.
2. M. A. Ouamri, G. Barb, D. Singh and F. Alexa, ”Load Balancing Optimization in Software-Defined Wide Area Networking (SD-WAN) using Deep Reinforcement Learning,” 2022 International Symposium on Electronics and Telecommunications (ISETC), Timisoara, Romania, 2022, pp. 16, doi: 10.1109/ISETC56213.2022.10010335. keywords: Deep learning;Wide area networks;Costs;Simulation;Process control;Reinforcement learning;Load management;SD-WAN;CPE;Load Balancing;artificial intelligence;MADQN,
3. I. Maity, R. Dhiman and S. Misra, ”EnPlace: Energy-Aware Network Partitioning for Controller Placement in SDN,” in IEEE Transactions on Green Communications and Networking, vol. 7, no. 1, pp. 183-193, March 2023, doi: 10.1109/TGCN.2022.3175901. keywords: Control systems;Internet of Things;Energy consumption;Switches;Measurement;Games;Delays;SDN;controller placement;network partitioning;game theory;IoT;energy efficiency,
4. P. -T. Tivig and E. Borcoci, ”Critical Analysis of Multi-Controller Placement Problem in Large SDN Networks,” 2020 13th International Conference on Communications (COMM), Bucharest, Romania, 2020, pp. 489-494, doi: 10.1109/COMM48946.2020.9141969. keywords: Control systems;Optimization;Delays;Process control;Peer-to-peer computing;Telecommunication traffic;software defined networking;controller placement problem;open issues,
5. S. Li, Z. Li and W. Zhang, ”A Dynamic Association Strategy for Controller Load Balancing in Software-Defined Networks,” 2022 23rd Asia-Pacific Network Operations and Management Symposium (APNOMS), Takamatsu, Japan, 2022, pp. 01-06, doi: 10.23919/APNOMS56106.2022.9919948. keywords: Costs;Network topology;Scalability;Heuristic algorithms;Control systems;Load management;Nash equilibrium;SDN;multiple controllers;dynamic controller association;load balancing;switches association migration.
6. M. R. Belgaum, S. Musa, M. M. Alam and M. M. Su’ud, ”A Systematic Review of Load Balancing Techniques in SoftwareDefined Networking,” in IEEE Access, vol. 8, pp. 98612-98636, 2020, doi: 10.1109/ACCESS.2020.2995849. keywords: Load management;Quality of service;Switches;Systematics;Measurement;Software;Artificial intelligence;conventional;load balancing;review;SDN;software-defined networking;systematic
7. Y. Fazea and F. Mohammed, ”Software Defined Networking based Information Centric Networking: An Overview of Approaches and Challenges,” 2021 International Congress of Advanced Technology and Engineering (ICOTEN), Taiz, Yemen, 2021, pp. 1-8, doi: 10.1109/ICOTEN52080.2021.9493541. keywords: Information-centric networking;Computer architecture;Internet;Software defined networking;Information centric network;ICN;Software defined network;SDN;SDN based ICN
8. N. H, M. Kalmat, F. Khan, S. Shinde and N. D. G, ”Dynamic Controller Placement in Software Defined Networks using Spectral Clustering,” 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, 2023, pp. 1-6, doi: 10.1109/ICCCNT56998.2023.10306706. keywords: Heuristic algorithms;Software algorithms;Clustering algorithms;Process control;Jitter;Control systems;Throughput;SDN;OpenFlow;POX;Mininet;Spectral clustering
9. Aakanksha, Sharma., V., Balasubramanian., Joarder, Kamruzzaman. (2023). A Novel Dynamic Software-Defined Networking Approach to Neutralize Traffic Burst. Computers, 12(7):131-131. doi: 10.3390/computers12070131
10. Mohammad, Rostami., Salman, Goli-Bidgoli. (2024). An overview of QoSaware load balancing techniques in SDN-based IoT networks. 13:89-89. doi: 10.1186/s13677-024-00651-7
11. Rajanikanth, P., Nagaraj, T., Ashwin, N., Venkatesh, S., Nagaraju, T., & Somashekharappa, A. (2024). Hierarchical Routing with Optimization Algorithm for SDN: Enhancing Network Lifetime and QoS. Revue d’Intelligence Artificielle,•38(3),•1027.
12. Xiao, J., Pan, X., Liu, J., Wang, J., Zhang, P., & Abualigah, L. (2024). Load balancing strategy for SDN multi-controller clusters based on load prediction. The Journal of Supercomputing, 80(4),•5136-5162.
13. Muhammed, Nura, Yusuf., Kamalrulnizam, Bin, Abu, Bakar., Babangida, Isyaku., Fadhil, hussein, mukhlif. (2023). Distributed Controller Placement in Software-Defined Networks with Consistency and Interoperability Problems. Journal of Electrical and Computer Engineering, 2023:6466996:1-6466996:33. doi:10.1155/2023/6466996
14. [14] Arun, M., et al. ”Anomaly Detection in IoT Network Using SDN Principles.” 2024 5th International Conference on Electronics and Sustainable Communication Systems (ICESC). IEEE, 2024.
15. G. D. Nithin, D. Yathis Kumar, S. Nithin, K. S. Sree Hari Nathan and R. Gandhiraj, ”Network Provisioning in SDN: Optimizing Network Resources for Enhanced Performance,” 2024 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET), Chennai, India, 2024, pp. 1-6, doi: 10.1109/WiSPNET61464.2024.10533095. keywords: Wireless communication;Performance evaluation;Signal processing algorithms;Computer architecture;Jitter;Throughput;Delays; Packet prioritization;Congestion control;Network flow management;Latency reduction,
16. Krishnan, Prabhakar, and Krishnashree Achuthan. ”CloudSDN: enabling SDN framework for security and threat analytics in cloud networks.” Ubiquitous Communications and Network Computing: Second EAI International Conference, Bangalore, India, February 8–10, 2019, Proceedings 2. Springer International Publishing, 2019.