Review of SDN-based load-balancing methods, issues, challenges, and roadmap

Review Paper

Mohit Chandra Saxena
Research Scholar,
School of Computer Science and Engineering, Galgotias University, Greater Noida, India
mohit.chandra_phd20@galgotiasuniversity.edu.in

Munish Sabharwal
Dean, School of Computer Science and Engineering,
Galgotias University, Greater Noida, India
dean.scse@galgotiasuniversity.edu.in

Preeti Bajaj
Vice Chancellor,
Lovely Professional University, Phagwara, Punjab, India
preetibajaj@ieee.org

Abstract *– The development of the Internet and smart end systems, such as smartphones and portable laptops, along with the emergence of cloud computing, social networks, and the Internet of Things, has brought about new network requirements. To meet these requirements, a new architecture called software-defined network (SDN) has been introduced. However, traffic distribution in SDN has raised challenges, especially in terms of uneven load distribution impacting network performance. To address this issue, several SDN load balancing (LB) techniques have been developed to improve efficiency. This article provides an overview of SDN and its effect on load balancing, highlighting key elements and discussing various load-balancing schemes based on existing solutions and research challenges. Additionally, the article outlines performance metrics used to evaluate these algorithms and suggests possible future research directions.*

Keywords*: Software-defined network, Load balancing, Data Plane, Control plane, Networking, IP, NFV*

1. INTRODUCTION

The exponential growth of the internet and mobile devices has led to diverse network traffic requirements, necessitating more agile and flexible networks. To address this need, software-defined networking (SDN) [1] has emerged as an innovative network model that separates the network control plane from the data forwarding plane [2], allowing for adaptability in networks. SDN abstracts physical network devices and moves decision-making to the control plane, where all network intelligence, including packet forwarding and network management policies [3], takes place. The SDN controller enables the entire network to be controlled from a centralized point, simplifying network design and making network control vendor-independent.

Load balancing is an essential aspect of network design and management, which aims to distribute traffic among network devices efficiently [4]. In traditional networks, a dedicated server is used for load balancing. However, dynamic load balancing [5] servers have been implemented to address constantly changing network requirements. Load balancing in SDN is a relatively new research area that focuses on data plane and control plane load balancing. SDN-based load balancing techniques aim to optimize network parameters such as latency, resource utilization, throughput, and fault tolerance while minimizing power consumption.

Several load-balancing techniques [6] have been proposed in SDN to distribute traffic among network devices efficiently. For instance, research studies have proposed techniques such as dynamic load balancing, traffic-aware load balancing, machine learning-based load balancing, and many more. The primary objective of this review paper is to provide a comprehen-

sive overview of load balancing in SDN. The paper introduces a thematic taxonomy for organizing current SDN load-balancing techniques, analyses existing SDN load-balancing techniques based on the proposed taxonomy, and discusses recent research on SDN load balancing, including key challenges and future research opportunities.

Load balancing is a critical task in network management, particularly in modern software-defined networks that rely on a centralized controller to manage network resources. Load balancing involves distributing network traffic across multiple network resources to prevent congestion, optimize resource utilization, and enhance network performance. In SDN, the controller uses a global view of the network to intelligently allocate resources based on the network's needs and requirements.

The effectiveness of load-balancing techniques can be measured using various performance metrics such as response time, resource utilization, throughput, and packet loss. Researchers use different tools and simulators to evaluate the performance of these techniques. Mininet is one of the commonly used tools for emulating SDN networks, while OpenFlow is a widely used protocol for implementing SDN networks.

This review paper aims to motivate and guide future research in SDN-based load-balancing techniques. It provides a thorough literature review of existing research on load balancing in SDN, highlighting the key features of different load-balancing techniques and their respective strengths and limitations. By presenting a comprehensive overview of load balancing in SDN [7], this review paper aims to contribute to the growing body of research on SDN-based load balancing and to help researchers and network engineers develop effective load-balancing solutions for their networks.

1.1. MOTIVATION AND RESEARCH QUERIES

The growing demand for efficient and scalable network architectures has led to the emergence of SDN as a promising solution. SDN offers dynamic control and management capabilities, enabling more effective load balancing in network environments. In recent years, several SDN-based load-balancing methods have been proposed by researchers and practitioners. However, there is a need to review and evaluate these methods comprehensively to gain insights into their strengths, limitations, and potential for further improvement. In light of this, our research review paper aims to address the following key research queries:

1. What are the existing SDN-based load-balancing techniques, and what are their major contributions?
2. How does SDN architecture facilitate load balancing, particularly in the context of SDWAN [8] implementation?
3. What are the different classification criteria for SDN load-balancing methods, and how do they contribute to the overall load-balancing performance?
4. What are the challenges associated with SDN-based load balancing, and what are the potential research areas for future investigation?
5. Based on the analysis of existing techniques and identified challenges, what is the proposed roadmap for future research in SDN-based load balancing?

1.2. RESEARCH HIGHLIGHTS AND CONTRIBUTIONS

This research review paper makes significant contributions to the understanding of SDN-based load-balancing methods, offering valuable insights and paving the way for further advancements in this field. The key highlights and contributions of this paper include:

Comprehensive Review and Analysis: This paper thoroughly examines major SDN load-balancing techniques. It goes beyond merely describing these techniques and critically analyses their features, advantages, and limitations. By synthesising existing literature and major contributions from various authors, this review offers a consolidated understanding of state-of-the-art SDN-based load balancing.

Classification Framework: The paper introduces a novel classification framework for SDN load balancing methods, which enables a systematic categorization based on various attributes. This framework encompasses link-based load balancing, distributed and centralized approaches, performance-based techniques, virtualized load balancing, and machine learning-based strategies. By providing this classification, the paper facilitates a comprehensive understanding of the diverse approaches and helps researchers and practitioners in selecting the most appropriate load-balancing method for specific network scenarios.

Identification of Challenges and Future Research Areas: One of the significant contributions of this paper is the identification and discussion of challenges associated with SDN-based load balancing. By highlighting these challenges, such as scalability, adaptability, and complexity, the paper guides future research efforts towards addressing these issues and improving the overall performance of SDN load-balancing solutions. Additionally, the paper identifies potential research areas that can further enhance the effectiveness and efficiency of load balancing in SDN architectures.

Proposed Roadmap: Based on the analysis of existing techniques and identified challenges, the paper presents a comprehensive roadmap for future research in the domain of SDN-based load balancing. This roadmap outlines key directions and priorities for further investigation, including the development of novel algorithms, integration of machine learning techniques, enhancement of scalability, and the exploration of

emerging technologies that can augment load balancing capabilities in SDN.

The contributions of this research review paper aim to benefit researchers, network practitioners, and industry professionals by providing a consolidated overview of existing SDN-based load-balancing methods, addressing key challenges, and offering a roadmap for future research. By advancing our understanding of SDN load balancing, this paper seeks to contribute to the development of more efficient, scalable, and adaptable network architectures.

1.3. ARTICLE ORGANIZATION

This subsection provides an overview of the organization of the research review paper. Fig. 1 below gives a summary of the flow of the paper.

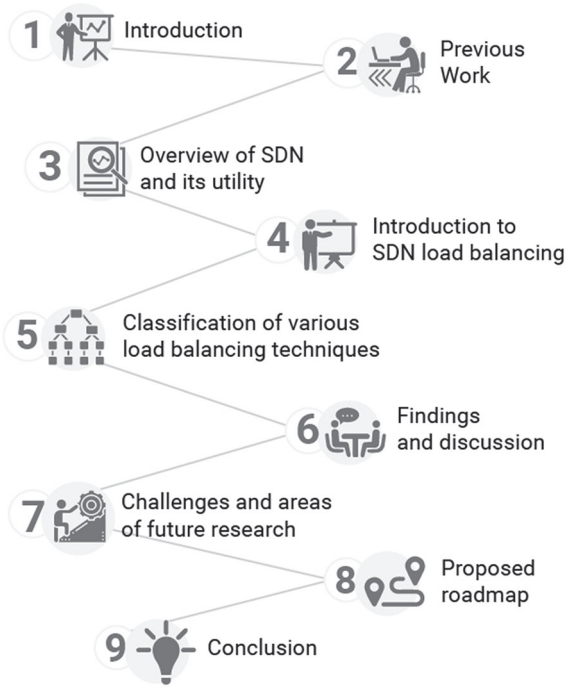

Fig 1. The flow of the paper

Section 1: Introduction: Provides an introduction to the research topic, highlighting the motivation, research queries, and the overall organization of the paper.

Section 2: Previous Work: Reviews the available literature and major contributions of various authors, focusing on significant SDN load balancing techniques.

Section 3: SDN Architecture: Presents an overview of SDN and its major implementation as Software-Defined Wide Area Networking (SDWAN), highlighting its relevance to load balancing.

Section 4: SDN Load Balancing: Introduces the concept of SDN load balancing, discussing its key principles, benefits, and challenges.

Section 5: Classification of SDN Load Balancing Methods: Provides a comprehensive classification framework for SDN load balancing techniques based on various attributes, such as link-based load balancing, distributed and centralized approaches, performance-based methods, virtualized load balancing, and machine learning-based strategies.

Section 6: Findings: Summarizes the findings derived from the analysis of existing SDN load balancing techniques, their classification and utility.

Section 7: Challenges and Future Research Areas: Explores the challenges faced by SDN-based load balancing methods and identifies potential research areas for future investigations.

Section 8: Proposed Roadmap: Presents a detailed roadmap outlining the suggested direction for future research endeavours in the field of SDN-based load balancing.

Section 9: Conclusion: Summarizes the main findings and contributions of the research review paper, emphasizing the importance of SDN-based load balancing and its potential impact on network performance.

2. RELATED WORK

Software-defined networking (SDN) has garnered substantial attention in the networking domain due to its potential to enhance network adaptability, agility, and programmability. Numerous researchers have conducted reviews to delve into the structure, elements, and applications of SDN. Rowshanrad et al. [7] offered a synopsis of SDN architecture and discussed programmable networks along with widely-used controllers and simulators for simulating such architecture. Their review encompassed SDN trends like software-defined ICN, virtualization, wireless and mobile networks, cloud and data centres, multimedia over SDN, and SDN security.

Fellow et al. [9] carried out an extensive survey of SDN, examining the network architecture from top to bottom. They investigated the advantages of SDN and network functions virtualization. Jammal et al. [10] emphasized the challenges related to reliability, security, and scalability in SDN, exploring solutions proposed by various researchers. Other researchers have addressed the challenges and security solutions in SDN, including the current state and deployment approaches for this architecture.

Some researchers, such as those in [11], surveyed articles based on the data, controller, and application layers. Farhady et al. [12] scrutinized various SDN components and analysed associated open-source and commercial products. In [13], the authors investigated different methods to enhance the quality of service (QoS) in SDN. Karakus et al. [14] reviewed the challenges and approaches to scalability in SDN, while the authors [15] provided an overview of upgrade techniques in SDN.

Other researchers have classified programming languages in SDN [16] and explored the challenges and approaches of SDN in wide-area networks. Jungmin

Son et al. [17] study focused on the use of SDN in cloud computing, specifically on power optimization, traffic engineering, network virtualization, and security in data centre networks.

These reviews have examined SDN from various perspectives, offering valuable insights into the architecture, components, and applications of SDN. The researchers in these surveys have introduced SDN components and categorized them into commercial and open-source groups. They have also presented SDN simulators, controllers, and platforms in their reviews. Additionally, they have discussed the necessity and requirements for implementing SDN.

Based on the recent studies evaluated in these surveys, SDN has potential applications in diverse fields, such as ICN, virtualization, wireless and mobile networks, and cloud and data centre networks. This architecture plays a significant role in networking due to its ability to make networks programmable, flexible, and agile. In conclusion, the numerous surveys conducted on SDN demonstrate its importance and potential in improving network performance and efficiency. Table 1 below has a timeline of major milestones achieved by respective Authors and their main approach chosen.

Table 1. Summary of Previous Work in a TimeLine

Year	Milestone	Author(s)	Approach/ Algorithm
2009	Introduction of SDN concept	-	-
2011	Proposal of OpenFlow protocol	McKeown et al.	-
2012	First commercial SDN product launched	-	-
2014	Release of first SDN standard (OpenFlow 1.3)	ONF	-
2015	First SDN-based products for data centers released	-	-
2016	Proposal of SDN-based load balancing architecture	Wang et al.	Adaptive dynamic load balancing algorithm
2016	Comprehensive survey on SDN and its applications	Yu et al.	-
2017	Proposal of load balancing method for VM migration	Wang and Chen	-
2017	Proposal of efficient load balancing scheme	Chang et al.	-
2017	Proposal of traffic-aware SDN-based load balancing approach for cloud datacenters	Yu et al.	-
2017	Proposal of SDN-based load balancing approach for scaling applications across geographically distributed data centers	Katsifodimos et al.	Machine learning-based approach
2017	Proposal of intelligent load balancing framework for SDN-based cloud computing	Ghorbani et al.	-
2017	Proposal of novel load-balancing algorithm based on SDN for cloud computing	Ouyang et al.	-
2017	Proposal of reinforcement learning-based load balancing algorithm for SDN	Park and Lee	Reinforcement learning-based approach
2018	Proposal of load balancing algorithm for SDN-based cloud computing	Wang et al.	-
2018	Comprehensive survey on load balancing in SDN	Liu et al.	-
2018	Proposal of adaptive load balancing algorithm based on deep learning in SDN	Wang et al.	Deep learning-based approach
2019	Proposal of machine learning-based load balancing algorithm for SDN	Shi et al.	Machine learning-based approach
2019	Comprehensive survey on load balancing in SDN for cloud computing	Ngo et al.	-
2019	Proposal of SDN-based load balancing algorithm for cloud computing	Zhou et al.	-
2019	Proposal of intelligent load balancing algorithm based on reinforcement learning in SDN	Huang et al.	Reinforcement learning-based approach
2019	Proposal of SDN-based load balancing scheme for edge computing	Wu et al.	-
2019	Comprehensive survey on dynamic load balancing for SDN	Li et al.	-
2020	Proposal of dynamic load balancing algorithm for SDN based on reinforcement learning	Tao et al.	Reinforcement learning-based approach
2020	Proposal of dynamic load balancing algorithm for SDN based on correlation analysis	Zhang et al.	Correlation analysis-based approach
2020	Proposal of energy-efficient load balancing algorithm for SDN	Zhang et al.	Energy-efficient approach
2020	Proposal of traffic-aware load balancing algorithm based on SDN	Li et al.	Traffic-aware approach
2021	Proposal of hybrid load balancing algorithm based on deep reinforcement learning in SDN	Guo et al.	Deep reinforcement learning-based approach

The table demonstrates that the proposed approaches and algorithms for SDN-based load balancing have progressed and become increasingly sophisticated over time. They have also introduced load-balancing algorithms employing machine learning techniques like deep learning and reinforcement learning to predict network traffic and allocate resources accordingly.

Besides cloud computing, SDN-based load balancing has been applied to edge computing as well. Wu et al. (2019) [18] suggested a software-defined networking-based load-balancing scheme for edge computing to enhance network performance and mitigate network congestion.

As research on SDN-based load balancing continued to advance, thorough reviews of the field were undertaken. Yu et al. (2016) and Ngo et al. (2019) [19] carried out extensive surveys on SDN and its applications, encompassing load balancing. Liu et al. (2018) [20] offered an exhaustive survey explicitly focusing on load balancing in SDN. Li et al. (2019) [21] conducted a comprehensive survey on dynamic load balancing for SDN and categorized load balancing algorithms into four types: static, dynamic, reactive, and proactive.

Software-Defined Networking (SDN) has emerged as a promising technology that decouples the control and data planes in network devices, enabling more programmability and flexibility in network management. Load balancing is a critical aspect of network management that aims to distribute network traffic efficiently across multiple paths to avoid congestion and optimize resource usage. Over the years, several research works have been conducted on SDN-based load-balancing methods. Some major path-breaking research works are as follows:

1. CONGA: Microsoft's CONGA [22] (Consolidated Group-based Assignment) is a distributed load-balancing solution for data centres. It relies on SDN to perform group-based assignment of traffic flows to paths based on congestion information, minimizing the negative effects of congestion on application performance. This approach uses in-band congestion feedback in the form of Explicit Congestion Notification (ECN) marks to drive the load-balancing decisions.
2. Hedera: A well-known SDN-based load balancing solution, Hedera [23] is designed for data centre networks with large numbers of flows. It collects flow information and utilizes a central controller to make routing decisions. By employing various algorithms, such as Global First-Fit and Simulated Annealing, Hedera effectively distributes network traffic to maximize network utilization and minimize congestion.
3. ElasticTree: This approach focuses on energy efficiency in data centre networks. ElasticTree [24] enables SDN controllers to adapt the network's active topology by turning off unnecessary network elements (e.g., switches) when they are not needed. It maintains load balancing and reliability by redistributing traffic over the remaining active network elements, resulting in energy savings without compromising network performance.
4. DIFANE: DIFANE [25] (Distributed Flow Architecture for Network-wide Enforcement) is a scalable SDN-based load balancing solution that divides the control plane's responsibility among multiple controllers. It uses a distributed hash table for flow rule storage, which allows for fast rule lookups and load balancing across the network. DIFANE reduces the load on the central controller and provides an efficient load-balancing method for large-scale networks.
5. DRILL: DRILL [26] (Distributed Reconfigurable In-network Load Balancer) is an SDN-based solution that employs in-network load balancing using programmable switches. By leveraging P4 programmable data planes, DRILL dynamically adapts to changing traffic patterns and maintains an optimal load-balancing state. It also provides flow-level fairness and low flow completion times.

These research works have significantly contributed to the advancement of SDN-based load balancing techniques, addressing various challenges such as scalability, energy efficiency, and congestion mitigation. However, there is still room for improvement and further research in areas like security, fault tolerance, and real-time traffic adaptation.

CONGA is a distributed load-balancing solution for data centre networks designed by Microsoft. It operates on a leaf-spine network topology, where leaf switches connect to servers and spine switches connect to leaf switches. The Architecture is depicted in Fig. 2 below.

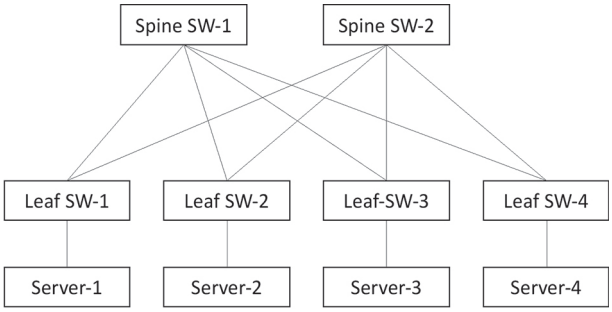

Fig 2. CONGA Architecture

The primary components of the CONGA architecture [27] include:

1. Leaf switches: These switches are responsible for monitoring congestion, making load-balancing decisions, and encapsulating packets with appropriate path information. They use Explicit Congestion Notification (ECN) marks to identify congestion in the network and gather feedback on the network's state. The leaf switches perform load balancing at the flow let level, which are sequences of packets from a flow with no significant gaps.
2. Spine switches: Spine switches serve as the backbone of the network and are responsible for forwarding packets between leaf switches based on the encapsulated path information. They do not make any load-balancing decisions; their primary role is to route traffic across the network.
3. Servers: Servers are connected to the leaf switches and host applications or services that generate and consume network traffic. They communicate with each other using standard TCP/IP protocols.

Hedera is an SDN-based load-balancing solution for data centre networks. It operates on a Fat-Tree topology,

which is a multi-rooted, hierarchical topology designed to provide high bandwidth and fault tolerance. The primary components of the Hedera architecture[28] include:

1. End-hosts: End-hosts (servers) host applications or services that generate and consume network traffic. They communicate with each other using standard TCP/IP protocols.
2. Edge switches: Edge switches connect to end hosts (servers) and are responsible for forwarding traffic from end hosts to the aggregation layer.
3. Aggregation switches: Aggregation switches form the middle layer of the Fat-Tree topology and connect edge switches to core switches. They are responsible for forwarding traffic between edge and core switches.
4. Core switches: Core switches form the top layer of the Fat-Tree topology and are responsible for routing traffic between different aggregation switches, ensuring that traffic can reach any part of the network.
5. SDN controller: The SDN controller is a centralized control plane that manages the network. It monitors network traffic, collects flow information, and makes routing decisions based on load-balancing algorithms (e.g., Global First-Fit or Simulated Annealing).

The Hedera architecture is depicted in Fig. 3 below:

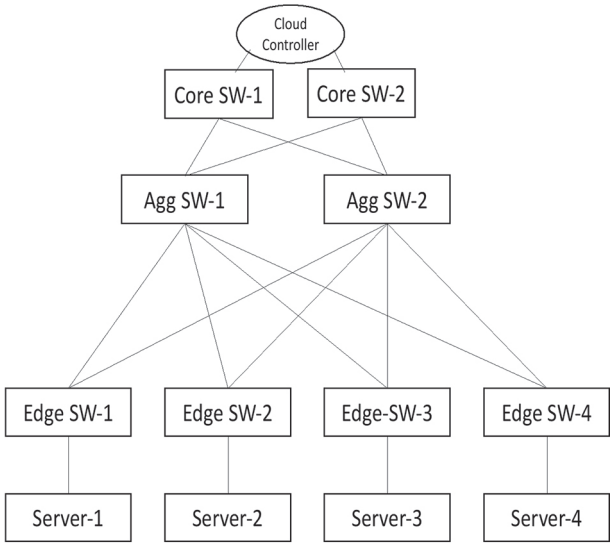

Fig 3. Fat tree Architecture formed by Hedera and ElasticTree approaches

ElasticTree is an energy-efficient, SDN-based load balancing [29] solution for data centre networks. It operates on a Fat-Tree topology just like Hedera depicted in Fig. 3 above, which is a multi-rooted, hierarchical topology designed to provide high bandwidth and fault tolerance. The primary components of the ElasticTree architecture include:

1. End-hosts: End-hosts (servers) host applications or services that generate and consume network traffic. They communicate with each other using standard TCP/IP protocols.
2. Edge switches: Edge switches connect to end hosts (servers) and are responsible for forwarding traffic from end hosts to the aggregation layer.
3. Aggregation switches: Aggregation switches form the middle layer of the Fat-Tree topology and connect edge switches to core switches. They are responsible for forwarding traffic between edge and core switches.
4. Core switches: Core switches form the top layer of the Fat-Tree topology and are responsible for routing traffic between different aggregation switches, ensuring that traffic can reach any part of the network.
5. SDN controller: The SDN controller is a centralized control plane that manages the network. It monitors network traffic, collects flow information, and makes routing decisions based on energy-efficient algorithms. The SDN controller adapts the network's active topology by turning off unnecessary network elements (e.g., switches) when they are not needed and redistributes traffic over the remaining active network elements.

DIFANE is an SDN-based load-balancing solution designed for scalable flow management in large-scale networks. The primary components of the DIFANE architecture include:

1. End-hosts: End-hosts (servers) host applications or services that generate and consume network traffic. They communicate with each other using standard TCP/IP protocols.

1. Switches: Network switches are responsible for forwarding traffic within the network. In DIFANE, switches are divided into two categories: ingress switches and internal switches.

1. Ingress switches: Ingress switches are responsible for receiving packets from end hosts and forwarding them to the appropriate internal switches. They also enforce flow rules received from the authority switches.

1. Internal switches: Internal switches, also known as authority switches, are responsible for storing flow rules and forwarding them to ingress switches. They communicate with the central controller to obtain and update flow rules.

1. Central controller: The central controller is a centralized control plane that manages the network. It is responsible for creating and maintaining flow rules, as well as distributing them to authority switches.

The Architecture of DIFANE corresponds to Fig. 4 below:

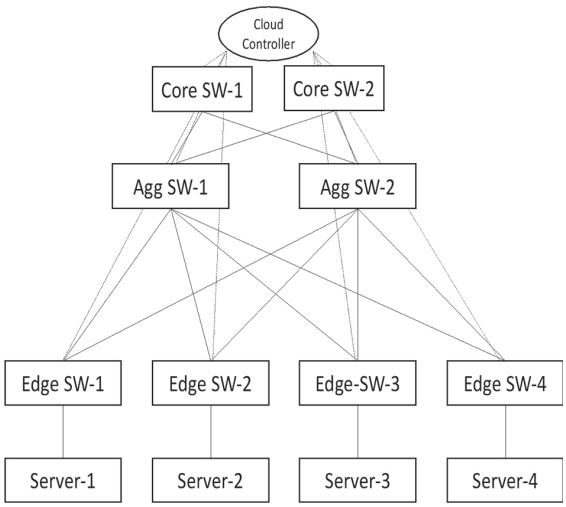

Fig 4. DIFANE Architecture

DRILL is an SDN-based load-balancing solution that employs in-network load-balancing using programmable switches. The primary components of the DRILL architecture [30] include:

1. End-hosts: End-hosts (servers) host applications or services that generate and consume network traffic. They communicate with each other using standard TCP/IP protocols.
2. Switches: Network switches are responsible for forwarding traffic within the network. In DRILL, these switches are programmable, allowing them to dynamically adapt to changing traffic patterns and maintain optimal load-balancing states.
3. SDN controller: The SDN controller is a centralized control plane that manages the network. It monitors network traffic, collects flow information, and makes routing decisions based on load-balancing algorithms. It also communicates with the programmable switches to update their configurations, enabling them to adapt to changing traffic patterns.
4. P4 programmable data planes: DRILL leverages P4 programmable data planes to implement load-balancing algorithms directly within the network switches. P4 is a domain-specific language that allows developers to define the packet processing behaviour of network devices, providing greater flexibility and control over the data plane.

DRILL architecture is defined in Fig. 5 below:

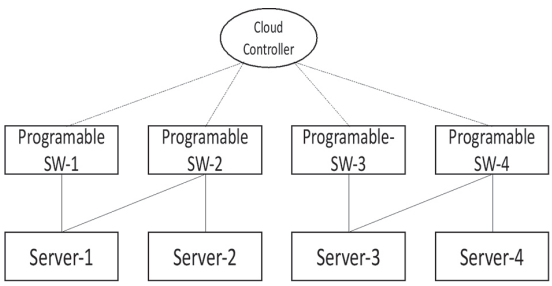

Fig 5. DRIL L architecture

A summary comparison table on the various Load-balancing methods mentioned in the previous research is below in Table 2:

Table 2. Comparison & Summary of various SDN load-balancing methods

Load Balancing Solution	Description	Key Features
CONGA	Microsoft's CONGA (Consolidated Group-based Assignment) is a distributed load-balancing solution for data centres that uses SDN to perform group-based assignment of traffic flows based on congestion information.	•Group-based assignment •In-band congestion feedback •Minimizes congestion impact on performance
Hedera	Hedera is an SDN-based load balancing solution designed for data centre networks with large numbers of flows. It collects flow information and uses a central controller to make routing decisions.	•Central controller •Global First-Fit and Simulated Annealing algorithms •Maximizes network utilization
Elastic Tree	ElasticTree focuses on energy efficiency in data centre networks, enabling SDN controllers to adapt the network's active topology by turning off unnecessary network elements when they are not needed.	•Energy efficiency •Adaptable network topology •Maintains load balancing and reliability
DIFANE	DIFANE (Distributed Flow Architecture for Network-wide Enforcement) is a scalable SDN-based load balancing solution that divides the control plane's responsibility among multiple controllers.	•Distributed control plane •Fast rule lookups •Load balancing across the network
DRILL	DRILL (Distributed Reconfigurable In-network Load balancer) is an SDN-based solution that employs in-network load balancing using programmable switches and leverages P4 programmable data planes to dynamically adapt.	•In-network load balancing •P4 programmable data planes •Flow-level fairness and low completion times

Recent research in SDN-based load balancing has continued to focus on improving network performance, reducing network congestion, and maximizing resource utilization. Tao et al. (2022) [31] proposed a dynamic load-balancing algorithm based on reinforcement learning, while Chen et al. (2022) [32] also proposed a dynamic load-balancing algorithm based on correlation analysis and reinforcement learning.

Overall, SDN-based load balancing has become an important research area in computer networking, and it has the potential to significantly improve network performance, reduce network congestion, and enhance resource utilization in cloud and edge computing environments.

3. ARCHITECTURE OF SDN

Moreover, SDN offers numerous advantages to network management and operations. It facilitates centralized network management, simplifying configura-

tion, monitoring, and troubleshooting. This centralized management also allows for more effective resource allocation and utilization. Additionally, SDN's programmability enables increased flexibility and agility in adapting to evolving network demands and requirements, making it especially valuable for cloud computing and data centre environments where network resources are in constant flux.

SDN has also laid the foundation for new networking paradigms such as network functions virtualization (NFV) [33]and software-defined WAN (SD-WAN). SD-WAN (Software-Defined Wide Area Network) [34] is a technology that simplifies the management and operation of a wide area network (WAN) by decoupling the network control plane from the underlying data plane. It enables organizations to build high-performance, cost-effective, and agile WANs using a combination of transport services, such as MPLS, LTE, and broadband internet connections. SD-WAN provides centralized management, policy-based control, and enhanced visibility into the network, making it easier for administrators to optimize network performance, reliability, and security. Fig. 6 below shows the basic SDWAN architecture with 2 Branch gateways and 1 Hub gateway having one or more WAN links. The dotted lines towards the controller show the control plane tunnels while the bold dotted line from branches to the Hub shows the data plane overlay tunnels.

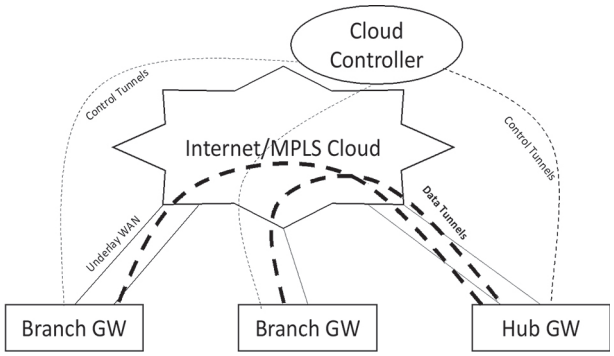

Fig 6. Basic SDWAN Architecture

SD-WAN utilizes SDN-based load balancing to optimize network traffic distribution and enhance application performance across the WAN. Here's how SD-WAN incorporates SDN-based load balancing:

1. Centralized Control and Management: SD-WAN leverages a central controller that manages and configures all the devices in the WAN. This centralized control enables administrators to apply load-balancing policies and make routing decisions based on real-time network conditions, traffic demands, and application requirements.
2. Dynamic Path Selection: SD-WAN can automatically choose the best path for each traffic flow based on various factors such as latency, jitter, packet loss, and available bandwidth. By intelligently distributing traffic across multiple links, SD-WAN ensures optimal utilization of available resources and prevents network congestion, improving overall network performance.
3. Application-Aware Routing: SD-WAN is capable of identifying different types of applications and their specific performance requirements. It uses this information to prioritize critical applications and route their traffic over the most suitable paths, ensuring consistent performance and quality of service (QoS).
4. Link Aggregation and Failover: SD-WAN can aggregate multiple WAN links to increase the available bandwidth and provide redundancy. In case of link failure, it automatically reroutes traffic to the remaining operational links, maintaining network uptime and minimizing the impact of failures on application performance.
5. Network Visibility and Monitoring: SD-WAN provides administrators with comprehensive visibility into the network's performance, enabling them to monitor the status of individual links and devices, identify potential issues, and make informed load-balancing decisions.

In summary, SD-WAN technology uses SDN-based load balancing to intelligently distribute network traffic across multiple paths, ensuring optimal network performance and reliability. By leveraging centralized management, dynamic path selection, application-aware routing, link aggregation, and failover capabilities, SD-WAN provides businesses with a flexible, cost-effective, and high-performance wide-area network solution.

Despite its advantages, SDN faces challenges. Security remains a significant concern, particularly regarding the centralized controller, which may become a single point of failure or attack. Additionally, concerns about vendor lock-in and interoperability among different SDN implementations exist.

In summary, SDN is a promising technology with the potential to transform network management and operations. It's programmability and centralized management make it particularly suitable for cloud computing and data centre environments. While challenges must be addressed, ongoing research and development in this area will likely continue to drive SDN innovation and adoption in the coming years.

Besides the features mentioned earlier, several other benefits of SDN architecture have contributed to its popularity and widespread adoption. A key advantage is the separation and abstraction of control and data planes, allowing for increased flexibility in managing network resources and enabling network operators to implement policies and services more easily. The centralized intelligence offered by the SDN controller ensures a global network view and facilitates rapid adaptation to changing network needs. Fig. 7 below presents a three-layered SDN architecture.

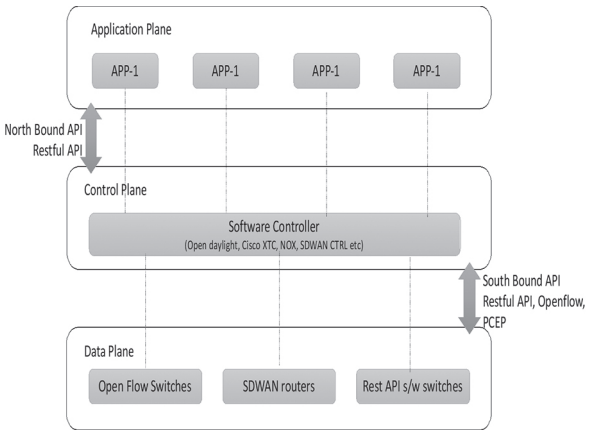

Fig 7. SDN Architecture

Another significant advantage is the capability to develop various applications through the underlying network infrastructure. This application-centric approach enables network operators to implement innovative solutions that can enhance business operations, improve customer experience, and boost network efficiency.

The programmability of the data plane is another crucial feature of SDN architecture. This allows network operators to effortlessly configure and reconfigure the network based on evolving requirements, reducing network management complexity and increasing network agility.

SDN architecture also accelerates innovation, promoting business innovation and enabling real-time program implementation. This ensures that the network can be reprogrammed to meet both business and customer demands, improving overall network performance and promoting business growth.

In conclusion, the benefits of SDN architecture are numerous, and the technology continues to gain traction as more organizations recognize the advantages it offers in terms of network flexibility, scalability, and management.

4. SDN LOAD BALANCING

Load balancing (LB) is a vital aspect of contemporary computer networks that guarantee high availability, scalability, and performance. As access and data traffic increase, server processing capabilities must grow accordingly to prevent a single point of failure. Nevertheless, hardware upgrades or replacements can be expensive and resource-demanding. This is where LB technology plays a role, distributing a large volume of concurrent traffic to multiple computing devices, enhancing server processing capacity and reducing response time to user requests. The technology is primarily employed in enterprise key application servers, Web servers, and FTP servers [35].

Traditionally, LB was referred to as a physical network component for evenly distributing network traffic and the task was carried out through specialized hardware devices based on factors like the server's current load, content relative to the requested location, or simple policies such as round-robin. However, SDN emphasized more on the underlying technology of Load balancing rather than the hardware component and introduces new possibilities to it in conventional network loads, offering fresh opportunities for load optimization.

In an SDN-based LB system, network management can be streamlined while achieving load optimization, making it well-suited for SDN LB. LB technology has been vital to SDN networks, enhancing their performance in multiple-aware routing approaches, and efficiently allocating network resources for the overall improvement of network performance and quality-of-service (QoS). Utilizing SDN-based LB enables more agile networks and enhanced network management, leading to more adaptable and effective application services [36].

Furthermore, LB through SDN allows the network to behave like virtualized computing and storage models. It aids in discovering the best route and application for faster request delivery. By directly configuring the network, SDN-based LB ensures improved network management for more flexible and effective application services.

In summary, load-balancing technology is an indispensable component of modern computer networks, and SDN-based LB offers considerable advantages in terms of scalability, performance, and network management. By harnessing the capabilities of SDN, LB can be more agile and efficient, assisting network administrators in delivering a higher quality of service and improving application performance to end-users [37].

5. CLASSIFICATION OF LOAD BALANCING TECHNIQUES

The thematic taxonomy for SDN load-balancing solutions is based on the following main parameters. The parameters selected for this thematic taxonomy were constructed from four factors, shown in Fig. 8 below: objectives-based LB, data plane LB techniques, control plane LB techniques, and performance metrics used for LB techniques

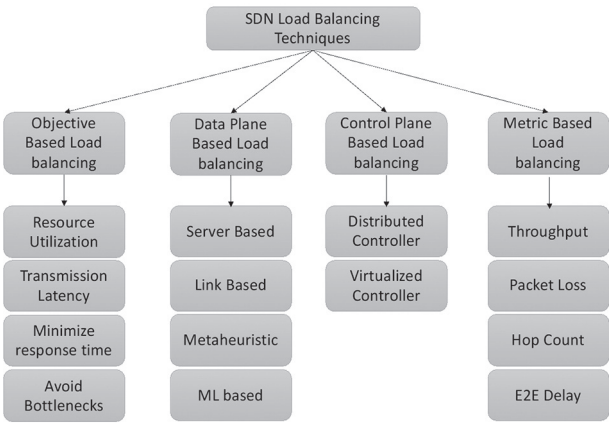

Fig 8. Thematic Taxonomy for Load Balancing Technologies in SDNs

5.1. OBJECTIVES-BASED LB

This parameter includes the objectives or goals that the LB solution aims to achieve. These objectives can be categorized into two main groups:

Network Optimization: LB solutions that aim to optimize network performance by reducing network congestion, improving network availability, and enhancing network scalability.

Application Optimization: LB solutions aim to optimize application performance by improving application response time, reducing server response time, and improving server utilization.

Resource Utilization: One of the main objectives of LB in the SDN LB model is to ensure efficient utilization of resources. Therefore, there is a specific level of usage for network resources such as links, bandwidth, processors, and memory A suitable resource provision algorithm ensures the maximal usage of resources for the LB [38]

Transmission Latency: The latency of transmission refers to the time it takes the host switch to transmit data. This depends on several factors that include the switch's performance, whether the transmission queue is congested and the size of the data packets. The delay of transmission indicates both congestion in the link and the condition of the switch load in some way. Thus, the SDN controller must collect the bytes that are transmitted within a specific period as well as the transmission rate. This parameter should be reduced [39].

Minimize response time: This is specified by the time interval between the time a server request or job is received and the time it is answered, or mission accomplished. Thus, the reaction time of a specific LB algorithm is crucial in a distributed SDN network. This parameter should be minimized

Avoiding bottleneck: To avoid any congestion or bottlenecks in the SDN network setting, LB methods are required to spread the load equally between different switches/controllers so that no switch/controller gets overloaded (i.e., among the bottlenecked switches of each link, the best is the one with the lowest capacity). Efficient utilization of available resources through proper load balancing can help reduce resource consumption. Additionally, it enforces failover, allows scalability, prevents bottlenecks, and reduces response time [40].

Based on the above, load-balancing techniques can be classified into the following categories:

5.1.1. Data plane LB techniques:

This parameter includes the different techniques used in the data plane for load balancing. These techniques can be categorized into two main groups:

Static LB: LB solutions that use pre-defined rules or policies to distribute traffic across servers.

Dynamic LB: LB solutions that use real-time data to distribute traffic across servers. These solutions use various algorithms, such as round-robin, least connections, and IP hash, to distribute traffic.

5.1.2. Control plane LB techniques:

This parameter includes the different techniques used in the control plane for load balancing. These techniques can be categorized into two main groups:

Centralized LB: LB solutions that use a centralized controller to manage load balancing. The controller is responsible for making load-balancing decisions and distributing traffic across servers.

Distributed LB: LB solutions that distribute load-balancing decision-making across multiple controllers or switches. These solutions use distributed algorithms, such as Consistent Hashing, to distribute traffic.

5.1.3. Performance metrics used for LB techniques:

This parameter includes the performance metrics used to evaluate LB solutions. These metrics can be categorized into two main groups:

Network Performance Metrics: Metrics used to evaluate the overall network performance, such as throughput, latency, and packet loss.

Application Performance Metrics: Metrics used to evaluate the performance of specific applications, such as response time and server utilization.

Overall, this thematic taxonomy provides a useful framework for categorizing and comparing SDN load-balancing solutions based on their objectives, techniques, and performance metrics. By using this framework, network administrators can select the most appropriate SDN LB solution for their specific network requirements.

5.2. DATA PLANE-BASED LOAD BALANCING

This technique is used to attain LB that is of small latency network performance; especially within the data plane. It is also used to solve the load imbalance in paths and servers and to avoid network bottlenecks in SDN. Data plane LB can be classified as servers LB and links LB are discussed in what follows

5.2.1. Server Load Balancing

Static server LB techniques are preconfigured and remain fixed until they are manually updated, whereas dynamic server LB techniques adapt to the current network conditions and traffic. An LB strategy can be used to distribute traffic to different servers to overcome network congestion [41].

Static server LB techniques include round-robin, IP hashing, and least connection, while dynamic server LB techniques include weighted round-robin, least traffic, and adaptive load balancing [42].

Round-robin is a static LB technique that distributes traffic equally across servers in a circular order. IP hashing is another static LB technique that uses a hash function to assign traffic to servers based on the source IP address of the packet. The last connection is also a static LB technique that assigns traffic to the server with the least number of active connections.

Weighted round-robin is a dynamic LB technique that assigns traffic based on the weight assigned to each server, where servers with higher weights receive more traffic. Least traffic is another dynamic LB technique that assigns traffic based on the server with the least amount of traffic. Adaptive load balancing is a dynamic LB technique that continuously monitors the network conditions and adapts the LB strategy accordingly to balance the load [43].

Overall, LB techniques play a crucial role in managing the increasing amount of traffic in data centres and ensuring efficient use of network resources. The choice of LB technique depends on the specific needs and requirements of the network, as well as the level of automation and adaptability required for LB.

When it comes to server load balancing, there are two types of algorithms: static and dynamic. Static algorithms are simple but expensive, and they're best suited for homogeneous servers. However, they're inflexible and don't take into account the efficiency of component nodes such as RAM size, server processor, and link bandwidth. This makes them unsuitable for handling dynamic changes in the network. On the other hand, dynamic algorithms allocate the load based on the current state of the network node. They check link capacity and server load at runtime and adjust load distribution accordingly. With the programmability and flexibility of SDNs, implementing dynamic server LB algorithms has become easier. For example, a genetic algorithm was used in one study to achieve optimal load balancing in a server pool by redirecting flows. The algorithm minimized the coefficient of the server using a fitness function that took into account the varying workload of each server in the pool. Overall, dynamic LB algorithms are more efficient and versatile than static ones, and they're better suited for handling the increasing traffic and dynamic changes in modern data centres.

5.2.2. Link-based Load Balancing

LB techniques for multiple path networks in SDNs have been extensively studied in the literature. Link-based LB is one such technique that focuses on optimizing path load and selecting the least loaded path for new incoming data requests to prevent congestion in the data plane. Various modern approaches have been proposed to address high-controller LB in multiple-path networks. Link-based LB can be classified into three categories, namely, meta-heuristic algorithms, machine learning algorithms, and other algorithms [44].

5.2.3. Meta-Heuristic Algorithms

Meta-heuristic algorithms are an effective approach to optimize network-wide management and overcome challenges in path load balancing. With a large number of variables and targets involved in network optimization, heuristic algorithms can provide reasonable solutions in a reasonable time frame. Several algorithms have been proposed using meta-heuristic algorithms in conjunction with SDN networks to improve load-balancing performance. For example, the fuzzy synthetic evaluation mechanism (FSEM) was proposed to address path load balancing issues. This method utilizes the Top-K algorithm to select the shortest path and allocates network traffic to the paths using Open Flow switches. The central SDN controller is responsible for installing flow-handling rules, and the FSEM enables dynamic path adjustments based on a global network view. The POX controller platform was used to implement this proposed method [45].

5.2.4. Machine Learning Based

By coupling algorithms with the SDN architecture, routing performance can be improved through the placement of centralized logic on the control plane, which allows for a global view of the network and the use of machine learning algorithms. Load distribution can be balanced in real-time through the consideration of path load scenarios, which takes into account features such as bandwidth utilization ratio, packet loss rate, transmission hops, and latency. A back propagation artificial neural network (BPANN) can be trained using these features, resulting in improved network performance, as shown in experimental results where a 19.3% reduction in network latency was achieved. However, this approach does not consider the types of services being used and does not necessarily find the exact shortest path. Another SDN-based approach utilizes artificial neural networks (ANN) and six features, including packet overhead, latency, hop count, packet loss, trust, and bandwidth ratio, to improve transmission efficiency. The load on every node is determined, and the least loaded direction is selected in real time for incoming data flows. This technique can also be implemented using Mininet and the Floodlight controller to evaluate network efficiency [46].

5.3. CONTROLLER-BASED LOAD BALANCING

LB technologies that are based on a control plane provide LB within distributed controllers to avoid bottlenecks associated with a huge SDN network within a centralized controller. Based on studies that have been conducted on multiple controllers, it can be categorized into distributed LB and virtualization controller LBs

5.3.1. Distributed Controller

Distributed controller LB architectures use one or more controllers in a network to address the challenges faced by a single controller network [47].

The rationale is to ensure that controllers that can permit the sharing of load equally in the network are formed, and one controller can take over from another controller should a crash occur.

Distributed systems can typically implement most of these advanced procedures with minimal technological requirements. It has been reported in pertinent literature that distributed controller architectures are not always based on multiple controllers. This type of network is physically distributed and of a different type.

5.3.2. Virtualised Controller

Virtualized controller LB using the slices technique is based on SDN network virtualization [48].

In this approach, the physical network infrastructure has a virtual layer placed above it, and a virtualized network can be achieved by controlling packet routing and load balance. Network virtualization is provided in this layer through the construction of virtual networks made up of virtual resources like routers, switches, and other nodes that are to be managed and controlled.

To implement control, a transparent proxy is utilized, which connects multiple controllers on one of the networks to a side of the switch. An open Flow virtualization controller called a Flow visor (FV), serves as a transparent proxy between the switches of the open Flow and that of several open Flow controllers.

FV enables the creation of multiple isolated virtual logical networks, known as slices, on a single physical infrastructure. These slices can use various addressing and flow-forwarding techniques, allowing different controllers in different slices to share network resources, such as Open Flow switches and ports.

5.4. METRIC-BASED LOAD BALANCING

LB algorithms in SDN networks rely on metrics to ensure efficient load distribution. Here are some of the most common metrics used in LB implementations:

Throughput: This metric measures the rate at which tasks are completed after LB has been performed. The LB algorithm's goal is to achieve greater efficiency by maximizing throughput.

Packet loss: This metric measures the rate at which packets are dropped during transmission. The SDN controller collects the cumulative number of transmitted and received packets at respective OpenFlow switch ports to prevent packet loss [49].

Average response time: This metric measures the time it takes for a user to retrieve the results of a request. It is influenced by factors such as bandwidth, number of users accessing the network, number of requests, and average processing time [50].

Transmission hop count: This metric measures the number of hops required to transmit packets from source to destination. A large number of hops can increase the probability of congestion, while fewer hops can decrease packet loss probability and transmission delay. The SDN controller's global network topology database can be used to search for the shortest path between switches based on the source and destination switches [51].

6. FINDINGS

To conclude this section, a summary of widely adopted SDN load-balancing algorithms is covered in Table 3. Here, we look at the performance objectives that the authors wanted to achieve as well as the selection criteria and mechanism used. Most of the literature works used centralized criteria as these works were based on SDN.

Table 3. Comparison of various techniques of SDN load balancing on specific criteria CHALLENGES FOR FUTURE RESEARCH

Strateg	Performanse Criteria	Selection Criteria	Description
DNQ [51]	Reduction of packet loss rate with different load	Centralized intelligent centre	Intelligent techniques used for path selection, important nodes, and flow forecasting
Least Connection [52]	Resource utilization optimization	Centralized and cooperative approach	Server with the least number of active connections is allocated more connections to balance traffic
SDSNLB [53]	Throughput, link load jitter	Centralized	Allocates network traffic to different flow paths for optimal and productive resource use
Dynamic Agent-based Load Balancing [54]	Efficient and adaptive	Centralized	Global visibility of SDN is used to efficiently migrate virtual machines in data centre networks
RLMD [55]	Node efficiency, node attractiveness, path quality, controller load balancing rate	Centralized and cooperative approach	Scheme for effective deployment of controllers and successful load balancing among them
Fuzzy Synthetic Dynamically Select the Evaluation Optimal Path Mechanism [56]	-	Centralized	Network flow is sent to flow paths under open flow switches for SDN controller to install flow handling
Switch Migration Based Decision-Making [57]	Response time, load distribution, and migration cost	Centralized approach	Chooses a master controller to enhance load balancing factor based on low cost
Adaptive Load Balancing Scheme [58]	Throughput and loss rate	Centralized and cooperative approach	New adaptive technique in data centres leveraging SDN for load balancing
Self-Adaptive Load Balancing [59]	Throughput testing, load balancing time, bandwidth utilization, and loss rate	Centralized approach	Ensures effective load balancing and distance between devices are considered
Double Deep Q Network Based VNF Placement Algorithm [60]	Path delay, running time of VNFIs, number of VNFIs, and utilization ratio of VNFIs	Centralized	Customized algorithm designed using gathered information to optimize network performance

7. CHALLENGES FOR FUTURE RESEARCH

Software-defined networking (SDN) has revolutionized the way network administrators manage and control their networks. One of the most significant applications of SDN is load balancing, which is the process of distributing traffic evenly across multiple servers to optimize resource utilization and ensure high availability of services. SDN-based load-balancing techniques have gained popularity in recent years due to their flexibility, scalability, and cost-effectiveness [62].

However, these techniques also face several challenges and issues that need to be addressed for their wider adoption and improved performance. In this article, we will explore these issues, challenges, and future research directions in SDN-based load balancing.

7.1. ISSUES AND CHALLENGES IN SDN-BASED LOAD BALANCING:

Controller overload: One of the most significant challenges in SDN-based load balancing is the controller's processing capacity, which can become a bottleneck when handling a large number of requests. The controller's processing power limits the number of switches and the amount of traffic that can be managed, leading to poor performance and increased latency.

Scalability: Another significant challenge is the scalability of SDN-based load balancing. As the number of switches and servers in the network increases, managing and controlling the network becomes increasingly complex and challenging. This complexity can lead to poor performance, increased latency, and even network outages.

Security: SDN-based load balancing also poses several security challenges, such as DDoS attacks, malware infections, and unauthorized access to the network. These security threats can compromise the availability and performance of the network, leading to significant financial losses.

Traffic engineering: SDN-based load balancing techniques must consider different traffic patterns and routing requirements to optimize network performance. However, designing efficient traffic engineering algorithms that can handle complex network topologies and diverse traffic patterns is a challenging task.

Service-level agreements: SDN-based load balancing must also ensure that service-level agreements (SLAs) are met, such as minimum response time, maximum latency, and minimum throughput. Meeting these SLAs can be difficult, especially when dealing with a large number of requests or unpredictable traffic patterns.

Dynamic load balancing for multiple controllers: In large-scale SDN networks with multiple controllers, there is a need for a dynamic load balancing mechanism that can handle burst traffic and adjust controller loads without compromising traffic balancing [63].

Network management for SD-IoT: With the increasing use of IoT devices, there is a need for suitable technologies to manage the massive amounts of data generated by these devices. SDN-based technologies can help distribute and monitor network traffic flows for load balancing and network delay minimization [64].

Hierarchical controller load balancing: Centralized SDN control using a single controller can result in a single point of failure and network collapse. Hierarchical load balancing using a super controller can help maintain global controller load information, but this requires time-consuming LB decisions [65].

Data plane fault tolerance and low-latency load balancing for SD-WAN: SDN provides opportunities to design custom, adaptive routing schemes for low end-to-end latency and failure recovery mechanisms. However, it remains unclear how to pick better paths in real time in the presence of connection failure or congestion [66].

Security challenges: Availability is a key security problem in SDN, and multiple controllers can cause cascade failures. Protecting the controller and building trust between controllers is a key issue, as is providing LB with improved protection against DDoS and long-awaited queues. The scalability of SDN also needs to be improved to prevent targeted attacks that can cause control plane saturation.

7.2 FUTURE RESEARCH DIRECTIONS IN SDN-BASED LOAD BALANCING:

Machine learning: Machine learning techniques can help optimize SDN-based load balancing by predicting traffic patterns and resource utilization, identifying network anomalies, and detecting security threats. These techniques can also help optimize traffic engineering algorithms and improve SLA compliance.

Blockchain: Blockchain technology can provide a secure and decentralized platform for SDN-based load balancing, allowing for greater transparency and accountability in network management. Blockchain can also enable more efficient and secure management of network resources and services.

Fog computing: Fog computing is a distributed computing paradigm that extends cloud computing to the edge of the network, where devices and sensors are located. Fog computing can help improve SDN-based load balancing by enabling faster response times, reducing latency, and improving resource utilization.

Network function virtualization: Network function virtualization (NFV) is the process of virtualizing network functions such as firewalls, load balancers, and routers. NFV can help optimize SDN-based load balancing by providing more flexible and scalable network services and reducing the complexity of network management.

Hybrid solutions: Hybrid SDN-based load-balancing solutions that combine traditional load-balancing techniques with SDN-based approaches can provide more efficient and reliable network management. Hybrid solutions can leverage the benefits of both approaches and provide better performance and scalability than either approach alone.

In conclusion, SDN-based load balancing is a promising technology that offers many benefits, including improved network performance, scalability, and flexibility. However, it also faces several challenges and issues that need to be addressed for its wider adoption and improved performance.

Future research directions in SDN-based load balancing include machine learning, blockchain, fog computing, network function virtualization, and hybrid solutions, among others.

8. PROPOSED ROADMAP

Introducing a new SDN-based load balancing technique, "Adaptive Multi-Objective Load Balancing (AMOLB)." This approach aims to optimize network performance by dynamically balancing multiple objectives, such as latency, throughput, energy efficiency, and resource utilization.

The AMOLB technique leverages the capabilities of SDN to monitor and control the network in real-time while using machine learning to adapt to changing network conditions.

Key components of the AMOLB technique:

- Centralized Controller: The centralized controller is responsible for managing the entire network, collecting real-time network statistics, and making load-balancing decisions based on the multi-objective optimization model.
- Multi-Objective Optimization Model: This model considers multiple objectives and assigns weights to each of them based on the network administrator's preferences or dynamic network requirements. The optimization model generates an optimal set of load-balancing policies that balance the objectives according to their assigned weights.
- Machine Learning Module: The machine learning module continuously analyses network traffic patterns and adjusts the weights assigned to different objectives based on real-time network conditions. It can also predict future traffic patterns and preemptively adjust the load balancing policies to maintain optimal network performance.
- Real-time Network Monitoring: The AMOLB technique relies on real-time network monitoring to collect network statistics, such as latency, throughput, and resource utilization. This data is used to update the multi-objective optimization model and make informed load-balancing decisions.
- Programmable Data Plane: The programmable data plane allows the AMOLB technique to implement dynamic load-balancing policies at the forwarding level. This ensures that traffic is optimally distributed across the network, considering the multiple objectives defined in the optimization model.

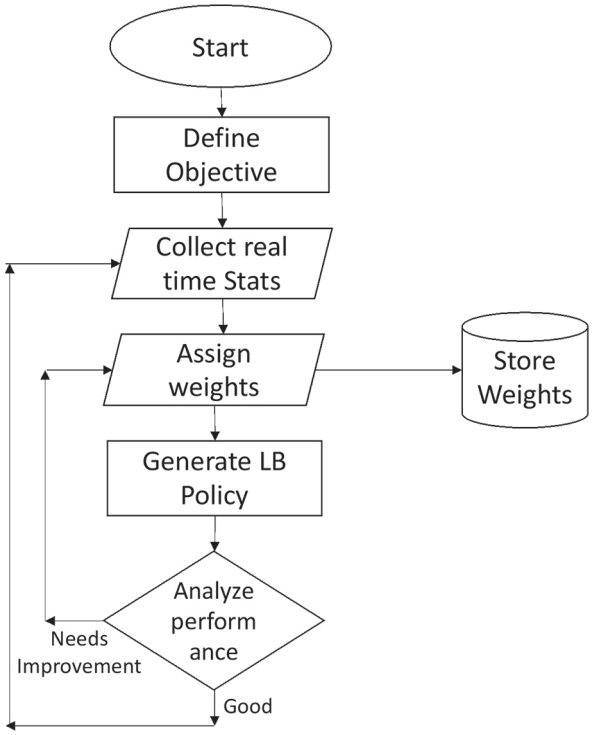

Fig. 9. Adaptive Multi-Object Load Balancing data flow

Fig. 9 above depicts the flow chart involving steps to implement the proposed AMOLB technique:

1. Define the objectives and their respective weights based on network requirements or administrator preferences.
2. Collect real-time network statistics using the centralized controller and programmable data plane.
3. Use the multi-objective optimization model to generate an optimal set of load balancing policies that consider the defined objectives and their weights.
4. Implement the load balancing policies across the network using the programmable data plane.
5. Continuously monitor the network and adjust the weights assigned to different objectives based on real-time network conditions using the machine learning module.
6. Periodically update the load balancing policies to maintain optimal network performance, considering the dynamic nature of network conditions and traffic patterns.

The pseudocode for the same is presented below in Fig. 9.:

```
// Step 1: Define Objectives and Weights

objectives ← defineObjectives() // Returns a dictionary with
objectives and their respective weights

// Step 2: Collect Real-time Network Statistics

networkStats ← collectNetworkStatistics() // Returns real-time
network statistics using the centralized controller and
programmable data plane

// Step 3: Multi-objective Optimization Model

loadBalancingPolicies ← generateOptimalPolicies(objectives,
networkStats) // Generates an optimal set of load balancing policies
based on the objectives and network statistics

// Step 4: Implement Load Balancing Policies

implementPolicies(loadBalancingPolicies) // Implements the load
balancing policies across the network using the programmable data
plane

// Step 5: Continuous Monitoring and Weight Adjustment

while true:

  updatedNetworkStats ← collectNetworkStatistics() // Collects
real-time network statistics

  if networkConditionsChanged(updatedNetworkStats):

    objectives ← adjustWeights(objectives, updatedNetworkStats)
// Adjusts the weights assigned to different objectives based on
real-time network conditions using a machine learning module

  // Step 6: Periodically Update Load Balancing Policies

  if timeToUpdatePolicies():

    loadBalancingPolicies ← generateOptimalPolicies(objectives,
updatedNetworkStats) // Generates updated load balancing policies
based on the adjusted objectives and network statistics

    implementPolicies(loadBalancingPolicies) // Implements the
updated load balancing policies across the network using the
programmable data plane
```

Fig 10. Pseudo-Code implementation for AMOLB

This technique offers a novel approach to SDN-based load balancing by considering multiple objectives and dynamically adapting to changing network conditions. By leveraging the capabilities of SDN and machine learning, the AMOLB technique can provide improved network performance, increased flexibility, and more efficient resource utilization.

9. CONCLUSION

Numerous load-balancing techniques have been proposed and implemented in SDN networks. These techniques can be classified into three main categories: proactive, reactive, and hybrid.

Despite the progress made in load-balancing research for SDN, several challenges and open issues still need to be addressed. For example, load balancing in a multi-controller environment remains a challenge, and new techniques like AMOLB are needed to address this issue. The security of SDN networks is also a concern, and load-balancing techniques need to be developed to prevent cyber-attacks that can compromise the network's availability and performance. Moreover, the Internet of Things (IoT) is expected to generate massive amounts of data, and load-balancing techniques must be developed to handle this data efficiently.

In conclusion, load balancing is an essential aspect of network management in SDN. It involves the intelligent allocation of network resources to improve network performance and avoid congestion. The proposed AMOLB technique offers a novel and efficient approach to SDN-based load balancing, considering multiple objectives and dynamically adapting to changing network conditions. This approach to SDN-based load balancing considers multiple objectives and dynamically adapts to changing network conditions. By leveraging the capabilities of SDN and machine learning, the AMOLB technique can provide improved network performance, increased flexibility, and more efficient resource utilization.

In conclusion, this paper has made a valuable contribution by thoroughly examining the challenges inherent in SDN-based load balancing. Through a comprehensive analysis of issues such as scalability, adaptability, and complexity, this research serves as a guiding resource for future investigations aimed at tackling these obstacles and optimizing the performance of SDN load-balancing solutions.

Moreover, this study has identified promising research avenues that hold the potential to augment the effectiveness and efficiency of load balancing within SDN architectures. By shedding light on these areas, the paper offers a roadmap for researchers to delve deeper and devise innovative solutions that can further enhance load-balancing techniques in SDN environments.

In summary, this research has provided essential insights into the existing challenges and future possibilities of SDN-based load balancing. By addressing these challenges head-on and exploring new research directions, we can collectively foster advancements in SDN technology and contribute to the continuous improvement of network performance and reliability. The findings of this paper lay a solid foundation for the ongoing pursuit of optimized load-balancing solutions in SDN domains, thus paving the way for more robust and efficient networking infrastructures in the future.

Researchers continue to explore new load-balancing techniques, like AMOLB, to address the challenges and open issues associated with SDN networks.

10.REFERENCES:

[1] A. Abdelaziz, A. Fong, A. Gani, U. Garba, S. Khan, A. Akhunzada, H. Talebian, K. K. R. Choo, "Distributed controller clustering in software-defined networks", PLoS One, Vol. e0174715, 2017, p. 12.

[2] N. Rowshanrad, "A survey on SDN, the future of networking", Journal of Advanced Computer Science & Technology, Vol. 3, No. 2, 2014, p. 232.

[3] P. Martinez-Julia, A. Skarmeta, "Empowering the Internet of things with software defined networking", White Paper, IoT6-FP7 European Research Project (accessed: 2020)

[4] A. A. Neghabi, N. Jafari Navimipour, M. Hosseinzadeh, A. Rezaee, "Load Balancing Mechanisms in the Software Defined Networks: A Systematic and Comprehensive Review of the Literature", IEEE Access, Vol. 6, pp. 14159-14178, 2018.

[5] W.-H. Liao, S.-C. Kuai, C.-H. Lu, "Dynamic Load-Balancing Mechanism for Software-Defined Networking", Proceedings of the International Conference on Networking and Network Applications, Hakodate, Japan, 2016, pp. 336-341.

[6] A. Neghabi, N. Navimipour, M. Hosseinzadeh, A. Rezaee, "Load balancing mechanisms in the software-defined networks: A systematic and comprehensive review of the literature", IEEE Access, Vol. 6, 2018, pp. 14159-14178.

[7] N. Rowshanrad et al. "A survey on SDN, the future of networking", Journal of Advanced Computer Science & Technology, Vol. 3, No. 2, 2014, p .232.

[8] M. Saxena, M. Sabharwal, P. Bajaj, "A novel method to enhance the reliability of transmission over secured sd wan overlay", Journal of Theoretical and Applied Information Technology, Vol. 101, No. 14, 2023.

[9] D. Kreutz et al. "Software-defined networking: a comprehensive survey", Proceedings of the IEEE, Vol. 103, No. 1, 2015, pp. 14-76.

[10] M. Jammal, T. Singh, A. Shami, R. Asal, Y. Li, "Software-defined networking: state of the art and research challenges", Computer Networks, Vol. 72, 2014, pp. 74-98.

[11] S. Scott-Hayward, S. Natarajan, S. Sezer, "A Survey of Security in Software Defined Networks", IEEE Communications Surveys & Tutorials, Vol. 18, No. 1, 2016, pp. 623-654.

[12] H. Farhady, H. Y. Lee, A. Nakao, "Software-defined networking: a survey", Computer Networks, Vol. 81, 2015, pp. 79- 95.

[13] A. Mirchev, "Survey of concepts for QoS improvement via SDN", Proceedings of the Seminars Future Internet and Innovative Internet Technologies and Mobile Communications, September 2015, pp. 33-40.

[14] D. A. Karakus, "Quality of Service (QoS) in software defined networking (SDN) a survey", Journal of Network and Computer Applications, Vol. 80, 2017, pp. 200-218.

[15] D. Li, S. Wang, K. Zhu, S. Xia, "A survey of network update in SDN", Frontiers of Computer Science, Vol. 11, No. 1, 2017, pp. 4-12.

[16] A. Mushtaq, R. Mittal, J. McCauley, M. Alizadeh, S. Ratnasamy, S. Shenker, "Datacenter congestion control: Identifying what is essential and making it practical", ACM SIGCOMM Computer Communication Review, Vol. 49, No. 3, 2019, 32-38.

[17] O. Michel, E. Keller, "SDN in wide-area networks: a survey", Proceedings of the 4th International Conference on Software Defined Systems, Valencia, Spain, 8-11 May 2017, pp. 37-42.

[18] T. Hu et al. "Controller load balancing mechanism based on distributed policy in SDN", ACTA ELECTONICA SINICA, Vol. 46, No. 10, 2018, p. 2316.

[19] D. T. T. Hien, T. D. Ngo, D. D. Le, H. Sekiya, V. H. Pham, K. Nguyen, "A software defined networking approach for guaranteeing delay in Wi-Fi networks", Proceedings of the 10th International Symposium on Information and Communication Technology, December 2010, pp. 191-196.

[20] J. Cui, Q. Lu, H. Zhong, M. Tian, L. Liu, "A load-balancing mechanism for distributed SDN control plane using response time", IEEE Transactions on Network and Service Management, Vol. 15, No. 4, 2018, pp. 1197-1206.

[21] L. Li, Q. Xu, "Load balancing research in SDN: A survey", Proceedings of the IEEE International Conference on Electronics Information and Emergency Communication, Macau, China, 21-23 July 2017, pp. 403-408.

[22] M. Alizadeh et al. "CONGA: Distributed congestion-aware load balancing for datacenters", Proceedings of the ACM conference on SIGCOMM, August 2014, pp. 503-514.

[23] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, A. Vahdat, "Hedera: Dynamic flow scheduling for data centre networks", Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation, 2010.

[24] A. S. Thorat, S. K. Sonkar, "A Review on Energy Efficient Load Balancing Techniques for Secure and Reliable Cloud Eco-system", International Journal of Advance Research and Innovative Ideas in Education, Vol. 2, No. 1, 2016.

[25] M. Yu, J. Rexford, M. J. Freedman, J. Wang, "Scalable flow-based networking with DIFANE", Proceedings of the ACM SIGCOMM 2010 Conference on SIGCOMM, 2010, pp. 351-362.

[26] B. Chang, A. Akella, L. D'Antoni, K. Subramanian, "Learned Load Balancing", Proceedings of the 24th International Conference on Distributed Computing and Networking, January 2023, pp. 177-187.

[27] N. Katta, M. Alizadeh, J. Rexford, D. Walker, "Infinite cache flow in software-defined networks", Proceedings of the third workshop on Hot topics in Software Defined Networking, 2016.

[28] R. Pries, M. Jarschel, D. Schlosser, M. Klopf, & P. Tran-Gia, "Power consumption analysis of data center architectures", Proceeding of Green Communications and Networking: First International Conference, Revised Selected Papers, October 2011, pp. 114-124.

[29] M. Al-Fares, M. Loukissas, A. Vahdat, "A scalable, commodity data center network architecture", Proceedings of the ACM SIGCOMM Conference on Data Communication, 2018, pp. 63-74.

[30] C. Lim, "Enhancing robustness of per-packet load-balancing for fat-tree", Applied Sciences, Vol. 11, No. 6, 2021, p. 2664.

[31] J. Tao, S. Liu, C. Liu, "A Traffic Scheduling Scheme for Load Balancing in SDN-Based Space-Air-Ground Integrated Networks", Proceedings of the IEEE 23rd International Conference on High Performance Switching and Routing, Taicang, Jiangsu, China, 6-8 June 2022, pp. 95-100.

[32] J. Chen et al. "ALBRL: Automatic Load-Balancing Architecture Based on Reinforcement Learning in Software-Defined Networking", Wireless Communications and Mobile Computing, Vol. 2022, 2022, pp. 1-17.

[33] J. Pei, P. Hong, M. Pan, J. Liu, J. Zhou, "Optimal VNF placement via deep reinforcement learning in SDN/NFV-enabled networks", IEEE Journal on Selected Areas in Communications, Vol. 38, 2019, p. 263-278.

[34] P. B. M. C. Saxena, "Evolution of Wide Area network from Circuit Switched to Digital Software defined Network", Proceedings of the International Conference on Technological Advancements and Innovations, Tashkent, Uzbekistan, 10-12 November 2021.

[35] Z. Shang, W. Chen, Q. Ma, B. Wu, "Design and implementation of server cluster dynamic load balancing based on OpenFlow", Proceedings of the International Joint Conference on Awareness Science and Technology & Ubi-Media, Aizu-Wakamatsu, Japan, 2-4 November 2013, pp. 691-697.

[36] H. Babbar, S. Parthiban, G. Radhakrishnan, S. Rani, "A genetic load balancing algorithm to improve the QoS metrics for software defined networking for multimedia applications", Multimedia Tools and Applications, Vol. 81, No. 7, 2022, pp. 9111-9129.

[37] T. Koponen et al. "Onix: A distributed control platform for large-scale production networks", Proceedings of the 9th USENIX conference on Operating systems design and implementation, 2010.

[38] C. Trois, M. D. Del Fabro, L. C. de Bona, M. Martinello, "A survey on SDN programming languages: Toward a taxonomy", IEEE Communications Surveys & Tutorials, Vol. 18, No. 4, pp. 2687-2712.

[39] K. E. Jungmin, "SDN in wide-area networks: a survey", Proceedings of the 4th International Conference on Software Defined Systems, 2017, pp. 37-42.

[40] K. Benzekki, A. El Fergougui, A. E. Elalaoui, "Software-defined networking (SDN): a survey", Security and communication networks, Vol. 9, No. 18, 2016, pp. 5803-5833.

[41] J. Son, R. Buyya, "A taxonomy of software-defined networking (SDN)-enabled cloud computing", ACM Computing Surveys, Vol. 51, No. 3, 2018, pp. 1-36.

[42] L. Zhuo, C. L. Wang, F. C. Lau, "Load balancing in distributed web server systems with partial document replication", Proceedings of the International Conference on Parallel Processing, Vancouver, BC, Canada, 21 August 2022, pp. 305-312.

[43] S. Abuthahir, S. C. B. Jaganathan, R. Saha, "An Efficient Enhanced Dynamic Load Balancing Weighted Round Robin Algorithm for Virtual Machine in Cloud Computing", Journal of Algebraic Statistics, Vol. 13, No. 2, 2022, pp. 2121-2128.

[44] C. Yu, Z. Zhao, Y. Zhou, H. Zhang, "Intelligent Optimizing Scheme for Load Balancing in Software Defined Networks", Proceedings of the IEEE 85th Vehicular Technology Conference, Sydney, NSW, Australia, 4-7 June 2017, pp. 1-5.

[45] S. Kaur, J. Singh, N. S. Ghumman, "Network programmability using POX controller", Proceeding of the International Conference on Communication, Computing & Systems, Vol. 138, August 2014, p. 70

[46] C. Fancy, & M. Pushpalatha, "Performance evaluation of SDN controllers POX and floodlight in mininet emulation environment", Proceedings of the International Conference on Intelligent Sustainable Systems, Palladam, India, 7-8 December 2017, pp. 695-699.

[47] F. Benamrane, R. Benaini, "An East-West interface for distributed SDN control plane: Implementation and evaluation", Computers & Electrical Engineering, Vol. 57, 2017, pp. 162-175.

[48] R. Schmidt, C. Y. Chang, N. Nikaein, "FlexVRAN: A Flexible Controller for Virtualized RAN Over Heterogeneous Deployments", Proceedings of the IEEE International Conference on Communications, Shanghai, China, 20-24 May 2019, pp. 1-7.

[49] N. L. Van Adrichem, C. Doerr, F. A. Kuipers, "Opennetmon: Network monitoring in OpenFlow software-defined networks", Proceedings of the IEEE Network Operations and Management Symposium, May 2014, pp. 1-8.

[50] S. Mohmmad, M. A. Shaik, K. Mahender, R. Kanakam, B. P. Yadav, "Average Response Time (ART): Real-Time Traffic Management in VFC Enabled Smart Cities", IOP Conference Series: Materials Science and Engineering, Vol. 981, No. 2, 2020, p. 022054.

[51] R. Jamal, L. C. Fourati, "Implementing shortest path routing mechanism using OpenFlow POX controller", Proceedings of the International Symposium on Networks, Computers and Communications, Hammamet, Tunisia, 17-19 June 2014, pp. 1-6.

[52] A. Jalili, M. Keshtgari, R. Akbari, "A new framework for reliable control placement in software-defined networks based on multi-criteria clustering approach", Soft Computing, Vol. 24, 2020, p. 2897- 2916.

[53] A. Jalili, M. Keshtgari, R. Akbari, R. Javidan, "Multi criteria analysis of controller placement problem in software defined networks", Computer Communications, Vol. 133, 2019, p. 115-128.

[54] S. Jamali, A. Badirzadeh, M. S. Siapoush, "On the use of the genetic programming for balanced load distribution in software defined networks", Digital Communications and Networks, Vol. 5, 2019, pp. 288-296.

[55] A. Javadpour, "Providing a way to create balance between reliability and delays in SDN networks by using the appropriate placement of controllers", Wireless Personal Communications, Vol. 110, 2020, pp. 1057-1071.

[56] X. Jia, Y. Jiang, Z. Guo, Z. Wu, "Reducing and balancing flow table entries in software-defined networks", Proceedings of the IEEE 41st Conference on Local Computer Networks, Dubai, United Arab Emirates, 7-10 November 2016, pp. 575-578.

[57] X. Jia, Y. Jiang, Z. Guo, J. Sun, "A low overhead flow-holding algorithm in software-defined networks", Computer Networks, Vol. 123, 2017, pp. 170-180.

[58] E. R. Jimson, K. Nisar, M. H. A. Hijazi, "The state of the art of software defined networking (SDN) issues in current network architecture and a solution for network management using the SDN", International Journal of Technology Diffusion, Vol. 10, 2019, pp. 33-48.

[59] S. Askar, "Adaptive load balancing scheme for data centre networks using software defined network", Science Journal of University of Zakho, Vol. 4, 2016, pp. 275-286.

[60] M. Priyadarshini, J. Mukherjee, P. Bera, S. Kumar, A. Jakaria and M. Rahman, "An adaptive load balancing scheme for software-defined network controllers", Computer Networks, Vol. 164, No. 106918, 2019.

[61] L. Wang, W. Mao, J. Zhao, Y. Xu, "A double deep Q-learning approach to online fault-tolerant SFC placement", IEEE Transactions on Network and Service Management, Vol. 18, No. 1, 2021, pp. 118-132.

[62] N. Hai, D. Kim, "Efficient load balancing for multi-controller in SDN-based mission-critical networks", Proceedings of the IEEE 14th International Conference on Industrial Informatics, Poitiers, France, 19-21 July 2016, p. 420-425.

[63] M. Mathur, M. Madan, M. C. Saxena, "A Proposed Architecture for Placement of Cloud Data Centre in Software Defined Network Environment", International Journal of Engineering and Advanced Technology, Vol. 1, No. 2, 2012, pp. 104-116.

[64] J. Zhao, M. Tong, H. Qu, J. Zhao, "An Intelligent Congestion Control Method in Software Defined Networks", Proceedings of the IEEE 11th International Conference on Communication Software and Network, Chongqing, China, 12-14 June 2019, pp. 51-56.

[65] C. Yu, J. Lan, Z. Guo, Y. Hu, "Optimizing the routing in software-defined networks with deep reinforcement learning", IEEE Access, Vol. 6, 2018, pp. 64533-64539.

[66] P. Sun, Y. Hu, J. Lan, L. Tian, M. T. Chen, "Time-relevant deep reinforcement learning for routing optimization", Future Generation Computer Systems, Vol. 99, 2019, pp. 401-409.