Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2, 218-230
2025
Publisher: Learning Gate
DOI: 10.55214/25768484.v9i2.4456

Leveraging SDN for scalable and sustainable fat tree networks: A multi-objective performance and energy efficiency evaluation of an 8-pod fat tree data center

Sura Fawzi[1], Norashidah Md Din[2*]
[1,2]Institute of Energy Infrastructure, Universiti Tenaga Nasional, 43000 Kajang, Selangor, Malaysia; surafawzi89@com (S.F.) norashidah@uniten.edu.my (N.M.D.).

Abstract: Modern data centers increasingly rely on Software-Defined Networking (SDN) to address challenges related to scalability, performance, and efficient resource management. This research investigates the scalability and performance optimization of fat-tree topology within SDN environments, focusing on the impact of a sleep mode technique on network efficiency and energy consumption. Using the Mininet emulator, an 8-pod fat-tree network is simulated and compared against traditional routing methods like Equal-Cost Multi-Path (ECMP) and Dijkstra's algorithm. The findings show that the sleep mode technique improves bandwidth utilization and throughput by reducing energy consumption during low-traffic periods without significantly affecting data flow. In contrast, Dijkstra's algorithm exhibited reduced throughput due to inefficient path management, while ECMP did not fully optimize load balancing or energy efficiency. The sleep mode approach efficiently redistributes traffic across active switches, preventing congestion and outperforming both Dijkstra and ECMP in terms of average load. The results demonstrate that implementing sleep mode in fat-tree SDN networks enhances both network performance and energy efficiency, offering a practical solution for large-scale data center operations. These findings provide valuable insights for optimizing SDN-based traffic management in modern network infrastructures.

Keywords: *8-pod Fat Tree, Modern Data Centers, Scalability, Sleep Mode, Software-Defined Networking (SDN).*

1. Introduction

In recent years, Software-Defined Networking (SDN) has emerged as a revolutionary approach to network management, enabling centralized control, flexibility, and automation. SDN decouples the control plane from the data plane, offering network administrators the ability to program and dynamically manage network behavior via software applications [1-5]. This has led to improved network efficiency, scalability, and fault tolerance, particularly in large-scale data center networks. One of the most widely adopted network topologies in modern data centers is the fat tree, which provides scalable, fault-tolerant, and efficient communication patterns for large numbers of hosts. The integration of SDN with the fat tree topology can significantly enhance the performance of data centers by enabling dynamic traffic management and resource allocation, addressing the challenges posed by high traffic loads and network congestion.

The 8-pod fat tree topology is a scaled-up extension of the 4-pod fat tree architecture used in data centers, where the network is divided into multiple pods [6, 7] each containing access, aggregation, and core switches. This hierarchical structure ensures that there are multiple paths between any two nodes in the network, offering high fault tolerance and efficient traffic distribution [8, 9]. The 8-pod fat tree refers to a k=8 network composed of 8 pods, each with 2 switches in the access layer and 2 switches in the aggregation layer linked to 4 core switches [10-12]. Further scaling up of the fat tree is possible to fit large-scale data centers that have multiple hosts and massive traffic. The design ensures that the fat tree has efficient load balancing, fault tolerance, and

History: Received: 15 November 2024; Revised: 17 January 2025; Accepted: 22 January 2025; Published: 27 January 2025
* Correspondence: norashidah@uniten.edu.my

performance optimization of the network, hence being specifically fit for a high-demand environment [13-15]. The topology also facilitates load balancing and fault tolerance through its redundant paths, ensuring that network performance remains resilient even in the event of switch or link failures.

Mininet, a widely used network emulator, provides an ideal environment for simulating and testing SDN-based networks like the 8-pod fat tree topology [16-19]. By enabling the creation of virtual networks with real OpenFlow switches, Mininet allows researchers and practitioners to experiment with SDN controllers, such as ONOS controller and OpenFlow protocol to manage the flow of traffic and control network behavior. Mininet's ability to scale from small topologies to large-scale, complex networks makes it a valuable tool for evaluating SDN protocols and testing network optimization techniques, such as load balancing, traffic engineering, and fault tolerance.

One of the key benefits of combining SDN with an 8-pod fat tree topology is the ability to dynamically control traffic paths through the network. SDN controllers can use protocols such as OpenFlow to adjust the flow tables in real-time, enabling efficient routing of data and minimizing network congestion. By dynamically selecting paths based on network conditions, such as congestion or link failure, SDN allows for more flexible and efficient traffic management compared to traditional network protocols like Spanning Tree Protocol or Routing Information Protocol. This can significantly reduce packet loss and latency while improving overall throughput, especially in environments with fluctuating traffic loads.

This article presents an in-depth analysis of the 8-pod fat tree topology within an SDN framework, using Mininet as the simulation platform. The primary focus is on the design, implementation, and evaluation of SDN-driven traffic management techniques in an 8-pod fat tree network. Using the SDN controller, the paper explores how dynamic traffic management, load balancing, and fault tolerance can be implemented to optimize network performance. Additionally, the potential role of energy-saving techniques like sleep mode is briefly discussed, although it remains a secondary consideration to the primary analysis of SDN-based performance improvements.

2. Literature Review

Traffic management and energy efficiency has become a critical consideration in modern data center networks, where large numbers of switches and links contribute to significant power consumption. Techniques such as sleep mode have been proposed as a means of optimizing energy usage in SDN-enabled networks. By turning off or putting idle network components (such as switches or links) into low-power states during periods of low traffic [20-22] data centers can reduce their overall energy consumption without sacrificing performance, it represents an important optimization that can be integrated into SDN networks to improve energy efficiency, particularly in large-scale, high-density topologies like the fat tree.

Fat tree with 8-pod k-array topology is a highly scalable, fault-tolerant network architecture often used in modern data centers to optimize traffic management and ensure high bandwidth availability [23, 24]. Based on the k-ary fat tree model, where k=8, the network consists of 8-pods, each comprising access, aggregation, and core switches. This hierarchical structure provides redundancy and multiple paths for data, ensuring high availability and preventing congestion, crucial for large-scale SDN environments. The fat tree network design allows for effective load balancing and fault tolerance, as traffic can be dynamically rerouted in case of link or switch failures. Additionally, SDN controllers like OpenFlow can leverage the topology's redundancy for real-time traffic optimization, balancing load and avoiding network bottlenecks.

In SDN, the controller manages the flow of data by programming flow tables in the switches, dynamically adjusting routes to ensure optimal performance based on metrics like latency, throughput, and link utilization [25]. The ability of SDN to react to network conditions in real time enhances the scalability and resilience of the 8-pod fat tree topology, making it ideal for modern data center environments. The use of SDN allows for dynamic path management,

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 218-230, 2025
DOI: 10.55214/25768484.v9i2.4456

enhancing network performance while enabling energy-saving strategies such as sleep mode to minimize power consumption during periods of low traffic demand.

Mininet was chosen for testing the 8-pod fat tree topology due to its lightweight, flexible, and cost-effective nature. It allows for the creation of virtual networks, simulating the fat tree structure with the required number of pods, switches, and hosts. The virtual environment enables efficient experimentation with SDN configurations and performance metrics, providing valuable insights without the need for extensive physical infrastructure [26].

Recent works have explored various aspects of SDN in fat tree topologies, particularly in relation to traffic engineering, fault tolerance, and energy efficiency. Al-Fares, et al. [27] for instance, introduced the fat tree topology as a scalable and fault-tolerant architecture suitable for data centers. Research work by Jin, et al. [28] and Kumari, et al. [29] extended this work by showing how SDN can improve traffic engineering and load balancing in fat tree networks, enhancing both resource utilization and network performance. Energy efficiency issues were investigated by Zhang, et al. [30] and Cheng, et al. [31] using sleep mode in SDN that deactivate switches and reduce power usage during off-peak hours without compromising network reliability. Several studies have demonstrated the energy-saving potential such as Safa, et al. [32] where the study utilized SDN-enabled sleep mode techniques in fat tree networks. Work in Hassan, et al. [33] developed frameworks that dynamically adjust the operational state of switches, ensuring energy efficiency without affecting performance. A comprehensive model for optimizing energy usage in 8-pod fat tree topologies is presented in Sood and Sood [34] while Alizadeh, et al. [35] proposed hybrid SDN architectures combining traditional and SDN-based approaches to optimize both performance and energy consumption.

Further research has focused on the scalability of SDN-based 8-pod fat tree networks. Zhang, et al. [36] showed how SDN controllers can manage energy-saving algorithms to reduce power usage during low-traffic periods. Similarly, work in Ghorbani, et al. [37] and Michelogiannakis, et al. [38] proposed dynamic power management strategies, leveraging real-time traffic conditions to reduce energy consumption in large-scale data centers. These studies highlight how SDN's programmability allows for fine-grained control over energy optimization, enabling significant reductions in the network's overall power footprint. The integration of SDN's flexibility with sleep mode techniques is gaining recognition as a promising solution for achieving both high performance and energy efficiency in large-scale data center networks.

Dynamic routing methods like Open Shortest Path First (OSPF) within SDN were studied by Rego, et al. [39]. When comparing SDN with traditional networks, the study highlighted improvements in network characteristics such as arrival rate, round-trip time, and QoS. Although OSPF in SDN did not achieve the same performance as conventional networks, it still led to relatively better QoS performance. In Dobrijevic, et al. [40] an SDN architecture in a smart factory's network was proposed, offering adaptable data flow management during system failures. Another study by Haile and Zhang [41] examined data center load balancing through four distinct scenarios and focusing on queue variations over time to improve QoS. The results showed increased bandwidth and throughput but with a rise in latency.

Pang, et al. [42] introduced the Horizon model, which enhances SDN's QoS by using a Markov method to predict link usage and monitor network activity. The findings indicated that Horizon reduces traffic delays in data centers to meet QoS requirements, although it doubled SDN's delay time. Additionally, Huang [43] employed Ternary Content Addressable Memory (TCAM) to handle user requests, with less emphasis on QoS. Multicast and unicast requests were used to improve throughput.

Recent studies have also explored bio-inspired algorithms. Researchers in Lv, et al. [44] extensively examined how bio-inspired techniques can be applied in SDN controllers. Another notable approach by Ammal and Chandra [45] suggested ant colony optimization for optimizing flow route discovery in SDN networks, particularly for multimedia services. In Ammal, et al. [46] a bio-inspired algorithm was proposed combining smell-sensing agents and Dijkstra's algorithm,

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 218-230, 2025
DOI: 10.55214/25768484.v9i2.4456

inspired by the scent detection abilities of dogs. This approach enhances the optimization of network paths. Similarly, Sabr, et al. [47] proposed a bio-inspired method for optimizing traffic engineering in an Internet-of-Vehicles environment using an SDN controller. The researchers applied the artificial bee colony algorithm to perform intelligent traffic management in dynamically configured networks. The results showed that the artificial bee colony approach outperformed other methods.

This research expands on existing studies by exploring the application of sleep mode in a large-scale fat tree network, i.e. 8-pod fat tree SDN topology, focusing on optimizing energy consumption through sleep mode strategies using a bio-inspired metaheuristic technique. The goal is to offer sustainable solutions for modern data center environments.

3. Materials and Methods

In this study, we investigate the potential of utilizing sleep mode in SDN to reduce energy consumption in large-scale data centers. Specifically, we explore its integration within the 8-pod fat tree topology, a hierarchical network structure known for its scalability and fault tolerance. By incorporating sleep mode algorithm into large scale 8-pods fat tree SDN data center, we aim to optimize bandwidth monitoring, sleep mode decisions, and path selection, ensuring that energy consumption is minimized without compromising network performance. The performance of this approach is evaluated through Mininet emulation, where we compare the results to traditional routing techniques such as Equal-Cost Multi-Path (ECMP) and Dijkstra's algorithm.

3.1. Step 1: Network Design and Implementation

This section details the implementation of the 8-pod fat tree topology in Mininet for simulating an SDN-enabled data center network. The fat tree is a scalable, fault-tolerant clos network comprising three hierarchical layers: access, aggregation, and core. In the 8-pod configuration, each pod includes access switches, aggregation switches, and core switches, all interconnected to provide efficient traffic management and fault tolerance. Mininet emulates this topology, with SDN controllers such as OpenFlow or ONOS managing traffic flow and routing decisions across the network. Each switch, whether access, aggregation, or core, is programmed with flow tables that specify how incoming packets are processed based on packet headers like IP or VLAN IDs.

The emulation experiment was conducted using an 8-pods large scale fat-tree topology with the SDN ONOS controller. The network was built using low-cost commodity Ethernet switches. In a k-ary fat-tree network, the topology consists of k pods, with each pod containing two layers of switches. Each switch has k ports, where the edge layer of each switch connects to hosts, while the remaining ports link to aggregation layer switches. At the core of the network, core switches are connected to the aggregation layer switches, forming the central layer of the structure which is then connected to the edge switches and hosts as depicted in Figure 1.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 218-230, 2025
DOI: 10.55214/25768484.v9i2.4456

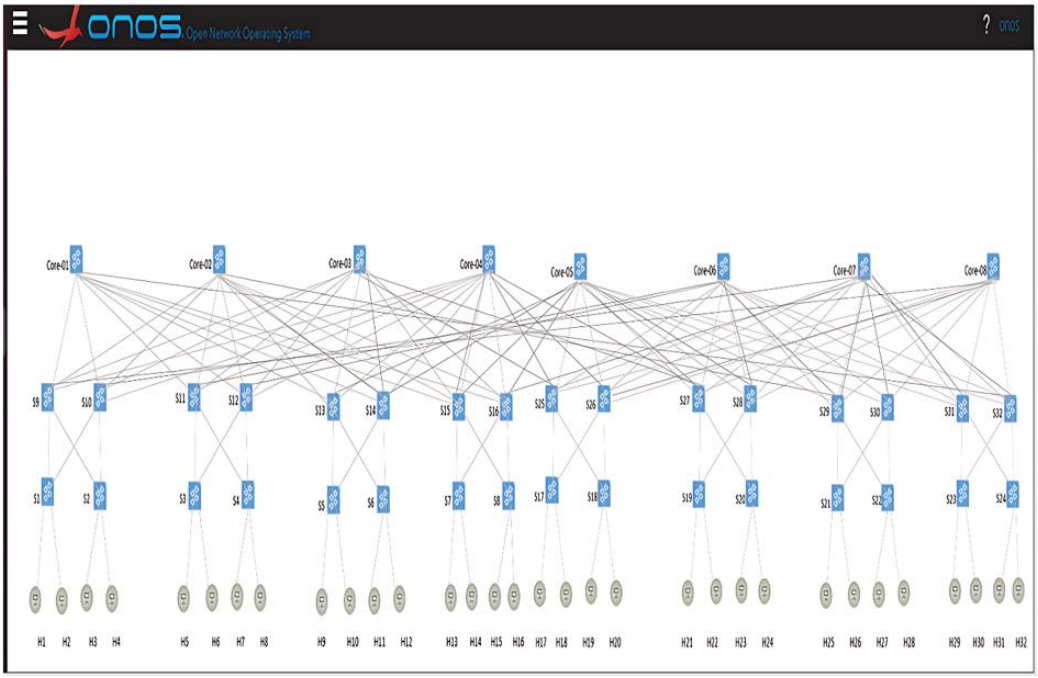

Figure 1.
Fat Tree Network when K= 8-pods.

3.2. Step 2: Integrating Mininet with Sleep Mode Implementation in Large-Scale Fat Tree Networks

Mininet integrates with the SDN through the OpenFlow and ONOS controller to manage network flows, optimize routing, and monitor performance. OpenFlow enables the SDN controller to remotely program switch flow tables for dynamic traffic management. In this research, we use OpenFlow 1.3 to control the flow tables in the 8-pod fat tree topology. When a packet arrives at a switch, the switch checks its flow table. If no matching entry is found, the switch sends a packet-in message to the controller, which computes the action and installs the flow entry. We use the ONOS controller, ideal for managing large-scale networks, which supports distributed control and high availability. ONOS provides centralized network visibility and real-time traffic control via OpenFlow.

The ONOS controller handles the flow table configuration, dynamic path calculation, and load balancing. This setup allows for real-time traffic management and performance testing in the 8-pod fat tree topology. Energy efficiency is also considered, with sleep mode techniques to reduce power consumption. The SDN controller can dynamically manage sleep mode by monitoring traffic patterns and directing switches to low-power states during low traffic periods, optimizing energy use without compromising performance. This approach, combining SDN flexibility with the scalable fat tree architecture, enables efficient traffic management and network optimization in large-scale data center networks.

The sleep mode framework in an SDN-enabled data center (SDN-DC) uses a metaheuristic sleep mode technique. This framework has two key phases: marking and matching. In the marking phase, the SDN controller monitors the fat tree network, identifying switches with low traffic and marking them as sleep candidates. These switches are then considered for sleep mode. In the matching phase, the controller evaluates the marked switches and reroutes traffic to the best available active path based on current traffic conditions. The SDN architecture, consisting of the application, control, and data planes, enables this process. The control plane uses OpenFlow APIs to manage flow tables and make decisions about sleep mode activation and traffic rerouting, optimizing energy efficiency while maintaining network performance.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 218-230, 2025
DOI: 10.55214/25768484.v9i2.4456

The sleep mode algorithm within an SDN-enabled data center uses a metaheuristic approach, where the SDN controller C dynamically manages network switches S based on traffic conditions and energy level. Figure 2 provides the pseudocode for implementing this algorithm, utilizing the concept of sleep mode for energy conservation. Controller C monitors traffic on each switch S in the network. If traffic on a switch is low, the controller designates the switch as a sleep candidate and activates sleep mode, while tracking its energy consumption.

In sleep mode, incoming packets are redirected to neighboring active switches. For active switches, packets are forwarded as usual, and energy consumption is calculated for both forwarding and redirection. The controller periodically recalculates the optimal routing based on traffic patterns and ensures packet rerouting through active paths when a switch is asleep. Over a set period T, the controller computes the average energy consumption, E, for the network to evaluate energy efficiency. This data provides an energy efficiency metric, reflecting the success of the sleep mode strategy in reducing power usage. The system continues to operate with periodic updates to sleep mode decisions based on network traffic changes.

For each switch S in network graph G:

```
    C.broadcastIncomingPackets(S)
    C.monitorSwitchStatus(S)
    if traffic(S) is low:
       C.putSwitchToSleep(S)
       C.calculateEnergyUsage(S)
       C.redirectPacketsToNeighboringActiveNode(S)
    else:
       C.forwardPackets(S)
       C.calculateEnergyUsage(S)
    C.monitorNetworkActivity()
    C.findOptimalRoutingPath(S)
    avgEnergy = C.calculateAverageEnergyUsage(timeT)
    end for
    captureEnergyForNetwork()
    returnEnergyMetrics()
 end if
 end for
```

Figure 2.
The Metaheuristic Sleep Mode Algorithm.

3.3. Step 3: Modeling Analysis

To emulate the SDN fat tree topology with sleep mode in Mininet, firstly we created the fat-tree topology, and secondly, we implemented the sleep mode algorithm. The network and algorithm model analysis were conducted on an Ubuntu 14.04 system with the ONOS controller running in a VirtualBox environment. The host machine used for the simulation is equipped with an Intel Core i7-8565U CPU (1.80 GHz) and 16 GB of RAM. To evaluate the proposed sleep mode strategy, a k=8 fat tree network was created, consisting of 32 switches and 32 hosts. The network was controlled by ONOS version 2.5.1, using Open vSwitch (OVS) with OpenFlow 1.3, and traffic was generated using the Iperf tool to

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 218-230, 2025
DOI: 10.55214/25768484.v9i2.4456

simulate TCP and UDP traffic. During the simulation, key SDN network metrics such as throughput, bandwidth utilization, link usage, and latency were monitored. Connectivity between data center hosts was analyzed while the controller interacted with the Mininet topology. Wireshark, a network protocol analyzer, was used to capture and examine network packets in real time.

4. Results and Discussion

The effectiveness of the proposed method was assessed by analyzing the performance metrics which are the throughput, bandwidth utilization, energy efficiency and average load. The results indicate significant improvements in the SDN-DC, particularly in throughput, bandwidth, average delay, and energy savings. This following sub-sections present the outcomes of the emulations conducted with the 8-pod fat tree topology in Mininet.

4.1. Throughput

Throughput is a crucial metric that measures the network's ability to efficiently handle data traffic. In large-scale data centers, maintaining high throughput is vital to meet the demands of diverse applications, particularly in high-traffic scenarios. Figure 3 presents a comparison of throughput between the sleep mode strategy and the ECMP technique, highlighting the impact of our method on network efficiency. Figure 3 shows that our proposed sleep mode technique outperforms the ECMP technique, in maximizing throughput, especially under heavy traffic. While ECMP balances traffic across multiple paths, it lacks the fine-grained optimization that the sleep mode provides, resulting in lower throughput as the network scales. In contrast, our method efficiently manages traffic flows, ensuring higher throughput and better load distribution in large-scale data centers. Additionally, a comparison with Dijkstra's algorithm Figure 4 highlights the advantages of our approach. While Dijkstra excels at shortest-path routing, it struggles to maintain high throughput in large networks like the 8-pod fat-tree topology. The sleep mode method, on the other hand, adapts dynamically to traffic conditions, optimizing throughput as the network size and traffic volume grow. Figure 4, illustrates the throughput measurements of sleep mode versus Dijkstra method.

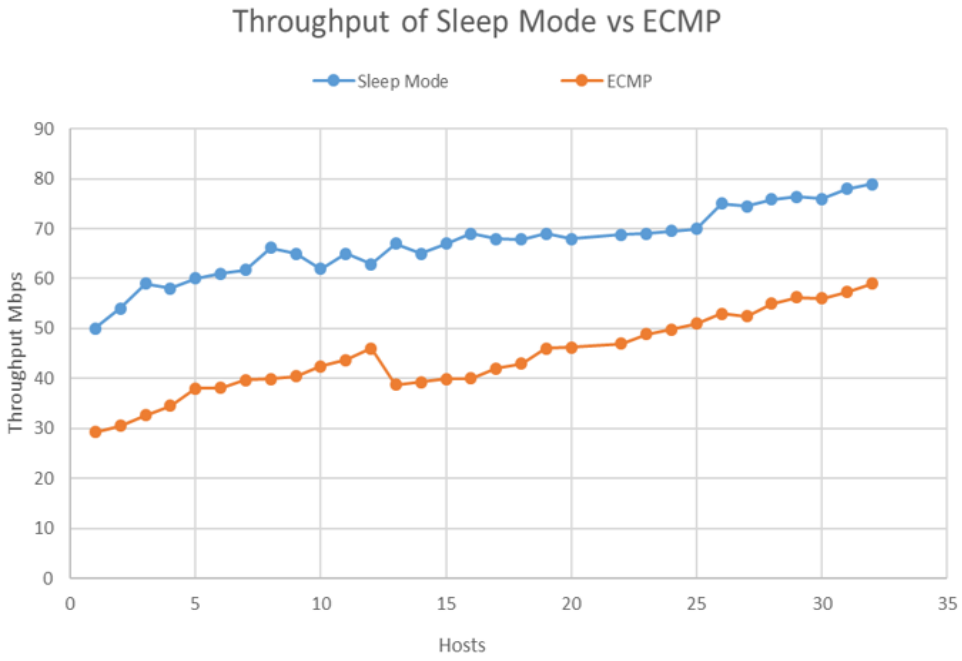

Figure 3.
Throughput of sleep mode versus ECMP.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 218-230, 2025
DOI: 10.55214/25768484.v9i2.4456

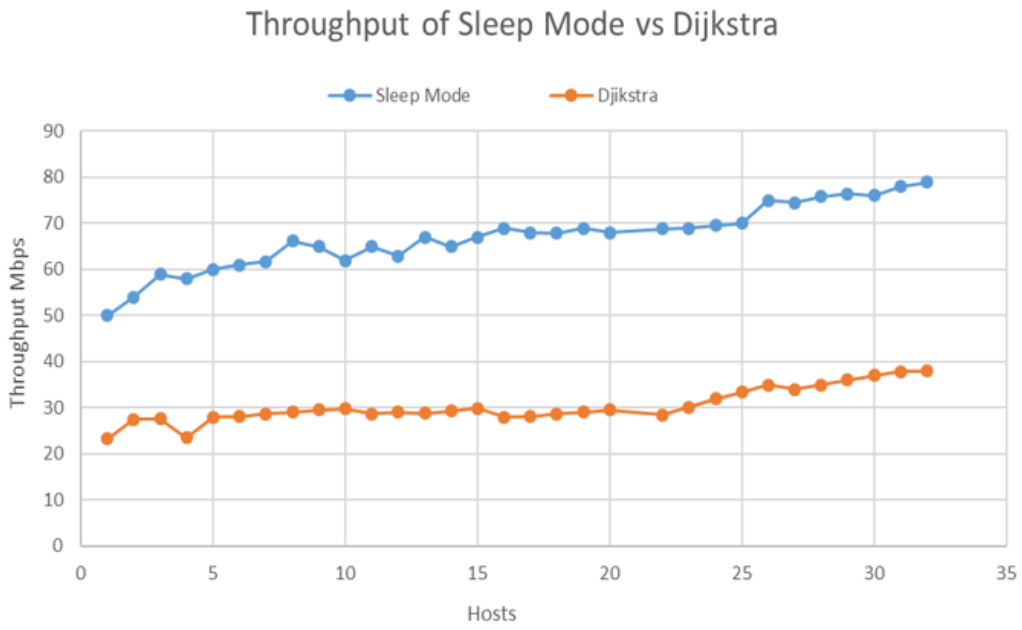

Figure 4.
Throughput of Sleep Mode versus Djikstra

4.2. Bandwidth Utilization

Figure 5 depicts the results of bandwidth utilization with the 8-pod fat-tree network configuration. The Mininet 8-pod fat-tree network environment provides an ideal testing ground for understanding how different traffic management algorithms perform under increased network demands. In this case we see that traffic management via sleep mode improves the bandwidth utilization in the network. This enhancement is critical for large data centers where efficient bandwidth utilization is essential for supporting diverse applications.

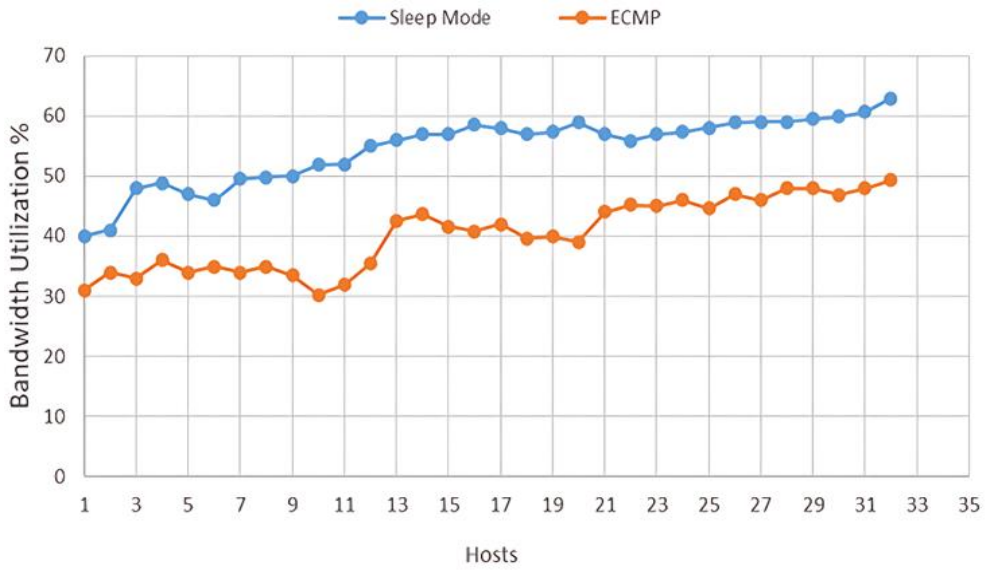

Figure 5.
Bandwidth utilization using sleep mode versus ECMP.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 218-230, 2025
DOI: 10.55214/25768484.v9i2.4456

In Figure 6, a comparison of the metaheuristic sleep mode technique with Dijkstra's algorithm further highlights the advantages of the sleep mode method. The sleep mode approach consistently delivered better bandwidth utilization performance, particularly in large-scale data centers, where Dijkstra struggled with scalability.

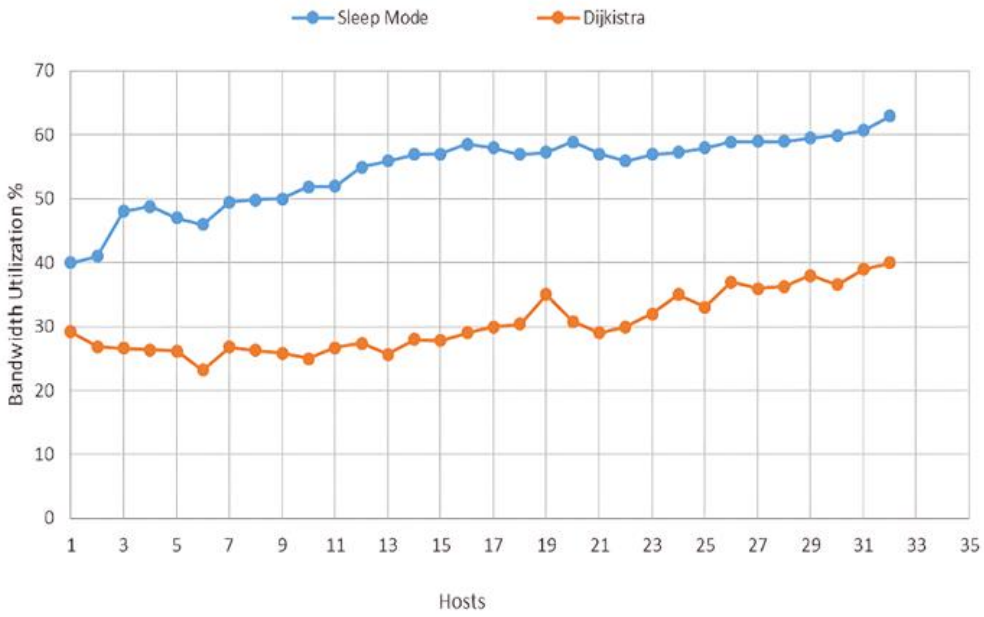

Figure 6.
Bandwidth Utilization Using Sleep Mode versus Dijkstra

4.3. Energy Efficiency

As illustrated in Figure 7, the sleep mode technique demonstrates far greater energy efficiency compared to ECMP. This is achieved by minimizing unnecessary active network components, allowing only the required ones to operate, and optimizing traffic flow effectively. Consequently, energy consumption is reduced, leading to lower operational costs and promoting a more sustainable SDN-based data center. Figure 8 demonstrates the energy consumptions of 8-pods fat tree sleep mode compared with the Dijkstra's method. The results showed that the sleep mode consumes less energy under different traffic loads compared to Dijkstra's.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 218-230, 2025
DOI: 10.55214/25768484.v9i2.4456

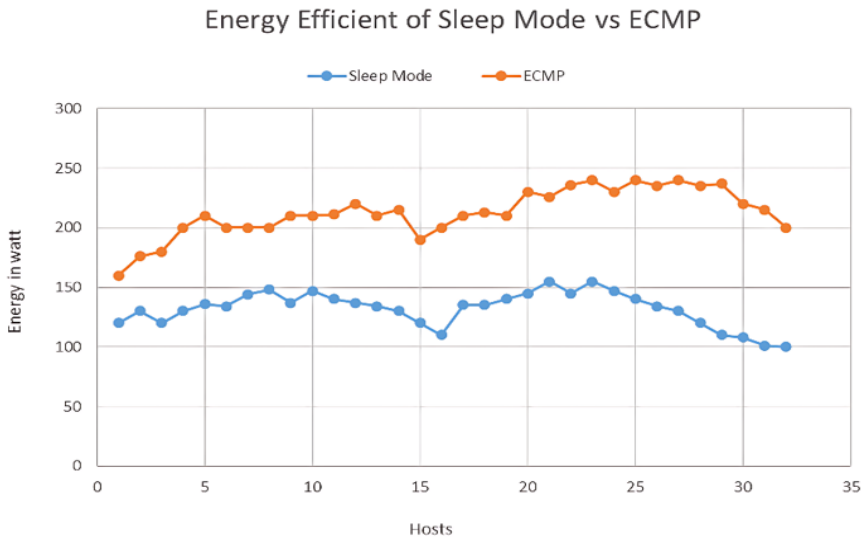

Figure 7.
Bandwidth of sleep mode versus ECMP.

4.4. Average Load

In this experiment, we used Wireshark to monitor and measure the average load on nodes, capturing network traffic between hosts within the 8-pod fat-tree topology. We analyzed the traffic flows comparing results with and without the implementation of the sleep mode routing algorithm. Figure 9 shows the average load over time, illustrating the impact of sleep mode on network efficiency that outperformed the ECMP and Dijkstra's methods respectively.

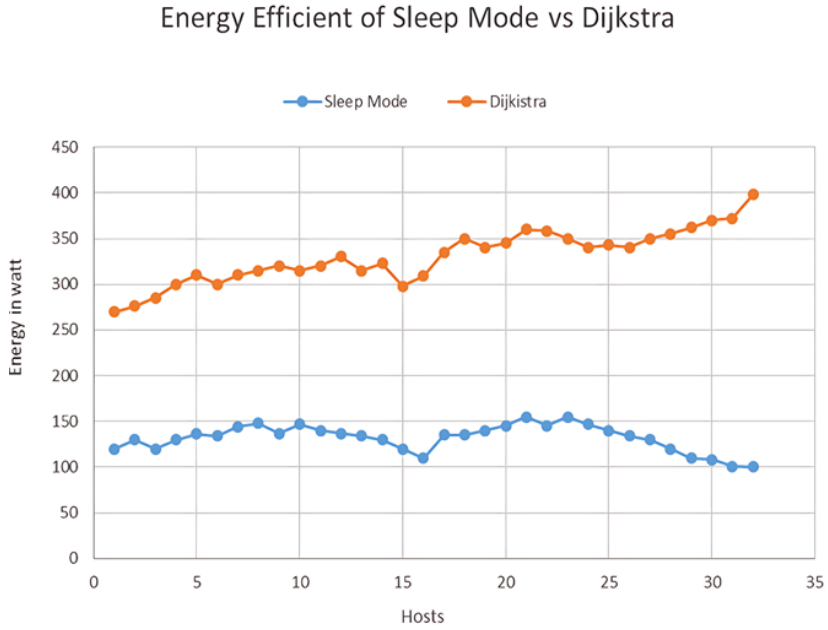

Figure 8.
Average load versus time.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 2: 218-230, 2025
DOI: 10.55214/25768484.v9i2.4456

5. Conclusion

In this study, we explore the scalability of fat tree topology, which features a hierarchical structure of access, aggregation, and core switches. Using the Mininet emulator, we simulate and evaluate the performance of an 8-pod fat tree network. The performance of the 8-pod fat tree SDN network with the sleep mode technique was compared to traditional routing methods: ECMP and Dijkstra's algorithm. The sleep mode strategy resulted in improved bandwidth utilization and throughput, as switches in sleep mode reduced energy consumption without significantly affecting data flow. In contrast, Dijkstra showed reduced throughput due to less efficient path management, and ECMP did not fully optimize throughput or load balancing during low-traffic periods. Regarding average load, the sleep mode technique outperformed both Dijkstra and ECMP by efficiently redistributing traffic across active switches, preventing congestion. While ECMP improved load distribution, it didn't account for energy-efficient sleep mode operations, limiting its effectiveness. Overall, the sleep mode strategy enhanced both network performance and energy efficiency, outperforming Dijkstra and ECMP in a large-scale SDN setup. This article provides a comprehensive analysis of the 8-pod fat-tree topology within an SDN framework, utilizing Mininet as the simulation platform. It focuses on the design, implementation, and evaluation of SDN-based traffic management techniques in an 8-pod fat-tree network.

6. Recommendation

This study evaluates the performance of the 8-pod fat tree network using SDN and a sleep mode technique. It finds that sleep mode outperforms traditional routing strategies like ECMP and Dijkstra's algorithm in throughput, bandwidth efficiency, and load distribution. The approach improves energy efficiency in large-scale data centers. Future work could focus on applying sleep mode in diverse real-world conditions, integrating it with energy-efficient hardware, and automating its activation. Additionally, further research can explore cross-layer optimizations, multi-tenancy support, and the impact on latency.

Transparency:

The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.

Competing Interests:

The authors declare that they have no competing interests.

Authors' Contributions:

All authors contributed equally to the conception and design of the study. All authors have read and agreed to the published version of the manuscript.

Copyright:

References

[1] M. A. Ja'afreh, H. Adhami, A. E. Alchalabi, M. Hoda, and A. El Saddik, "Toward integrating software defined networks with the Internet of Things: a review," *Cluster Computing*, vol. 25, no. 1, pp. 1-18, 2022. https://doi.org/10.1007/s10586-021-03402-4

[2] E. Molina and E. Jacob, "Software-defined networking in cyber-physical systems: A survey," *Computers & Electrical Engineering*, vol. 66, pp. 407-419, 2018. https://doi.org/10.1016/j.compeleceng.2017.05.010

[3] N. M. Yungaicela-Naula, "Towards security automation in software defined networks," *Computer Communications*, vol. 183, pp. 64-82, 2022. https://doi.org/10.1016/j.comcom.2021.12.004

[4] W. Rafique, L. Qi, I. Yaqoob, M. Imran, R. U. Rasool, and W. Dou, "Complementing IoT services through software defined networking and edge computing: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1761-1804, 2020. https://doi.org/10.1109/COMST.2020.2973142

[5] A. Akhunzada, "Secure and dependable software defined networks," *Journal of Network and Computer Applications*, vol. 61, pp. 199-221, 2016. https://doi.org/10.1016/j.jnca.2015.11.004

[6] J. Mishra, J. Sheetlani, and K. H. K. Reddy, "Data center network energy consumption minimization: A hierarchical FAT-tree approach," *International Journal of Information Technology*, vol. 14, no. 1, pp. 507-519, 2022. https://doi.org/10.1007/s41870-021-00607-0

[7] Z. Lu, J. Lei, Y. He, Z. Li, S. Deng, and X. Gao, "Energy optimization for Software-Defined data center networks based on flow allocation strategies," *Electronics*, vol. 8, no. 9, p. 1014, 2019. https://doi.org/10.3390/electronics8091014

[8] H. Bhatia *et al.*, "Interactive investigation of traffic congestion on fat-tree networks using treescope," in *Computer Graphics Forum*, 2018, vol. 37, no. 3: Wiley Online Library, pp. 561-572.

[9] X. Gao, "NEMO: Novel and efficient multicast routing schemes for hybrid data center networks," *Computer Networks*, vol. 138, pp. 149-163, 2018. https://doi.org/10.1016/j.comnet.2018.03.017

[10] H. M. S. Khan, H. Khan, C. M. A. Hayat, H. Tayyab, and K. Ali, "An enhanced cost-effective and scalable network architecture for data centers," *Spectrum of Engineering Sciences*, vol. 2, no. 4, pp. 1-32, 2024. https://doi.org/10.46397/se.2024.2.4.1

[11] D.-m. Yu, Z. Zhang, Y.-h. Deng, L.-x. Lin, T.-j. He, and G.-l. He, "Flexible, highly scalable and cost-effective network structures for data centers," *Journal of Network and Computer Applications*, vol. 210, p. 103542, 2023. https://doi.org/10.1016/j.jnca.2023.103542

[12] A. N. Quttoum, "Interconnection Structures, Management and Routing Challenges in Cloud-Service Data Center Networks: A Survey," *International Journal of Interactive Mobile Technologies*, vol. 12, no. 1, 2018.

[13] K. C. Okafor, I. E. Achumba, G. A. Chukwudebe, and G. C. Ononiwu, "Leveraging fog computing for scalable IoT datacenter using spine-leaf network topology," *Journal of Electrical and Computer Engineering*, vol. 2017, no. 1, p. 2363240, 2017. https://doi.org/10.1155/2017/2363240

[14] S. Wang, "Impact of network topology on the performance of DML: Theoretical analysis and practical factors. In " presented at the IEEE INFOCOM 2019—IEEE Conference on Computer Communications. IEEE. https://doi.org/10.1109/INFOCOM.2019.8737495, 2019.

[15] Z. Zhang, Y. Deng, G. Min, J. Xie, L. T. Yang, and Y. Zhou, "HSDC: A highly scalable data center network architecture for greater incremental scalability," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 5, pp. 1105-1119, 2018. https://doi.org/10.1109/TPDS.2018.2877990

[16] S. Fawzi and N. M. Din, "A Mininet emulation study for SDN fat tree data center sleep mode routing algorithms," *Edelweiss Applied Science and Technology*, vol. 8, no. 6, pp. 8956-8967, 2024.

[17] J. Gomez, E. F. Kfoury, J. Crichigno, and G. Srivastava, "A survey on network simulators, emulators, and testbeds used for research and education," *Computer Networks*, vol. 237, p. 110054, 2023. https://doi.org/10.1016/j.comnet.2023.110054

[18] S. Khan, A. Akram, H. Alsaif, and M. Usman, "Emulating software defined network using mininet-ns3-WIFI integration for wireless networks," *Wireless Personal Communications*, vol. 118, no. 1, pp. 75-92, 2021. https://doi.org/10.1007/s11277-021-08490-z

[19] M. N. A. Sheikh, I.-S. Hwang, M. S. Raza, and M. S. Ab-Rahman, "A qualitative and comparative performance assessment of logically centralized SDN controllers via Mininet emulator," *Computers*, vol. 13, no. 4, p. 85, 2024. https://doi.org/10.3390/computers13040085

[20] B. G. Assefa and Ö. Özkasap, "A survey of energy efficiency in SDN: Software-based methods and optimization models," *Journal of Network and Computer Applications*, vol. 137, pp. 127-143, 2019. https://doi.org/10.1016/j.jnca.2019.01.001

[21] C. Cheng, J. Dou, and Z. Zheng, "Energy-efficient SDN for internet of things in smart city," *Internet of Things and Cyber-Physical Systems*, vol. 2, pp. 145-158, 2022.

[22] X. Chen, X. Wang, B. Yi, Q. He, and M. Huang, "Deep learning-based traffic prediction for energy efficiency optimization in software-defined networking," *IEEE Systems Journal*, vol. 15, no. 4, pp. 5583-5594, 2020. https://doi.org/10.1109/JSYST.2020.2986819

[23] T.-L. Kao, Y. T. Miao, and S.-Y. Wang, "OPS: A fairness link allocation based on SDN in datacentre networks," *International Journal of Communication Networks and Distributed Systems*, vol. 25, no. 3, pp. 307-332, 2020. https://doi.org/10.1504/IJCNDS.2020.107374

[24] Y. Zhang, L. Cui, and Y. Zhang, "A stable matching-based elephant flow scheduling algorithm in data center networks," *Computer Networks*, vol. 120, pp. 186-197, 2017. https://doi.org/10.1016/j.comnet.2017.04.027

[25] L. Qin, W. Wei, and X. Diao, "FlowDecider: End-Host Driven Proactive Load Balancing for Data Center Networks," in *2021 IEEE 21st International Conference on Communication Technology (ICCT). https://doi.org/10.1109/ICCT52987.2021.9686394*, 2021: IEEE, pp. 931-936.

[26] J. Son and R. Buyya, "Priority-aware VM allocation and network bandwidth provisioning in software-defined networking (SDN)-enabled clouds," *IEEE Transactions on Sustainable Computing*, vol. 4, no. 1, pp. 17-28, 2018. https://doi.org/10.1109/TSUSC.2017.2767888

[27] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, pp. 63-74, 2008. https://doi.org/10.1145/1402946.1402952

[28] Y. Jin, X. Liu, and J. Wang, "Traffic engineering in SDN-based Fat Tree networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 3, pp. 1137-1150, 2018. https://doi.org/10.1109/TNSM.2018.2852793

[29] D. Kumari, P. Sharma, and A. Singh, "Energy-efficient traffic management in SDN-enabled data center networks," *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 8, no. 1, pp. 13-25, 2019. https://doi.org/10.1186/s13677-019-0154-7

[30] L. Zhang, S. Li, and K. Yang, "Energy-efficient routing in SDN-enabled data center networks," *Future Generation Computer Systems*, vol. 106, pp. 123-135, 2020. https://doi.org/10.1016/j.future.2019.11.025

[31] Z. Cheng, T. Yang, and X. Wu, "Scalability and fault tolerance in SDN-based Fat tree networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 1, pp. 110-123, 2021. https://doi.org/10.1109/TPDS.2020.2998528

[32] M. Safa, W. Liao, and S. Wang, "Energy-efficient traffic management for data center networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 3, pp. 387-401, 2015. https://doi.org/10.1109/TNSM.2015.2432746

[33] M. Hassan, R. Soni, and V. Jindal, "Energy optimization in SDN-based data centers: A comprehensive study," *Journal of Network and Computer Applications*, vol. 179, p. 102999, 2022. https://doi.org/10.1016/j.jnca.2021.102999

[34] P. Sood and M. Sood, "SDN and mininet: Some basic concepts," *International Journal of Advanced Networking and Applications*, vol. 7, no. 2, pp. 2690-2693, 2015.

[35] M. Alizadeh, T. Edsall, and S. Dharmapurikar, "CONGA: Distributed congestion-aware load balancing for datacenters," in *Proceedings of ACM SIGCOMM*. *https://doi.org/10.1145/2619239.2626317*, 2014, pp. 503-514.

[36] H. Zhang, J. Zhang, and W. Bai, "Resilient datacenter load balancing in the wild," in *Proceedings of ACM SIGCOMM*. *https://doi.org/10.1145/3098822.3098835*, 2017, pp. 1-14.

[37] S. Ghorbani, Z. Yang, and P. Godfrey, "DRILL: Micro load balancing for low-latency data center networks," in *Proceedings of ACM SIGCOMM*. *https://doi.org/10.1145/3098822.3098841*, 2017, pp. 17-30.

[38] G. Michelogiannakis, K. Z. Ibrahim, J. Shalf, J. J. Wilke, S. Knight, and J. P. Kenny, "Aphid: Hierarchical task placement to enable a tapered fat tree topology for lower power and cost in hpc networks," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. *https://doi.org/10.1109/CCGrid.2017.00067*, 2017: IEEE, pp. 228-237.

[39] A. Rego, S. Sendra, J. M. Jimenez, and J. Lloret, "OSPF routing protocol performance in Software Defined Networks," in *2017 Fourth International Conference on Software Defined Systems (SDS)*. *https://doi.org/10.1109/SDS.2017.25*, 2017: IEEE, pp. 131-136.

[40] O. Dobrijevic, M. Santl, and M. Matijasevic, "Ant colony optimization for QoE-centric flow routing in software-defined networks," in *2015 11th International Conference on Network and Service Management*. *https://doi.org/10.1109/CNSM.2015.7341403*, 2015: IEEE, pp. 274-278.

[41] G. B. Haile and J. Zhang, "Dynamic load balancing algorithm in SDN-based data center networks," *International Journal of Engineering Research & Technology (IJERT)*, vol. 10, pp. 1-6, 2021.

[42] J. Pang, G. Xu, X. Fu, and K. Zhao, "Horizon: a QoS management framework for SDN-based data center networks," *Annals of Telecommunications*, vol. 72, pp. 597-605, 2017. https://doi.org/10.1007/s12243-017-0579-2

[43] M. Huang, "Dynamic routing for network throughput maximization in software-defined networks," in *IEEE INFOCOM 2016—The 35th Annual IEEE International Conference on Computer Communications, IEEE*. *https://doi.org/10.1109/INFOCOM.2016.7524626*, 2016, pp. 1-9.

[44] Z. Lv, J. Lloret, and H. Song, "Guest editorial software defined internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3504-3510, 2021. https://doi.org/10.1109/TITS.2021.3061271

[45] R. A. Ammal and S. V. Chandra, "Bio-inspired algorithms for software defined network controllers," in *2018 International CET Conference on Control, Communication, and Computing (IC4)*. *https://doi.org/10.1109/IC4.2018.8660841*, 2018: IEEE, pp. 306-310.

[46] A. R. Ammal, P. Sajimon, and S. Vinodchandra, "Application of smell detection agent based algorithm for optimal path identification by SDN controllers," in *Advances in Swarm Intelligence: 8th International Conference, ICSI 2017, Fukuoka, Japan, July 27–August 1, 2017, Proceedings, Part II 8*. *https://doi.org/10.1007/978-3-319-60627-4_45*, 2017: Springer, pp. 502-510.

[47] A. A. Sabr, M. A. Mohammed, and D. A. Bayz, "Advanced greedy hybrid bio-inspired based shortest path routing algorithm for SDN controller over VANET: Issues and challenges," in *ITM Web of Conferences*. *https://doi.org/10.1051/itmconf/20236401017*, 2024, vol. 64: EDP Sciences, p. 01007.