Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6, 8956-8967
2024
Publisher: Learning Gate
DOI: 10.55214/25768484.v8i6.3921

A Mininet emulation study for SDN fat tree data center sleep mode routing algorithms

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

Abstract: Big-data networks are raising energy use and associated ecological effects and have become a significant cause for concern in the last decade. With the expected significant increase in the traffic demand in data centres escalates the need to be energy efficient. In this scenario, the implementation of sleep mode in software defined network (SDN) data centres could provide an energy aware solution. Energy consumed of the network can be reduced by putting underutilized network devices or components to sleep. Employing sleep mode aligns with sustainability goals and can reduce the environmental impact of data centers by lowering carbon emissions. In this work meta heuristic algorithm is incorporated at the SDN central controller in a fat tree-based data centre for bandwidth usage monitoring, sleep decisions and path selection using Mininet emulation. The Mininet emulation performance study was conducted based on comparison between metaheuristic and Dijkstra method. The proposed sleep mode method obtained enhancement in the performance parameter with uses less energy in the network.

Keywords: *Data center, Energy efficiency, Mininet, Sleep mode, Software defined network.*

1. Introduction

SDN makes it possible for complex network applications to be executed software-wise by successfully separating the network control plane from the data plane. Smart cities, the emergence concerning the Internet of Things (IoT), applications that utilise big data, and sophisticated artificially intelligent networks, as well as the growth in data centre networks' size, information processing the ability, and efficiency to meet the demands of high traffic in computing services [1]. Data centres are using more energy, and this has an impact on the environment [2]. The foundation of the software-defined network concept is the division of the control plane into the data plane. Although an effectively centralised controller oversees the network, the data plane controls the forwarded data function [3-6]. In addition to being utilised by network technology vendors and large corporations, SDN offers a strong feature of network flexibility [7-8]. Many network applications, including traffic engineering, energy efficiency, dynamic routing, security, and load balancing, have been made possible by the ease of use and portability of network management. While a data plane ahead. packets in response to commands from the control plane, a control plane uses the actions. An overview of the network is provided by a centralised control plane, which facilitates network management. Three layers make up the fat-tree topology [9] the central layer from base to top, the aggregation layer, and the edge layer. The number of pods within a k-ary fat tree is k. There are two switch layers in each pod. There are k ports on each switch. A switch's edge layer ports connect to hosts, while the remaining ports are connected to switches in the aggregation layer [10-17]. Core switches are located in the fundamental layer, and each one has a port connected to a k-ary.

* Correspondence: surafawzi89@gmail.com

2. Literature Review

In recent decades, for economic and environmental reasons, the idea of cost-effective green networking has emerged as a crucial concern for companies and researchers [18, 19]. In order to reduce energy consumption, the most significant equivalent strategy for network energy-efficiency techniques concentrated on routing optimisation models and network traffic with a variety of constraints [20, 21]. Using a variety of simulation tools, numerous researchers have tried to increase networks' energy efficiency. An efficient method for assessing efficacy, investigating and troubleshooting SDN rules, and resolving network-related research concerns is the network simulation [22]. The process of mimicking an actual network's behaviour in a controlled environment is known as network emulation. It entails simulating the properties of network connections, including packet loss, jitter, latency, and bandwidth, and other network impairments in order to evaluate and test the efficiency, accuracy, and scalability of apps, protocols, or network devices as if they were running in a live network. Innovating with SDN network is limited to popular simulators like GNS3 and OMNet++, NS2, and NS3. To validate while assessing the advantages of DC Network, an appropriate emulation tool is recommended such as Mininet. Mininet [23] is an emulation tool useful for studying the concerning the use of SDN frameworks. It enables quick development and detailed testing tool, making it ideal for educational purposes, research work, in addition to the validation evaluation using tools like Mininet enables precise testing and validation of the exact SDN control software that will be deployed in production environments, as seen in examples like and Google and BSN Labs [24]. This approach ensures that the software behaves as expected under various network conditions, allowing organizations to identify potential issues and optimize performance before actual deployment. Tropea et al. [25] utilized the same SDN, control software in both emulated and production environments, companies can significantly reduce the risks associated with software rollouts, leading to smoother transitions and more reliable network operations. Various research has been done using different tools for simulating the network recent research on energy efficiency in networking has leveraged a variety of simulation tools to optimize and evaluate energy-saving strategies. Some studies have extensively used NS-2 and NS-3, which are popular discrete-event network simulators, to model and evaluate energy-efficient strategies in networking environments. For instance, Alwasel et al. [26] focused on reducing energy consumption through optimised data aggregation and management of power techniques, simulating energy-efficient methods of routing in wireless sensor networks using NS-3. This greatly increased the network's operational lifetime.

An SDN-enabled optical Data Centre Network (DCN) with adaptable Quality of Service, or QoS, providing for multi-tenant applications was proposed and experimentally examined by Yu et al. [27]. NS2 and NS3 were used to simulate an SDN-enabled optical data centre, where the network structure and statistics were tracked. These statistics could be used to automatically slice and reconfigure the network under an SDN controller's control. The experimental findings, which were confirmed by NS2 and NS3, showed that dynamic QoS provisioning is made possible by adaptable highest importance distribution for network traffic flows. Additionally, the real-time network statistics-driven automatic load balancing function greatly improved network performance while preserving high QoS. TPSMA (A two-phase Sensing Process Algorithm), an optimisation method inspired by nature and used in networks of wireless sensors (WSNs), was first presented by Abidin et al. [28]. The two stages of the TPSMA technique—marking and matching—are analogous to predators demarcating the borders of their domains. The results demonstrated that using TPSMA approach in WSN alerts is an affordable way to provide wide coverage with minimal power consumption. Lin et al. [29] used OMNeT++ to model dynamic energy management in data center networks, developing algorithms that reduce energy consumption by selectively powering down idle servers and networking devices, achieving substantial energy savings without degrading performance (Computer Networks).

Utilising OMNET ++, Let et al. [30] suggest a two-phase SDN-based routing mechanism that seeks to reduce energy consumption while maintaining a certain degree of quality of service for user flows. The results show that the suggested routing technique significantly reduces energy

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921

consumption for crowded traffic with mice-type flows. It can meet users' QoS needs while offering efficient link load balancing. Additionally, Monteiro et al. [31] evaluated the sustainable flow in wireless sensors networks (WSNs) using MATLAB and suggested a meta-heuristic algorithm for route selection optimisation dependent within the dolphin echolocation method (DEA). The research introduced a specialized SDN-WSN optimization model incorporating the DEA algorithm, designed with a unique pseudo-code tailored to the specific application. This model demonstrated that the DEA algorithm outperformed other competing algorithms in solving the optimization problem. Although SDN-based approaches have been suggested for efficient path selection in WSNs, they still face challenges related to exploration and exploitation. Torkzadeh et al. [32] presents SD-IoAV used MATLAB, a composite architecture designed to integrate SDN with the Internet of Autonomous Vehicles (IoAV). Because of the geographical dispersion of IoAV, this integration presents significant challenges. To manage the underlying communications efficiently, multiple SDN controllers have been installed throughout widely distributed SDN domains. Simulations conducted in MATLAB indicate that the suggested approach effectively manages energy.

Emulation, unlike simulations, is a cost-effective way to run real code under genuine communicating and computing conditions. A network emulator, especially Mininet, provide a virtual network in which developers and network engineers can simulate multiple network circumstances and configurations without requiring physical hardware. This enables extensive testing, troubleshooting, and optimisation of network configurations and software applications prior to installation in real-world conditions. Mininet has gained significant attention due to its capability to emulate Software-Defined Networking (SDN) environments realistically, making it an ideal tool for testing energy-efficient algorithms in real-time and identifying the most effective SDN routing strategies. A software-defined data centre (SDDC) and a mininet were used by Son et al. [33] suggested segment-routing-based energy-aware routing technique. By optimising link utilisation and preserving link residual capacity, the method seeks to maximise throughput. The findings showed that, in comparison to the highest segment label depth, the segment label stack's length decreased. The analysis was carried out by contrasting other current techniques in a single-controller established with the energy-aware routing method in an environment with multiple controllers. Based on their investigation of traffic-related energy consumption, Kaur et al. [34] proposed a framework for attaining energy conservation in Software-Defined Networks (SDNs). To solve the energy efficiency issue related to traffic stabilisation, author created an IP-based model using Mininet. Furthermore, they suggested a heuristic algorithm intended to maximise network efficiency and traffic flow. In order to lower consumption of energy in SDN-based data centres, Aslam et al. [35] concentrated on increasing network efficiency by utilising the Software-defined Networking (SDN) elastic tree framework. Research aimed to enhance existing optimization techniques within the elastic tree model while introducing new factors, such as power control mechanisms informed by traffic pattern analysis.

A virtual data center network is developed using the Mininet emulator [36], enabling the simulation of real-world conditions. In order to optimise network efficiency and energy consumption, the optimiser incorporates a Floodlight controller that monitors traffic flow and is designed using multiple models. The ONOS controller, a sleep mode technique, and the Mininet are used in this study to simulate a fat-tree topology. Network performance is assessed in a number of scenarios using mininets. By utilising the sleep mode feature of SDN with a bioinspired technique called TPSMA (the Territorial Predator Scent Marking Algorithm), the research aims to create an energy-saving plan for data centres. While increasing network performance and efficiency, this strategy seeks to lower energy consumption.

3. Materials and Methods

In this research, we will explore the use of sleep mode in software-defined networks (SDN) to cut down on data centres' energy usage, addressing the environmental impact of increasing traffic. We will integrate a metaheuristic algorithm into the SDN controller within a Fat Tree architecture

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921

for efficient bandwidth monitoring, sleep decisions, and path selection. Additionally, we will conduct performance evaluations through Mininet emulation, comparing the results with traditional methods. This approach aims to reduce energy usage while maintaining optimal network performance.

3.1. Step 1: A Conceptual Design and Implementation of Sleep Mode in Software Defined Network Data Center SDN-DC

Due to the complexity of connections in the real-world network, we propose a hybrid energy efficiency approach in SDN that combines the key benefits of traffic-aware sleep mode and dynamic load balancing to achieve energy-efficient and performance-aware network operation. In this strategy, switches/ nodes that are not being used or are underutilized can be put into sleep mode, while the remaining resources can be used for load balancing and this will lead to energy efficient. More specifically, to lower total power consumption, we employ particular network topological features like fat tree structures using TPSMA as well as traffic engineering solutions. SDN controllers and switches have two modes: sleep mode and wake mode. Switches and controllers use the most energy when in wake-up mode. Additionally, they consume a minimum of energy when they are in sleep mode. Since it requires some time to connect and establish the traffic when turning the controller or switch on from the off mode, the idle switch or controller may choose to sleep instead of shutting off. In SDN-DC, sleep mode is therefore the best option.

In data center SDN traffic aware energy-efficient routing techniques are classified according to the power saving characteristics they rely on and the topology architecture they use. Energy-efficient strategies can be categorised as basic or customised to a specific network architecture, depending on the SDN topology assumption. The major objective in sleep mode technique to turn off unused switches and redirect the traffic to the active switch, thus will enhance the performance and QoS in SDN. Energy efficiency in SDN-DC is divided into three categories: energy of switches, links, and controllers. Shang et al. [10] worked on SDN load distribution to improve the energy of the SDN and the controllers. To solve the energy problem, the author implemented EERAS Energy Efficiency Routing Algorithm Solution, and model the energy efficiency based on packet flow as follows:

$$E = n \sum i=1 (e1i * Eswi + e2i * Eci + e3i * L) + Elink$$

where Eswi is the switch energy, Eci controller energy, and E link is the link energy, packet queue caused by L the latency, and eli which is a constant from the simulation represent the energy coefficient for switch i, e2i is energy coefficient for controller i, e3i is the energy coefficient related to latency. In SDN data center the biggest part of the consumed energy is from the switches and links of the SDN, the energy of each DC component such as controller, switch and link and each part is based on packet flow. In this work we focus on the energy efficiency in SDN, we define our topology as undirected graph G = (N; L), where N is the network's node set consisting of the two sets the host set H and the set S for SDN switches, thus, N = H, S, and L indicates the set of links connecting with each node N in fat tree topology, D represents the traffic demand between the SDN hosts. E known as energy used in the network, T is given time per each network activity.

3.2. Step 2: Mininet Emulation Study for SDN Fat Tree Sleep Mode

Mininet is a network emulator that lets researchers and engineers build and test virtual networks upon a single machine. Mininet supports a variety of topologies for representing different network layouts [43]. To emulate SDN software defined network fat tree topology with sleep mode in Mininet, first we create fat-tree topology, second we implement sleep mode technique which saves energy by turning off / minimising the power of idle network components. Fat-Tree topologies are widely used in data centres due to their ability to scale and fault tolerance Figure. 1, illustrate the. It is composed of three layers: core, aggregation and edge layer. Each layer has a set of switches, and the network is designed to provide multiple paths for data between hosts, which is important for load

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921

balancing and redundancy. To set up the Mininet, first we install the virtual machine VM, a virtual machines describes a software emulation of an actual computer that runs the same operating system along with apps as the real machine. It enables multiple VMs to run on the same physical host, sharing resources such as memory, CPU, and storage whereas remaining isolated from one another. Virtual machines VMs have become popular for server merging, testing applications, and running multiple operating systems on identical hardware. Secondly we install Mininet, and ensure the required dependencies and tools, such as OpenFlow and ONOS controller. We create a custom fat tree topology using a python script. This script will specify the number of pods, switch links, and host connections. Fat tree topology with k pods consists of (k/2) ^2 core switches. k pods, each with (k/2) aggregation switches and (k/2) edge switches. Each edge switch connects to (k/2) hosts. from mininet.topo import topo file class fat tree topo (Topo): Mininet example for emulating fat tree topology using python script as follows:

```
def build(self, k=4):
core_switches = []
agg_switches = []
# Create aggregation and edge switches, and hosts
for pod in range(k):
agg_switches.append([])
edge_switches.append([])
for i in range(k // 2):
agg_switch = self.addSwitch(f'a{pod}_{i+1}')
# Connect edge switches to hosts
for j in range(k // 2):
host =   self.addHost(f'h{pod}_{i}_{j+1}')
self.addLink(edge_switch, host)
```

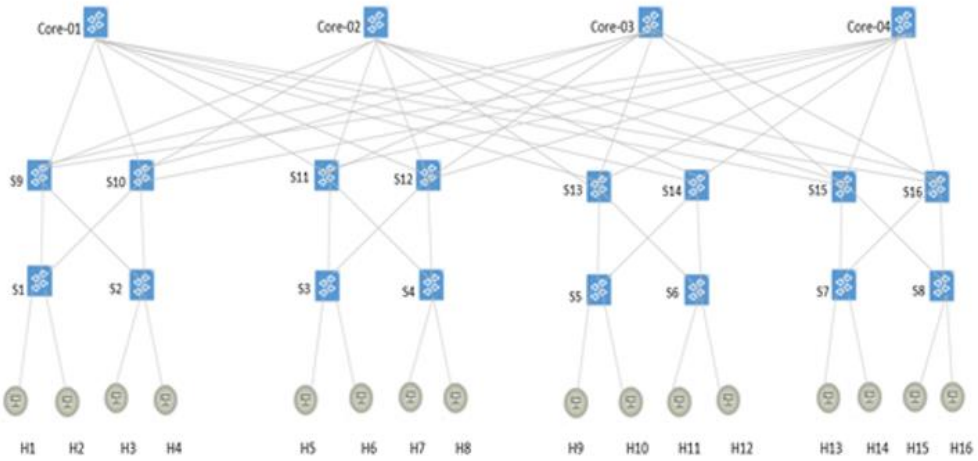

Figure 1.
Fat tree network.

3.3. Step 3: Traffic Model

In this study, we simulate the traffic of an SDN-based data center using the Iperf tool. To begin, we measure bandwidth utilization by generating UDP traffic in Mininet. The proposed method is applied to analyze network performance, where we create traffic loops in each pod between various hosts, such as from h1 to h4, followed by h5 to h8, h9 to h12, and h13 to h16, based on the total number of hosts. Each switch or node's bandwidth utilization is assessed

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921

according to our experimental approach. After completing the UDP traffic tests, we generate TCP traffic same scenario in order to evaluate the throughput of the SDN-DC. The results obtained from the TCP traffic tests provide insights into the overall performance of the data center in terms of throughput, offering a comprehensive view of its capabilities under different traffic loads.

3.4. Step 4: Algorithm Implementation

This This section presents the implementation of a sleep mode algorithm in an SDN fat-tree topology and provides an overview of the evaluation environment used in the study. The Mininet emulator was employed to simulate a Fat-Tree topology. A k=4 Fat-Tree network with 20 switches and 16 hosts was constructed using ONOS controller version 2.5.1 and Open vSwitch (OVS) with OpenFlow version 1.3. To generate UDP and TCP traffic in the SDN environment, the Iperf tool was utilized. The Mininet tests were conducted with the ONOS controller using the OpenFlow protocol on an Ubuntu host. Bandwidth utilization and latency were used as performance indicators, and energy efficiency was measured in Mininet. This study compares the proposed technique with Dijkstra's method to evaluate the solution. First we implement the sleep mode algorithm, and next we generate the traffic using Iperf tool.

Algorithm 1: Sleep Mode
Initializing the network elements
G is network graph, N is the network nodes, S is the network switches
C is the network controller
C distribute the traffic
Monitor switch status and put in sleep mode if traffic is light
for switch in S:
traffic = C.get_switch_traffic(switch)
  if traffic < traffic_threshold:
C.set_switch_sleep_mode(switch)
  else:
C.set_switch_active_mode(switch)
Calculate energy consumption and total energy
for switch in S:
status = C.get_switch_status(switch)
total_energy += C.calculate_switch_energy(switch, status)
Calculate optimal routing paths for each node
for node in N:
optimal_path = C.compute_routing_path(node)
C.update_routing_table(node, optimal_path)
Calculate average energy usage over time T
energy_data
for t in range(T):
energy_data.append (total_energy)
avg_energy = sum(energy_data) / len(energy_data)
Log and return energy consumption
C.log_energy_usage(avg_energy)
return avg_energy
end if
end for

The software-defined networking SDN framework is made up of three different layers: data plane, the control plane, and the application layer, which are all linked together via flow tables and OpenFlow APIs. The application layer, or API, is an open domain in which new applications can be created using extensive network information. The control layer in an SDN network makes decisions by forwarding info through switches with southbound APIs to applications and

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921

organisational policies above via northbound APIs. The sleep mode technique is utilised at the control layer to help the controller identify nodes alongside the least traffic, put them to sleep, and reroute incoming data packets towards the next active node during the matching phase. In SDN-based Fat Tree network, path selection is controlled by the SDN controller, which also manages switches entering sleep mode for energy efficiency. When a switch is placed in sleep mode, it stops participating in traffic forwarding to save power. The controller monitors the network and ensures that data is routed around switches that are in sleep mode. Instead of using the sleep-mode switches in the routing path, the controller reconfigures the flow tables to route traffic through other active, awake switches following the SDN routing. This ensures that the network operates efficiently, with minimal energy consumption, while maintaining traffic flow by avoiding sleep-mode devices.

4. Results and Discussion

We evaluated the proposed method by measuring the bandwidth utilization, the traffic generated through the Iperf tool in Mininet, applying the proposed method for performance analysis. The results indicate that the SDN data center's performance improved in terms of bandwidth utilization, average delay, and energy efficiency. This section presents the findings from the Mininet emulator when all hosts are active.

4.1. Bandwidth

From the bandwidth utilization result we can see the bandwidth increasing while the number of devices increase with sleep mode method. Figure. 2 Illustrate the bandwidth utilization versus hosts, in compare with Dijkstra's utilization, from the result sleep mode outperformed Dijkstra's method. percent.

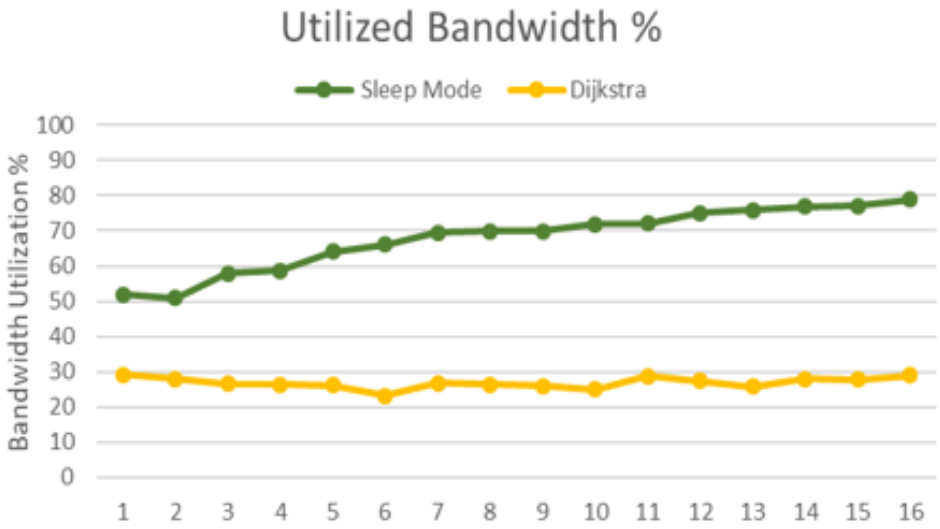

Figure 2.
Bandwidth versus hosts.

4.2. Throughput

From Figure. 3, the result indicates the throughput results using TCP traffic with SDN data centre, result demonstrate the difference between Dijkstra's and sleep mode techniques. The sleep mode technique improves SDN-DC traffic performance efficiency by increasing throughput and improving the outcome as compared to the Dijkstra algorithm.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921

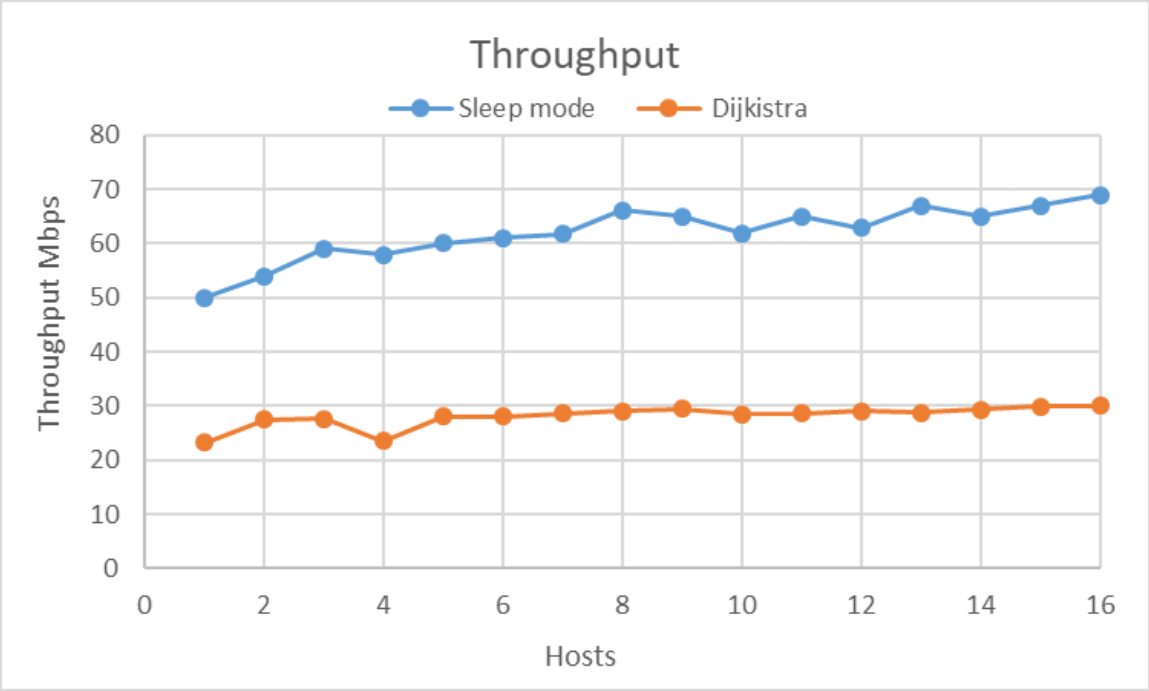

Figure 3.
Throughput versus hosts.

4.3. Energy Efficiency

From the result in Figure 4, the result of energy illustrates the enhancement in the usage of energy as the traffic demand grow and when compared to Dijkstra's algorithm, the proposed sleep mode method has a significant improvement in energy efficiency versus traffic demand. The proposed method efficiently decreases the energy use about 45% resulting from pathfinding computations through the implementation of sleep mode as a novel strategies and optimisations method.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921

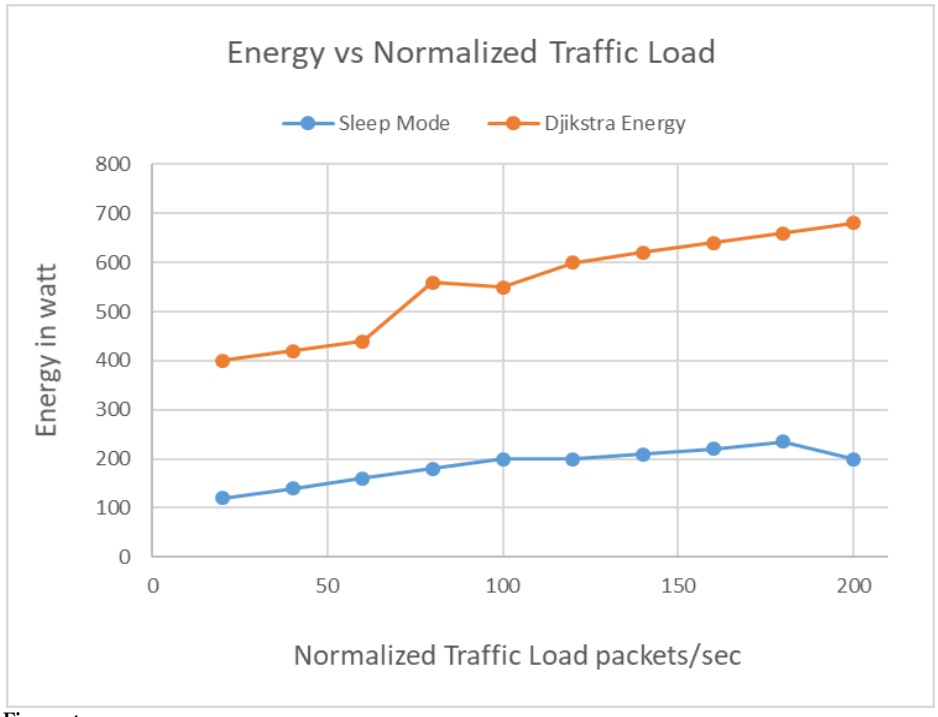

Figure 4.
Energy efficiency versus traffic load.

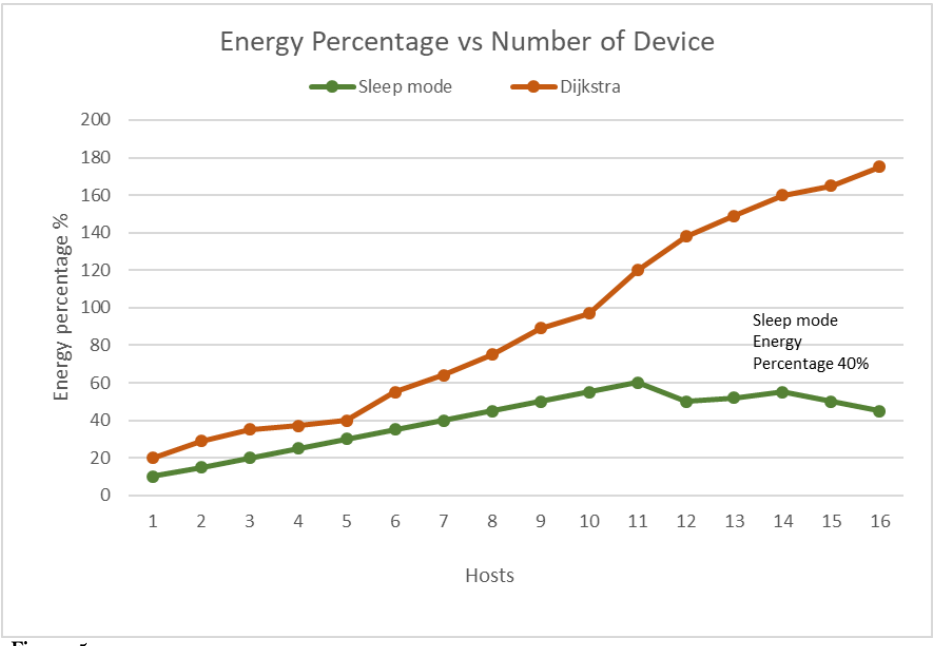

Figure 5.
Energy efficiency versus hosts.

From Figure. 5, Illustrate the Energy versus hosts, as we can see the energy increases while hosts increased, as compared to the Dijkstra's algorithm. sleep mode outperforms Dijkstra's Figure.6, Illustrate the sleep mode versus Dijkstra's, result indicate the sleep mode saved up to 40% more

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921

energy efficient improved as compare with Djikstra's, thus the proposed sleep mode method offers significant potential for achieving good energy efficiency in various applications. By intelligently managing the power consumption of electronic devices during idle or low activity periods, this method helps to reduce overall energy consumption and network performance as it is increases the performance parameters factors. The sleep mode method leverages the principles of power management and load balancing with the smart meta heuristic algorithms to determine the optimal times and conditions for transitioning devices into sleep mode state.

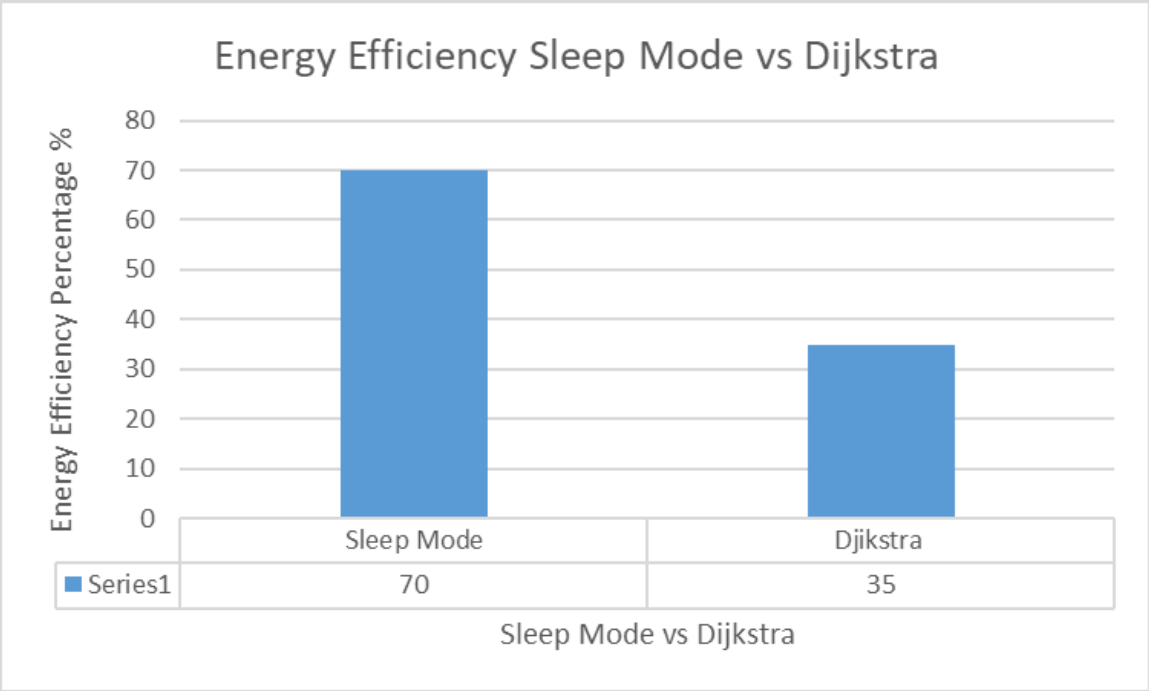

Figure 6.
Energy efficiency in sleep mode versus Dijkstra.

5. Conclusion

The proposed method demonstrates a remarkable improvement in energy efficiency. Compared to Dijkstra's algorithm, it significantly reduces the energy consumption associated with pathfinding computations through the integration of novel strategies and optimizations. This approach not only lowers overall energy usage but also enhances network performance without sacrificing the accuracy or quality of the pathfinding solution. An essential aspect of this research is the use of Mininet, a versatile network emulator that enables the simulation of large-scale network topologies, such as data centers, in a highly controlled environment. By utilizing Mininet, the proposed method can be rigorously tested and analyzed in a realistic, yet flexible, setting. This ensures that the improvements in energy efficiency and network performance are thoroughly validated before practical implementation. Overall, this advancement not only promotes a more sustainable and eco-friendly approach but also offers significant cost savings in energy-intensive applications. As a result, the proposed method represents a major leap forward in pathfinding algorithms, outperforming the conventional Dijkstra's algorithm in terms of energy efficiency.

6. Recommendation

Further studies should focus on improving energy efficiency at a larger scale using Mininet. To address the challenges of traffic load and network efficiency, it would be beneficial to explore advanced solutions such as dynamic load balancing, traffic prediction algorithms, and more

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921

sophisticated sleep mode strategies. These approaches could optimize resource utilization and reduce energy consumption while improving the general performance of massive data centre networks.

References

[1] B.G. Assefa and O. Oz asap,"A survey of energy efficiency in sdn : Software-based methods and optimization models," Journal of Network andComputerApplications,2019.

[2] Salman O, El hajj IH, Kayssi A, Chehab A (2016) SDN controllers: a comparative study. In: Proceedings of the 18th Mediterranean electro technical conference MELECON, Limassol, Cyprus, pp 18–20.

[3] D. K reutz, F. M. V. Ramos, P. E. Ver ´ıssimo, C. E. Rothenberg, S. Azodol molky, and S. Uhlig, "Software- defined networking: A comprehensive survey,"Proceedings of the IEEE, vol.103,no.1,pp. 14–76,Jan2015

[4] Go line lli ES (2015) A software defined networking evaluation approach to distributing load. Master's Thesis submitted to the Department of Informatics, University of Oslo.

[5] Metzler, Jim, and Ashton Metzler. "The 2015 Guide to SDN and NFV." January, available at: https://www. a10networks. com/sites/default/files/resource-files/2015Ebook-A10-all. pdf [accessed: 20 Aug 2016] (2015).

[6] Zhong H, et al (2015) An efficient SDN load balancing scheme based on variance analysis for massive mobile users. Hindawi Publishing Corporation Mobile Information Systems, Article ID 241732.

[7] W. Zhang, X. Zhang, H. Shi, and L. Zhou, "An efficient latency monitoring scheme in software defined networks,"Future Generation ComputerSystems,Elsevier,vol.83,pp.303–309,2018.

[8] M. Cokic and I. Seskar, "Software defined network management for dynamic smart grid traffic,"Future Generation Computer Systems, Elsevier,vol.96,pp.270–282,2019.

[9] C. Qiu, S. Cui, H. Yao, F. Xu, F. R. Yu, and C. Zhao, "A novel qos enabled load scheduling algorithm based on rein for cement learning in software-defined energy internet,"Future Generation Computer Systems, Elsevier, vol.92, pp.43–51,2019.

[10] Shang, Fengjun, Lin Mao, and Wenjuan Gong. "Service-aware adaptive link load balancing mechanism for Software-Defined Networking." Future Generation Computer Systems 81 (2018): 452-464.

[11] Hacham,Sura, Norashidah Md Din, and Nagaletchumi Balasubramanian. "A Bio-Inspired Territorial Predator Scent Marking Algorithm in Software-Defined Data Center." 2022 4th International Conference on Smart Sensors and Application (ICSSA). IEEE, 2022.

[12] Chaudhary, Rajat, and Neeraj Kumar. "EnFlow: An energy-efficient fast flow forwarding scheme for software-defined networks." IEEE Transactions on Intelligent Transportation Systems 22.8 (2020): 5293-5309.

[13] Son, Jungmin, and Rajkumar Buyya. "A taxonomy of software-defined networking (SDN)-enabled cloud computing." ACM computing surveys (CSUR) 51.3 (2018): 1-36.

[14] Shirmarz, Alireza, and Ali Ghaffari. "Performance issues and solutions in SDN-based data center: a survey." The Journal of Supercomputing 76.10 (2020): 7545-7593.

[15] Son, Jungmin. Integrated provisioning of compute and network resources in Software-Defined Cloud Data Centers. Diss. University of Melbourne, Parkville, Victoria, Australia, 2018.

[16] Kaur, Karamjeet, Veenu Mangat, and Krishan Kumar. "A review on Virtualized Infrastructure Managers with management and orchestration features in NFV architecture." Computer Networks 217 (2022): 109281.

[17] Okafor, K. C., et al. "Towards complex dynamic fog network orchestration using embedded neural switch." International Journal of Computers and Applications 43.2 (2021): 91-108.

[18] Singh, Sarabjit. "SD-WAN service analysis, solution and its applications." (2018).

[19] Jain, Vanita, Vivek Yatri, and Chaitanya Kapoor. "Software defined networking: State-of-the-art." Journal of High-Speed Networks 25, no. 1 (2019): 1-40.

[20] B. G. Assefa and O. Ozkasap, "RESDN: A novel metric and method for energy efficient routing in software defined networks," IEEE Trans. Netw. Service Manage., vol. 17, no. 2, pp. 736–749, Jun. 2020.

[21] W. Chen, X. Yin, Z. Wang, X. Shi, and J. Yao, "Placement and routing optimization problem for service function chain: State of art and future opportunities," Commun. Comput. Inf. Sci., vol. 1254, pp. 176–188, Oct. 2020.

[22] M. F. Tuysuz, Z. K. Ankarali, and D. Gözüpek, "A survey on energy efficiency in software defined networks," Comput. Netw., vol. 113, pp. 188–204, Feb. 2017.

[23] J. Zhang, M. Ye, Z. Guo, C.-Y. Yen, and H. J. Chao, "CFR-RL: Traffic engineering with reinforcement learning in SDN," IEEE J. Sel. Areas Commun., vol. 38, no. 10, pp. 2249–2259, Oct. 2020.

[24] Gomez, Jose, et al. "A survey on network simulators, emulators, and testbeds used for research and education." Computer Networks 237 (2023): 110054.

[25] Tropea, Mauro, and Nunzia Palmieri. "Software defined networking emulator for network application testing." Open Architecture/Open Business Model Net-Centric Systems and Defense Transformation 2022. Vol. 12119.

SPIE, 2022.
[26] Alwasel, Khaled Salh. Modeling and simulation of data-driven applications in SDN-aware environments. Diss. Newcastle University, 2021.
[27] Yu, Yinbo, et al. "Fault management in software-defined networking: A survey." IEEE Communications Surveys & Tutorials 21.1 (2018): 349-392.
[28] H. Z. Abidin and N. M. Din, "Provisioning an energy efficient with maximum coverage WSN through biological inspired sensor node placement," 2014 IEEE 2nd International Symposium on Telecommunication Technologies (ISTT), Langkawi, Malaysia, 2014, pp. 341-345, doi: 10.1109/ISTT.2014.7238232.
[29] Lin, Shan, et al. "Software-defined networking enabled optical data center network with flexible QoS provisioning." Optics Communications 530 (2023): 129129.
[30] Let, G. Shine, et al. "Software-defined networking routing algorithms: issues, QoS and models." Wireless Personal Communications 131.3 (2023): 1631-1661.
[31] Monteiro, Rui Pedro da Cunha. Green communications: an environment to support energy-aware networks developments. Diss. 2023.
[32] Torkzadeh, Samaneh, Hadi Soltanizadeh, and Ali A. Orouji. "Energy-aware routing considering load balancing for SDN: a minimum graph-based Ant Colony Optimization." Cluster Computing 24.3 (2021): 2293-2312.
[33] Son, Jungmin, and Rajkumar Buyya. "A taxonomy of software-defined networking (SDN)-enabled cloud computing." ACM computing surveys (CSUR) 51.3 (2018): 1-36.
[34] Kaur, Kuljeet, Sahil Garg, Georges Kaddoum, Neeraj Kumar, and Francois Gagnon. "SDN-based Internet of autonomous vehicles: An energy-efficient approach for controller placement." IEEE Wireless Communications 26, no. 6 (2019): 72-79.
[35] H. Aslam and S. Munawar, "Smart Energy Conservation in Data Centers Using Machine Learning Based Software-Defined Networking," 2023 6th International Conference on Energy Conservation and Efficiency (ICECE), Lahore, Pakistan, 2023, pp. 1-10, doi: 10.1109/ICECE58062.2023.10092487. https://mininet.org/.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 8, No. 6: 8956-8967, 2024
DOI: 10.55214/25768484.v8i6.3921