International Journal of Communication Networks and Information Security
2024, 16(4)
ISSN: 2073-607X,2076-0930
https://ijcnis.org/
Research Article

An efficient hybrid software-defined networking (HSDN) approach is proposed to optimise the distribution of network traffic in traffic engineering

Dr. K Shanthi Latha[1], Dr.Maram Ashok[2]
Assistant Professor, Computer science and Engineering, Malla Reddy College of Engineering , Hyderabad,Telangana, India.
rshanthilatha@gmail.com
Principal & Professor, Computer science and Engineering, Malla Reddy College of Engineering , Hyderabad,Telangana, India.
principal@mrce.in

ARTICLE INFO

Received: 30 Aug 2024
Accepted: 7 Oct 2024

ABSTRACT

Traffic engineering (TE) is a very efficient technique for optimizing the distribution of network traffic, leading to improved performance of a hybrid software-defined network (SDN). Traditionally, TE systems have mostly used heuristic methods to centrally optimize the setting of link weights or traffic splitting ratios while dealing with static traffic demand. It is important to realise that as the network grows and management gets trickier, centralised traffic engineering (TE) methods have a hard time keeping up with the huge amount of data they have to process and take a long time to find the best way to route traffic when there are problems or changes in the demand for traffic on the network. Aim to improve the implementation of dynamic and efficient routing in traffic engineering (TE). ring (TE). Efficient hybrid SDN (hSDN) schemes are crucial for the preservation of global information and resource allocation to several applications running in the network. These schemes specifically focus on topology identification, traffic categorization, energy management, and load balancing algorithms. Therefore, in order to enhance network performance by enhancing traffic engineering (TE), it becomes crucial to provide appropriate resources to applications in the network. To accomplish a worldwide optimization goal, An interactive setting for training routing agents with access to partial link use data. To improve the distribution of credit in a multi-agent system, A differential reward assignment method. The purpose of this mechanism is to motivate agents to make choices that are more optimal. The comprehensive simulations conducted on real traffic traces demonstrate the superiority of improving traffic engineering (TE) performance, especially in scenarios when traffic demands vary or network outages happen.

Keywords:-Software Defined Networks, Traffic engineering, network monitoring, network management, network measurement.

1 INTRODUCTION

As a result of the rapid increase in Internet traffic, TE has become more prominent in its efforts to balance traffic and enhance network performance [1]. Currently, the efficiency of typical dispersed networks is mostly limited by the shortest route routing protocols they use. Fortunately, the introduction of SDN architecture allows for the separation of control plane and data plane, which enables TE to develop adaptable solutions for more efficient traffic routing. However, implementing a fully equipped SDN network, which replaces all outdated routers with SDN switches, presents both economic and technological challenges [3]. Consequently, Internet Service Providers (ISPs) routinely use hybrid SDN, which involves deploying SDN switches partly in conventional dispersed networks, as a realistic means of achieving more intelligent traffic engineering. Multiple studies have shown that traffic engineering in hybrid software-defined networking may produce network performance that is comparable to a fully SDN-enabled network [4].

Prior solutions for hybrid software-defined networking mostly focused on the development of different heuristics [4]–[7]. Humans often create these heuristics to optimize routing strategies for specific traffic demands. Consequently, the routing rules based on these heuristics always experience a decline in performance due to their inability to adjust to the constantly changing network environment, such as varying traffic needs or network connection failures. Furthermore, the significant computational and communication burden makes it unfeasible for these heuristics to quickly recalculate and implement a suitable routing plan in a network environment that is constantly changing.

Essential algorithms have shown significant promise in addressing dynamic decision-making challenges by allowing both experience-driven and model-free control [8]. Instead of relying on human-designed rules, reinforcement learning (RL) trains an intelligent agent to dynamically and efficiently develop optimal strategies based on various contexts. Reinforcement learning may autonomously gather a substantial amount of relevant knowledge by iteratively engaging with a simulated environment and learning from mistakes without the need for external guidance. The agent's collected knowledge enables it to uncover concealed patterns within previous data and build a direct correlation between dynamic settings and optimum strategies.

Several innovative studies have sought to use established methods to tackle the TE issues in the hybrid SDN [9], [10]. However, when the network becomes larger, the number of possible actions rises fast, making it difficult to accurately determine the best routing option in real-time with only one agent [11]. This study uses a multi-agent framework to create a distributed traffic engineering system in a hybrid software-defined networking environment. Previous research used a single-agent reinforcement learning framework to create a centralized TE system.

To tackle the traffic engineering problem, The placement of SDN nodes in an HSDN environment, which combines centralized and decentralized paradigms, in order to tackle the traffic engineering problem. The Intelligent Node Placement technique leverages the node's degree and traffic characteristics to strategically place SDN nodes in the network, thereby enhancing network flexibility. The incorporation of flexibility into the network guarantees multipath routing, resulting in a reduction of the maximum link utilization (MLU) and the underutilization of connections, hence improving traffic engineering inside the network. This study distinguishes itself from prior research by using a non-greedy approach to placing SDN nodes and considering all possible pathways for data routing. Table 1 presents a comparison between the proposed INP method and the existing research on SDN node placement in HSDN.

The article structures the remaining portion as follows: Section 2 provides an overview of the existing research and solutions in the field of TE. Section 3 presents the problem definition and a detailed explanation of our proposed approach, CMRL, which encompasses both the offline training phase and the online inference phase. In Section 4, the real-world results of several algorithms are shown on real network topologies and traffic traces when traffic demands change or the network goes down. Ultimately, The conclusion and propose potential avenues for further study in Section 5.

2 RELATED WORK

This section showcases many studies that examine traffic engineering and traffic categorization within the context of software-defined networking. Traffic engineering is a critical process that has been a prominent subject for improving the internet's performance for many years. Karakus et al. [5] conducted comprehensive research on quality of service (QoS) in networks that use OpenFlow technology. They examined existing studies that focused on improving QoS. The text provided a comprehensive summary of the interconnections between QoS and SDN. The research suggests that the notion of software-defined networking may enhance quality of service. Additionally, it scrutinizes the Quality of Service functionalities of OpenFlow and outlines the various obstacles and issues that require resolution to improve QoS capabilities.

The results showed an increase in the data transfer rate and a reduction in the packet delivery time for the designated data streams [6]. In their work, Yan et al. [7] conducted a comprehensive examination of the methods used in traffic classification and evaluated the machine learning algorithms typically utilized in traffic classification inside SDNs. In addition, they provided an overview of the overall difficulties encountered while classifying traffic, as well as suggestions for improving traffic classification in software-defined networking. In a corporate setting, Amaral et al. [8] introduced a straightforward framework for gathering traffic using SDN. The captured internet traffic using the OpenFlow protocol and then employed ensemble learning techniques to analyze the obtained data. The findings demonstrated that these algorithms achieved a high level of accuracy, indicating that the suggested solution is well-suited for identifying traffic flows of relevance in small company networks.

Lashkari et al. [9] introduced a method for identifying and analyzing Tor traffic by examining the timing patterns of the data transfers. The work focused only on statistical characteristics derived from the timing information. The authors demonstrated the efficacy of using time-based characteristics for detecting Tor traffic. Furthermore, These characteristics to categorize the traffic and distinguish various applications. Furthermore, studies have demonstrated that the flow timeout significantly affects the efficiency of categorization. Rezaei et al. [10] introduced a comprehensive framework for traffic classification using deep learning. They also offered specific instructions for the various stages of the traffic classification process, including data collection and model selection. The authors highlighted the use of deep learning in traffic categorization and discussed unresolved issues and obstacles in this field. Choubey et al. [11] developed a technique for classifying traffic based on artificial neural network models and one-dimensional convolutional neural network (CNN) models. The proposed technique used Andrew Moore's dataset, which included data from two separate sites. The results of the proposed system showed that both models achieved high accuracy without encountering overfitting, while their testing durations were shorter compared to machine learning models. Xu et al. [12] proposed a network architecture that combines a deep learning-powered traffic classification system with SDNs. The traffic categorization system as a virtualized network function (VNF) to reduce the load on the SDN controller and prevent network paralysis from any VNF failure. The proposed system aims to improve the quality of service by assigning separate network resources to each application. The experiments demonstrate that the proposed model surpasses existing classification approaches, and the controller can allocate suitable route paths for different flows.

Table 1: Table of comparisons for the HSDN traffic engineering projects

Reference	Problem	Objective	Typologies Used
13	TE-issue in HSDN	Minimizing MLU while reducing packet losses & improvenetwork utilization	Exodus, Abovenet
14	TE-issue in HSDN	Reduction of network	SNDLIB

		capacity upgrade through the use of SDNenabled routers	
15	Load balancing in hSDN	Dijikstra-Repeat path algoritm is proposed to enable loop free multipath routing while minimizing congestion	SNDLIB
16	Maximizing Network Control Ability (NCA)& network flexibility with minimizing cost	Traditional routing switches are upgraded in order to achieve NCA & flexibility with limited budget	Exodus, Ebone , Above net
17	HSDN deploy ment problem with resource constraints	HybridScore strategy is proposed which diminishes re-quired switch upgrading	RealInternet Enterpr ise datase
Propsoed Model	TE-improvement in hSDN	mproving network utilization through SDN nodes placement which addresses minimizing MLU and improving under-utilization in the network	SNDLIB

3 SYSTEM DESIGN AND IMPLEMENTATION

Evaluate the routing in conventional networks and the opportunistic placement of SDN nodes, while also conducting a thorough examination of the proposed architecture. The performance comparisons using test scenarios with simulations of random graphs. The proposed method utilizes comprehensive network knowledge, whereas the dispersed algorithm depends on local information, and the hybrid technique integrates both. The experimental findings clearly show that the hybrid algorithm is better at lowering route costs, effectively balancing optimization and efficiency. The centralized method ranks in the second position; however, the distributed approach results in greater expenses owing to the constraint of restricted local knowledge. This study provides valuable insights into optimizing route computation and contributes to the development of future breakthroughs in software-defined networking.

3.1 Routing with OSPF

Consider a network consisting of 4 nodes, as seen in Figure 1. An implicit server connects each node. Let's suppose that the cost of each of the options is equivalent. Each row denotes the traffic originating from the designated node, A, B, C, or D. Let X, a 4x4 matrix, represent the network's use. The shortest pathways for nodes A, B, C, and D are as follows: The shortest pathways for nodes A, B, C, and D are as follows: A-B, A-B-D, A-C for node A; B-A, B-D, B-C for node B; C-A, C-D, C-B for node C; and D-B-A, D-B, D-C for node D.

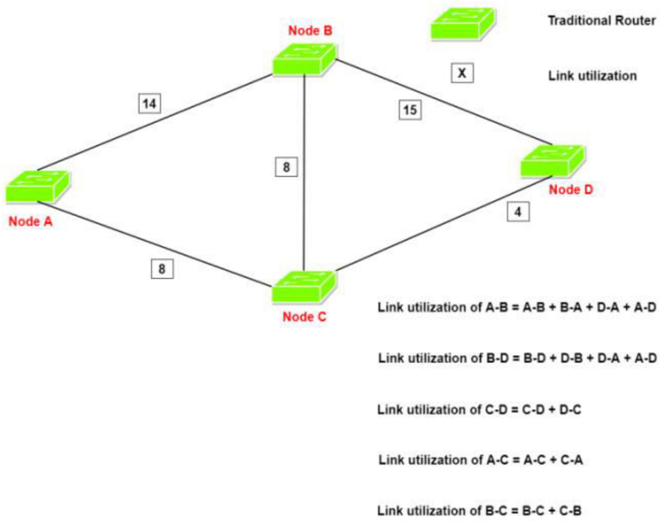

Figure 1: A conventional network, OSPF routing

Calculate the mean of all the linkages to identify the extensively used connections. Calculating the network's average link usage and round it to 10. We calculate the standard deviation from the average value. The standard deviation is 4.19, rounded to 4. Propose a link, Lp, that exhibits higher or lower usage when the standard deviation (σ) is deducted from the mean link utilization for overutilized or underused connections. Figure 1 demonstrates that connections with usage exceeding 14 and falling below 6 are either overutilized or underutilized. Links B-D and C-D are the ones experiencing excessive or insufficient use. The traffic utilization matrix X indicates that both A and B transfer the same amount of traffic, specifically 12 units. Additionally, C transfers 8 units of traffic, and D transfers 7 units. In order to optimize traffic distribution across the network, it is crucial to strategically position SDN nodes to minimize the highest level of link use. Transmitting data across heavily used connections may lead to a decline in performance, necessitating the need to handle such situations in the network.

Table 2: Symbol table used

Variables	Meaning
Lavg	Average link utilization in the network
σ	Standard deviation
K	Number of links in the network

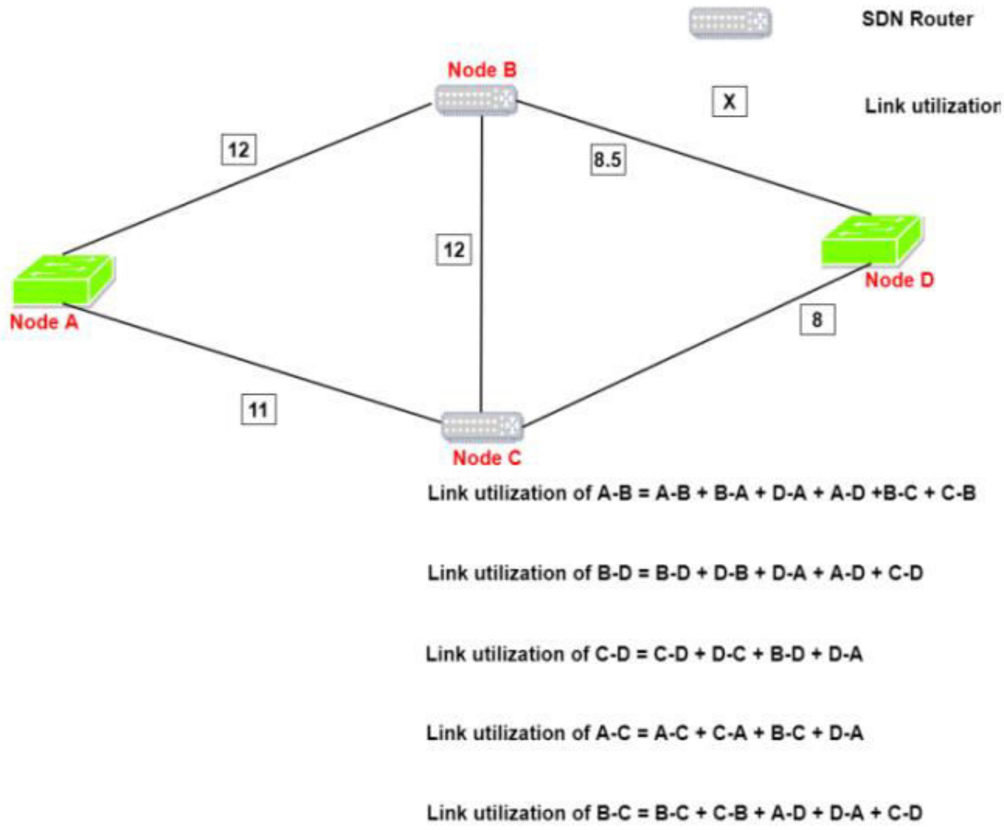

Figure 2: Placement of SDN nodes in HSDN determined by greed

$$L_{avg} = \frac{Load\_of\_all\_the\_links\_in\_the\_network}{K}$$

$$\sigma = \sqrt{\frac{(L_1 - L_{avg})^2 + (L_2 - L_{avg})^2 + ... + (L_k - L_{avg})^2)}{K}}$$

$$Lp - Lavg \geq \sigma$$

3.2 Greedy based placement

As shown in Figure 2, the greedy-based method for SDN deployment recommends placing nodes in areas with the highest degree. Placing nodes at the greatest degree guarantees that flow splitting (multipath routing) occurs over several connections, resulting in consistent use of all the links. This configuration recommends positioning SDN nodes at nodes B and C. This design allows for the distribution of data originating from nodes B and C over many outgoing connections. Spliting data between nodes B and C into either two or three separate paths. One route utilizes the OSPF protocol and has a distance of 1, whereas another path traverses the B-C link with a distance of 2. The intermediate node A/D has the highest route length of 3. Nevertheless, if node B has to transfer data to node A, there are only two viable channels. The first option is the shortest, which involves using the B-C connection as an intermediary. Avoid the second approach, B-D-C-A, as it does not provide the shortest path from D to A via D-C-A, potentially leading to loops. Therefore, SDN nodes should avoid transmitting data to outdated nodes that share the same routes as the ones currently in use. This example maintains a constant division ratio of SDN nodes at 0.5. Figure 2 displays the total link usage of the network with the following values: A-B = 12, B-D = 8.5, B-C = 12, A-C = 11, and C-D = 8. Therefore, by strategically placing SDN nodes in the network, The traffic engineering by reducing the minimum link utilization and optimizing the use of lightly loaded connections in the network.

3.3 Intelligent Node Placement

Most studies have placed nodes based on the number of outgoing routes they contain. Placing nodes solely based on outgoing channels is ineffective, as it doesn't lead to enhanced traffic engineering until a node experiences extensive use. After running OSPF in a conventional network, the intelligent node placement strategy considers the traffic matrix, node degree, and the establishment of loop-free pathways.

3.4 Illustrative Example-INP

The suggested positioning of SDN nodes considers the node's degree and the volume of incoming and outgoing traffic throughout the placement process. Initially, we create a list to identify the nodes under consideration for replacement. The traffic matrix consists of nodes A and B.

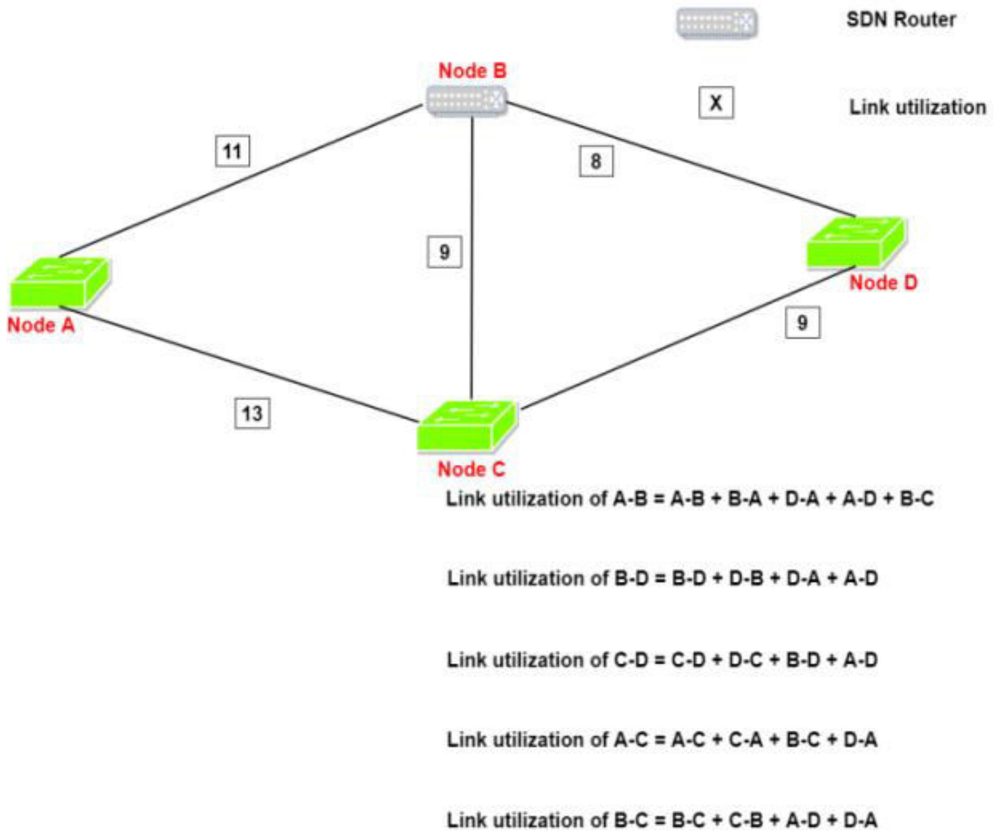

Figure 3: Proposed INP-scheme

The node is facing increased traffic pressure compared to C and D. Moreover, the previous phase's analysis of the node's traffic matrix determines the shortest route for both the node's origin and intermediate traffic. This information to replace the outdated node. Select node B for replacement because it has a higher degree and greater traffic needs than node A, which only has a degree of 2. This stage involves the substitution of a traditional node with a software-defined networking node, as seen in Figure 3. Figure 3 provides information that considers both high and low usage, leading to an improvement in traffic engineering. Comparing the proposed INP to the greedy-based placement in Fig. 2, which places SDN nodes at the maximum degree, reveals that the INP requires only a few strategically positioned SDN nodes with reduced costs to improve TE in the network. During the process of flow splitting, it is important for SDN nodes to avoid sending data to legacy nodes when there is a conflict between different (centralized or decentralized) planes. This becomes particularly crucial when the packet's previous path aligns with the shortest route.

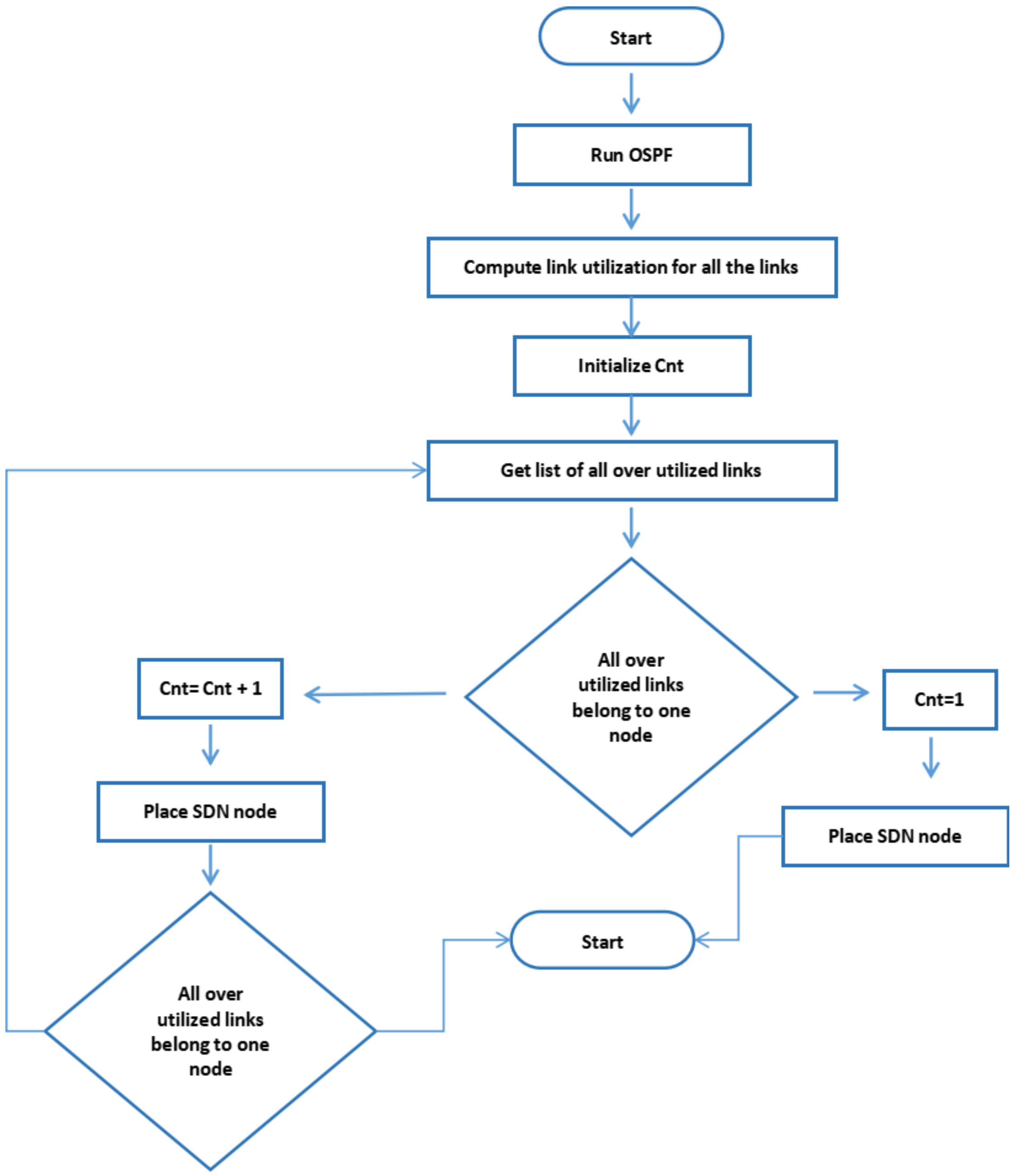

Figure 4: Flowchart for the INP scheme

The controller in HSDN obtains topology information by using either layer 2 or layer 3 protocols. Network nodes communicate by exchanging Link State Advertisements (LSAs) to gather information about the network's structure. Upon receiving the link state advertisements from all nodes, the node obtains the network's whole topology. To transport data from the sender to the recipient, each node uses the shortest path algorithm. Once the routing process concludes, Traffic matrix to calculate the link usage for each connection. Computing the average utilization of the links by evaluating the overall use of each connection. As previously mentioned, determine the standard deviation after computing the average. Links with usage below or above the link utilization subtracted from the standard deviation are considered underutilized or heavily used links, respectively. Currently, maintain a tally to track the number of extensively used connections. If all the heavily used connections connect to the same node, could implement a single SDN placement to reduce the number of links needed. Figure 4 illustrates that if connections connect to distinct nodes, must replace all nodes with SDN nodes. Given that a link signifies a connection between two nodes, excessive use of a connection necessitates identifying the nodes that require replacement. First, prioritize a node based on the number of heavily used connections it possesses. Furthermore, the network grants a node a higher priority if it generates a significant amount of traffic. The degree of connectivity between nodes in the network determines priority. Figure 4 illustrates the sequential process of positioning SDN nodes incrementally within the HSDN framework.

4 RESULTS AND DISCUSSION

4.1 Simulation Environment

In this study, conduct the simulation experiment using an actual network architecture from SNDLIB. SNDLIB contains both the network's topological data and traffic demand information. Figures 5a–5c demonstrate the use of three distinct network typologies: Abilene, Atlanta, and Geant. The Abilene topology has an average node degree of 2.5 and encompasses a total of 132 requests. Atlanta has a mean node degree of 4 and a total of 210 requests. The Geant topology has an average node degree of around 3.27 and consists of 462 requests. Table 3 offers a thorough compilation of all the relevant network topology data utilized in the study. The network using a directed graph (V,E), with V representing the topology's vertices and E representing its edges. The INP approach using MATLAB R2016b on a Windows 10 OS. This study compares its findings with both the OSPF and greedy-based methodologies carry out several repetitions of each experiment to measure the average performance.

Table 3: SNDLIB typologies used

Topology Name	Nodes	Links
Abilene	12	15
Atlanta	15	22
Geant	22	36

4.1 Maximum Link Utilization

Compare the cumulative distribution function to the mean link utilization (MLU). The greedy-based techniques have a total of 3 SDN nodes in Abilene, 4 SDN nodes in Atlanta, and 6 SDN nodes in Geant. Factors such as traffic patterns, link usage, and node degree determine the placement of SDN nodes in the INP framework. This significantly reduces the need for SDN nodes compared to greedy-based systems. Specifically, the Atlanta, Abilene, and Geant networks necessitate 2, 2, and 4 nodes, respectively. Due to its limited understanding of link loads and traffic patterns, OSPF's performance suffers in all analyzed topology scenarios. OSPF optimizes connection consumption by only routing the shortest routes. Strategically install software-defined networking nodes using greedy algorithms to enhance performance compared to the Open Shortest Path First (OSPF) routing protocol. As a result of the widespread use of SDN nodes are widely used in the network's infrastructure, the greedy method sometimes exceeds the recommended INP.

Figure 5a: Topology of Abilene, Image-Source

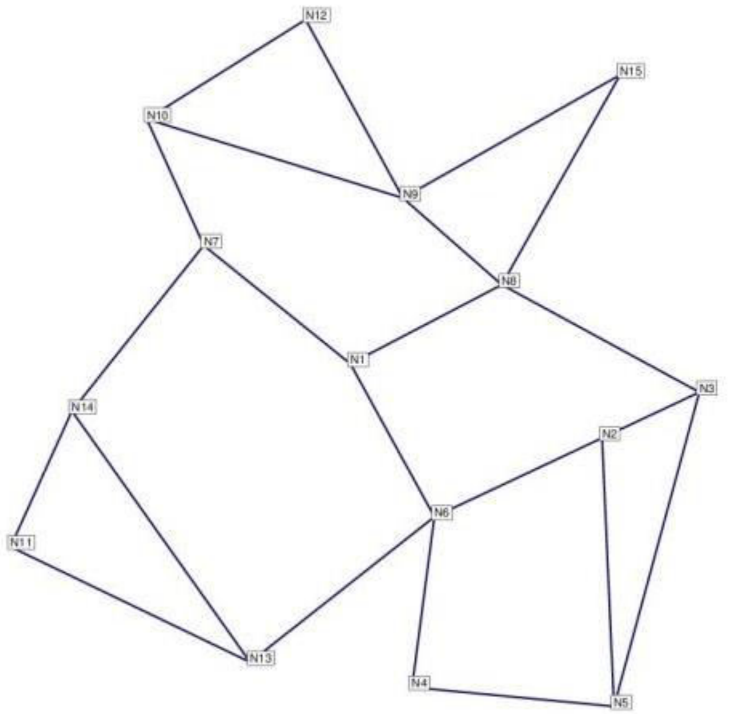

Figure 5b: Atlanta topology, Source Image

Figure 5c: Geant topology, Source Image

Figure 5: Atlanta, Abilene, and Geant topologies were used in the assessment

INP consistently outperforms other topologies in almost all scenarios examined. INP has a decreased mean length of utterance in comparison to both OSPF and the Greedy method. The suggested INP demonstrates an average decrease of 15–19% in the MLU of the investigated topologies in comparison to OSPF routing. INP has superior performance compared to greedy techniques, with an observed improvement of about 5-7% across different topologies.

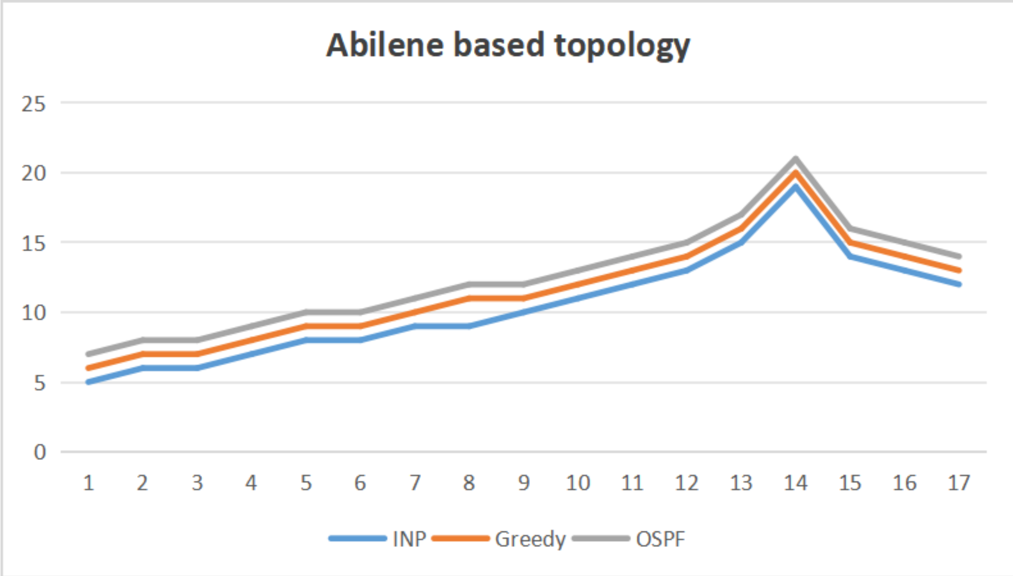

Figure 6: Maximum Abilene topology link usage graph

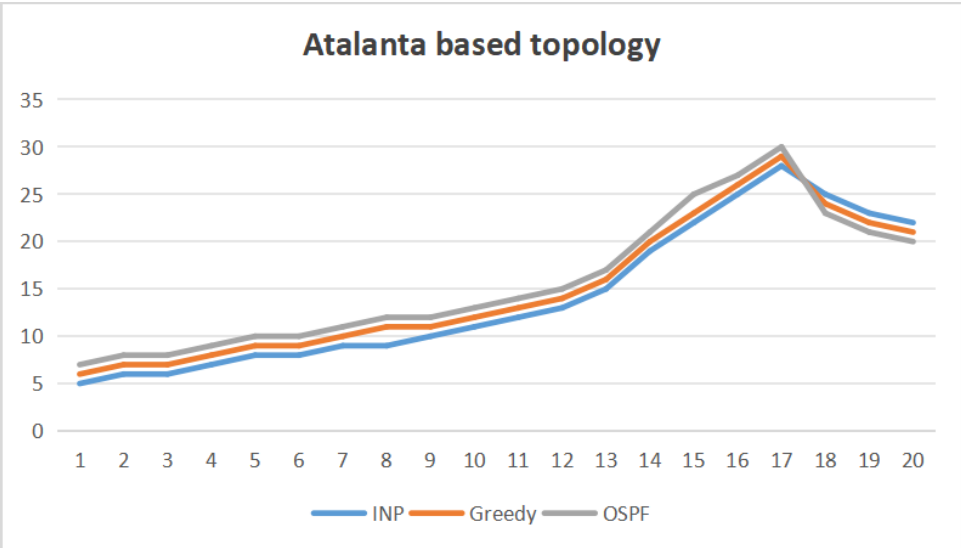

Figure 7: Atlanta topology's graph of maximum link usage

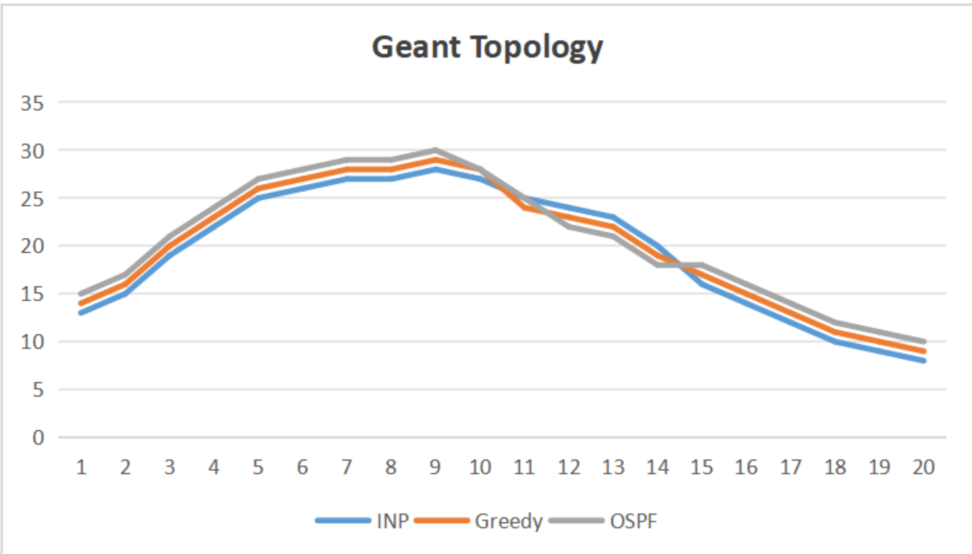

Figure 8: Geant topology's maximum link consumption graph

5 CONCLUSION:

Advancing HSDN traffic engineering technology is critical for facilitating its wider use. One example of this is the Google B4 network. The B4 network utilizes the distinctive characteristics of HSDN to accomplish the TE requirements that conventional networks cannot fulfill. This paper introduces a systematic structure for managing and optimizing the flow of network traffic in software-defined networks. This study examines traffic engineering technology from two perspectives: traffic measurement and traffic management. In the context of network measurement, each layer of the network has network parameters that are associated with and indicate the present condition of the network. Generally, monitor the measurements of both the application layer and the network layer. The traffic matrix is a reliable and precise data source that effectively reflects the current state of the network. Nevertheless, more research is required to enhance the accuracy of traffic matrix prediction technology. The primary objective of network management is to implement effective traffic scheduling techniques based on flow monitoring technologies to meet particular needs and optimize network performance. In the context of the IP/HSDN hybrid network, Analyze four particular components: QoS guarantee, traffic load balancing, traffic management, and energy-saving scheduling. Future projections indicate that traffic engineering based on multiprotocol label switching (MPLS) in hybrid software-defined networking (HSDN) and the study of TE performance will become increasingly important.

References

[1] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, ―A Detailed Investigation and Analysis of Using Machine Learning Techniques for Intrusion Detection,‖ IEEE Commun. Surv. Tutorials, vol. 21, no. 1, pp.686–728, 2019, doi: 10.1109/COMST.2018.2847722.

[2] V. Kapoor and R. Yadav, ―A Hybrid Cryptography Technique for Improving Network Security,‖ IJCA, vol. 141, no. 11, pp. 25–30, May 2016, doi: 10.5120/ijca2016909863.

[3] C. Yu, J. Lan, Z. Guo, and Y. Hu, ―DROM: Optimizing the Routing in Software-Defined Networks With Deep Reinforcement Learning,‖ IEEE Access, vol. 6, pp. 64533–64539, 2018, doi:10.1109/ACCESS.2018.2877686.

[4] N. Awadallah Awad, ―Enhancing Network Intrusion Detection Model Using Machine Learning Algorithms,‖ Computers, Materials & Continua, vol. 67, no. 1, pp. 979–990, 2021, doi:10.32604/cmc.2021.014307.

[5] Muthukumaran V., V. V. Kumar, R. B. Joseph, M. Munirathanam, and B. Jeyakumar, ―Improving Network Security Based on Trust-Aware Routing Protocols Using Long Short-Term Memory-Queuing SegmentRouting Algorithms:,‖ International Journal of Information Technology
Project Management, vol. 12, no. 4, pp. 47–60, Oct. 2021, doi:10.4018/IJITPM.2021100105.

[6] S. Akbar, J. A. Chandulal, K. N. Rao, and G. S. Kumar, ―Improving network security using machine learning techniques,‖ in 2012 IEEE International Conference on Computational Intelligence and Computing Research, Coimbatore, India: IEEE, Dec. 2012, pp. 1–5. doi: 10.1109/ICCIC.2012.6510197.

[7] R. Ahmad, R. Wazirali, and T. Abu-Ain, ―Machine Learning for Wireless Sensor Networks Security: An Overview of Challenges and Issues,‖ Sensors, vol. 22, no. 13, p. 4730, Jun. 2022, doi:10.3390/s22134730.

[8] S. Anbalagan et al., ―Machine-Learning-Based Efficient and Secure RSU Placement Mechanism for Software-Defined-IoV,‖ IEEE Internet Things J., vol. 8, no. 18, pp. 13950–13957, Sep. 2021, doi:10.1109/JIOT.2021.3069642.

[9] S. Nanda, F. Zafari, C. DeCusatis, E. Wedaa, and B. Yang, ―Predicting network attack patterns in SDN using machine learning approach,‖ in 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks, Palo Alto, CA: IEEE, Nov.
2016, pp. 167–172. doi: 10.1109/NFV-SDN.2016.7919493.

[10] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan, ―Machine Learning in Wireless Sensor Networks: Algorithms, Strategies, and Applications,‖IEEE Commun. Surv. Tutorials, vol. 16, no. 4, pp. 1996–2018, 2014, doi: 10.1109/COMST.2014.2320099.

[11] J. Ramprasath and V. Seethalakshmi, ―Secure access of resources in software-defined networks using dynamic access control list,‖ Int J Communication, vol. 34, no. 1, p. e4607, Jan. 2021, doi:10.1002/dac.4607.

[12] V. Vimal et al., ―Enhance Software-Defined Network Security with IoT for Strengthen the Encryption of Information Access Control,‖ Computational Intelligence and Neuroscience, vol. 2022, pp. 1–10, Oct.2022, doi: 10.1155/2022/4437507.

[13] S. Shin, L. Xu, S. Hong, and G. Gu, ―Enhancing Network Security through Software Defined Networking (SDN)‖.

[14] A. Ahmad, E. Harjula, M. Ylianttila, and I. Ahmad, ―Evaluation of Machine Learning Techniques for Security in SDN,‖ in 2020 IEEE Globecom Workshops (GC Wkshps, Taipei, Taiwan: IEEE, 2020, pp. 1–6. doi: 10.1109/GCWkshps50303.2020.9367477.
[15] J. A. Pérez-Díaz, I. A. Valdovinos, K.-K. R. Choo, and D. Zhu, ―A Flexible SDN-Based Architecture for Identifying and Mitigating LowRate DDoS Attacks Using Machine Learning,‖ IEEE Access, vol. 8, pp. 155859–155872, 2020, doi: 10.1109/ACCESS.2020.3019330.
[16] S. Latif, Z. Zou, Z. Idrees, and J. Ahmad, ―A Novel Attack Detection Scheme for the Industrial Internet of Things Using a Lightweight Random Neural Network,‖ IEEE Access, vol. 8, pp. 89337–89350,2020, doi: 10.1109/ACCESS.2020.2994079.
[17] ―CICIDS2017.‖ https://www.kaggle.com/datasets/cicdataset/cicids2017 (accessed Sep. 18, 2023).
[18] Z. Cui, R. Ke, Z. Pu, and Y. Wang, ―Stacked Bidirectional and Unidirectional LSTM Recurrent Neural Network for Forecasting Network-wide Traffic State with Missing Values.‖ arXiv, May 23, 2020. Accessed: Sep. 19, 2023. [Online]. Available:http://arxiv.org/abs/2005.11627.