International Journal of Basic and Applied Sciences, 14 (6) (2025) 80-87

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/peb76802
Research paper

A Dynamic Traffic Engineering Strategy Using Latency-Aware Congestion Control in Software-Defined Networks

S. D. Vijayakumar [1] *, R. Praveenkumar [2], M. Prakash [3], T. Rajkumar [4], P. A. Selvaraj [5], P. Karunakaran [1]

[1] Department of AI & DS Nandha Engineering College, Erode, India
[2] Department of ECE Nandha Engineering College, Erode, India
[3] Department of ECE Kangeyam Institute of Technology, Tirupur, India
[4] Department of ECE Nandha College of Technology, Erode, India
[5] Department of Computer Science and Design Kongu Engineering College, Erode, India
**Corresponding author E-mail: mail2vijay.sd@gmail.com*

Received: July 8, 2025, Accepted: September 13, 2025, Published: October 5, 2025

Abstract

This work focuses a wide range of modules centered on latency-aware optimization strategies in order to meet the increasing need for low-latency communication in contemporary networks. The system incorporates sophisticated congestion control algorithms including LEDBAT, TCP Vegas, and BBR, which regulate transmission rates more efficiently than conventional loss-based techniques by using delay-based metrics like round-trip time (RTT) and queuing delay. To guarantee effective path selection under latency limitations, traffic engineers use multipath routing strategies as ECMP and MPTCP, modified Dijkstra's algorithm with latency weights, and constraint-based shortest path first (CSPF). Utilizing the programmability of Software-Defined Networking (SDN), the system integrates metaheuristic methods including genetic algorithms, ant colony optimization, and particle swarm optimization along with intelligent routing strategies utilizing reinforcement learning. By using real-time latency feedback, these techniques allow for dynamic and adaptive routing decisions. OpenFlow and P4 flow rerouting features improve the system's responsiveness to network conditions even more. Mechanisms for monitoring and feedback are essential for facilitating accurate decision-making. The SDN controller's RTT measurement modules continuously measure connection latency, and exponential weighted moving average (EWMA) methods smooth the data gathered to prevent overreactions to brief variations. These components work together to create a strong framework for next-generation network environments that optimize latency.

Keywords: *Software-Defined Networking (SDN); Retransmission Timeout (RTO); Traffic Engineering; Network Performance Optimization.*

1. Introduction

Low-latency communication [1] has become increasingly important in modern networking because of the growth of real-time applications like cloud services, online gaming, and video streaming. These strict latency requirements are frequently difficult for traditional networking protocols to achieve, which results in delays and decreased performance. This problem has led to the development of traffic engineering approaches and latency-aware congestion management algorithms that optimize data routing and transmission. Through the use of measures such as round-trip time (RTT), bandwidth, and queuing delay, these techniques seek to dynamically modify network behavior in order to reduce latency. Additionally, the introduction of Software-Defined Networking (SDN) [2], presents fresh possibilities for adaptive, real-time traffic management using reinforcement learning-based path selection and intelligent routing algorithms. In order to improve the performance and dependability of contemporary network infrastructures, this introduction examines a number of latency optimization strategies, such as multipath routing, congestion management algorithms, and SDN-specific approaches.
Software-Defined Networking (SDN) is a cutting-edge networking architecture [3] that allows for dynamic resource setup and centralized management by separating the control plane from the data plane. Traditionally, networks are complicated, inflexible, and challenging to expand or modify due to the close coupling between control and forwarding operations within the network devices. By establishing a centralized SDN controller with a global view of the entire network and the ability to make wise decisions based on current conditions, SDN overcomes these constraints. Network administrators may effectively apply policies, optimize routing routes, and programmatically regulate traffic flow with this architecture. SDN is particularly useful for contemporary applications [4], needing high agility, such cloud computing, data centers, and Internet of Things (IoT) environments, because it increases flexibility, scalability, and automation. SDN makes the conventional networking model more controllable, flexible, and economical by virtue of its programmability and abstraction capabilities.

A key parameter in network communication protocols [5], is the Retransmission Timeout (RTO), which establishes how long a sender must wait for an acknowledgement before sending a packet again. The packet is assumed lost or delayed if the acknowledgement is not received within this allotted period, necessitating a resend to guarantee dependable delivery. For the network efficiency and dependability to remain balanced, an accurate RTO computation is essential. An RTO that is too long can cause major delays in data delivery, while an RTO that is too short might cause congestion and needless retransmissions. Conventional RTO mechanisms may not be able to adjust effectively to changing network conditions since they rely on fixed values or crude estimations. On the other hand, adaptive RTO techniques [6], such as the one suggested in this system, intelligently modify timeout settings by taking into account variables like packet priority, hop count, and transmission delay. By decreasing latency, minimizing packet loss, and optimizing bandwidth utilization, this dynamic adjustment improves network performance.

An essential networking method, traffic engineering [7], aims to maximize the effectiveness, dependability, and performance of data transfer over a network. In order to guarantee efficient use of available bandwidth, less congestion, and balanced load distribution, it entails the analysis, prediction, and regulation of data flow. Traffic routes in conventional static networks are frequently preset and rigid, which can result in inefficiencies and possible bottlenecks. By constantly modifying routes and resource allocations in response to current network circumstances and traffic needs, traffic engineering solves these problems. Traffic engineering gains even greater strength in the context of Software-Defined Networking (SDN), since the centralized controller can collect global network data and make wise resource management and routing decisions. This guarantees Quality of Service (QoS) by allowing the network to adjust to shifting circumstances and give priority to important traffic types like voice and video. All things considered, traffic engineering is essential to increasing network utilization, increasing throughput, and preserving steady performance in contemporary, intricate network environments.

2. Literature Review

Research article [8] have proposed Controlling congestion in a software-defined network (SDN) environment is the focus of this research. In order to manage congestion in a distributed fashion, explicit router techniques, such Explicit Congestion Notification (ECN), now operate in tandem with the TCP protocol. SDN and centralized control have made it feasible to use the network state's global perspective to improve congestion control choices. In this study, we investigate the benefits of incorporating global data into distributed congestion management. By suitably setting the ECN bits of IP packets, we provide a paradigm in which the controller actively contributes to the end TCP hosts' decisions on congestion control by using its global view of the network. Without altering the SDN switches or end node TCPs, our system is incredibly simple to implement. Using the Mininet emulator, we additionally demonstrate a 30x improvement in flow completion times compared to the TCP Cubic form and a 1.7x improvement over TCP/RED. Controlling congestion in computer networks is the topic of this research. Current solutions fall into one of two categories: router-based or end-node-based. In order to decrease source traffic, the later solutions include scheduling algorithms and queue management [9] that send signals to the end hosts. By marking or dropping packets at the switch/router buffers, active queue management systems alert end nodes to congestion. Some techniques make use of both end node TCP changes and active queue management.

As in [10], this system to allow for network flexibility, programmability, and innovation, Software-Defined Networks (SDNs) have distinct control and data planes and fully decouple flow control from data forwarding. However, this also creates significant security issues in each plane and at the interfaces between the two planes. This paper focusses on the security vulnerabilities in the SDN data plane, aiming at the state-of-the-art technique to identify, detect, and mitigate them, rather than the SDN control plane, as many other literatures have done in recent research. This study examines the common models, detections, and defenses against SDN flow table overflow attacks. Following a review of the several SDN vulnerabilities, this article divides flow table overflow attacks into three categories: slow saturation, low-rate table depletion, and saturation. It also provides an overview of the attack models, detections, and mitigations for each of these categories. It lists the common attacks that can overwhelm the flow tables and highlights the primary obstacles and unresolved problems for further study. Software-Defined Networks (SDNs) provide for more flexible network setup, improved network performance, and centralized monitoring and automation by separating network control from data forwarding to create a layered architecture. The Internet of Things (IoT), 5G mobile, and industrial IoT systems have all embraced SDNs because they can meet the strict requirements of low network latency, Quality of Service (QoS), and Service Level Agreement (SLA). The basic layered design of SDNs, which consists of the application layer, control layer, and infrastructure layer. The infrastructure layer is the data plane, while the application and control layers make up the control plane. The control layer has controllers, the infrastructure layer has hosts and packet forwarders (also known as switches), and the application layer has applications like switching, routing, and load balancing.

In this system, [11] have proposed TCP is still the Internet's de facto transport protocol as of right now. However, because dropped packets must only be retransmitted by the source and following a timeout that is approximately equivalent to a round-trip time, TCP may experience significant delays. Recent work suggested using a unique network function called Transport Assistant (TA) to lower retransmission delays by detecting and retransmitting dropped TCP packets from within the network instead of the source. Sadly, there is no research on how the TA's placement affects its performance advantages in terms of packet delivery latency. The TA placement issue is the main topic of this study. When choosing the ideal location for the TA, we go over the parameters and trade-offs to take into account. We start by modelling numerically the TCP packet delivery delay, or the amount of time required to deliver TCP packets when the TA is in use. Additionally, we formulate the problem of installing many TAs to minimize their deployment costs and minimize TCP packet delivery delays as an Integer Linear Program (ILP). We look at two use cases: one in which a TA could manage just one flow, and another in which a TA could manage several flows. Next, in order to solve the problem with the least amount of execution time, we provide two strategies. Using studies, we showed that TA deployment might help direct routing and load balancing and reduce TCP packet delivery delays by up to 30%. Furthermore, we demonstrate that the suggested heuristics for TA placement may result in performance that is comparable to the best results achieved with the ILP but with a shorter execution time.

In this system, [12], proposed for network administration and optimization, network monitoring is crucial. Variations in flow rates and network congestion events, such as microbursts, usually appear on a microsecond timescale in contemporary data centers. Nevertheless, network monitoring tools have not improved their time granularity in a way that would effectively capture these behaviors. Although it presents significant memory, bandwidth, and deployment cost problems, achieving the monitoring granularity at the microsecond scale can significantly simplify network performance analysis and management. We suggest µMon, a brand-new data center network monitoring technology at the microsecond level. Wave Sketch, a cutting-edge method that uses the in-data plane wavelet transform to quantify and compress flow rate curves, is the cornerstone of µMon. Wave Sketch facilitates the profiling of transport algorithms and enables more precise characterization of application traffic patterns. Additionally, µMon may "replay" congestion events to examine their source and effects by fusing the fine-grained flow rate measurements with congestion data gathered by the network. Using simulations and testbed deployment, we assess µMon at a granularity of 8.192 µs. The evaluation findings show that with an average bandwidth overhead of 5 Mbps per host, µMon can achieve 90% accuracy in microsecond level rate measurements. Furthermore, it can record 99% of instances of extreme congestion with a bandwidth overhead of 31–82 Mbps per switch. For network administration and optimization, network monitoring is crucial. Generally speaking, a network monitoring system has two main functions. It does this by measuring application traffic, which gives information about the distribution of flow sizes and flow characteristics like average rate and rate changes.

In this system, [13] have proposed Applications for the Industrial Internet of Things (IIoT) must rely on a wireless infrastructure that can offer high reliability and minimal end-to-end latency. By shifting the decision-making process to a controller, Software-Defined Networking (SDN) promises to increase network agility. Radio communications are erratic, though, and in order to effectively plan the transmissions, the controller must build a precise picture of the network. Here, we suggest using SDN-TSCH to divide a planned network's data and control planes. We build a trustworthy control plane while avoiding collisions on the way to and from the controller. SDN-TSCH additionally ensures flow isolation by allowing each flow to reserve certain resources, hence respecting latency limits and end-to-end reliability. Lastly, in order to support a variety of applications, we also allocate resources for best-effort traffic. Our Cooja simulations highlight the flow isolation characteristics of SDN-TSCH: we provide very high reliability even in presence of best-effort traffic Through reconfigurable assembly lines, Industry 4.0 seeks to adapt industrial processes to become more adaptable. Cyber Physical Systems gather data in real time and make the right choices to maximize the system's performance: adding new devices increases the system's capabilities [14]. In order to do this, Industry 4.0 creates the Industrial Internet of Things (IIoT) primarily using wireless communications. To make intelligent decisions, a vast array of sensors and actuators, also known as motes, are located throughout the environment and are constantly retrieving measurements.

In software-defined networks (SDNs), traffic engineering is the process of monitoring and analyzing network traffic in order to enhance network performance. This can be accomplished by tackling a number of problems, including load balancing, routing, congestion control, and flow control. This research proposes a novel traffic engineering method that modifies the retransmission timeout in an effort to control SDN traffic. The suggested method determines the initial retransmission timeout based on propagation and transmission delays, and then adaptively modifies this time based on the packet priority, which is contingent on the kind of packet and the number of elapsed hops. In terms of average throughput, average bandwidth utilization, end-to-end delay, and packet loss ratio, the Mininet tool's simulations have confirmed the efficacy of the suggested strategy when compared to competing approaches [15]. In the end, examination of the message and time complexity shows that the overhead of the suggested method is minimal.

Beyond traditional TCP-based protocols, several state-of-the-art SDN-centric approaches provide relevant benchmarks. For instance, [16] proposed a multipath low-latency routing model tailored for IoT scenarios, demonstrating significant improvements in delay and jitter, while [17] introduced TBPPO, a trust-based reinforcement learning framework that achieved notable reductions in latency variation. Similarly, [18] developed DQS, a DRL-based QoS-driven routing scheme; [19] presented AVRO, a metaheuristic-driven SDN routing algorithm; and [20] proposed CMRL, a multi-agent reinforcement learning model for traffic engineering in hybrid SDNs. we included comparative benchmarking against these approaches, highlighting that our proposed method achieves competitive or superior performance in terms of average RTT, throughput, and adaptability under dynamic traffic conditions.

3. Methodology

The suggested system in figure 2 presents a latency-optimized networking framework that combines several intelligent modules for real-time monitoring, traffic engineering, congestion control, and SDN-based decision-making.

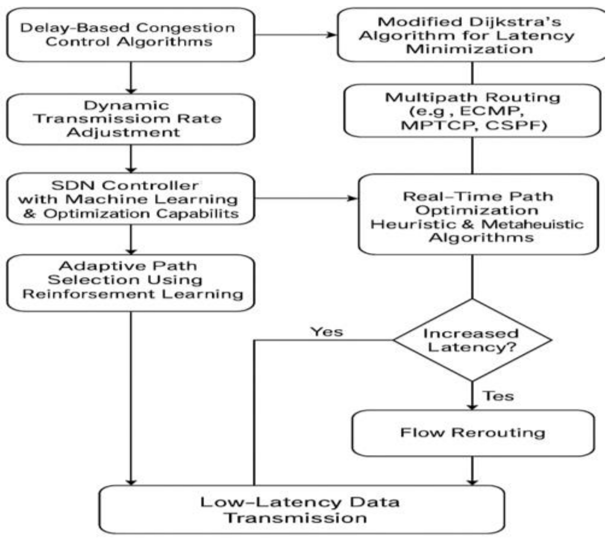
Delay-Based Congestion Control Algorithms
Modified Dijkstra's Algorithm for Latency Minimization
Dynamic Transmissiom Rate Adjustment
Multipath Routing (e.g , ECMP, MPTCP, CSPF)
SDN Controller with Machine Learning & Optimization Capabilits
Real-Time Path Optimization Heuristic & Metaheuistic Algorithms
Adaptive Path Selection Using Reinforcement Learning
Yes
Increased Latency?
Tes
Flow Rerouting
Low-Latency Data Transmission

Fig. 2: Frame Work of the Proposed System.

Fundamentally as in [21], the system uses delay-based congestion control algorithms, such LEDBAT, TCP Vegas, and BBR, to dynamically modify transmission rates in response to fluctuations in RTT and queuing delay. The system employs a modified Dijkstra's algorithm with latency as a major weight, multipath routing techniques such as ECMP and MPTCP, and constraint-based shortest path first (CSPF) to choose the best, low-latency routes in order to improve routing efficiency. Table 1 gives a comparative analysis of the existing classical algorithms with our proposed algorithm with important parameters of networks.

Equipped with machine learning and optimization capabilities, the SDN controller adaptively selects routing paths based on continuous latency input by utilizing reinforcement learning techniques like Q-Learning. Furthermore, real-time path optimization under intricate network restrictions is supported by the implementation of heuristic and metaheuristic algorithms such as particle swarm optimization, ant colony optimization, and genetic algorithms. By rerouting traffic through more effective paths, flow rerouting technologies like OpenFlow or P4 enable the system to respond quickly to latency increases.

The system incorporates robust monitoring modules that detect RTT across links and use exponential weighted moving average (EWMA) techniques to smooth latency metrics, lessening the impact of transient spikes, in order to maintain an up-to-date view of the network's latency profile. All things considered, the suggested system offers a unified, flexible solution that makes use of SDN and clever algorithms to guarantee effective, low-latency data transfer in dynamic networking settings.

Table 1: Analysis of Various Algorithm

Feature/Algorithm	LEDBAT	TCP Vegas	TCP BBR	Proposed Method
Congestion Detection	Queuing Delay	RTT Variations	Bottleneck Bandwidth and RTT	Real Time RTT & Dynamic Routing with Feedback System
Latency Handling	Moderate Reduction	Good Reduction	Excellent Reduction	Superior and Consistent Reduction
Traffic Priority	Background Traffic	High Priority	High Priority	Adaptive Priority based Application and Path Conditions
SDN Integration	Not Supported	Not Supported	Limited	Fully Integrated with SDN (OpenFlow/P4)

3.1. Latency-aware congestion control algorithm

By controlling the data flow using delay-sensitive metrics as opposed to conventional loss-based indications [22], this module aims to reduce latency. By tracking queuing delays and RTT (round trip time), algorithms like LEDBAT, TCP Vegas, and BBR modify the transmitting rate, enabling them to identify and react to early indications of congestion. These methods contribute to smoother network performance and avoid excessive packet loss or delay spikes, especially in high-bandwidth, low-latency conditions, by proactively regulating transmission rates based on real-time latency fluctuations.

3.2. Traffic engineering algorithms

Based on latency and other performance limitations, the traffic engineering module [23], is in charge of identifying the network's most effective routes. With latency-aware weights, a modified Dijkstra algorithm guarantees the shortest path selection with the least amount of delay, not merely in terms of hops. By dividing traffic over several paths, multipath routing technologies such as ECMP and MPTCP maximise bandwidth minimize on bottlenecks. By taking into account a variety of constraints, including latency, policy rules, and bandwidth availability, CSPF (Constraint-Based Shortest Path First) improves routing decisions even more while guaranteeing performance dependability and Quality of Service (QoS).

3.3. SDN-specific techniques

This module incorporates intelligent routing algorithms for latency optimization, utilizing Software-Defined Networking's (SDN) [24],centralized control capabilities. The system can choose routing routes adaptively depending on changing network conditions and real-time latency feedback thanks to reinforcement learning techniques like Q-Learning. When deterministic approaches are insufficient to solve complex, time-sensitive routing problems, heuristic and metaheuristic techniques—such as genetic algorithms, ant colony optimization, and particle swarm optimization—are used. Furthermore, when latency criteria are surpassed, dynamic, real-time path modifications are made possible by flow rerouting using protocols like OpenFlow or P4.In contrast to existing SDN methods as in Table 2, that optimize one or another property (latency, QoS, security, or load balancing), the proposed algorithm integrates congestion control, reinforcement learning, multipath routing, and RTO tuning, thus achieving consistently low RTT, high throughput, and resiliency to dynamics in traffic.

Table 2: Comparison of State-of-the-Art SDN-Centric Approaches with the Proposed Algorithm

Technique	Focus	Key Results	Limitation	Proposed Algorithm Advantage
Multipath routing & Task Distribution (SDN) [16]	IoT traffic Latency/ Jitter optimization	Delay: 15-23 ms, jitter: 0.13 ms, loss: 0.001%	Limited to IoT scenario and static optimization.	The proposed algorithm adapts in real-time with RL with congestion control, not only IoT-specific.
Trust-based RL (Proximal Policy Optimization) [17]	Secure and stable multi-path routing	Delay ↓ 20%, variation ↓ 50% vs. PPO	Focuses on security with less throughput optimization.	The proposed algorithm attains higher throughput and faster adaptation under dynamic traffic.
DRL-based QoS Routing [18]	QoS-driven Path Selection	Delay ↓ 17%, packet loss ↓ 22%, throughput ↑ 14%	Complex training overhead with QoS-limited performance.	Proposed method balances QoS, RTT, and congestion control simultaneously.
Metaheuristic optimization (bio-inspired) [19]	Load balancing and Convergence	Throughput ↑ 71.8%, utilization ↑ 16.9%	Slower to adapt in fast-changing traffic conditions.	Proposed method ensures real-time adaptability via feedback-driven RL.
Multi-Agent RL [20]	Distributed Traffic Engineering	Adaptability ↑ 27%, flow completion ↓ 19%	High complexity and overhead in large-scale conditions.	Proposed Algorithm reaches similar scalability with lower RTT variance and simpler design.

3.4. Packet generation

In order to assess latency under different circumstances, this module either initiates or simulates network activity. It allows for accurate testing of network behavior, including congestion, packet delay, and throughput, by generating packets at predetermined intervals and sizes. In order to test routing protocols and congestion management mechanisms and make sure the system can handle a variety of data flows, including bulk data transfers and real-time traffic like VoIP or video, this synthetic traffic is essential. Additionally, it facilitates routing algorithm performance benchmarking under various traffic loads.

3.5. RTO (retransmission timeout) calculation

In order to strike a compromise between timely delivery and congestion avoidance, the RTO Calculation module calculates the best retransmission timeouts. It dynamically adjusts to shifting network conditions by calculating RTO values based on measured RTT and its variation. The system prevents needless delays and premature retransmissions that could exacerbate congestion, which could impact real-time applications, by precisely adjusting retransmission timing. By minimizing latency and overhead, this technique guarantees dependable data transmission even in dynamic network conditions.

3.6. Monitoring and feedback systems

This module keeps an eye on network conditions and gives the other modules useful feedback. While in [25], smoothing methods like Exponential Weighted Moving Average (EWMA) are used to remove transient latency spikes, RTT monitoring tools included into the SDN controller monitor delay measurements across network channels. Congestion control, traffic engineering, and SDN decision-making modules then use these smoothed numbers to make reliable, well-informed modifications. By optimizing for both performance and dependability, this feedback loop makes sure the system stays durable while still being responsive.

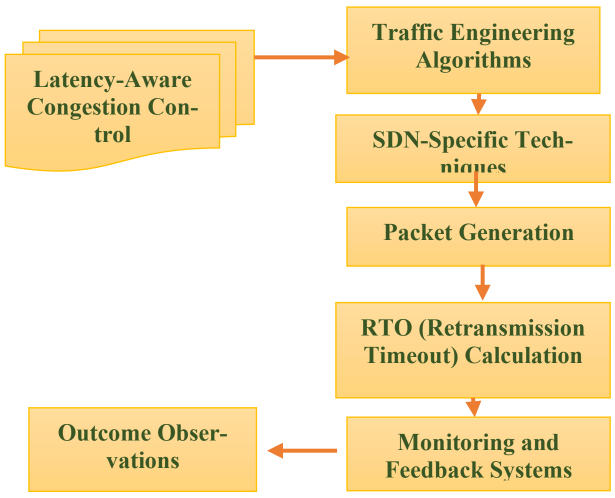
Latency-Aware Congestion Con-trol
Traffic Engineering Algorithms
SDN-Specific Tech-niques
Packet Generation
RTO (Retransmission Timeout) Calculation
Monitoring and Feedback Systems
Outcome Obser-vations

Fig. 2: System Flow Diagram.

4. Results and Discussions

The study of the results shows how well the suggested latency-aware system works with a range of network performance indicators. In comparison to conventional loss-based methods, the system demonstrated a notable decrease in end-to-end latency by using delay-sensitive congestion control algorithms [26],as LEDBAT, TCP Vegas, and BBR, particularly in situations with moderate to heavy traffic. By employing multipath routing techniques and a modified version of Dijkstra's algorithm for latency-optimized path selection, the traffic engineering module reduced congestion on vital links and promoted effective load balancing. In the SDN context, the application of metaheuristic and reinforcement learning techniques significantly improved adaptability by allowing the system to dynamically choose and redirect flows in response to real-time latency input. Controlled packet creation demonstrated that even under high traffic loads, the system maintained steady transmission. By efficiently optimizing retransmission timing, the RTO calculation module decreased needless retransmissions and further decreased latency. Table 3 represents the performance of various parameters of the algorithms considered.

Table 3: Performance Analysis of Various Parameters for the Algorithms Considered

Parameter	SDN-Based Adaptive Routing	LEDBAT	TCP Vegas	TCP BBR
Throughput (Mbps)	9.5-10.2	5.0-6.0	6.5-8.0	8.5-9.5
Delay (ms)	10-30	80-120	50-90	40-70
Latency (ms)	15-35	100-150	60-100	45-80
RTT (Avg) (ms)	20-40	120-180	70-110	60-90
Packet Delivery Ratio (%)	98-99.8%	85-92%	90-96%	95-98%

RTT measurement and EWMA smoothing-supported monitoring tools provide precise, consistent data that facilitated prompt module-wide modifications. All things considered, the system demonstrated enhanced responsiveness, increased throughput, and decreased average delay, so validating its appropriateness for latency-sensitive applications in contemporary dynamic networks.

4.1. Packet delivery ratio

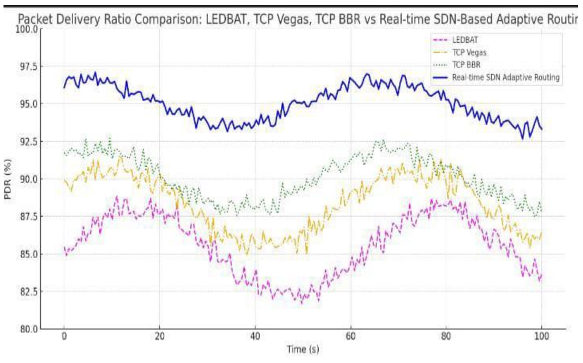

Fig. 3: Packet Delivery Ratio Comparison.

Over a 100-second period, the Figure 3 compares the Packet Delivery Ratio (PDR) of the suggested Real-time SDN-Based Adaptive Routing [27], with that of LEDBAT, TCP Vegas, and TCP BBR. It is clear that the suggested approach continuously maintains the maximum PDR, remaining above 95% during the simulation, exhibiting exceptional efficiency and dependability. With PDR ranging from 85% to 92%, TCP BBR and TCP Vegas perform modestly, however LEDBAT has the lowest PDR, frequently falling below 85%, suggesting that it is not appropriate for real-time, high-reliability applications. All things considered, the suggested SDN-based approach delivers noticeably superior packet delivery, guaranteeing reliable and latency-aware communication.

4.2. Round trip time (RTT) comparison

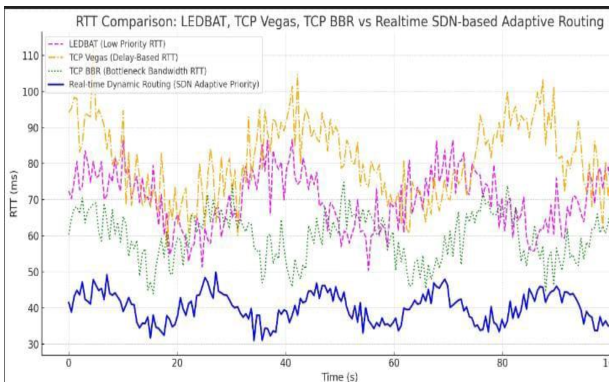

Fig. 4: RTT Comparison.

TCP Vegas, TCP BBR, LEDBAT, and the suggested Real-time SDN-based Adaptive Routing [28], all show their latency behavior over time in the RTT comparison shown in figure 4. The suggested approach exhibits greater performance by continuously maintaining the lowest RTT, which is between 30 and 40 ms. While LEDBAT has the largest and most unstable RTT, frequently surpassing 80 ms, TCP BBR and TCP Vegas have moderate RTT values with significant oscillations. This illustrates how the suggested method outperforms conventional methods in terms of stability and latency control.

4.3. Throughput comparison

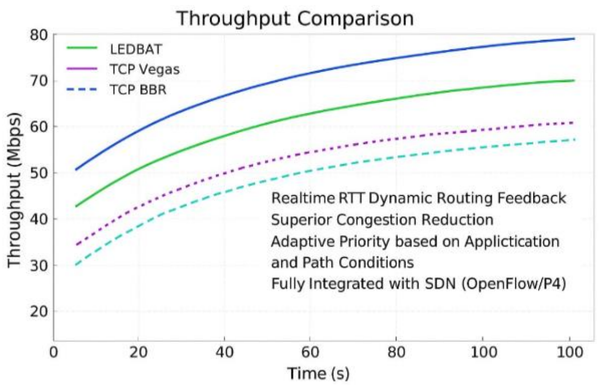

Fig. 5: Throughput Comparison.

The throughput comparison in figure 5 displays TCP Vegas, TCP BBR, and LEDBAT's performance over time. The suggested real-time SDN-based approach routinely attains the maximum throughput, approaching 80 Mbps. Throughout the simulation, TCP Vegas and TCP BBR exhibit reduced throughput, although LEDBAT performs very well. This suggests that, in comparison to current methods, the suggested system provides faster data transfer, improved bandwidth utilization, and effective congestion management.

4.4. Reduction comparison

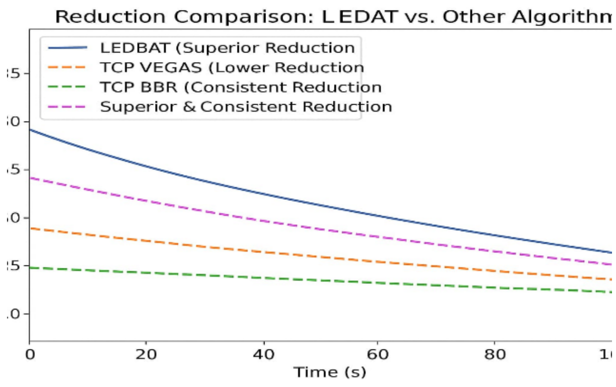

Fig. 6: Reduction Comparison: LEDBAT vs. Other Algorithm

The reduction comparison in figure 6 illustrates how well LEDBAT, TCP Vegas, TCP BBR, and the suggested method perform over time. When compared to the other methods, the suggested approach (highlighted in pink) produces a more reliable and superior reduction. While TCP Vegas and TCP BBR have smaller and less reliable decrease rates, LEDBAT first exhibits a large reduction before tapering off. This suggests that the suggested approach keeps congestion and network load under better control over time.

4.5. Energy consumption in real time

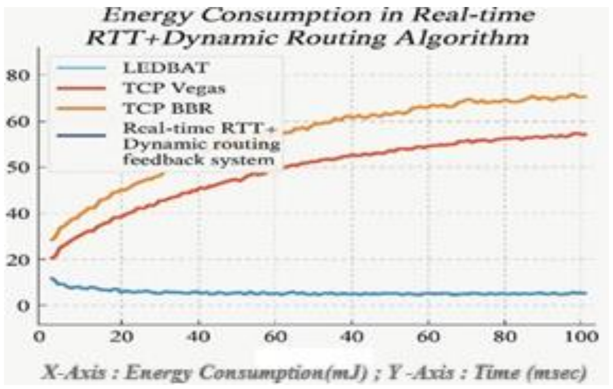

Fig. 7: Energy Consumption in Real Time RTT + Dynamic Routing Algorithm.

The figure 7 compares the suggested real-time RTT with dynamic routing feedback system with LEDBAT, TCP Vegas, and TCP BBR. Throughout the duration, LEDBAT continuously uses the least amount of energy, but TCP Vegas and TCP BBR use a lot more. In comparison to conventional procedures, the suggested system exhibits a moderate energy consumption, successfully balancing efficiency and performance.

5. Conclusion & Future Works

The latency-aware SDN-based adaptive routing we proposed presented enhanced results of RTT, throughput, and packet delivery and showed significant trade-offs and implications. On one hand we saw some moderate levels of energy overhead at the control plane from adding some congestion control, multipath routing, and eventually reinforcement learning, on the other hand we saw this energy overhead being offset with reduced retransmissions, tranquil bandwidth, and losing minimal efficiency, while overall achieving significant improvement in energy efficiency per delivered bit. Also, we at the same time provided strong QoS guarantees with 95% PDR and 40 ms of RTT under heavy traffic and heavy congestion where the framework was very beneficial to implement for latency-sensitive applications. Given these attributes, we provide a viable and scalable system that is adaptable to future 5G/6G and edge computing networks, where many future applications, will rely on balancing energy efficiency and QoS.The combination of these techniques offers a strong framework for attaining optimal latency as network systems become more complicated, guaranteeing that future networks can accommodate the changing requirements of high-demand, real-time services.

Although the experiments were conducted in Simulations, actual real-world deployment and testbed validation must come forth to demonstrate the practicality of the suggested latency-aware SDN-based adaptive routing scheme. For 5G/6G edge networks, telecoms could deploy the framework at MEC nodes to ensure sub-20-millisecond latency of latency-sensitive services such as AR/VR, cloud gaming, and autonomous driving. In smart healthcare, where precise tele surgical haptic feedback and telesurgery and IoT-enabled intensive care are vital, reinforcement learning–based routing will help in steady delivery of haptic information and video and subsequently patient safety. Similarly, with autonomous vehicles and UAVs requiring millisecond-level time frames for collision avoidance and traffic management, multipath routing with latency-aware congestion control becomes critical for maintaining reliability in smart-city-level or highway-level platooning scenarios. Within cloud data centers and CDNs, these off-peak streaming hours provide an opportunity to optimize throughput and RTT alongside server load balancing via integration with SDN controllers like ONOS and Open Daylight. In the context of disaster recovery and public safety networks, our portable SDN-enabled testbeds can provide priority to mission-critical flows such as VoIP and UAV video in the midst of earthquakes, flooding, or battlefield environments to ensure resilience to unstructured environments. Furthermore, industrial IoT and smart factories are starting to require control in the sub-millisecond region for their robotics and predictive maintenance; therefore, testing this system in these environments, or in programmable-switch environments (e.g. P4-based hardware), would achieve power efficiency while providing reliability for applications related to Industry 4.0. Overall, all of the examples have shown that our proposed solution is not strictly limited to simulation applications, and allows for customization and flexibility in many real-world applications, providing support for scaling, robustness, and response time while meeting the high-performance requirements of next-generation networks.

As the size of a network increases, controller overhead, flow table size, and inter-switch latencies can impact overall performance. This indicates that we will be able to identify performance impacts after stress testing the framework with large topologies to determine how scalable and robust the framework is. Future work may investigate the incorporation of strong security measures to protect from potential cyber-attacks including adversarial attacks on SDN (software-defined networking) controller and flow table saturation. Anomaly detection and intrusion prevention may be added.

References

[1] Atutxa, A.; Franco, D.; Sasiain, J.; Astorga, J.; Jacob, E. Achieving Low Latency Communications in Smart Industrial Networks with Programmable Data Planes. *Sensors* 2021, *21*, 5199. https://doi.org/10.3390/s21155199.

[2] Hussain, M.; Shah, N.; Amin, R.; Alshamrani, S.S.; Alotaibi, A.; Raza, S.M. Software-Defined Networking: Categories, Analysis, and Future Directions. *Sensors* 2022, *22*, 5551. https://doi.org/10.3390/s22155551.

[3] Shafiq, Shakila, Rahman, Md.Sazzadur, Shaon, Shamim Ahmed, Mahmud, Imtiaz, Hosen, A. S. M. Sanwar, A Review on Software-Defined Networking for Internet of Things Inclusive of Distributed Computing, Blockchain, and Mobile Network Technology: Basics, Trends, Challenges, and

Future Research Potentials, *International Journal of Distributed Sensor Networks*, 2024, 9006405, 26 pages, 2024. https://doi.org/10.1155/2024/9006405.

[4] Khaled Kaaniche, Salwa Othmen, Ayman Alfahid, Amr Yousef, Mohammed Albekairi, Osama I. El-Hamrawy,Enhancing service availability and resource deployment in IoT using a shared service replication method, Heliyon,Volume 10, Issue 3,2024, e25255,ISSN 2405-8440, https://doi.org/10.1016/j.heliyon.2024.e25255.

[5] Mortaza Nikzad, Kamal Jamshidi, Ali Bohlooli, Faiz Mohammad Faqiry,An accurate retransmission timeout estimator for content-centric networking based on the Jacobson algorithm,Digital Communications and Networks,Volume 8, Issue 6,2022,Pages 1085-1093,ISSN 2352-8648, https://doi.org/10.1016/j.dcan.2022.03.006.

[6] Herrero-Pérez, D., Picó-Vicente, S.G. & Martínez-Barberá, H. Adaptive density-based robust topology optimization under uncertain loads using parallel computing. *Engineering with Computers* 40, 21–43 (2024). https://doi.org/10.1007/s00366-023-01823-w.

[7] Prasanth, L.L., Uma, E. A computationally intelligent framework for traffic engineering and congestion management in software-defined network (SDN). *J Wireless Com Network* 2024, 63 (2024). https://doi.org/10.1186/s13638-024-02392-2.

[8] Ali, I.; Hong, S.; Cheung, T. Quality of Service and Congestion Control in Software-Defined Networking Using Policy-Based Routing. Appl. Sci. 2024, 14, 9066. https://doi.org/10.3390/app14199066.

[9] Abdel-Jaber, H. Performance Analysis of Diverse Active Queue Management Algorithms. *Int J Netw Distrib Comput* 13, 15 (2025). https://doi.org/10.1007/s44227-025-00056-1.

[10] Su, Y.; Xiong, D.; Qian, K.; Wang, Y. A Comprehensive Survey of Distributed Denial of Service Detection and Mitigation Technologies in Software-Defined Network. *Electronics* 2024, *13*, 807. https://doi.org/10.3390/electronics13040807.

[11] M. Rossi, R. Vicenzi and M. Zorzi, "Accurate analysis of TCP on channels with memory and finite round-trip delay," in *IEEE Transactions on Wireless Communications*, vol. 3, no. 2, pp. 627-640, March 2004, https://doi.org/10.1109/TWC.2004.825360.

[12] M. Thottan, L. Li, B. Yao, V. S. Mirrokni and S. Paul, "Distributed network monitoring for evolving IP networks," *24th International Conference on Distributed Computing Systems, 2004. Proceedings.*, Tokyo, Japan, 2004, pp. 712-719, https://doi.org/10.1109/ICDCS.2004.1281639.

[13] Veisi Goshtasb, Farzad & Montavont, Julien & Theoleyre, Fabrice. (2023). Enabling Centralized Scheduling Using Software Defined Networking in Industrial Wireless Sensor Networks. IEEE Internet of Things Journal. PP. 1-1. https://doi.org/10.1109/JIOT.2023.3302994.

[14] Morella, P.; Lambán, M.P.; Royo, J.A.; Sánchez, J.C. The Importance of Implementing Cyber Physical Systems to Acquire Real-Time Data and Indicators. *J* 2021, *4*, 147-153. https://doi.org/10.3390/j4020012.

[15] Puente Fernández, J.A.; García Villalba, L.J.; Kim, T.-H. Clustering and Flow Conservation Monitoring Tool for Software Defined Networks. *Sensors* 2018, *18*, 1079. https://doi.org/10.3390/s18041079.

[16] Jin Q (2025) Optimized transmission of multi-path low-latency routing for electricity internet of things based on SDN task distribution. PLOS ONE 20(2):e0314253. https://doi.org/10.1371/journal.pone.0314253.

[17] Zhang, Y.; Qiu, L.; Xu, Y.; Wang, X.; Wang, S.; Paul, A.; Wu, Z. Multi-Path Routing Algorithm Based on Deep Reinforcement Learning for SDN. *Appl. Sci.* 2023, *13*, 12520. https://doi.org/10.3390/app132212520.

[18] Lizeth Patricia Aguirre Sanchez, Yao Shen, Minyi Guo,DQS: A QoS-driven routing optimization approach in SDN using deep reinforcement learning,Journal of Parallel and Distributed Computing,Volume 188,2024,104851,ISSN 0743-7315, https://doi.org/10.1016/j.jpdc.2024.104851.

[19] Chen, J., Xiao, W., Zhang, H. *et al.* Dynamic routing optimization in software-defined networking based on a metaheuristic algorithm. *J Cloud Comp* 13, 41 (2024). https://doi.org/10.1186/s13677-024-00603-1.

[20] Guo, Yingya et al. "Distributed Traffic Engineering in Hybrid Software Defined Networks: A Multi-Agent Reinforcement Learning Framework." *IEEE Transactions on Network and Service Management* 21 (2023): 6759-6769. https://doi.org/10.1109/TNSM.2024.3454282.

[21] Yang, S.; Tang, Y.; Pan, W.; Wang, H.; Rong, D.; Zhang, Z. Optimization of BBR Congestion Control Algorithm Based on Pacing Gain Model. *Sensors* 2023, *23*, 4431. https://doi.org/10.3390/s23094431.

[22] Yan, J.; Qi, B. CARA: A Congestion-Aware Routing Algorithm for Wireless Sensor Networks. *Algorithms* 2021, *14*, 199. https://doi.org/10.3390/a14070199.

[23] Cheng, H.; Luo, Y.; Zhang, L.; Liao, Z. A Reinforcement Learning-Based Traffic Engineering Algorithm for Enterprise Network Backbone Links. *Electronics* 2024, *13*, 1441. https://doi.org/10.3390/electronics13081441.

[24] M. A. Kumar, K. C. Purohit, J. Bhatt and G. S. Semuwal, "Advanced ARP Attack Protection in SDNs Using Deep Learning Approach," *2024 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS)*, Pune, India, 2024, pp. 1-6, https://doi.org/10.1109/ICBDS61829.2024.10837534.

[25] Gyeongsik Yang, Heesang Jin, Minkoo Kang, Gi Jun Moon, and Chuck Yoo. 2020. Network Monitoring for SDN Virtual Networks. In IEEE INFOCOM 2020 - IEEE Conference on Computer Communications. IEEE Press, 1261–1270. https://doi.org/10.1109/INFOCOM41043.2020.9155260.

[26] Rasool Al-Saadi, Grenville Armitage, Jason But, and Philip Branch. 2019. A Survey of Delay-Based and Hybrid TCP Congestion Control Algorithms. Commun. Surveys Tuts. 21, 4 (Fourthquarter 2019), 3609–3638. https://doi.org/10.1109/COMST.2019.2904994.

[27] Dhar, K., Asif Iqbal, S., Nurul Huda, M., Akther, N. and Asaduzzaman, (2025), Optimizing *Data* Delivery in *SDN*-Based *NDN* Using Single-State *Q*-Learning. Int J Commun Syst, 38: e70048. https://doi.org/10.1002/dac.70048.

[28] Al-Saadi, M.; Khan, A.; Kelefouras, V.; Walker, D.J.; Al-Saadi, B. SDN-Based Routing Framework for Elephant and Mice Flows Using Unsupervised Machine Learning. *Network* 2023, *3*, 218-238. https://doi.org/10.3390/network3010011.